A Data-Driven Model Selection Approach to Spatio-Temporal Prediction*

Rocío Zorrilla[1], Eduardo Ogasawara[2],Patrick Valduriez[3], Fábio Porto[1]

[1]Laboratório Nacional de Computação Científica - LNCC

[2]Centro Federal de Educação Tecnológica Celso Sukow da Fonseca - CEFET-RJ

[3]INRIA & LIRMM

{romizc,fporto}@lncc.br, eogasawara@ieee.org, Patrick.Valduriez@inria.fr

Abstract. *Spatio-temporal Predictive Queries encompass a spatio-temporal constraint, defining a region, a target variable, and an evaluation metric. The output of such queries presents the future values for the target variable computed by predictive models at each point of the spatio-temporal region. Unfortunately, especially for large spatio-temporal domains with millions of points, training temporal models at each spatial domain point is prohibitive. In this work, we propose a data-driven approach for selecting pre-trained temporal models to be applied at each query point. The chosen approach applies a model to a point according to the training and input time series similarity. The approach avoids training a different model for each domain point, saving model training time. Moreover, it provides a technique to decide on the best-trained model to be applied to a point for prediction. In order to assess the applicability of the proposed strategy, we evaluate a case study for temperature forecasting using historical data and auto-regressive models. Computational experiments show that the proposed approach, compared to the baseline, achieves equivalent predictive performance using a composition of pre-trained models at a fraction of the total computational cost.*

1. Introduction

Successfully predicting the behavior of spatio-temporal phenomena based on past observations is essential for a wide range of scientific studies and real-life applications like precipitation nowcasting [Souto et al. 2018], and climate alert systems [Murat et al. 2018]. In support of these applications, traditional data processing and time series analysis approaches generate predictive models that aim for predictive accuracy at the cost of high execution time and utilization of computational resources [Hassani and Silva 2015].

More recently, a new class of systems, known as prediction serving systems, has emerged to support trained models scheduling warranting performance and run-time efficiency [Ghanta et al. 2019, Polyzotis et al. 2018]. For spatio-temporal phenomena, the focus of this paper, expressing a predictive query, involves specifying spatio-temporal constraints that define a region, a target variable whose values are to be inferred, and an evaluation metric for the performance of the predictive query. The query outcome then

*The authors thanks CAPES, CNPq, and FAPERJ for partially supporting the paper. This work is developed in the context of the HPDaSc INRIA-Brazil Associated Team.

exhibits the target variable's future values on the specified region, computed by predictive models that meet the metric evaluation threshold.

However, we argue that building a query plan to answer a spatio-temporal predictive query is hard from several perspectives. Among them, we are interested in the model selection and allocation problem: for a given spatio-temporal query region, a serving system must automatically build an appropriate plan that chooses between training models or pick pre-trained models for each query region spatial position.

We adopt a data-driven approach to guide the model selection problem. Considering the availability of historical data, the approach pre-processes the data by grouping sequences of the domain using a shape-based similarity measure, which only considers the temporal dimension. The approach trains time series models at each group's representatives sequence. It uses sequence shape similarity between points in the query region to identify candidate models. Finally, it uses a model recommendation strategy to indicate the ones that meet the metric evaluation criteria.

Our experiments explore the robustness of the domain partitioning and the predictive performance of the proposed model composition used to answer spatio-temporal predictive queries. Results indicate comparable predictive quality using a model composition based on cluster representatives, with a fraction of the computational cost. Moreover, our experiments show that a single clustering strategy, with a fixed number of partitions, may not fully reflect the spatial variations of time series shape throughout the data domain. We adopt a time series classification approach, using a deep learning model, to further improve the model selection.

The remaining of this paper is structured as follows. In Section 2, we describe the problem formulation; in Section 3, we introduce our proposal to tackle the problem described; in Section 4, we show the experimental results; in Section 5 we discuss related works; and finally, conclusions and future works are given in Section 6.

2. Problem Formulation

Let $\mathcal{D} = \{((x, y), s)$, with $(x, y) \in \mathbb{R}^2$ and $s = (s_1, s_2, \ldots, s_T)$ denote a univariate time series (u.t.s) with T time units $\}$, $\mathcal{D}$ represents a spatio-temporal domain. Let $\mathcal{G} = \{g_1, g_2, \ldots\}$ be a set of predictive models, based on forecasting techniques, that were trained with different u.t.s. $s \in \mathcal{D}$. Each model $g \in \mathcal{G}$ is represented as:

$$g = \langle s, A, \mathbf{p}, E_g, \Sigma_g \rangle, \tag{1}$$

where:

- s: input sequence (t.s.) divided in training, validation and test sub-sequences,
- A: forecasting technique,
- $\mathbf{p}$: parameters for the forecast technique,
- E_g: in-sample error [Hastie et al. 2009],
- Σ_g: implementation/execution quality metrics.

We use $g(s, t_p, t_f) = (s_{T+1}, \ldots, s_{T+t_f})$ to represent a forecast of t_f time units of s, indicating that t_p time units were used as validation t.s. to compute E_g. In this context, we are interested in processing a spatio-temporal predictive query (STPQ) Q:

$$Q = \langle R, t_p, t_f, Q_m \rangle, \tag{2}$$

where:

- R: represents the spatial region of interest,
- t_p: $\{s_{T-t_p-1}, \ldots, s_T\}$ validation time units,
- t_f: $\{s_{T+1}, \ldots, s_{T+t_f}\}$ forecast time units ($t_f \geq 1$),
- Q_m: evaluation metric for the predictive output.

We assume $\langle MSE\left\{E_g; s \in R\right\}, t_{train}, t_{eval}\rangle$ as an evaluation metric, bounded by Q_m. Thus, we focus on providing an efficient solution to selecting pre-trained models to compose an answer to a STPQ.

3. Our Proposal

Given the problem formulation described in Section 2, a possible solution could be to pre-train a predictive model for each t.s. in $\mathcal{D}$. It is sub-optimal as many points would never be queried, and as the t.s. change, the models need to be re-trained. Another costly option would be to train models at the query region points in run-time.

We propose a data-driven model selection approach that focuses on grouping historical data representing the behavior of the target variable in the domain. We argue that, by considering only a set of models generated over a t.s. representative, which generalizes the shape similarity of other t.s. in the domain, it is possible to preserve a predictive quality comparable to the baseline approach of using a model for every t.s.

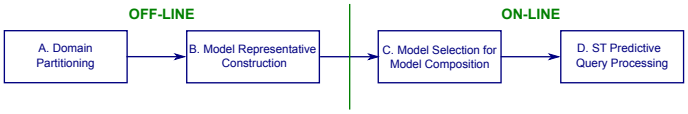

Figure 1. A two phase query processing approach

The approach is divided into two phases, offline and online (Figure 1). The offline phase comprises two steps: (A) the domain partitioning, based on t.s. clustering techniques; (B) the construction of predictive models at each group t.s. representative. The online phase is applied when processing an STPQ, comprises: (C) a process to select a set of pre-trained representative models, to schedule and run them; (D) an approach to compose the query output using forecasts of models allocated to every query region point.

The offline phase is also responsible for storing the domain partitioning and the pre-trained models for later retrieval in the online phase. This reduces the computational workload if we were to train a model on each point of a query region at run-time.

3.1. Domain Partitioning

This step aims to: partition the domain into groups with high shape similarity; and find a representative for each group. The k-medoids clustering algorithm can minimize local dissimilarity of each group, and yields an existing t.s. in the dataset as representative (medoid) [Liao 2005]. In this paper, the choice of the number of groups k aims to produce predictive models with accurate forecasts for similar t.s.

3.2. Model Representative Construction

In order to answer STPQ with acceptable predictive error, we consider using a model trained at each medoid. We refer to these k models as representative models and are

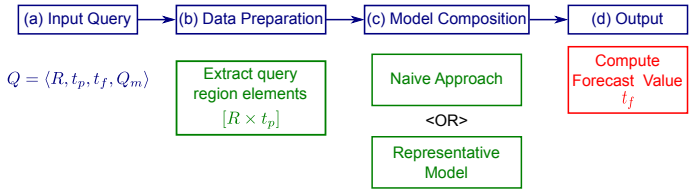

Figure 2. On-Line STPQ processing.

computed during the offline phase as follows. Let's assume a medoid series has size T. We first train a predictive model using $(T - t_p)$ time units and validate it with the immediate sequence of t_p time units, to compute the forecast error E_g. This model is then re-trained, including the t_p sub-sequence, becoming the representative model that can be used to make predictions of t_f time units for all t.s. in its group that fall within a particular query region.

3.3. Model Selection for Model Composition

We define "Model Composition" as the subset of predictive models that can compute the forecast value of each element in a region of interest R of the domain. The justification to implement this step is based on the intrinsic properties of the spatio-temporal data: the consistency and auto-correlation on nearby points in the domain makes difficult the task of finding an 'optimal' number of groups (k) [Liao 2005]. Within this step, we hypothesize that, if the representative predictive models manage to adequately predict a group of elements with similar shape patterns, then these models will allow us to obtain a prediction for a region of interest of the domain. We consider a model selection process based on the following strategies:

- Naive Approach (baseline): for each t.s. s_j in each group, we train its model g_j and calculate the corresponding forecast error. This costly strategy generates as many models as there are t.s. in the region.
- Representative Models: we propose that, given the t.s. representative in each group, we train its corresponding model in order to predict future values for each element in the group and evaluate a corresponding generalization error.

3.4. Spatio-Temporal Predictive Query Processing

The online phase is depicted in Figure 2, and described as follows:

(a) The query region R and the time units t_p (past) and t_f (future) are parsed from the input query.
(b) A $[R \times t_p]$ spatio-temporal sub-region is extracted from the original dataset, associating a t.s. of t_p time units for each point in R.
(c) A model composition is created using data about the domain partitioning from the offline phase. Algorithm 1 considers two strategies for model selection: (i) train a predictive model on each point in R, and (ii) intersect the query region R with the groups to find the representatives for every t.s. and load the pre-trained models.
(d) With the model composition of the previous step, the requested forecast for the t_f steps for each t.s. in R is computed using its corresponding representative model. Here, we highlight that the same model can generate different forecasts for different t.s., provided that the t.s. undergo a data transformation (e.g., normalization).

Algorithm 1 Apply a Model Selection Strategy

```
1:  function SELECT_MODEL_COMPOSITION(D, selection_id, t_p)
2:      model_comp ← ⊥
            /* Model Composition with Naive Approach */
3:      if is_naive_selection(selection_id) then
            /* Let model at each element predict its own element */
4:          model_comp ← load_trained_models_each(D, t_p)
5:      end if
            /* Model Composition with Representative Models */
6:      if is_representative_selection(selection_id) then
            /* User needs to supply value for k of partitioning scheme */
7:          k ← get_k_for_request(selection_id)
            /* Retrieve previously trained models at each representative */
8:          (medoids_with_models, D_part) ← load_models_at_medoids(D, k, t_p)
9:          for m ∈ medoids_with_models do
            /* Retrieve the elements associated to current representative
10:             cluster ← elements_represented_by(m, D_part)
            /* Let model at current representative predict these elements */
11:             model_comp ← set_predictor(cluster, m, model_comp)
12:         end for
13:     end if
14:     Return model_comp
15: end function
```

The online procedure takes as input the domain, the query parameters, and the model selection strategy. Then, for each element in the query region, the model composition obtained indicates which model performs the forecast.

4. Experiments and Results

We evaluate the following aspects of our proposal: the domain partitioning, the predictive quality of the representative models, the model composition and the query performance.

Experimental dataset. We use a subset of the Climate Forecast System Reanalysis (CFSR) dataset, which contains four daily air temperature observations from January 1979 to December 2015 covering the space between 8N-54S latitude and 80W-25W longitude [Saha et al. 2011]. We subset this data to include one year of readings in the Brazilian territory, then transform each t.s. into the tuple (latitude, longitude, daily average temperature values), with dimensions (90, 90, 365).

Computational environment. We use a Dell PowerEdge R730 server with 2 Intel Xeon E5-2690 v3 2.60GHz CPUs, 768GB RAM, running Linux CentOS 7.7.

4.1. Domain Partitioning Evaluation

We implemented k-medoids using the DTW similarity measure [Sakoe and Chiba 1978]. Computing k-medoids requires pairwise distances, which can be calculated beforehand as a 2-d matrix. We perform this expensive computational process only once.

We vary the number of groups from $k = 2$ up to $k = 150$ with a stride of two and calculate the Within-cluster Sum of Squares (WSS) for each value of k, obtaining a monotonically decreasing trend for WSS. This makes the choice of k difficult, a known problem for high volumes of data with low variability throughout neighbor points [Liao 2005]. Thus we consider three methods: the elbow method [Liao 2005], silhouette index [Rousseeuw 1987] and a fitting of the WSS curve by using a smooth cubic spline. These results are summarized in Table 1. Using a cubic spline, we can find the minimum value for the second derivative by fitting the values. We argue that this method is more

Table 1. Methods to find the optimal value for k.

Method	Optimal k
Elbow	4
Silhouette	8
Cubic spline for WSS	66

appropriate for our dataset, as the splines smoothed the variations that were preventing to find a higher value for k.

4.2. Predictive Quality of Models at Representatives

Here, we evaluate the accuracy of the forecasts computed on the test sub-sequence (t_f) by comparing them against the observational values available. We consider the Symmetric Mean Absolute Percentage Error (sMAPE) for forecast error evaluation and the Mean Squared Error (MSE) [Hyndman and Koehler 2006] for accumulated forecast.

In this section, we evaluate the predictive quality of Auto-Regressive Integrated Moving Average (ARIMA) models [Box and Jenkins 1976], enhanced with auto.ARIMA [Hyndman and Khandakar 2008] for parameter selection. These models fit the description in Section 2 and offer good trade-off between predictive accuracy and computational cost.

4.2.1. Evaluation of sMAPE Forecast Error

Considering the domain partitioning with $k = 8$, we have eight groups with 1013 ± 617 t.s. on average, yielding eight representative models. Using scatter plots diagrams of intra-cluster similarity and forecast error, we find that, for the group index zero (Figure 3.a), the maximum sMAPE value was the lowest among the eight groups. Conversely, for the group index four (Figure 3.b), the maximum sMAPE value was the highest.

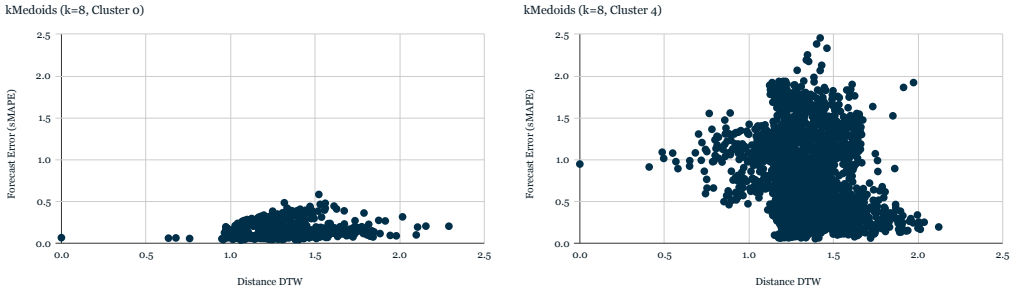

Figure 3. Forecast error for: (a) group 0: 0.16 ± 0.07 **(b) group 4:** 0.72 ± 0.35**.**

We don't observe a clear correlation between the DTW distances and the forecast error: as k increases, there is a tendency to obtain groups with more similarity between their elements (lower DTW distance) and also the predictions tend to be more accurate. It is noteworthy that lower values of k (8, 66) can produce some representatives that offer better predictions than, for example, the 'worst' (highest forecast error) representatives of the partitioning scheme with $k = 132$. Both these observations indicate that different spatial areas may need more precise partitioning than others.

4.2.2. Evaluation of MSE Forecast Error

Here we are interested in evaluating the MSE metric computed when forecasting an entire group of domain partitioning. We compare the following approaches:

- Naive Approach (baseline): for every t.s. in each group, train a model and calculate the forecast error. Then for each group, compute its corresponding MSE.
- Representative Models: given the k corresponding models for the representatives in a domain partitioning, use its representative model to forecast future values. Finally, compute the accumulated MSE values.

Table 2 shows the results of the MSE evaluation for $k = 8$. The columns are as follows: (1) cluster ID; (2) elapsed time to train models for all the t.s. in the group (baseline); (3) elapsed time to forecast t_f future units for the t.s. in the group (baseline); (4) accumulated MSE value for the baseline; (5) accumulated MSE value for the Representative Models; (6) percentage change of the MSE values between the approaches.

Table 2. Forecast Error Analysis with $k = 8$ and $t_f = 8$

cid	T. Train.(s)	T. For. (s)	Naive	Repr. Models	Δ (%)
0	2041.469	1.069	0.170	0.185	8.82
1	3447.608	1.299	0.689	0.926	34.38
2	2011.441	0.880	0.581	0.678	16.70
3	2685.912	1.238	0.413	0.492	19.13
4	14542.318	5.727	0.785	0.838	6.75
5	3231.718	1.375	0.407	0.437	7.37
6	1930.740	0.957	0.157	0.203	29.30
7	1811.335	0.853	0.388	0.551	42.01

We observe that the MSE of the Representative Models varies significantly between groups and is consistently larger than the MSE of the Naive Approach, by 6.75% to 42.01%. Moreover, we find that 76% of the domain t.s. can be predicted using only five models with a forecast error incremented by at most 20%.These results support our hypothesis that when considering more compact groups, each representative generalizes its elements better, and this generalization can be extended to predictive quality.

4.2.3. Elapsed Time for Training, Validation and Forecast

The computational cost for training and forecasting is also relevant. According to Table 2, the total time for training the models over all the t.s. in the Naive Approach is about 31500 seconds (8.75 hours). The average training time of an ARIMA model using a t.s. with 349 time units is then $31500/8100 \approx 3.9$ seconds. In our proposal, we consider re-training models. Thus, the total training time for a given partitioning can be estimated as $k \times (2 \times 3.9)$ seconds, about a minute for the domain partitioning with $k = 8$.

The results in this section support the hypothesis that: (1) the data distribution variation observed in the domain would point to a strategy based on multiple partitioning criteria; (2) model representatives can significantly reduce the model training cost with acceptable accuracy. Experiments in this section were repeated for all values of k considered in Section 4.1, and we found that $k = 132$ minimized the MSE metric. For these reasons, we will consider $k = \{8, 66, 132\}$ for multiple domain partitioning criteria.

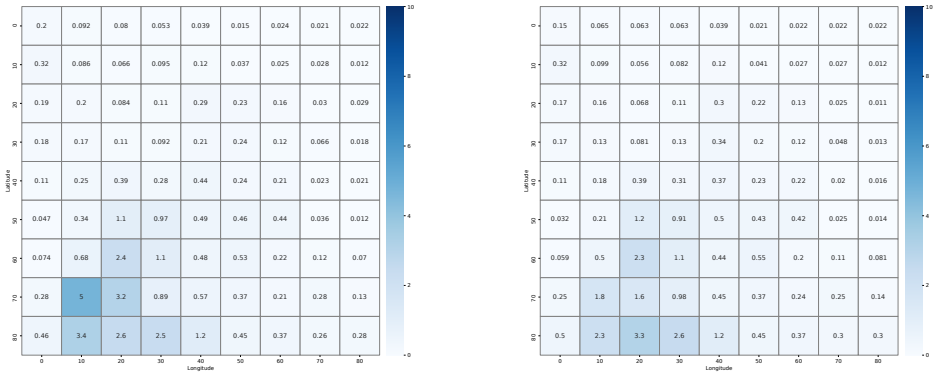

Figure 4. Model Composition with Representatives for: (a) $k = 66$; (b) $k = 132$

4.3. Processing Spatio-Temporal Predictive Queries

Here, we evaluate the predictive quality of a Model Composition over a region of interest R when processing an STPQ.

4.3.1. Predictive Quality of Model Composition

We consider multiple domain partitioning criteria and a Model Selection approach to be applied on query regions of fixed size $R = [10 \times 10]$ distributed uniformly over the domain, with these approaches:

- Naive Selection: For each t.s. in R, we select its pre-trained ARIMA model.
- Selection of Representative Models: For each t.s. in R, we determine its corresponding group and select the pre-trained ARIMA Representative Model.

The predictive quality of the Model Composition forecasts is evaluated using the accumulated MSE over the query region R.

Figures 4.a 4.b correspond to color maps of the MSE over different regions of the domain for $k = 66$ and $k = 132$, respectively, with a dark blue for the highest forecast error. Experimentally, we find that a problematic spatial region near the bottom left. Even there, using $k = (8, 66)$ may yield better results than $k = 132$ for some slices. This finding triggered the development that follows next.

4.3.2. Classifier for Model Selection

This section proposes a Model Selection approach that leverages the predictive quality variation of the Representative Models in domain partitioning. Here, the intuition is that by applying multiple partitioning to a domain, each t.s. would be mapped to a set of groups. Conversely, each domain sequence would be associated with a set of model representatives, and so the question is which one to pick.

We formulate the model selection proposal as a univariate time series (u.t.s.) classification problem: Given an unlabeled u.t.s. of t_p time units, assign it to one or more predefined classes. Then we are able to generate the Time Series Classification Dataset as $TSCD = \{(s_1, y_1), \ldots, (s_N, y_N)\}$ as a collection of pairs (s_i, y_i) where s_i is a u.t.s

with y_i as its corresponding one-hot label vector of the labels for its class [I. Fawaz et al. 2019].

In our context, each of these classes represents one of the available domain partitioning criteria. Considering $k = \{8, 66, 132\}$, we obtain 183 classes in total, after accounting for medoid repetition. In order to work with a balanced dataset, we extract for the $TSCD$ approximately 30 samples per class [Du et al. 2018]. We consider 5000 samples, divided in the percentages $60/20/20$ for training, validation, and test, respectively.

Considering the sequential aspect of time series data requires algorithms that can harness this temporal property to select a class label. In this work, we consider a classifier based on Neural Network models. After considering non-hybrid approaches that provided inferior classification accuracy [I. Fawaz et al. 2019], we opted for the hybrid architecture 1D Convolutional Neural Network – Long-Short Term Memory (1DCNN-LSTM) [Xu et al. 2020]. We considered variations for parameters such as learning rate and batch size, that affect the training time and how fast we achieve convergence in the validation loss function.

4.3.3. Evaluation of the Classifier for Model Selection

After training the Classifier presented in the previous section, we repeat the same experiments from Section 4.3.1 using the classifier as a Model Selection approach. For each t.s. in R, the classifier receives a t.s. of length t_p as input. As output, we obtain a Representative Label that corresponds to one of the Representative Models. With this model selection process, we repeat the forecast error analysis from Section 4.3.1.

The experimental results are summarized in Table 3: the first columns correspond to the Naive Composition; the next three columns correspond to the Representative Models Composition with the three values of k; the last column (highlighted) to the Classifier for Model Composition.

Table 3. MSE Forecast Error Summary including the Classifier.

Naive	k-Medoids			Classifier
	$k = 8$	$k = 66$	$k = 132$	
0.38 ± 0.61	0.48 ± 0.59	0.47 ± 0.86	0.39 ± 0.62	0.70 ± 0.81

Due to space restrictions, we omit the resulting colormap of the MSE of the forecast errors using the Classifier for Model Selection. We observed that the Classifier generates a composition with predictive quality comparable to the Naive Approach in some areas. However, the opposite is true for other regions, this can be explained by the limited knowledge of the classifier about the time series, as it receives t.s. of t_p time units.

Finally, we compare the execution of an STPQ using the proposed Model Composition, with the Naive Selection based on ARIMA models, over different query region sizes. Results are shown in Table 4, it is similar to Table 3 but with the query regions. We observe that, for the majority of the query regions considered, the forecast error of the Classifier for Model Selection is closer to the ARIMA Naive Selection.

Table 4. MSE Forecast Error for Spatio-Temporal Queries in the domain $\mathcal{D}$.

Query Region	Naive	k–Medoids			Classifier
		$k = 8$	$k = 66$	$k = 132$	
$[0, 20] \times [0, 20]$	0.158	0.089	0.174	0.160	0.190
$[20, 40] \times [35, 55]$	0.203	0.335	0.199	0.230	0.330
$[50, 70] \times [60, 80]$	0.170	0.584	0.203	0.188	0.274
$[15, 35] \times [65, 85]$	0.034	0.063	0.045	0.038	0.093
$[20, 50] \times [50, 80]$	0.122	0.203	0.147	0.135	0.202
$[15, 45] \times [20, 50]$	0.156	0.262	0.155	0.168	0.281
$[40, 55] \times [20, 40]$	0.483	0.707	0.530	0.541	0.618
$[65, 80] \times [50, 70]$	0.248	0.470	0.302	0.308	0.343
$[30, 60] \times [5, 20]$	0.137	0.353	0.205	0.147	0.391
$[10, 40] \times [55, 70]$	0.095	0.139	0.111	0.098	0.135

5. Related Works

In this work, we integrate tools designed for two types of knowledge fields: (i) time series classification and (ii) processing spatio-temporal predictive queries. The former gained attention in the last decade due to the accelerated advancement of deep learning techniques, many are discussed in a thesis aimed at deep learning for TSC [I. Fawaz et al. 2019], and the site `http://www.timeseriesclassification.com`, in efforts to reunite dataset and research papers on this evolving topic.

Common uses for spatio-temporal predictive queries in spatio-temporal data are predictive analytics to answer complex questions involving missing or future values, correlations, and trends, which can be used to identify opportunities or threats [Ghanta et al. 2019, Polyzotis et al. 2018]. The predictive functionality can help build introspective services for various resource management and optimization tasks [Crankshaw et al. 2017].

While we do not aim to propose a full Predictive Serving System [Crankshaw et al. 2017], it is worth exploring some of these systems to better understand the requirements behind model composition and model selection. The framework Clipper [Crankshaw et al. 2017] is designed to serve trained models at interactive latency, with two model selection policies based on multi-armed bandit algorithms for a trade-off between accuracy and computation overhead. Rafiki [Wang et al. 2018] is an inference service based on reinforcement learning that provides an online multi-model selection to compose ensembles.

Regarding massive data processing and model training, in [Mirzasoleiman 2021] are discussed techniques for dataset characterization in a reduced number of representatives elements, with data-efficient methods to extract representative subsets that generalize the full data. Finally, DJEnsemble [Pereira et al. 2021] investigates the prediction of spatio-temporal phenomena using deep-learning models; leveraging statistical properties of the t.s. to generate tiles in contrast of our shape-based approach.

6. Conclusions and Future Works

The main objective of this work is to develop an approach to make predictions, within some tolerated error margin, about future states of a spatio-temporal region, using carefully selected predictive models that have been trained with limited temporal data. To achieve this, we formulate the problem of model composition to process predictive queries and propose a solution where the model selection is guided by a data-driven approach backed by shape-based domain partitioning. The computational experiments were then designed to evaluate the proposal, considering the case study of temperature forecasting.

Within our proposal, both the domain partitioning (k–medoids) and the construction of Representative Models can be computed and persisted during an offline phase, quickly retrieved during an online phase, significantly reducing the elapsed time for processing predictive queries. In this regard, the choice of k becomes an important factor for the predictive quality, and three techniques to find optimal values of k were explored. We find that the intuitive choice of a large value of k may not always produce the best results: fewer groups may produce more accurate results for some elements of the query region.

The previous result motivated the proposal of a neural network classifier for model selection. In the offline phase, we allow the construction of representative predictive models for multiple partitioning criteria ($k = \{8, 66, 132\}$). For the online phase, the classifier matches the subset (t_p time units) of each u.t.s in the query region to one of the representatives, thus creating the model composition for a given predictive query.

We show that our proposal can process predictive queries with significantly lower response time, while maintaining comparable predictive quality. To evaluate this experimentally, we used sMAPE forecast errors accumulated over query regions with MSE. Results indicate 20% and 45% relative increases for $k = 66$ and the Classifier approach, respectively, with a gain in computational efficiency of two orders of magnitude as a trade-off. We recognize that the Classifier needs to be improved, e.g., by considering a domain with a larger volume of data and understanding its classification accuracy.

Our proposal opens up several research directions. The calculation of pairwise DTW distances can be enhanced by grouping time series with an incremental process for the DTW matrix [Oregi et al. 2017]. For the domain partitioning task, we could consider non-crisp partitioning techniques [Izakian et al. 2015], producing more than one representative for a given element. This work did not focus on forecast time for the online phase as the ARIMA models deliver predictions in milliseconds; however, more complex models would imply significant service times. Therefore, a natural follow-up would include a multi-objective optimization process.

References

Box, G. and Jenkins, G. M. (1976). *Time Series Analysis: Forecasting and Control*. Holden-Day.

Crankshaw, D., Wang, X., Zhou, G., Franklin, M. J., Gonzalez, J. E., and Stoica, I. (2017). Clipper: A low-latency online prediction serving system. In *NSDI'17 - USENIX*, pages 613–627, Boston, MA. USENIX Association.

Du, S. S., Wang, Y., Zhai, X., Balakrishnan, S., Salakhutdinov, R. R., and Singh, A. (2018). How many samples are needed to estimate a convolutional neural network? In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Ghanta, S., Subramanian, S., Khermosh, L., Sundararaman, S., Shah, H., Goldberg, Y., Roselli, D., and Talagala, N. (2019). ML health monitor: taking the pulse of machine learning algorithms in production. In *Applications of Machine Learning*, volume 11139, pages 191 – 202. International Society for Optics and Photonics, SPIE.

Hassani, H. and Silva, E. S. (2015). Forecasting with big data: A review. *Annals of Data Science*, 2(1):5–19.

Hastie, T., Tibshirani, R., and Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction, 2nd Edition*. Springer.

Hyndman, R. J. and Khandakar, Y. (2008). Automatic time series forecasting: The forecast package for r. *Journal of Statistical Software, Articles*, 27(3):1–22.

Hyndman, R. J. and Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4):679 – 688.

I. Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., and Muller, P.-A. (2019). Deep learning for time series classification: A review. *Data Min. Knowl. Discov.*, 33(4):917–963.

Izakian, H., Pedrycz, W., and Jamal, I. (2015). Fuzzy clustering of time series data using dynamic time warping distance. *Engineering Applications of Artificial Intelligence*, 39:235–244.

Liao, T. W. (2005). Clustering of time series data: A survey. *Pattern Recognition*, 38(11):1857 – 1874.

Mirzasoleiman, B. (2021). Efficient machine learning from massive datasets.

Murat, M., Malinowska, I., Gos, M., and Krzyszczak, J. (2018). Forecasting daily meteorological time series using arima and regression models. *International Agrophysics*, 32(2):253–264.

Oregi, I., Pérez, A., Del Ser, J., and Lozano, J. A. (2017). On-line dynamic time warping for streaming time series. In *Machine Learning and Knowledge Discovery in Databases*, pages 591–605, Cham. Springer International Publishing.

Pereira, R., Souto, Y., Chaves, A., Zorrilla, R., Tsan, B., Rusu, F., Ogasawara, E., Ziviani, A., and Porto, F. (2021). *DJEnsemble: A Cost-Based Selection and Allocation of a Disjoint Ensemble of Spatio-Temporal Models*, page 226–231. ACM, NY, USA.

Polyzotis, N., Roy, S., Whang, S. E., and Zinkevich, M. (2018). Data lifecycle challenges in production machine learning: A survey. *SIGMOD Rec.*, 47(2):17–28.

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53 – 65.

Saha, S., Moorthi, S., Wu, X., Wang, J., Nadiga, S., and Becker, E. (2011). Ncep climate forecast system version 2 (cfsv2) selected hourly time-series products.

Sakoe, H. and Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49.

Souto, Y. M., Porto, F., de Carvalho Moura, A. M., and Bezerra, E. (2018). A spatiotemporal ensemble approach to rainfall forecasting. In *IJCNN, 2018*, pages 1–8.

Wang, W., Gao, J., Zhang, M., Wang, S., Chen, G., Ng, T. K., Ooi, B. C., Shao, J., and Reyad, M. (2018). Rafiki: Machine learning as an analytics service system. *Proc. VLDB Endow.*, 12(2):128–140.

Xu, G., Ren, T., Chen, Y., and Che, W. (2020). A one-dimensional cnn-lstm model for epileptic seizure recognition using eeg signal analysis. *Frontiers in Neuroscience*, 14:1253.