Scalable Incremental Similarity-based Learning with Neural Filters for Large Scale Timeseries Classification Systems

1st Nils Schaetti
Information Science Institute
GSEM/CUI
University of Geneva
Geneva, Switzerland
nils.schaetti@unige.ch

2nd Pasquale De Rosa
Information Science Institute
GSEM/CUI
University of Geneva
Geneva, Switzerland
pasquale.derosa@unige.ch

3rd Stéphane Montavon
Veterinary Department
Swiss Armed Forces
Bern, Switzerland
smontavon@bluewin.ch

4th David Deillon
Alogo Analysis SA
Renens, Switzerland
david@alogo-analysis.ch

5th Michel Deriaz
Information Science Institute
GSEM/CUI
University of Geneva
Geneva, Switzerland
michel.deriaz@unige.ch

***Abstract*—Automatic health monitoring and activity recognition systems provide specific information for caregivers and health professionals to prevent injury or disease. With the improvement of sensor technologies, wireless communication and machine learning, systems can now be aware of changes in the user's state and its environment in order to provide activity-aware automatic health predictions and information. However, these systems face the challenge to recognize specific patterns and activities in a large and quickly evolving database of sensor data. In this paper, an innovative approach to classify timeseries in a large dynamical space of classes is evaluated. The algorithm is based on the Conceptor Framework (CF) and close to the Reservoir Computing paradigm in machine learning. Compared to traditional approaches for classification, Conceptors allow the recognition of a large number of dynamical patterns incrementally without interference with previous learning and can easily adapt to new classes and users on the fly. For this empirical study, we gathered a dataset of timeseries extracted from sensors placed on horses during training sessions and evaluated various methods on a task of predicting which horse is the source of a given series. The dataset was gathered as a part of the HorseTrack[1] project which aims to create a system able to automatically detect injuries in horses. Our results show that incremental similarity-based learning with Conceptors is able to solve effectively this task with short training time while classical ESNs are not able to perform better than the majority baseline.**

***Index Terms*—Timeseries Classification, Horses, Sensor, Health Systems, Reservoir Computing, Conceptors**

I. Introduction

The advancement of sensor technology allowed sensors and computing power to be condensed into small devices that can be embedded and therefore used to track the health status of people or animals. With the extensive use and availability of fast and wireless communication systems such as mobile phones and their ultra-fast connections, it has become possible to track and analyze the activity of a huge amount of users providing them with predictions and information adapted to their immediate environment. The capacity of these activity-aware systems has been further increased by the availability of efficient machine learning algorithms and information retrieval systems able to provide extremely accurate prediction and to search in databases of patterns and categories.

However, these systems still face the challenge to retrieve, compare and identify patterns in very large databases while having the capacity to learn incrementally without interfering with previous training and to adapt quickly to new users and features. The difficulty of this challenge is further increased with health monitoring systems where patterns are dynamical and data composed of time-related samples. In the field of machine learning and artificial neural networks, recurrent models are well suited for this kind of data and are therefore interesting to evaluate on timeseries classification tasks and dynamical pattern recognition applied to automatic sensor-based health systems.

RNNs are known to be very difficult to train due to the vanishing gradient problem. To overcome this issue, a simple RNN model known as Echo State Network (ESN) was introduced in late 2000s [1] with similar methods in computational neurosciences [2]. This set of models is known by the scientific community as Reservoir Computing (RC). The idea behind RC is simple and consists in randomly creating an RNN and train only the output layer. Empirical data showed that it is capable to achieve very good results in practice. As the training

[1]Co-funded by Innosuisse

process targets only one simple output layer, linear methods can be used such as the common Ridge Regression (RR). Therefore ESNs are quick and easy to train in comparison to other recurrent models such as Long Short-Term Memory (LSTM) [3] and Gated Recurrent Units (GRU) [4], both based on the Stochastic Gradient Descent (SGD) algorithm.

The Conceptor Framework introduced in mid-2010s [5] proposes to extend ESN-based recurrent models by defining mathematically how a dynamical pattern occupies the hidden space inside an ESN. This geometrical description of input patterns is referred to as *conceptors* and can be used as a neural filter to re-generate patterns or to identify new patterns. Moreover, a set of logical operations can be defined on top of *conceptors* in order to implement some basic logical and conceptual reasoning such as pattern abstraction, similarity comparison and incremental life-long learning.

However, only few studies in the existing literature compared conceptor-based RNN and classical ESN models on real word data. In this study, we then decided to compare these two approaches on a timeseries classification task. The dataset was extracted from Alogo Move Pro[2] gathered on horses during training sessions and consists of 187 samples of 20-dimensional timeseries data such as 3-dimensional accelerations and angles. In these experiments, models have to identify which horse was the source of a timeseries in a closed set of 34 possible candidates. For each model, we performed a 10-fold cross validation and then searched for significant differences between folds using two-sided statistical t-tests.

This paper is organized as follows. In the next section, we discuss related work on timeseries classification, sensor data and conceptors. In section 3, we present the dataset we used and its main properties. In section 4, we present the ESN model, the Conceptor Framework (CF) and the method we used to classify timeseries. In section 5, we propose to apply these two models to timeseries classification with various parameters and we analyse these results. Finally, in section 6, we discuss the CF proposal and compare it to the ESN model and other baselines such as Discrete Fourier Transform (DFT) and Singular Value Decomposition (SVD), we also give suggestions for future work.

II. Related Work

In the scientific literature, a wide range of works illustrated the possible applications of ESNs and reservoir computing to classification problems. In [6], the authors investigated the ability of ESNs to classify digits of the well-known MNIST database, demonstrating that those models were capable of achieving a good efficiency in terms of the loss generated, also thanks to a suitable preprocessing of images.

In [7], the authors analyzed the suitability of ESN-based Reservoir Computing models to classify text documents of the Reuters C50 dataset based on authorship, proving that ESNs can be able to achieve real state-of-the-art results on this field, also if considered in comparison with more traditional models such as the Support Vector Machines (SVMs). In natural language processing, ESN has also been applied to cross-domain authorship attribution [8] and author profiling on social network [9]. ESNs have also been applied to a large variety of scientific fields such as robotics [10]–[13], temporal series classification and forecasting [14]–[17]. Conceptor networks have been applied to timeseries prediction [18], image classification [19] and natural language processing [20].

Furthermore, several other works in the related literature described the possible use of sensors for classification tasks, among which is certainly worth mentioning the one [21] where inertial measurement units (IMUs) were attached to the wrists, feet and pelvis of a specific set of climbers in order to automatically detect and classify their climbing activities.

Lastly, other recent studies that emerged in the field of sensor-based automatic classification were those from Leutheuser et al. [22], where a hierarchical, multi-sensor based classification system was developed for the distinction of a large set of individual daily life activities (DLAs), Howcroft et al. [23], where wearable sensors were adopted to classify and analyze the risk of serious health concerns in the elders, and Haladjian et al. [24], where a classification algorithm was developed in order to detect soccer goalkeepers' training exercises using a wearable sensor attached to their gloves.

III. Dataset

The data has been collected with Alogo Move Pro sensors, manufactured by Alogo Analysis, specifically designed to record kinetic data and trajectories of horses during their training sessions. This dataset was gathered as a part of the HorseTrack project which aims to develop and implement a set of applicable methods able to identify horses as well as to detect and predict injuries.

Each session data, completed by one horse, has been subsequently registered in the form of a timeseries with a sampling rate of 10 milliseconds. However, for our training purposes and in order to enhance the efficiency of the learning process, we decided to downsample the original timeseries with a resampling factor of 0.1, making sure that the resulting information loss had no significant impact on the scope of our study.

The original raw collected dataset is composed of 34 columns, 20 of which contain the data used to train the models. More specifically, the sensor recorded the acceleration data (ax, ay, az), the rotation angles around the 3D axes (roll, pitch and yaw), the position vector in the ECEF coordinates system (rx, ry, rz), the velocity vector (vx, vy, vz) and additional information such as intensity, power, energy, speed, distance and height.

The original data presented a split of the timeseries in four gait types assumed by the horses during the respective sessions which are standing, walk, canter and trot. However, this split in the raw dataset was not taken into account for our research purposes, and we trained our models on the entire timeseries. For this reason, we decided to normalize each session independently in order to prevent the training process

[2]https://alogo.io

from being affected by the differences observed in values for each gait.

Moreover, we observed that a relevant number of horses (40, the 54%) completed only one session, determining possible biases in data which could negatively impact on the effectiveness of our models: given this assumption, we decided to apply a further preprocessing of data in order to not consider the horses with a single session generated. The preprocessed dataset presented a total of 187 sessions (over an original number of 227) completed by 34 horses with an average timeseries length of 44,498 timesteps and a total length (for all sessions) of 1,468,434.

The dataset is very biased as the two horses with the greatest number of session have 14 sessions each and the next five horses have 9 sessions. This bias adds a difficulty to the task of classification and we added a majority classifier as the main baseline. This baseline always predicts one horse which has 14 sessions and consequently reaches an accuracy of 7.4%.

IV. Echo State Networks and the Conceptor Framework

A. Echo State Networks

Echo State Networks are composed of three parts. The first is called reservoir and is a set of randomly connected neurons with recurrent connections. The second is the input layer which connect network's inputs to the reservoir, also generated randomly. The third is the output layer which is the target of the training process. The state at time t of this reservoir layer is defined by equation 1 and represented by a non-linear activation vector $\mathbf{h}(t)$.

$$\mathbf{h}(t) = (1-a)\mathbf{h}(t-1) + af(\mathbf{U}\mathbf{x}(t) + W\mathbf{h}(t-1) + \mathbf{b}) \quad (1)$$

with $\mathbf{h}(t) \in \mathbb{R}^{n_h}$, where n_h is the size of the reservoir layer. Connections between inputs $\mathbf{x}(t)$ and the reservoir are represented by the matrix $\mathbf{U} \in \mathbb{R}^{n_h \times n_x}$. n_x is the dimension of the input temporal signal. The matrix $\mathbf{W} \in \mathbb{R}^{n_h \times n_h}$ represents weights of reservoir's internal connection. The null state $\mathbf{h}(t) = 0$ is usually used for initialisation. The leak rate is an essential parameter that allows to adapt the network's dynamic to the one of the task at hand. Finally, $\mathbf{b}$ is the bias vector to the reservoir's units. Regarding the output of the ESN, it is defined by,

$$\hat{\mathbf{y}}(t) = g(\mathbf{W}^{out}\mathbf{h}(t)) \quad (2)$$

$\mathbf{W}^{out} \in \mathbb{R}^{n_y \times n_h}$ is the matrix of output connection weights with n_y as the number of outputs. Function g is usually the identity function. The training process consists in solving a system of linear equation in order to minimise the quadratic error $E(\mathbf{Y}, \mathbf{W}^{out}\mathbf{H})$ between network's hidden states and the target output to be learned. The output matrix $\mathbf{W}^{out}$ can then be obtained using equation 1 to gather reservoir states in a first step, and in a second step using linear methods to $\mathbf{W}^{out}\mathbf{H} = \mathbf{Y}$, where $\mathbf{Y} \in \mathbb{R}^{n_y \times \tau}$ and $\mathbf{H} \in \mathbb{R}^{n_h \times \tau}$ are respectively the matrices containing all the target outputs and the corresponding reservoir states. $\mathbf{W}^{out}$ can be effectively computed using the well-known Ridge Regression to minimise the output weights and avoid overfitting.

$$W^{out} = \mathbf{Y}\mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I})^{-1} \quad (3)$$

The parameter λ is the regularisation factor to fine tune for the task at hand and $\mathbf{I}$ is the identity matrix $\mathbf{I} \in \mathbb{R}^{n_h \times n_h}$.

To evaluate ESNs on our timeseries classification task, we used a *spectral radius* equal to 0.95 and a ridge parameter λ fine-tuned if the network seemed to be facing overfitting. To evaluate the impact of the leak rate and the reservoir size we evaluated the various ESN-based models with different values for these parameters. For each evaluation, we initialized the random number generator used for the generation of $\mathbf{W}$ with the same seed in order to minimise additional factors.

B. The Conceptor Framework

The Conceptor Framework was proposed to establish a new and fresh view on the neuro-symbolic integration problem [5]. The scientific problem describes the challenge faced by artificial intelligence practitioners to implement high-level logical and conceptual reasoning based on neural network low-level dynamic.

The main idea behind the Conceptor Framework is to characterize the patterns of neural activation in a dynamical network using neural filters called *conceptors*. When a neural network such as ESNs, described with equation 1, is driven by different input dynamical patterns $\mathbf{x}_a(t), \mathbf{x}_b(t), ...$, the corresponding states of the reservoir $\mathbf{h}_a, \mathbf{h}_b, ...$ are restricted in different subspace $\mathbf{R}_a, \mathbf{R}_b, ...$ of the reservoir state space.

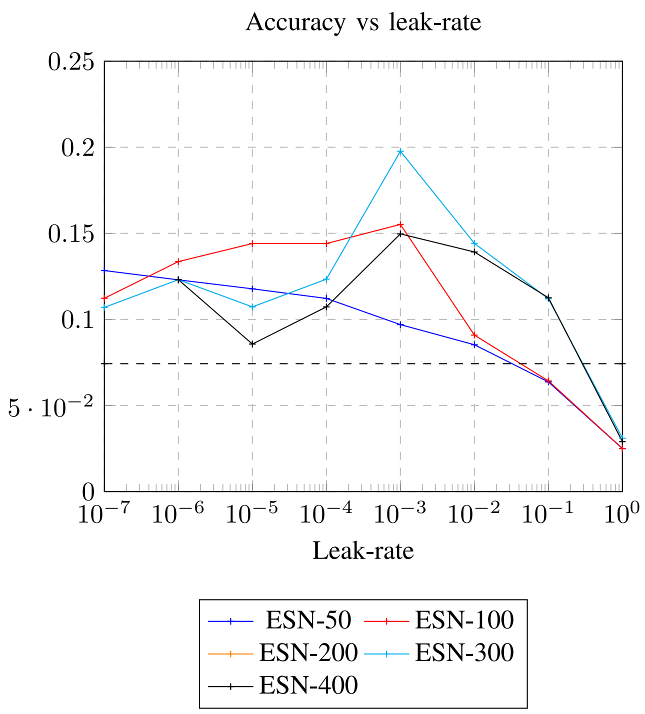
These different subspaces correspond to the different input patterns and are characteristics to them. For each of these region, we can learn incrementally neural filters $\mathbf{C}_a, \mathbf{C}_b, ...$ called *conceptors* which can be used afterwards to re-generate the corresponding pattern by restricting the neural dynamic of an ESN to the subspace $\mathbf{R}$. It can also be used to recognize a pattern by quantifying how a new input pattern populate the different subspaces learned by $\mathbf{C}_a, \mathbf{C}_b, ...$.

More formally, when an ESN is driven by an input pattern $\mathbf{x}(t)$, the resulting timeseries of reservoir states $\mathbf{h}(1), \mathbf{h}(2), ...$ can be seen as a cloud of points in the n_h-dimensional hidden state space. Using principal component analysis (PCA), it is possible characterize this cloud by an n_h-dimensional ellipsoid. Mathematically, the *correlation matrix* $\mathbf{R}$ of the reservoir state points represents this geometrical interpretation. The *singular vectors* $\mathbf{v}_1, ..., \mathbf{v}_{n_h}$ and the *singular values* $\sigma_1, ..., \sigma_{n_h}$ of $\mathbf{R}$ represent respectively the directions and the lengths of the different axes of the ellipsoid representation of input patterns.

The next step to compute the conceptor is to normalized these lengths σ_i to obtain

$$s_i = \frac{\sigma_i}{\sigma_i + \alpha^{-2}} \quad (4)$$

with $\alpha \geq 0$ defined as the *aperture* parameter. As a result, all s_i are lower or equal to 1 and the new ellipsoid is located



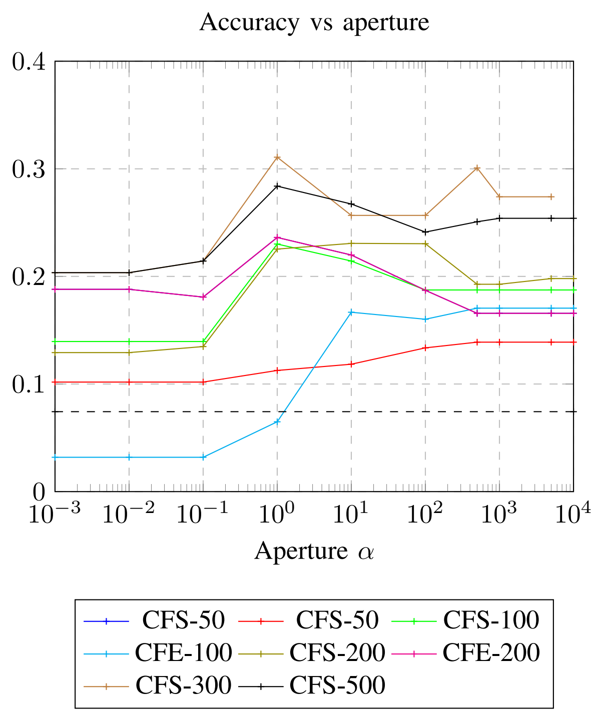


Fig. 1: 10-Fold cross-validation accuracy of ESN and Conceptors models on the HorseTrack dataset with various leaky rates, apertures and reservoir sizes.

inside the unit sphere. The resulting $\mathbf{C} \in \mathbb{R}^{n_h \times n_h}$ describes this normalized ellipsoid and is called the *conceptor matrix*. This matrix can be easily deduced from $\mathbf{R}$ using a linear method such as,

$$\mathbf{C} = \mathbf{R}(\mathbf{R} + \alpha^{-2}\mathbf{I})^{-1} \tag{5}$$

where I is the identity matrix $I \in \mathbb{R}^{n_h \times n_h}$. To understand *aperture*, one can see a conceptor $\mathbf{C}$ as the matrix which minimizes the loss function

$$\lambda(\mathbf{C}) = ||\mathbf{h}(t) - \mathbf{C}\mathbf{h}(t)||^2 + \alpha^{-2}||\mathbf{C}||^2 \tag{6}$$

where $||\mathbf{C}||^2$ is the sum of squared matrix weights. The *aperture* regulate the balance between the two different costs. The first cost ($||\mathbf{h}(t) - \mathbf{C}\mathbf{h}(t)||^2$) seeks to transform the $\mathbf{C}$ matrix into the identity map ($\mathbf{I}$) while the the second push $\mathbf{C}$ towards the zero map.

1) Conceptor Logic: Conceptors can be submitted to a set of logical operation referred as *Conceptor Logic*. The two conceptors $\mathbf{C}_1$ and $\mathbf{C}_2$ obtained from two different patterns can be joined together by the OR operation defined as

$$\mathbf{C_1} \vee \mathbf{C_2} := (\mathbf{R}_1 + \mathbf{R}_2)(\mathbf{R}_1 + \mathbf{R}_2 + \mathbf{I})^{-1} \tag{7}$$

where $\mathbf{C_1} = C(\mathbf{R}_1, 1)$ and $\mathbf{C_2} = C(\mathbf{R}_2, 1)$. The result of this OR operation can be seen as the conceptor which would have been obtained by merging the two data sources which were used previously to compute $\mathbf{C_1}$ and $\mathbf{C_2}$. A NOT operation is also defined as

$$\neg\mathbf{C} := \mathbf{R}^{-1}(\mathbf{R}^{-1} + \mathbf{I})^{-1} \tag{8}$$

Here, the conceptor $\neg\mathbf{C}$ can be seen as resulting from a data source which inversion co-variance coefficients compared to the data source used to compute $\mathbf{C}$ directly.

C. Timeseries classification

In this study, we seek to classify session gathered from sensor and processed as timeseries under different classes representing the horses on which they were collected. Here n_c is number of horses (and then of classes) and the evaluation is done with the following procedure. First, we used a part of the dataset as the training set which we used to compute n_c conceptors incrementally using the OR logical operation, or to train an ESN. At the end of the training phase we have n_c conceptors. The other part of the dataset is used for evaluation and for each session, we used either the ESN as a classifier, an *evidence-based measure* or a *similarity-based measure* to compute the predicted class. We repeat this procedure 10 times to compute the 10-fold cross validation and the corresponding $p-value$ to find significant differences with various baselines.

1) Evidence-based classification: A first approach to classify timeseries used in our study is to defined the predicted class of a new unseen session as the one with the highest *evidence* which is the addition of *positive* and *negative evidences*. The *positive evidence* for class i and a state point at time t is defined as $pev(t) = \mathbf{h}(t)\mathbf{C}_i\mathbf{h}(t)^T$ while the *negative evidence*

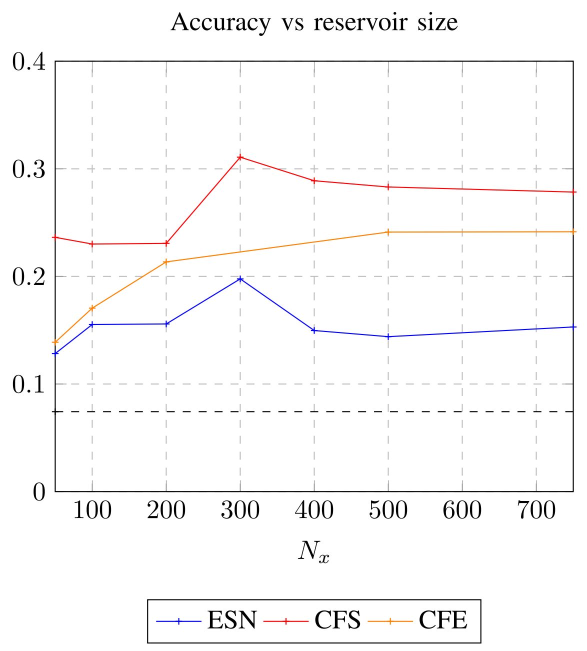


Fig. 2: 10-Fold cross-validation accuracy of ESN and Conceptors models on the HorseTrack dataset with various leaky rates, apertures and reservoir sizes.

is defined by $nev(t) = \mathbf{h}(t)\mathbf{C}_i^- \mathbf{h}(t)^T$. The conceptor $\mathbf{C}_i^-$ is the one obtained by,

$$\mathbf{C}_i^- = \neg \bigvee \{\mathbf{C}_1, ..., \mathbf{C}_{i-1}, \mathbf{C}_{i+1}, ..., \mathbf{C}_{n_c}\} \tag{9}$$

The conceptor $\mathbf{C}_i^-$ can be interpreted representing the fact that the pattern under investigation does not belong to any of classes than the class i. The *combined evidence* is defined by $ev(t) = pev(t) + nev(t)$. The predicted class is the one with the highest average evidence over time.

2) Similarity-based classification: A second approach to timeseries classification based on conceptor is by computing the similarity between conceptors in a profile-based paradigm.

The main idea is to compute the similarity using matrices $\mathbf{V}_i$ and $\mathbf{S}_i$

$$sim(\mathbf{C}, \alpha, i, j) = \frac{\|(\mathbf{S}^i)^{1/2}(\mathbf{V}^i)^T\mathbf{V}^j(\mathbf{S}^j)^{1/2}\|}{\|diag(\mathbf{S^i})\|\|diag(\mathbf{S^j})\|} \tag{10}$$

Where $\mathbf{V}_i$ and $\mathbf{S}_i$ result from the SVD of $C(\mathbf{R}_i, \alpha)$.

For each sample, the predicted class in the one with the highest similarity to the corresponding conceptor learned during the training phase.

V. Results

To evaluate ESNs and Conceptor networks and analyze their behaviours, we looked first for the right parameters for the leak rate (ESN) and the aperture (CF). The leak-rate is very important for ESNs as it regulates the dynamics of the network. It was also essential to determine the temporal properties of timeseries used in this study.

A. Leak-rate

To analyse the dynamical behaviors of ESNs on the HorseTrack dataset we evaluated the accuracy on the task of timeseries classification with leak-rates varying between 1.0 and 10^{-7}. For each leak-rate value, matrices $\mathbf{U}$, $\mathbf{W}$ and $\mathbf{b}$ were kept constant.

The left side of figure 1 shows the 10-fold cross-validation accuracy for five ESNs with different reservoir sizes between 50 ($n_h = 50$) and 400 units ($n_h = 400$). The best results (19.8%, 15.6% and 15.5%) are achieved respectively by the ESNs with 300, 200 and 100 units and a leak-rate of 10^{-3}, 10^{-3} and 10^{-2}. The effect of this leak-rate value cannot be observed for the ESN with 50 units. However, this lower performance can be due to an insufficient number of units.

For the other ESNs, the accuracy quickly increases from leak-rate 1.0 to 10^{-3} and slowly decreases after this value. Regarding statistical significant, we observed after computing the p-value, that only the best accuracy (19.8%) achieved an accuracy significantly different ($p = 3.79\%$) from the majority classifier represented by the black doted line.

In view of these first results, it is clear that this task is very difficult to solve for neural networks such as the ESNs. It could be due to temporal and statistical properties of the timeseries which could not contain the necessary information to identify horses, or to a lack of performance on the ESN side. We then wanted to compare these results to an improved version of ESNs based on conceptor matrices.

B. Aperture

The aperture regulates the opening of the conceptor to input data by modifying the lengths of the axis of the hyper-ellipsoid. This geometrical object represents the pattern created by the input data into the reservoir states space. It should be optimized for this task and we there wanted to evaluate how different conceptor networks behave under different aperture values.

We then tested the 10-CV accuracy of seven different conceptor network. Five based on the similarity metrics (CFS) and three on the evidence function (CFE). For each model, we evaluated their performance with aperture varying between 10^{-3} and 10^4. The right side of figure 1 shows the 10-fold cross-validation accuracy of the evaluated models.

The highest accuracy is reached by similarity-based Conceptor Network with 300 units with 31.1% accuracy. The second and third best results are obtained respectively with the same model with an aperture of 10 and also a similarity-based network with 500 units with 28.4% and 25.7%.

For evidence-based models, the best accuracy is reached by a model with 200 units and an aperture of 100 with 21.3%. Most models achieve their highest accuracy with an aperture of 1.0. Similarity-based models achieved their best results with an aperture of 1.0 while evidence-based models reached highest accuracy with aperture of 10 or 100 but obtained similar results with aperture 1.0. We can conclude that the aperture parameter seems independent of the function used to compute the final prediction.

Classifier	Accuracy
Majority	7.4 %
DFT	3.4 %
SVD	3.8 %
ESN ($n_x = 300$)	19.8 %
Conceptor ($n_x = 300$) (Evidence)	23.6 %
Conceptor ($n_x = 200$) (Similarity)	31.1 %

TABLE I: Comparison of 10-fold cross-validation accuracy of baselines, ESN and Conceptor models on HorseTrack dataset along with the number of parameters to be learned of each model.

All results presented above showed significant differences with the majority classifier with p-values below 1.0%.

C. Reservoir size

In order to evaluate the influence of increasing the number of units in the reservoir and the possible overfitting, the performed evaluations with varying reservoir sizes between 50 and 750 for each kind of models (ESN, CFS and CFE). For each reservoir size, we took the best accuracy found with various apertures. Figure 2 shows the result of these experiments. For CFS, the accuracy slowly grows up to 300 units and slightly decrease but it cannot be stated if it exists significant differences with models above 300 units.

For CFE, the accuracy slowly increases up to 750 units but stay above the results obtained by CFS. For ESN, results are clearly below ones obtained with CFE and CFS with a top performance at 300 units. The accuracy decreases afterwards and these results could be the sign of overfitting. The top performance at 300 units is the only result significantly higher than the majority classifier. If the cause of this drop is overfitting, we can observe than CFE and CFS are not impacted by this issue.

D. Comparison

Table I shows the comparison of our different models with the majority baseline and two common methods based on Discrete Fourier Transform (DFT) and Singular Value Decomposition (SVD). For the first method, we transform the timeseries into vector representation of Fourier coefficents and used these features to find the most similar horse observed during the training phase. This most similar horse was returned as the one predicted by the model. For SVD, we transformed timeseries into a vector representation composed on singular vectors and values and used the same methods as DFT to find the most similar horse in the training set.

The best method found during our evaluation is the Conceptor network based on the similarity measure with 31.08% which outperformed the second best method, the Conceptor network based on evidence with 23.62%. These two Conceptor networks outperformed the best ESN model which obtained 19.76%. These three models obtained results significantly higher than the majority classifier with respectively $p = 0.21\%$, $p = 0.38\%$ and $p = 3.79\%$ The two baselines based on DFT and SVD obtained result not significantly different than the majority classifier and similar to a random classifier.

VI. Conclusions

In this paper, we investigated the possibility to use Echo State Networks-based neural models and Conceptor methods to allow an automatic health monitoring system, aimed to prevent injuries or disease on horses, to incrementally and scalably identify horses based on sensor data. Multi-dimensional data were collected from sensor during horse training sessions and models had to identify which horse was the source of the timeseries. We demonstrated that the extended version of the Reservoir Computing paradigm known as Conceptor network is able to outperform classical ESNs on this task. We evaluated two different ways to perform the classification, the first based on a similarity measure and the second on a function known as evidence. All Reservoir Computing-based methods evaluated in this study had very short training time that did not exceed a few minutes.

Our results show that the classification based on the generalized cosine function outperformed evidence-based models and that similarity-based models can obtained interesting results in timeseries classification. The useful conclusion of this study is the impact of the aperture parameter which stay similar for the different classification methods and the overall performance of conceptor-based Reservoir Computing which can be used with Conceptor Logic to incrementally learn and identify dynamical patterns.

Beyond these preliminary results, an important range for improvement remains possible. In the future, we would like to evaluate methods based on self-supervised *Convolutional Neural Networks*, and classical methods in timeseries classification such as Piecewise Aggregate Approximation (APP) or Auto-Correlation Coefficients (ACC). To evaluate these methods, we will not base our evaluation on a simple accuracy metrics, but on Mean Reciprocal Rank (MRR) as the final system should be able to propose a set of possible candidates. In addition to this metrics, we will also evaluate methods based on the time necessary to analyse a new incoming session.

References

[1] Herbert Jaeger. Short term memory in echo state networks. gmd-report 152. In *GMD-German National Research Institute for Computer Science (2002), http://www. faculty. jacobs-university. de/hjaeger/pubs/STMEchoStatesTechRep. pdf*. Citeseer, 2002.

[2] Wolfgang Maass, Thomas Natschläger, and Henry Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural computation*, 14(11):2531–2560, 2002.

[3] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[4] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[5] Herbert Jaeger. Controlling recurrent neural networks by conceptors. *arXiv preprint arXiv:1403.3369*, 2014.

[6] Nils Schaetti, Michel Salomon, and Raphaël Couturier. Echo state networks-based reservoir computing for mnist handwritten digits recognition. In *2016 IEEE Intl Conference on Computational Science and Engineering (CSE) and IEEE Intl Conference on Embedded and Ubiquitous Computing (EUC) and 15th Intl Symposium on Distributed Computing and Applications for Business Engineering (DCABES)*, pages 484–491. IEEE, 2016.

[7] Nils Schaetti. Behaviors of reservoir computing models for textual documents classification. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2019.

[8] Nils Schaetti. Bidirectional echo state network-based reservoir computing for cross-domain authorship attribution. Notebook for PAN at CLEF, 2018.

[9] Nils Schaetti and Jacques Savoy. Comparison of neural models for gender profiling. In *Proceedings of the 14th international conference on statistical analysis of textual data (Jun 2018)*, volume 470, 2018.

[10] Junichi Kuwabara, Kohei Nakajima, Rongjie Kang, David T Branson, Emanuele Guglielmino, Darwin G Caldwell, and Rolf Pfeifer. Timing-based control via echo state network for soft robotic arm. In *The 2012 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2012.

[11] Paul G Plöger, Adriana Arghir, Tobias Günther, and Ramin Hosseiny. Echo state networks for mobile robot modeling and control. In *Robot Soccer World Cup*, pages 157–168. Springer, 2003.

[12] Eric A Antonelo, Benjamin Schrauwen, Xavier Dutoit, Dirk Stroobandt, and Marnix Nuttin. Event detection and localization in mobile robot navigation using reservoir computing. In *International Conference on Artificial Neural Networks*, pages 660–669. Springer, 2007.

[13] Eric Antonelo, Benjamin Schrauwen, and Dirk Stroobandt. Mobile robot control in the road sign problem using reservoir computing networks. In *2008 IEEE International Conference on Robotics and Automation*, pages 911–916. IEEE, 2008.

[14] Francis Wyffels and Benjamin Schrauwen. A comparative study of reservoir computing strategies for monthly time series prediction. *Neurocomputing*, 73(10-12):1958–1964, 2010.

[15] Paulin Coulibaly. Reservoir computing approach to great lakes water level forecasting. *Journal of hydrology*, 381(1-2):76–88, 2010.

[16] Aida A Ferreira, Teresa Bernarda Ludermir, Ronaldo RB de Aquino, Milde MS Lira, and Otoni Nóbrega Neto. Investigating the use of reservoir computing for forecasting the hourly wind speed in short-term. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 1649–1656. IEEE, 2008.

[17] Xiaowei Lin, Zehong Yang, and Yixu Song. Short-term stock price prediction based on echo state networks. *Expert systems with applications*, 36(3):7313–7317, 2009.

[18] Zheng Xu, Ling Zhong, and Anguo Zhang. Phase space reconstruction-based conceptor network for time series prediction. *IEEE Access*, 7:163172–163179, 2019.

[19] Yuhuang Hu, MS Ishwarya, and LC Kiong. Classify images with conceptor network. *arXiv preprint arXiv:1506.00815*, 2015.

[20] Tianlin Liu, João Sedoc, and Lyle Ungar. Correcting the common discourse bias in linear representation of sentences using conceptors. *arXiv preprint arXiv:1811.11002*, 2018.

[21] Jeremie Boulanger, Ludovic Seifert, Romain Hérault, and Jean-Francois Coeurjolly. Automatic sensor-based detection and classification of climbing activities. *IEEE Sensors Journal*, 16(3):742–749, 2015.

[22] Heike Leutheuser, Dominik Schuldhaus, and Bjoern M Eskofier. Hierarchical, multi-sensor based classification of daily life activities: comparison with state-of-the-art algorithms using a benchmark dataset. *PloS one*, 8(10):e75196, 2013.

[23] Jennifer Howcroft, Edward D Lemaire, and Jonathan Kofman. Wearable-sensor-based classification models of faller status in older adults. *PLoS one*, 11(4):e0153240, 2016.

[24] Juan Haladjian, Daniel Schlabbers, Sajjad Taheri, Max Tharr, and Bernd Bruegge. Sensor-based detection and classification of soccer goalkeeper training exercises. *ACM Transactions on Internet of Things*, 1(2):1–20, 2020.