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

Rocío Zorrilla, Eduardo Ogasawara, Patrick Valduriez, Fábio Porto

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

Rocío Zorrilla[1 [0000−0001−6096−0925]], Eduardo Ogasawara[2 [0000−0002−0466−0626]], Patrick Valduriez[3 [0000−0001−6506−7538]], and Fábio Porto[1 [0000−0002−4597−4832]]

[1] Laboratório Nacional de Computação Científica – LNCC, Brazil.
romizc@lncc.br – http://www.lncc.br
[2] Centro Federal de Educação Tecnológica Celso Sukow da Fonseca – CEFET-RJ, Brazil.
[3] INRIA & LIRMM, France

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.

Keywords: Spatio-temporal · Time-series · Predictive models.

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 [32], and climate alert systems [21]. 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 [12, 35].

* 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.

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 [10, 18, 25]. Inspired by the tradition of database systems, predictive serving systems are expected to support prediction requests through a declarative query interface [5]. 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 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 such that their composition meets the specified performance constraints and covers the requested spatial area for prediction.

In practice, for large spatial domains, such as the Brazilian territory in a weather forecast service, it is not feasible to hold pre-trained models for each possible point of interest. Complementarily, training models at run-time may not be feasible under stringent elapsed-time query constraints, such as in nowcasting applications [26].

This work proposes reusing pre-trained models built in a reduced set of spatio-temporal points, that probably fall outside the query spatio-temporal region, in order to answer predictive queries. By using pre-trained models, we shall meet the real-time prediction execution constraints. However, as the models may have been trained outside the query region, the procedure shall guarantee that the prediction error produced by the composition of models is minimized.

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 $s = (s_1, s_2, \ldots, s_T)$ denotes 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 univariate time series $s \in \mathcal{D}$. 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 (time series) divided in training, validation and test subsequences,
- A: forecasting technique,
- $\mathbf{p}$: parameters for the forecast technique,
- E_g: in-sample error [13],
- 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 time series 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. This process can be integrated into a more general query processing framework outside the scope of this work. Moreover, $\mathcal{D}$ is a dataset that is directly processed in *raw* by queries in database systems like [31].

3 Our Proposal

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

In this paper, 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 time series representative, which generalizes the shape similarity (variations according to the temporal dimension) of other time series in the domain, it is possible to preserve a predictive quality comparable to the baseline approach of using a model for every time series We could then process an STPQ efficiently while maintaining a low error margin.

In our approach, the domain $\mathcal{D}$ comprises univariate time series and their (lat, lon) positions. However, when building or training models for each time series, we do not include the spatial positions as features. This effectively decouples the spatial component from the domain, allowing us to identify and cluster similar time series based purely on their temporal evolution.

Therefore, in Equation (1), the focus is on the temporal characteristics of the data:

$$g = \langle s, A, p, E_g, \Sigma_g \rangle$$

For spatio-temporal predictive queries, as shown in Equation (2), spatial information is incorporated separately to apply the time series models across various locations:

$$Q = \langle R, t_p, t_f, Q_m \rangle$$

Here, R represents the spatial region of interest, guiding the use of the predictive models without altering the underlying time series analysis.

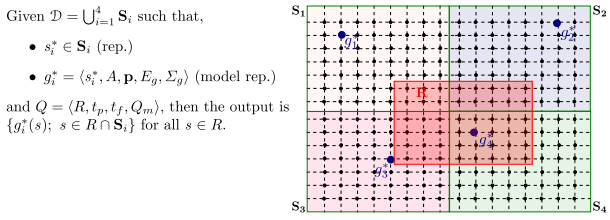

Fig. 1: Our Model Composition Approach

To illustrate our proposal, Figure 1 shows a domain that has been partitioned into four groups $S = \{S_1, S_2, S_3, S_4\}$. Here, the query region R has 35 univariate time series and intersects with the four groups. Within our proposal, we only need to train four models to process the STPQ. Note that three models were

trained outside of R. The approach is divided into two phases, offline and online

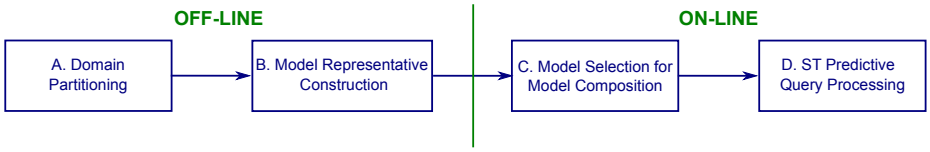

Fig. 2: A two phase query processing approach

(Figure 2). The offline phase comprises two steps: (A) the domain partitioning, based on time series clustering techniques; (B) the construction of predictive models at each group time series representative. The online phase is applied when processing a spatio-temporal predictive query. It consists of: (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 the 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. As a result, we can reduce the computational workload and execution elapsed-time if we were to train a model on each point of a query region in run time.

3.1 Domain Partitioning

This step aims to: partition the domain into groups with time series with high shape similarity among themselves; and find a representative. By using k-medoids as a clustering algorithm, each group found can minimize its local dissimilarity and be represented by a medoid that corresponds to an existing time series in the dataset [1]. In this paper, the number of groups k is chosen to produce k corresponding predictive models that produce accurate forecasts for similar time series

Usually, for k-medoids, the choice of k should strike a balance between minimizing the computational cost in using few representatives while maximizing the accuracy when assigning each time series data to its cluster. In the context of the off-line partitioning step, we are not interested in reducing the computational cost. Instead, we want to find the corresponding predictive models that produce accurate forecasts for similar time series

The k-medoids algorithm requires the user to specify k. When using a clustering technique for high volumes of data and low variability of the data values throughout neighbor points, it is difficult to determine the optimal number of groups [19]. We consider three methods to find an optimal value for k: the elbow method [1], silhouette index [27] and a fitting of the WSS curve by using a smooth cubic spline [4].

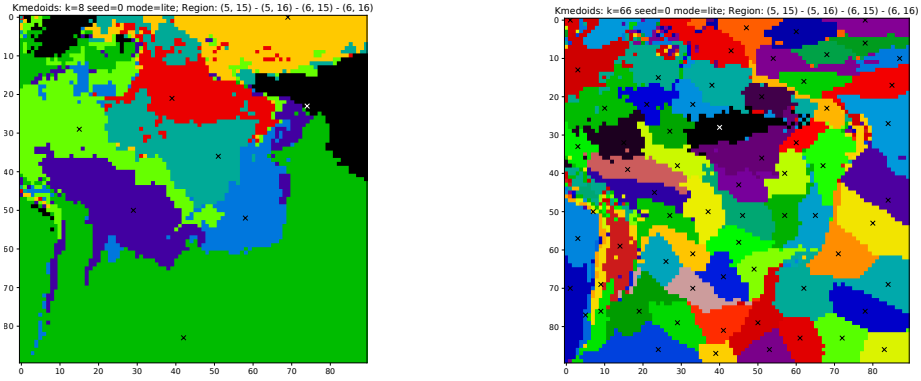

Fig. 3: Groups obtained with k-Medoids using $k = 8$ (left) and $k = 66$ (right). Corresponding medoids are marked with a '$\times$'.

3.2 Model Representative Construction

In order to answer STPQ with acceptable predictive error and reasonable query evaluation time, we consider using a model trained at each medoid. We refer to these k models as representative models and are 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 of validation, becoming the representative model that can be used to make predictions of t_f time units for all time series in its group that fall within a particular query region.

3.3 Model Selection for Model Composition

For the scope of this work, we define "Model Composition" as the subset of predictive models that can compute the forecast value of each element in a region of interest on the domain with increased accuracy. 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). Even when we consider the elements only in the temporal dimension, this difficulty persists [2]. Within this step, our hypothesis consists in assuming 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, based on limited information about its past. In order to find this model composition, we consider a model selection process based on the following strategies:

– Naive Approach: for each time series s_j in each group, we train its model g_j and calculate the corresponding forecast error. We consider this the baseline

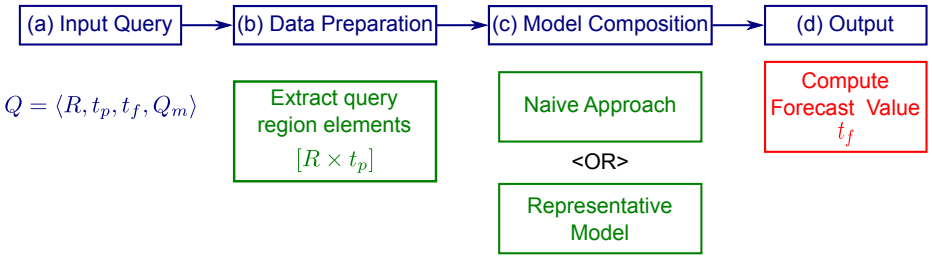

Fig. 4: On-Line STPQ processing.

strategy as it generates as many models as there are time series in the region, and requires a high computational cost.

– Representative Models: we propose that, given the time series 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 4, 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 time series 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 time series 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 time series in R is computed using its corresponding representative model. Here, we highlight that the same model can generate different forecasts for different time series, provided that the time series undergo a data transformation (e.g., normalization). The forecast is produced by the inverse transformation of the model output.

The online procedure can also be represented by Algorithm 2. As input, the procedure takes 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, and the known in-sample error of the model is attributed.

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
```

Algorithm 2 Process Online Predictive Query

```
1: function PROCESSQUERY(D, R, t_p, t_f, selection_id)
       /* Obtain a model composition, also load available models /*
2:   model_comp ← select_model_composition(D, selection_id)
       /* Extract t_p past time units for region R /*
3:   region_data ← extract_region(D, R, t_p)
4:   query_out ← ⊥
5:   for element ∈ region_data do
       /* model composition determines representative (medoid) to use /*
6:     representative ← find_repr(model_comp, element)
       /* representative has trained model, do forecast of t_f steps /*
7:     forecast ← predict(representative.model, element, t_f)
       /* annotate the current element with forecast series and known error /*
8:     error ← representative.error
9:     annotate(element, forecast, error)
       /* the query result has a set of the annotated elements in R /*
10:    query_result ← add_element(element, query_out)
11:  end for
       /* Compute the MSE of the errors for a single error metric over R /*
12:  error_mse ← combine_errors_mse(query_out)
13:  annotate(query_out, error_mse)
       /* Output is the forecast and error at each element of R, as well as the MSE /*
14:  Return query_out
15: end function
```

4 Experiments and Results

In this Section, we describe the experimental validation of the methodology presented, following the steps: the domain partitioning, the predictive quality of the representative models, the model composition and the query performance. We show how each step is applied to the use case of temperature forecasting, with the corresponding presentation and analysis of the results of each step.

Experimental dataset. We use a subset of the Climate Forecast System Reanalysis (CFSR) dataset, which contains four daily air temperature obser-

vations from January 1979 to December 2015 covering the space between 8N-54S latitude and 80W-25W longitude [29]. We subset this data to include one year of readings in the Brazilian territory, then transform each time series 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 of RAM, and running Linux CentOS 7.7. for all experiments.

4.1 Domain Partitioning Evaluation

We implemented k-medoids using the Dynamic Time Warping similarity measure [30], and compared with a regular partitioning technique (baseline) based solely on the geometry of the domain (k rectangles). The representative time series for each technique are the medoid and centroid, respectively. Computing k-medoids requires pairwise distances, which can be calculated beforehand as a 2-d matrix. In our proposal, we perform this expensive computational process only once, for a quick retrieval later.

For each of the two partition techniques, 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. For k-medoids, we obtain a monotonically decreasing trend for the WSS curve. This makes the choice for an optimal k difficult, a known problem for high volumes of data with low variability throughout neighbor points [19].

In Figure 5, it is possible to observe the decreasing behavior of the WSS curve for higher values of k.

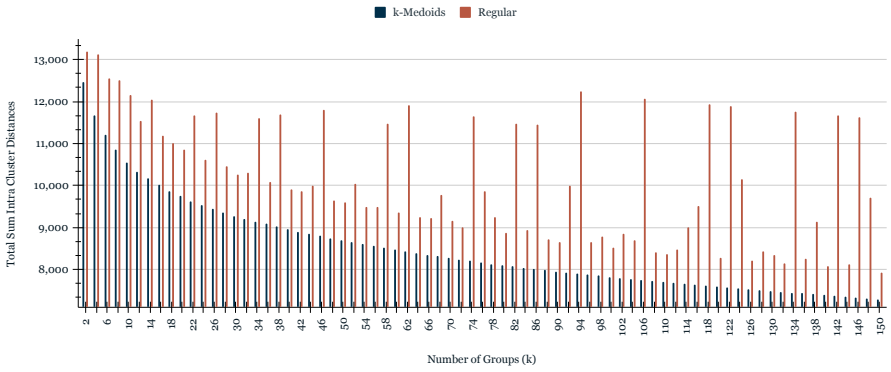

Fig. 5: Total Within cluster Sum of Squares of k-Medoids and Regular Partitioning Techniques.

It is particularly important for our problem, where there is low variation in the spatial data distribution of the different time series The Table 1 summarizes the findings of applying the available methods to find optimal values for k (Section 3.1). The monotonous trend of k-medoids allowed for calculating

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

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

the minimum value for the second derivative, by fitting the values of the WSS using a cubic smooth spline. We argue that this method is more appropriate for our dataset, as it highlights the decreasing trend in the intra-cluster cost as k increased. It was possible because the splines smoothed the small variations that were preventing the other methods from finding a higher value for k. This value gives us a reasonable number of representatives to use in the next phases of the methodology. Also, since we applied two other evaluation methods for selecting k that produced two other domain partitioning schemes, we can also consider these groups when evaluating compositions relevant to our proposed model selection approach.

4.2 Predictive Quality of Models at Representatives

Here, we are interested in evaluating the accuracy of the forecast values computed on the test sub-sequence (t_f) by comparing them against the observational values available. In this work, we consider the Symmetric Mean Absolute Percentage Error (sMAPE) for forecast error evaluation and the Mean Squared Error (MSE) [15] for accumulated forecast.

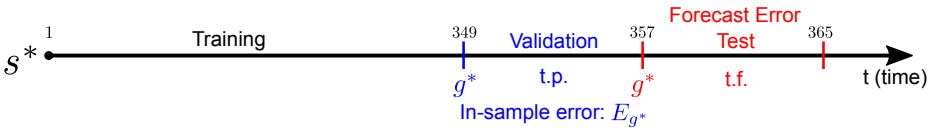

Fig. 6: Splitting a Sequence to train and test a model.

In order to assess the predictive quality of a model on a representative time series, we train the corresponding k models in a domain partitioning and evaluate the following metrics:

– sMAPE: Forecast error of the representative model when forecasting each time series in its respective group.

– MSE: Accumulated forecast error (sMAPE) by combining the previous forecasts within each group.

In this section, we evaluate the predictive quality of Auto-Regressive Integrated Moving Average (ARIMA) models [3]. These models fit the description in Section 2 and offer a good trade-off between predictive accuracy and computational cost [21]. We leverage auto.ARIMA [14] implementation to choose optimal ARIMA parameters.

Evaluation of sMAPE Forecast Error Considering the domain partitioning with $k = 8$, we have eight groups with 1013 ± 617 time series on average, yielding eight representative models. In order to explore the relationship between the intra-cluster similarity and forecast error, we gather the forecast errors within each group and generate scatter plots diagrams, with the Dynamic Time Warping distance of each sequence to its medoid in the x-axis and the sMAPE metric in the y-axis. We found that, for the group index zero (Figure 7), the maximum

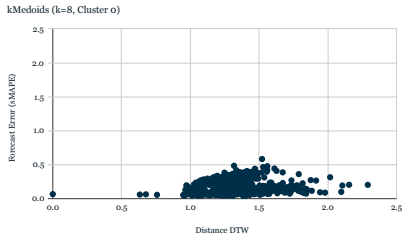

Fig. 7: Forecast Error $= 0.159 \pm 0.073$.

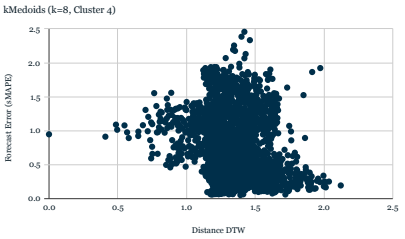

Fig. 8: Forecast Error $= 0.718 \pm 0.347$.

sMAPE value was the lowest among the eight groups. Conversely, for the group index four (Figure 8), the maximum sMAPE value was the highest.

We observe that there is not a clear correlation between the Dynamic Time Warping distances and the forecast error. If we consider all the representatives, then as k increases, there is a tendency to obtain groups with more similarity between their elements (lower Dynamic Time Warping distance) and also the predictions tend to be more accurate. An additional important observation here is 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.

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:

- ARIMA or Naive Approach (Baseline): We train a model using a time series and calculate the corresponding forecast error for every time series in each group. Then for each group, we compute its corresponding MSE value.
- Representative Models (Proposal): given the k corresponding models for the representatives in a domain partitioning, we use its representative model to forecast future values; finally, we compute the accumulated MSE values.

Considering the domain partitioning with $k = 8$, Table 2 highlights the results of the MSE evaluation. The columns are as follows: (1) cluster/group ID; (2) elapsed time to train models for all the time series in the group (naive approach); (3) elapsed time to forecast t_f future units for the time series in the group (naive approach); (4) accumulated MSE value for the Naive Approach; (5) accumulated MSE value for the Representative Models; (6) percentage change of the MSE values between the approaches.

Col. 1. Group or Cluster Id.
Col. 2. Elapsed time to train models for all the time series in the group (naive approach).
Col. 3. Elapsed time to forecast t_f future units for the time series in the group (naive approach).
Col. 4. Accumulated MSE value for the Naive Approach.
Col. 5. Accumulated MSE value for the Representative Models.
Col. 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)	ARIMA	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 time series can be predicted using only five models with a forecast error incremented by at most 20% of the Naive Approach, which would consider 8100 different models for the same predictions. These results support our hypothesis that when considering more compact groups, each representative generalizes its elements better, and this generalization can be extended to the predictive quality.

Elapsed Time for Training, Validation and Forecast An additional aspect in the evaluation of the Representative Models is the computational cost for training and forecasting. According to Table 2, the total time for training the models over all the time series in the Naive Approach is about 31500 seconds (8.75 hours). Thus, the average training time of an ARIMA model using a time series with 349 time units is $31500/8100 \approx 3.9$ seconds. In our proposal, we consider train and re-train models for k representative time series 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 different regions of the domain would point to a strategy based on multiple partitioning criteria; (2) by using model representatives, we can significantly reduce the model training cost while keeping acceptable forecast errors. Additionally, 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.

4.3 Processing Spatio-Temporal Predictive Queries

Our proposed online phase (See Fig. 2) comprises two steps: (D) Model Selection for Model Composition and (E) STPQ Processing. Here, we evaluate the predictive quality of a Model Composition over a region of interest R when processing an STPQ.

Model Composition Evaluation To assess the predictive quality of a 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. We consider these approaches for Model Selection:

- Naive Selection: For each time series in R, we select its pre-trained ARIMA model. Here, we introduce kNN as an additional predictive model for comparisons of forecast accuracy, so we obtain two sets of experimental results.
- Selection of Representative Models: For each time series 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. These results are summarized in Table 3: the first column corresponds to the Model Composition using Naive Selection; the following three columns represent the Model Composition formed by the Selection of Representative Models for a domain partitioning, varying $k = \{8, 66, 132\}$. We present the mean and variance of the MSE for the 81 query regions.

Table 3: MSE Forecast Error Summary.

ARIMA	kNN	k-Medoids		
		$k = 8$	$k = 66$	$k = 132$
0.38 ± 0.61	0.62 ± 0.91	0.48 ± 0.59	0.47 ± 0.86	0.39 ± 0.62

We use color maps to help the visualization of the MSE over different regions of the domain. Figure 9 and Figure 10 correspond to $k = 66$ and $k = 132$, respectively. Each color map shows the relative magnitude of the values, with a dark blue for the highest forecast error and a constant palette throughout figures.

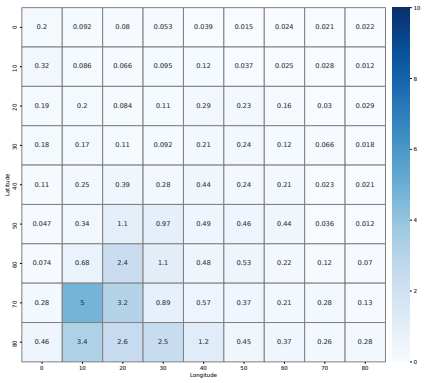

Fig. 9: Model Composition with Representatives ($k = 66$).

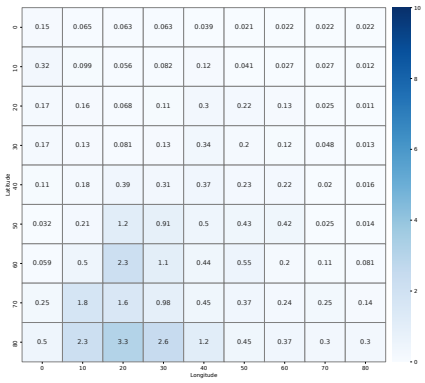

Fig. 10: Model Composition with Representatives ($k = 132$).

Experimentally, we find that a spatial region near the bottom left results in larger forecast errors. Even there, using k (8, 66) may yield better results than $k = 132$ for some slices. This finding triggered the development and evaluation that follows next.

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 time series 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 extend the problem formulation presented in Section 2. Consider two domain partitioning criteria $\mathcal{D} = \cup_{i=1}^{m} \mathbf{P_i}$ and $\mathcal{D} = \cup_{j=1}^{n} \mathbf{Q_j}$, where $m \neq n$; the set of representatives on the partitioning considered is $\mathcal{R} = \{p_i, \ldots, p_m, q_1, \ldots, q_n\}$,

and $\mathcal{G}_{(\mathcal{R})}$ the set of their predictive models. Then, for a given $s \in \mathcal{D}$:

$$\exists\, \hat{s} \in \mathcal{R},\ \text{such that},\ \min_{\substack{\hat{s}\in\mathcal{R} \\ s\in\mathcal{D}}} d_{DTW}(\hat{s}, s). \tag{3}$$

We formulate the model selection proposal as a univariate time series (univariate time series) classification problem: Given an unlabeled univariate time series of t_p time units, assign it to one or more predefined classes. From (3), 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 [11].

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 [8]. 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 [16], we opted for the hybrid architecture 1D Convolutional Neural Network – Long-Short Term Memory (1DCNN-LSTM) [7, 34]. 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. From now on, we will consider only the last model in Table 4 (CNN1D-LSTM(2)) that presented the higher accuracy.

Table 4: Models' metrics on Test Set.

Model	Layers	Accuracy	Loss
CNN1D-LSTM(1)	6	57.740	2.673
CNN1D-LSTM(2)	6	64.759	1.865

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 using the classifier as a Model Selection approach. For each time series in R, the classifier receives a time series 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.

The experimental results are summarized in Table 5: it extends the Table 3 with the last column (highlighted) representing the Classifier for Model Selection.

Table 5: MSE Forecast Error Summary including the Classifier.

ARIMA	kNN	k-Medoids			Classifier
		$k = 8$	$k = 66$	$k = 132$	
0.38 ± 0.61	0.62 ± 0.91	0.48 ± 0.59	0.47 ± 0.86	0.39 ± 0.62	0.70 ± 0.81

We show the colormap for the MSE of the forecast errors computed in different regions of the domain using the Classifier for model composition in Figure 11. For comparison, we also add the colormap for the ARIMA (baseline) Approach.

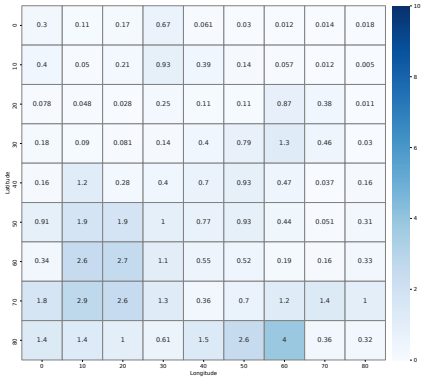

Fig. 11: Forecast Error with Model Composition by Classifier.

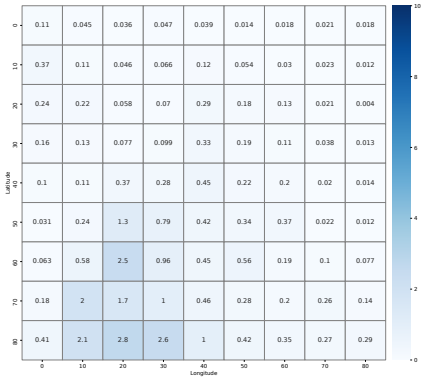

Fig. 12: Model Composition with ARIMA, naive approach.

We observed that the Classifier generates a composition with predictive quality comparable to the Naive Approach in some areas. Moreover, the classifier quality is reflected in a few regions of the domain that exhibit a smaller forecast error than when using the composition based on Selection of Representative Models directly. 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 time series 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 and kNN regressions for univariate time series, over different query region sizes. Results are shown in Table 6, it is similar to Table 5 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 6: MSE Forecast Error for Spatio-Temporal Queries in the domain $\mathcal{D}$.

Query Region	ARIMA	kNN	k–Medoids			Classifier
			$k = 8$	$k = 66$	$k = 132$	
$[0, 20] \times [0, 20]$	0.158	0.209	0.089	0.174	0.160	0.190
$[20, 40] \times [35, 55]$	0.203	0.258	0.335	0.199	0.230	0.330
$[50, 70] \times [60, 80]$	0.170	0.145	0.584	0.203	0.188	0.274
$[15, 35] \times [65, 85]$	0.034	0.067	0.063	0.045	0.038	0.093
$[20, 50] \times [50, 80]$	0.122	0.210	0.203	0.147	0.135	0.202
$[15, 45] \times [20, 50]$	0.156	0.198	0.262	0.155	0.168	0.281
$[40, 55] \times [20, 40]$	0.483	0.707	0.707	0.530	0.541	0.618
$[65, 80] \times [50, 70]$	0.248	0.190	0.470	0.302	0.308	0.343
$[30, 60] \times [5, 20]$	0.137	0.208	0.353	0.205	0.147	0.391
$[10, 40] \times [55, 70]$	0.095	0.226	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 [16], and the site http://www.timeseriesclassification.com, in efforts to reunite dataset and research papers on this evolving topic.

For time series clustering, the use of k-medoids with Dynamic Time Warping as similarity measure was used with success in several applications [22, 28]. In our work, we validate k–Medoids as an appropriate algorithm for our dataset, but we shifted our focus away from the Euclidean distance in favor of Dynamic Time Warping; the former failed to capture temporal misalignments.

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 [10,25]. The predictive functionality can help build introspective services for various resource management and optimization tasks [9].

While we do not aim to propose a full Predictive Serving System [6], it is worth exploring some of these systems to better understand the requirements behind model composition and model selection. The framework Clipper [6] 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 [33] 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 [20] 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. The work focuses on extracting representative subsets for training machine learning models, and developing theoretically rigorous opti-

mization techniques. Finally, DJEnsemble [24] investigates the prediction of spatio-temporal phenomena using deep-learning models. However, instead of our shape-based approach, they partition the domain into tiles based on the statistical properties of the time series 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.

Experimentally, we find that the k-medoids method can efficiently group time series in a domain according to the Dynamic Time Warping distance. Also, the resulting medoids generalizes the temporal evolution of their group. Therefore it can be used to train representative models that take a univariate time series as input. 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 univariate time series 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.

Results from the forecast error analysis support using a time series classifier to leverage potential gains in predictive performance when using multiple partitioning schemes. However, 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 Dynamic Time Warping distances can be enhanced by grouping time series with an incremental process for the Dynamic Time Warping matrix [23]. For the domain partitioning task, we could consider non-crisp partitioning techniques [17], 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 (see Table 2); however, more complex models would imply significant service times. Therefore, a natural follow-up would include a multi-objective optimization process.

References

1. Aggarwal, C.C., Reddy, C.K.: Data Clustering: Algorithms and Applications. Chapman and Hall/CRC, 1st edn. (2013)
2. Aghabozorgi, S., Seyed Shirkhorshidi, A., Ying Wah, T.: Time-series clustering - a decade review. Inf. Syst. **53**(C), 16–38 (Oct 2015). https://doi.org/10.1016/j.is.2015.04.007
3. Box, G., Jenkins, G.M.: Time Series Analysis: Forecasting and Control. Holden-Day (1976)
4. Burden, R.L., Faires, D.J., Burden, A.M.: Numerical Analysis. CENGAGE Learning, Boston, US, 10 edn. (2016)
5. Crankshaw, D., Gonzalez, J., Bailis, P.: Research for practice: Prediction-serving systems. Commun. ACM **61**(8), 45–49 (Jul 2018). https://doi.org/10.1145/3190574
6. Crankshaw, D., Wang, X., Zhou, G., Franklin, M.J., Gonzalez, J.E., Stoica, I.: Clipper: A low-latency online prediction serving system. In: 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17). pp. 613–627. USENIX Association, Boston, MA (2017)
7. Du, Q., Gu, W., Zhang, L., Huang, S.L.: Attention-based lstm-cnns for time-series classification. In: Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems. p. 410–411. SenSys '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3274783.3275208
8. Du, S.S., Wang, Y., Zhai, X., Balakrishnan, S., Salakhutdinov, R.R., Singh, A.: How many samples are needed to estimate a convolutional neural network? In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 31. Curran Associates, Inc. (2018), https://proceedings.neurips.cc/paper/2018/file/03c6b06952c750899bb03d998e631860-Paper.pdf
9. Filippo, A.D., Lombardi, M., Milano, M.: Methods for off-line/on-line optimization under uncertainty. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18. pp. 1270–1276. International Joint Conferences on Artificial Intelligence Organization (7 2018). https://doi.org/10.24963/ijcai.2018/177, https://doi.org/10.24963/ijcai.2018/177
10. Ghanta, S., Subramanian, S., Khermosh, L., Sundararaman, S., Shah, H., Goldberg, Y., Roselli, D., Talagala, N.: ML health monitor: taking the pulse of machine learning algorithms in production. In: Zelinski, M.E., Taha, T.M., Howe, J., Awwal, A.A.S., Iftekharuddin, K.M. (eds.) Applications of Machine Learning. vol. 11139, pp. 191 – 202. International Society for Optics and Photonics, SPIE (2019). https://doi.org/10.1117/12.2529598

11. Gulli, A., Pal, S.: Deep Learning with Keras. Packt Publishing (2017)
12. Hassani, H., Silva, E.S.: Forecasting with big data: A review. Annals of Data Science **2**(1), 5–19 (2015). https://doi.org/10.1007/s40745-015-0029-9
13. Hastie, T., Tibshirani, R., Friedman, J.H.: The elements of statistical learning: data mining, inference, and prediction, 2nd Edition. Springer series in statistics, Springer (2009)
14. Hyndman, R.J., Khandakar, Y.: Automatic time series forecasting: The forecast package for r. Journal of Statistical Software, Articles **27**(3), 1–22 (2008). https://doi.org/10.18637/jss.v027.i03
15. Hyndman, R.J., Koehler, A.B.: Another look at measures of forecast accuracy. International Journal of Forecasting **22**(4), 679 – 688 (2006). https://doi.org/https://doi.org/10.1016/j.ijforecast.2006.03.001
16. Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., Muller, P.A.: Deep learning for time series classification: A review. Data Min. Knowl. Discov. **33**(4), 917–963 (jul 2019). https://doi.org/10.1007/s10618-019-00619-1
17. Izakian, H., Pedrycz, W., Jamal, I.: Fuzzy clustering of time series data using dynamic time warping distance. Engineering Applications of Artificial Intelligence **39**, 235–244 (2015). https://doi.org/https://doi.org/10.1016/j.engappai.2014.12.015
18. Lee, Y., Scolari, A., Interlandi, M., Weimer, M., Chun, B.G.: Towards high-performance prediction serving systems. In: NIPS Machine Learning Systems Workshop (2017)
19. Liao, T.W.: Clustering of time series data: A survey. Pattern Recognition **38**(11), 1857 – 1874 (2005). https://doi.org/https://doi.org/10.1016/j.patcog.2005.01.025
20. Mirzasoleiman, B.: Efficient machine learning from massive datasets (May 2021), http://web.cs.ucla.edu/~baharan/research.htm
21. Murat, M., Malinowska, I., Gos, M., Krzyszczak, J.: Forecasting daily meteorological time series using arima and regression models. International Agrophysics **32**(2), 253–264 (2018). https://doi.org/10.1515/intag-2017-0007
22. Nakagawa, K., Imamura, M., Yoshida, K.: Stock price prediction using k-medoids clustering with indexing dynamic time warping. Electronics and Communications in Japan **102**(2), 3–8 (2019). https://doi.org/https://doi.org/10.1002/ecj.12140
23. Oregi, I., Pérez, A., Del Ser, J., Lozano, J.A.: On-line dynamic time warping for streaming time series. In: Ceci, M., Hollmén, J., Todorovski, L., Vens, C., Džeroski, S. (eds.) Machine Learning and Knowledge Discovery in Databases. pp. 591–605. Springer International Publishing, Cham (2017)
24. Pereira, R., Souto, Y., Chaves, A., Zorrilla, R., Tsan, B., Rusu, F., Ogasawara, E., Ziviani, A., Porto, F.: DJEnsemble: A Cost-Based Selection and Allocation of a Disjoint Ensemble of Spatio-Temporal Models, p. 226–231. Association for Computing Machinery, New York, NY, USA (2021), https://doi.org/10.1145/3468791.3468806
25. Polyzotis, N., Roy, S., Whang, S.E., Zinkevich, M.: Data lifecycle challenges in production machine learning: A survey. SIGMOD Rec. **47**(2), 17–28 (dec 2018). https://doi.org/10.1145/3299887.3299891
26. Ravuri, S.V., Lenc, K., Willson, M., Kangin, D., Lam, R., Mirowski, P.W., Fitzsimons, M., Athanassiadou, M., Kashem, S., Madge, S., Prudden, R., Mandhane, A., Clark, A., Brock, A., Simonyan, K., Hadsell, R., Robinson, N.H., Clancy, E., Arribas, A., Mohamed, S.: Skilful precipitation nowcasting using deep generative models of radar. Nature **597**, 672 – 677 (2021)
27. Rousseeuw, P.J.: Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics **20**, 53 – 65 (1987). https://doi.org/https://doi.org/10.1016/0377-0427(87)90125-7

28. Ruiz, L., Pegalajar, M., Arcucci, R., Molina-Solana, M.: A time-series clustering methodology for knowledge extraction in energy consumption data. Expert Systems with Applications **160**, 113731 (2020). https://doi.org/https://doi.org/10.1016/j.eswa.2020.113731
29. Saha, S., Moorthi, S., Pan, H.L., Wu, X., Wang, J., Nadiga, S., Tripp, P., Kistler, R., Woollen, J., Behringer, D., Liu, H., Stokes, D., Grumbine, R., Gayno, G., Wang, J., Hou, Y.T., ya Chuang, H., Juang, H.M.H., Sela, J., Iredell, M., Treadon, R., Kleist, D., Delst, P.V., Keyser, D., Derber, J., Ek, M., Meng, J., Wei, H., Yang, R., Lord, S., van den Dool, H., Kumar, A., Wang, W., Long, C., Chelliah, M., Xue, Y., Huang, B., Schemm, J.K., Ebisuzaki, W., Lin, R., Xie, P., Chen, M., Zhou, S., Higgins, W., Zou, C.Z., Liu, Q., Chen, Y., Han, Y., Cucurull, L., Reynolds, R.W., Rutledge, G., Goldberg, M.: The NCEP climate forecast system reanalysis. Bulletin of the American Meteorological Society **91**(8), 1015 – 1058 (01 Aug 2010). https://doi.org/10.1175/2010BAMS3001.1
30. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Transactions on Acoustics, Speech, and Signal Processing **26**(1), 43–49 (1978). https://doi.org/10.1109/TASSP.1978.1163055
31. da Silva, A.C., Lustosa, H.L.S., da Silva, D.N.R., Porto, F.A.M., Valduriez, P.: SAVIME: an array DBMS for simulation analysis and ML models prediction. J. Inf. Data Manag. **11**(3) (2020), https://periodicos.ufmg.br/index.php/jidm/article/view/24223
32. Souto, Y.M., Porto, F., de Carvalho Moura, A.M., Bezerra, E.: A spatiotemporal ensemble approach to rainfall forecasting. In: 2018 International Joint Conference on Neural Networks, IJCNN 2018, Rio de Janeiro, Brazil, July 8-13, 2018. pp. 1–8 (2018)
33. Wang, W., Gao, J., Zhang, M., Wang, S., Chen, G., Ng, T.K., Ooi, B.C., Shao, J., Reyad, M.: Rafiki: Machine learning as an analytics service system. Proc. VLDB Endow. **12**(2), 128–140 (Oct 2018). https://doi.org/10.14778/3282495.3282499
34. Xu, G., Ren, T., Chen, Y., Che, W.: A one-dimensional cnn-lstm model for epileptic seizure recognition using eeg signal analysis. Frontiers in Neuroscience **14**, 1253 (2020). https://doi.org/10.3389/fnins.2020.578126
35. Yang, C., Clarke, K., Shekhar, S., Tao, C.V.: Big spatiotemporal data analytics: a research and innovation frontier. International Journal of Geographical Information Science **34**(6), 1075–1088 (2020). https://doi.org/10.1080/13658816.2019.1698743