Improving the Classification Performance of Liquid State Machines Based on the Separation Property

Emmanouil Hourdakis and Panos Trahanias

Institute of Computer Science,
Foundation for Research and Technology – Hellas (FORTH),
Heraklion, Greece

Abstract. Liquid State Machines constitute a powerful computational tool for carrying out complex real time computations on continuous input streams. Their performance is based on two properties, approximation and separation. While the former depends on the selection of class functions for the readout maps, the latter needs to be evaluated for a particular liquid architecture. In the current paper we show how the Fisher's Discriminant Ratio can be used to effectively measure the separation of a Liquid State Machine. This measure is then used as a fitness function in an evolutionary framework that searches for suitable liquid properties and architectures in order to optimize the performance of the trained readouts. Evaluation results demonstrate the effectiveness of the proposed approach.

Keywords: Liquid State Machines, Separation, Fisher Discriminant Ratio.

1 Introduction

Liquid State Machines (LSMs) are based on the concept of using biologically realistic neural networks for processing continuous input channels of information [9,10]. Their powerful computational abilities can be attributed to the capacity of these networks to transform the temporal dynamics of an input signal into a high dimensional spatio-temporal pattern, which preserves recent and past information about the input. This information can be retrieved with a high degree of accuracy using certain types of classifiers.

Following their introduction [9], LSMs have been used in various pattern classification tasks, including speech recognition [14] and movement prediction [2]. The notion behind LSMs has also been extended to problem domains outside computational modeling, where researchers use physical mediums for the implementation of the liquid, such as a bucket of water [5] or real cell assemblies [3].

An LSM consists of three components: *(i)* the liquid, i.e. a pool M of spiking neurons that accepts input from different sources and outputs a series of spike trains, *(ii)* a filter L that is applied on the output of the liquid in order to create a state matrix S, and *(iii)* one or more memoryless readout maps that are trained to extract information from S. The main conception behind this setup is that the complex dynamics of the input are transformed by the liquid to a high dimensional space, in a way that

L. Iliadis and C. Jayne (Eds.): EANN/AIAI 2011, Part I, IFIP AICT 363, pp. 52–62, 2011.

preserves their recent and past properties. This can be compared to a pool of water with a stone thrown in it. The disturbances that are caused in the liquid could be used by a trained observer to deduce the properties of the motion of the stone before entering the water.

To improve the classification performance of an LSM, one must ensure that for two different input histories, the liquid states produced are significantly different [8]. This property, known as separation, has recently received increased attention in the literature, due to its close correlation with the performance of the LSM. In [9] the separation between two different liquid states is calculated by measuring the Euclidean distance of their state vector, i.e. the filtered neuron output sampled at one time instance. A similar geometric interpretation has been given by [7], who measure the separation of the liquid as the Euclidean distance between the centroids of the states that belong to different classes. In [3], the authors use spike train distance metrics instead of Euclidean distance. From the perspective of a classification system, it has been suggested that the rank of the state matrix S can be used to measure the quality of the liquid [8]. According to this measure, the larger the number of linear independent variables produced by a liquid state, the better the classification that can be performed by the LSM.

Attempts to improve the performance of an LSM in the literature have shown that it is very difficult to devise a proper measure or structural criterion to optimize the quality of the liquid. For example in [18] the authors have concluded that randomly generated liquids outperform any attempt to structurally modify the LSM. In [17] the authors use both reinforcement learning and genetic algorithms in order to optimize the classification performance of the LSM. Finally in [16] the authors use the centroid separation measure in order to drive the synaptic modification of the LSM.

In the current paper we propose a criterion that measures the separation of the liquid states that correspond to different classes based on the class means and variances. The classification performance of an LSM is subsequently improved by employing an evolutionary framework to minimize the introduced criterion. Experimental results attest on the performance and accuracy of the proposed approach.

2 Separation Measure

Similarly to Support Vector Machines, the liquid of the LSM acts as a kernel that transforms the low-dimensional space of an input signal to the spatio-temporal space of the liquid. When used for classification, it is important that this transformation yields liquid states that are well separated across different classes. Thus an appropriate measure of the quality of the LSM is a criterion that incorporates the liquid means and variances in order to compute the separation of the data. For a classification task of ω_i different classes we define three quantities.

(a) The between-class scatter matrix:

$$S_b = \sum_{i=1}^{M} P_i \, (\mu_i - \mu_0)(\mu_i - \mu_0)^T \tag{1}$$

where $\mu_0 = \sum_{i=1}^{M} P_i \mu_i$ is the global mean vector for all M classes, P_i is the a priori probability of class ω_i, and μ_i is the mean of the liquid states that correspond to class ω_i.

(b) The within-class scatter matrix:

$$S_w = \sum_{i=1}^{M} P_i \Sigma_i \tag{2}$$

where Σ_i is the covariance matrix of the liquid states that correspond to class ω_i, and P_i is as eq. 1.

(c) The covariance matrix S_m with respect to the μ_0 global mean:

$$\boldsymbol{S_m = S_w + S_b} \tag{3}$$

The trace of the S_m matrix adequately describes the sum of the variances of the features around the global mean, while the trace of the S_w matrix is the sum of the feature variances computed for all classes. Consequently, the separation between different classes can be determined using the following quantity:

$$\boldsymbol{FDR = trace\{S_w^{-1} S_m\}} \tag{4}$$

which is the generalization of the Fisher Discriminant Ratio [FDR, 19] to more than two classes.

To apply the FDR measure on an LSM we sample the spatio-temporal patterns of the liquid's action potentials with respect to each input. These are used to construct the state matrix S:

$$\boldsymbol{S} = \begin{bmatrix} \boldsymbol{s_{11}} & \cdots & \boldsymbol{s_{1j}} \\ \vdots & \ddots & \vdots \\ \boldsymbol{s_{i1}} & \cdots & \boldsymbol{s_{ij}} \end{bmatrix} \tag{5}$$

where $\boldsymbol{s_{ij}}$ is the filtered output of the j^{th} spiking neuron in the liquid and i is the index of the time window in which the outputs of the neurons are sampled. For each task, we create n matrices S, each corresponding to a different class. The FDR measure is then calculated based on eqs. 1-4.

3 Genetic Algorithm Based Liquid Evolution

The separation of the liquid is positively correlated with the performance of the LSM [8]. Consequently, one can improve the classification performance of the trained readouts by designing a liquid that has a high separation measure. To accomplish this we minimize the FDR (eq. 4) of a liquid using genetic algorithms (GAs). GAs are a stochastic optimization method that can optimize complex functions by exploiting their parameter space [6]. Due to their ability to find good solutions in multi-modal and non-differentiable functions they have been extensively used to optimize the performance of neural networks. In these cases researchers encode properties such as the architecture, weights or neuronal models of the network and exploit them in order to minimize some

objective optimization function (see [15] for a review). To evolve the LSM we utilize three types of properties: *(i)* the parameters of the neurons in the liquid, *(ii)* the architecture of the liquid, and *(iii)* the local properties of each architecture. The effect these parameters have on the liquid performance is discussed in the following.

The first part of the chromosome encodes the firing threshold of all neurons in the liquid and the mean of the Gaussian noise added to the neurons' output on every step. These two parameters control the responsiveness and generalization properties of each neuron. In the first case, if the firing threshold of a neuron is low, then it will require to integrate more spikes before firing a post-synaptic potential, making the liquid less responsive to the perturbations of the low input signals. The second parameter controls the mean of the white noise added to each neuron, which affects the generalization properties of the training.

The second part of the chromosome encodes three different architectures for the LSM. Each architecture specifies a different way for connecting the inputs to the liquid, and the inter-liquid connectivity (Fig. 1). In architecture 1, all input neurons are connected to all neurons within the liquid. Thus all task information is integrated in overlapping liquid locations. This is the original setup suggested by [9]. In the second architecture the neurons that encode the input from different sources project to different locations in the liquid. In this case, the resulting LSM will produce a state vector whose entries correspond to particular input properties. The third architecture also incorporates the properties of the input, but discriminates it depending on whether they are temporally varying or static throughout the classification task. Temporally varying inputs project to different locations within the liquid, while the constant input signals are propagated to all neurons.

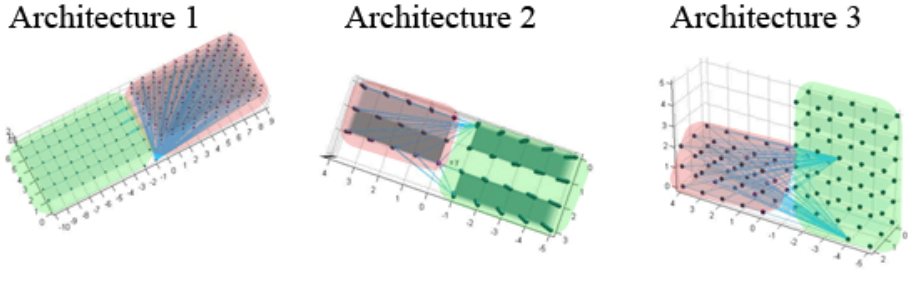

Fig. 1. The three architectures used in the genetic algorithm, shown from a different perspective in order to highlight how the components are connected together. In each architecture the liquid component of the LSM is shaded with red, while the inputs with green.

The last part of the chromosome encodes the properties common to all architectures. These include the size of the liquid map and the locality of the connections in the liquid (i.e. the size of the neighborhood that each neuron is allowed to connect to). Because of the exponential descending output of the dynamic synapses of the LSM [10], the last parameter affects the chaotic dynamics of the liquid, i.e. the period in which a certain input has an effect on the liquid state.

To evolve the chromosomes we use 3 different operators, mutation, crossover and selection. Mutation was implemented by adding a random number drawn from a

Gaussian distribution, with zero mean and standard deviation that starts from 1 and decreases linearly until it reaches 0 in the final generation. To perform crossover, the GA selects (with probability 0.5) a bit from each parent chromosome in order to form a child. Selection was implemented using a roulette wheel function.

4 Experimental Results

In the current section we evaluate the performance of the FDR measure and the GA optimization framework on a number of different classification tasks. For this reason, we focus on two issues: *(i)* the ability of the proposed measure to predict the quality of the liquid in an LSM, and *(ii)* whether the optimization framework can reduce the error of the readouts by minimizing the FDR.

4.1 Measure Evaluation

To evaluate the FDR measure we use three classification tasks, which incorporate different methods for encoding the input. This is important since diverse input encodings can have a different effect on the liquid dynamics. Population codes [11] provide a consistent representation of the input by using distributed and partially overlapping neuron groups in order to encode the values of a variable. In contrast, rate codes [11] produce a higher homogeneity when used as input because they employ the same neuron to represent different input values. Consequently in the three tasks discussed below, rate codes are used to encode the input in two cases, whereas population codes are used in the third case. The liquid in all the aforementioned classification tasks follows the initialization and topology settings discussed in [9]. To evaluate the measure on different classification methods we employ four different readouts: the first readout is implemented with a multi-layer perceptron using the backpropagation rule to train the weights [12]. The second readout implements linear regression [4]. The third readout implements a classifier that uses least squares to find the regression coefficients [4], while the fourth the p-Delta rule on a parallel perceptron layer [1].

For the first classification task, we compare the performance of the two most popular measures in the literature, namely the centroids and rank measures, against the FDR. For this reason we use an LSM with one linear regression readout to classify whether the rate of the input is above a particular value, in this case five Hertz. Input is encoded as a random Poisson rate and applied for 1000ms to a pool of Leaky Integrate and Fire [13] spiking neurons. The liquid used for this purpose consists of a pool of 125 spiking neurons, arranged in a 3 dimensional grid. States are sampled every 10ms for 1 second and input to the 4 readout units.

The FDR, Rank and Centroids measures were *(a)* calculated for the two state vectors S_1 and S_2 from eq. 5 that correspond to classes ω_1 and ω_2, and *(b)* compared against the performance of a linear regression readout for 16 different simulations (Fig. 2).

A good measure should be positively correlated with the error of any trained readout that is used to extract information from the liquid. As Fig. 2 shows there is a clear correlation between the value of the error of a readout map and the value of the FDR measure (for both cases high values close to 1 are colored with red shades, while low values close to 0 are colored with blue). Furthermore, the results presented in Fig. 2 show

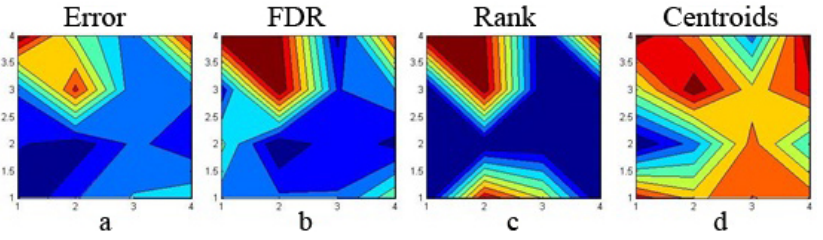

Fig. 2. Evaluation of the FDR, Rank and Centroids measures against their ability to predict the performance of the linear readout map of an LSM. In each subplot the x,y axes correspond to different configurations for an LSM. The color in each x,y entry corresponds to the value of the error (for subplot a) or the negative value of the measure (for subplots b,c,d).

that the proposed measure outperforms the Centroids measure and, at the same time, performs better than the Rank measure. By comparing Fig. 2a and Fig. 2b, it is evident that the FDR measure can predict with satisfying accuracy the performance of the linear regression readout and, therefore, the separation of the liquid in the LSM (readout error/FDR correlation was 0.86). Due to space limitations we do not provide the corresponding contour plots for the other three readouts, although we note that the results were similar to the ones presented in Fig. 2.

The second task requires the LSM to classify two different motions of an object, based on the projection of its image on a 9x9 grid of receptive field neurons. The output of the retina field is encoded as a group of 81 neurons, each one corresponding to a different cell. These neurons fire random Poisson spikes of 30Hz when their corresponding cell in the retina field is occupied, and at a rate of 5Hz otherwise. This output is then projected to a liquid with 63 neurons, where we record the post-synaptic potentials of the neurons for 3 sec (3000ms). Information from the liquid response is used to classify whether the movement on the retina belongs to either one of the two behaviors for 9 different simulations. The error is calculated by subtracting the readout value from the actual behavior being performaned for each step of the simulaton, and normalized to 1. Due to space limitations, results are not presented for this task in the form of contour plots but rather as graphs of the errors of the readouts against the FDR (Fig. 3).

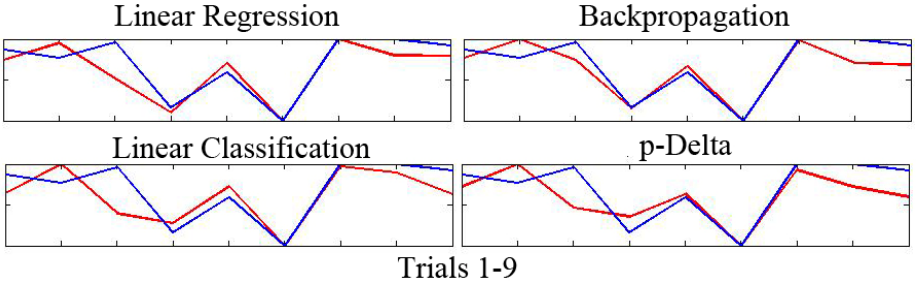

Fig. 3. Graphs of the FDR measure (blue line) against the readout error (red line) of linear regression, backpropagation, linear classification and pDelta methods. The x-axis corresponds to the 9 different trials of the simulation, while the y-axis shows the output of the corresponding error and measure.

As Fig. 3 illustrates, FDR follows quite closely the corresponding error in all cases. The correlation between the FDR measure and the readout error was in all cases above 0.8, indicating a close relationship between the two.

For the third task we use an LSM that must classify the type of three different objects, a circle, a square and a hexagon. To encode the input we first sharpen each image using a Laplacian filter and consequently convolve it with 4 different Gabor filters with orientations π, $\frac{\pi}{2}$, 2π and $-\frac{\pi}{2}$ respectively. The four convolved images from the input are projected into four neuronal grids of 25 neurons, where each neuron corresponds to a different location in the Gabor output. Information from the liquid response is classified by the above mentioned four readouts for 9 different simulations (Fig. 4).

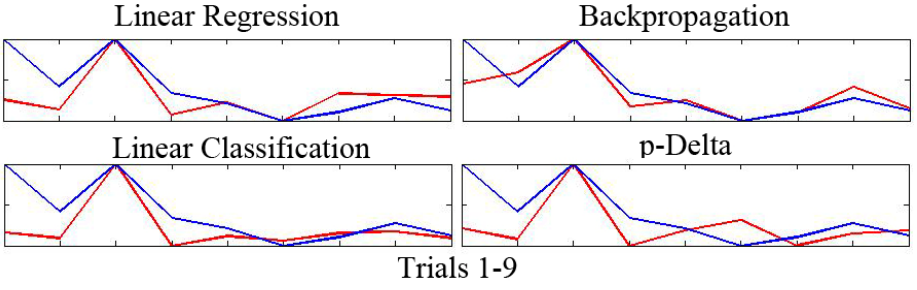

Fig. 4. Graphs of the FDR measure (blue line) against the readout error (red line) of linear regression, backpropagation, linear classification and pDelta methods. The x-axis corresponds to the 9 different trials of the simulation, while the y-axis shows the output of the corresponding error and measure.

The results presented above for all three tasks, indicate that the FDR measure can describe the quality of the liquid over a broad range of tasks and input encodings.

4.2 Liquid Optimization

In the current section we evaluate the extent to which the aforementioned GA framework can optimize the performance of an LSM. For this reason, we consider an additional classification task that requires the integration of temporal information in the liquid states. More specifically, we consider a binary classification task in which an LSM must classify whether the end point of a moving planar robotic arm is closer (or not) than a predefined distance to a given target location. The difficulty of the task lies in the fact that the time-varying control model of the arm must be combined with the static signal of the end point location and produce discrete liquid states, even in cases where the input dynamics are not so different.

Input in this case consists of two different channels. The first encodes the spatial location of an end point position in (x, y) space coordinates and the second the inverse kinematics of different arm trajectories. Input joint positions are generated by creating different trajectories using a two-link planar arm based on one start position (Fig. 5a), three speed profiles (Fig. 5b) and five random ending positions for the train and test sets (Fig. 5c,d). The training set consists of the trajectories between the initial

position (Fig. 5a) and a random end position (Fig. 5c). The test set is generated using a different set of ending positions (Fig. 5d).

To determine the trajectory between a starting and ending position, a random speed profile is chosen from the templates in Fig. 5b. The joint configurations of the robot across the pathway of a trajectory are obtained using an iterative solution to the inverse kinematics problem based on the pseudo-inverse of the robot's Jacobian.

To encode the target position we use a population code with 10 neurons for each dimension (i.e. the x, y coordinates). Thus for the two dimensional space 20 input neurons are used. The simulated robotic arm that is employed in the experiments consists of 2 joints, namely elbow and shoulder, which are also encoded using population codes.

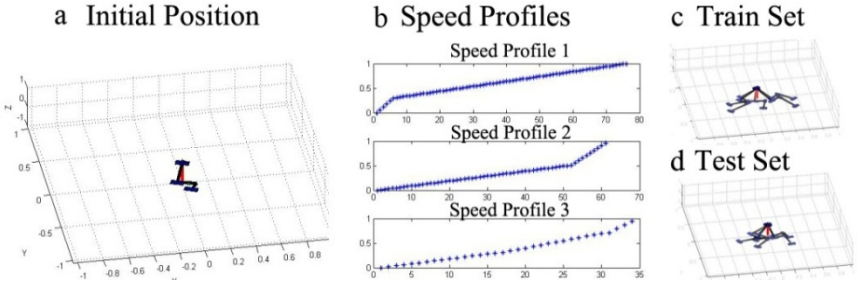

Fig. 5. a. The initial position of the robot's arm. **b.** The three speed profiles used to generate random movements. **c.** The five different configurations of the arm of the robot for the five ending positions of the train set. **d.** The five different configurations of the arm for the test set.

The classification task we consider requires the LSM to predict whether in the next location, the end point of the robot's arm will be closer than a predefined distance to the object in x, y coordinates. To classify a given location correctly, the LSM must make a prediction on the speed of the arm at any given time. Hence, the liquid state must integrate information about the location of the robot's end-point effector position in previous time steps. To generate the different liquid states we have run 100 simulations for the train set kinematics and 30 simulations for the test set kinematics. To learn the classification task, these liquid states were input to 4 readout units, namely a feedforward neural network, a parallel perceptron layer, a linear regression and a linear classification readout.

Even though the results reported here regard the optimal individual produced by the genetic algorithm, it should be noted that similar results were obtained for all chromosomes in the last generation. The GA was run for 18 generations in order to minimize the FDR criterion (Fig. 6). As the four rightmost plots in Fig. 6 show, while the genetic algorithm was used to minimize the FDR measure, it also reduced the error on all four readout maps.

The subplots in Fig. 6 demonstrate how the GA was able to optimize the performance of the LSM by reducing the classification error of the readouts. The error was reduced from 0.3 to 0.1 in all four readouts maps (Fig. 6, right four subplots), simply by optimizing the FDR measure on the liquid (Fig. 6, left subplot) during the evolutionary process. The improvement in the liquid performance is also evident when we examine the output of the four classifiers in the optimal individual produced by the GA. As Fig. 7 shows, all the readouts are able to classify different movements with a high degree of accuracy.

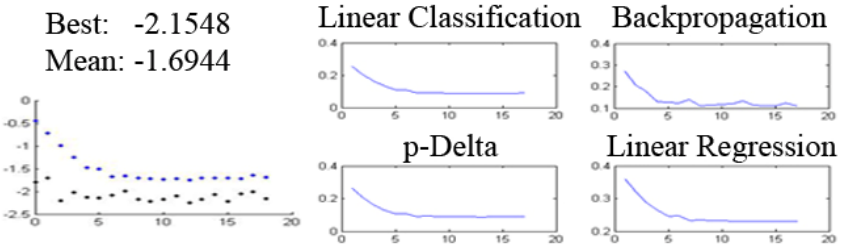

Fig. 6. The results from the evolution of the LSM for 18 generations of the 100 individuals. The left plot shows the best and mean fitness (y-axis) of the population on each generation (x-axis). The right set of four plots shows the average error of the four readouts across the generations (y-axis) of the 100 individuals (x-axis) in the final population.

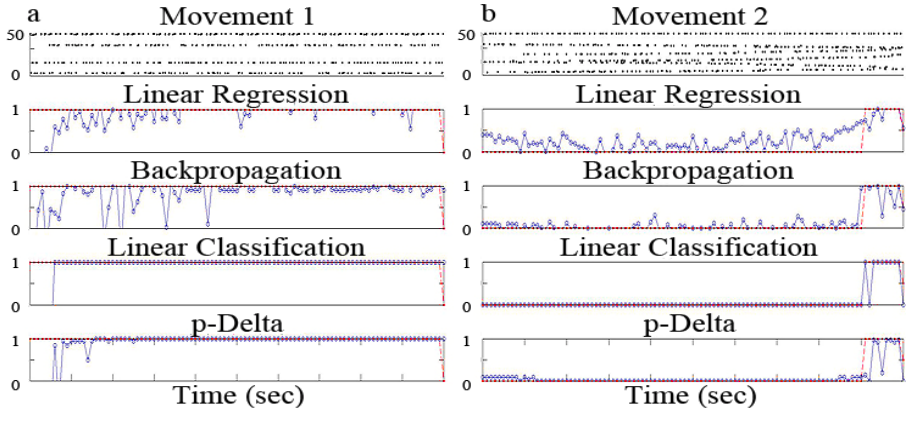

Fig. 7. The output of the four trained readouts for two sample movements (four bottom subplots) and the input projected to the liquid (top subplot)

In the first example (plot a), the robot's arm never reaches the target location in a distance closer than required. In the second (plot b), it approximates the end location in the final 10 simulation steps. In both plots, we show the stimulus input to the liquid (top graph of plots a,b), the output of the four readouts (blue lines in bottom four graphs of plots a,b) and the target for each readout (red lines in bottom four graphs of plots a,b). Each graph is labeled with the corresponding readout map. The x-axis represents the simulation time in 100ms intervals for all graphs.

5 Discussion

In the current paper we presented a method for improving the computational capabilities of LSMs by focusing on the separation property of the liquid. This was measured using the Fisher's Discriminant Ratio, a measure that is maximized when the class means are far from each other, and the class variances are as small as possible. To evaluate the FDR measure against a broad range of classification tasks, we

incorporated different types of neuron encodings for the input. As the results show, the FDR criterion accurately predicts the performance of the readouts without having any knowledge of the algorithm used to train them. Due to this fact, the GA, by minimizing the value of the FDR criterion, also improved the performance of all four classifiers.

In contrast to other criterions, the FDR is a supervised measure, i.e. requires the class labels in order to compute the quantities in eqs. 1-3. Consequently, the evolutionary framework that was presented is also supervised. We consider this a benefit of the method, since it allows the design of liquid architectures that will be suited for the specific dynamics of the classification task.

The proposed methodology will be used to develop a large-scale model of movement imitation and observational learning. In this context, the FDR measure can be employed to optimize the performance of the LSMs that underpin the agent's perception and motor control system. We also plan to investigate whether projecting the liquid states along the eigenvectors of the argument of the FDR measure, can further improve class separation.

References

1. Auer, P., Burgsteiner, H., Maass, W.: A learning rule for very simple universal approximators consisting of a single layer. Neural Nets 21, 786–795 (2008)
2. Burgsteiner, H., Kroll, M., Leopold, A., Steinbauer, G.: Movement prediction from real-world images using a liquid state machine. Applied Intelligence 27(2), 99–109 (2007)
3. Dockendorf, K.P., Park, I., He, P., Principe, J.C., DeMarse, T.B.: Liquid state machines and cultured cortical networks: The separation property. BioSystems 95(2), 90–97 (2009)
4. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification. Wiley, Chichester (2000)
5. Fernando, C., Sojakka, S.: Pattern recognition in a bucket. Advances in ALife, 588–597 (2003)
6. Fogel, D.B.: An introduction to simulated evolutionary optimization. IEEE Trans. Neural Networks 5, 3–14 (1994)
7. Goodman, E., Ventura, D.: Spatiotemporal pattern recognition via liquid state machines. In: Intl. Joint Conf. Neural Networks, IJCNN, pp. 3848–3853 (2006)
8. Legenstein, R., Maass, W.: Edge of chaos and prediction of computational performance for neural circuit models. Neural Nets 20(3), 323–334 (2007)
9. Maass, W., Natschlaeger, T., Markram, H.: Real-time computing without stable states: A new framework for neural computation based on perturbations. Neural Computation 14(11), 2531–2560 (2002)
10. Natschlager, T., Markram, H., Maass, W.: Computer models and analysis tools for neural microcircuits. A practical guide, pp. 123–138 (2003)
11. Rieke, F.: Spikes: Exploring the neural code. MIT Press, Cambridge (1999)
12. Rumelhart, D., Hinton, G., Williams, R.: Learning Internal Representations by Error Propagation. In: Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. 1. MIT Press, Cambridge (1986)
13. Stein: Some models of neuronal variability. Biophysical Journal 7, 37–68 (1967)
14. Verstraeten, D., Schrauwen, B., Stroobandt, D.: Isolated word recognition with the liquid state machine: a case study. In: 13th European Symp. Artificial Neural Nets (ESANN), vol. 95(6), pp. 435–440 (2005)

15. Yao, X.: Evolving artificial neural networks. Proc. of the IEEE 87(9), 1423–1447 (1999)
16. Norton, D., Ventura, D.: Improving the performance of liquid state machines through iterative refinement of the reservoir. Neurocomputing 73, 2893–2904 (2010)
17. Kok, S.: Liquid State Machine Optimization, Master Thesis, Utrecht University (2007)
18. Matser, J.: Structured liquids in liquid state machines, Master Thesis, Utrecht University (2010)
19. Fisher, R.A.: "The Use of Multiple Measurements in Taxonomic Problems" (PDF). Annals of Eugenics 7(2), 179–188 (1936)