Use of the separation property to derive Liquid State Machines with enhanced classification performance

Emmanouil Hourdakis a, b, *, Panos Trahanias a, b

a Institute of Computer Science, Foundation for Research and Technology — Hellas (FORTH), Heraklion, Greece
b Department of Computer Science, University of Crete, Heraklion, Greece

ARTICLE INFO

Available online 2 December 2012

Keywords:
Liquid State Machines
Separation
Fisher Discriminant Ratio

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.

© 2012 Elsevier B.V. All rights reserved.

1. Introduction

Liquid State Machines (LSMs) are based on the concept of using biologically realistic neural networks for processing continuous input channels of information [1,2]. 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 [1], LSMs have been used in various pattern classification tasks, including speech recognition [3] and movement prediction [4]. 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 [6].

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 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 [7]. This property, known as separation, has recently received increased attention in the literature, due to its close correlation with the performance of LSMs. In Ref. [1] the separation between two different liquid states is calculated by measuring the Euclidean distance of their state vectors, i.e. the filtered neuron output sampled at one time instance. A similar geometric interpretation has been given in Ref. [8], in which case the separation of the liquid is measured as the Euclidean distance between the centroids of the states that belong to different classes. In Ref. [6], 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 [7]. 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 Ref. [9] the authors have concluded that randomly generated liquids outperform any attempt to structurally modify the LSM. In Ref. [10] the authors use both reinforcement learning and genetic algorithms in order to optimize the classification performance of the LSM. Finally in Ref. [11] the authors use the centroid separation measure in order to drive the synaptic modification of the LSM.

In a previous work, we have shown how the Fisher Discriminant Ratio can measure the separation of a Liquid State Machine.
based on the class means and variances [12]. In the current paper we capitalize on our previous work and proceed further to evaluate the measure against additional classification tasks, and also use it in an evolutionary framework in order to optimize the quality of an LSM. More specifically, in Section 2 we outline the proposed measure of separation and formulate it mathematically. The classification performance of an LSM is subsequently improved by employing an evolutionary framework to minimize the introduced criterion (Section 3). Section 4 presents experimental results, which attest on the performance and accuracy of the proposed approach, while Section 5 discusses the benefits of the proposed measure and provides suggestions for future work.

2. Separation property

Similarly to Support Vector Machines, 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.

This separation can be geometrically quantified by employing simple criteria to describe how class vectors scatter throughout the domain space. In the current work, we estimate the separation property based on two quantities. The first requires the means of different classes to be as far away as possible from each other, while the second that the class variances are minimal. The reason that these two measures signify the extent to which different classes are separated is illustrated in Fig. 1, for a 2-dimensional, two class case.

Fig. 1 shows how, in the 2-dimensional space, the means and variances of different classes can be used to measure the separation of the dataset. The first measure refers to the means of the two classes, and ensures that the baricenters of the data points are geometrically as far away as possible (Fig. 1a). The second requires that the class variances are small, so that their respective points will not overlap (Fig. 1b). To illustrate this concept, Fig. 1c shows how two classes with large variances can overlap, despite them having well separated means.

In the following section we describe how the separation property can be implemented computationally, and applied to an LSM in order to measure its classification capacity.

2.1. Measure formulation

Based on the separation property discussed in the previous section, we obtain in the current section a formulation of the relevant measures that is directly amenable to implementation. Accordingly, to evaluate a classification task that refers to \( \omega_i \) different classes, we define three quantities:

(a) The between-class scatter matrix:

\[
S_b = \sum_{i=1}^{M} P_i (\mu_i - \mu_0) (\mu_i - \mu_0)^T
\]

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 \( \omega_i \), and \( \mu_i \) is the mean of the liquid states that correspond to class \( \omega_i \). The \( S_b \) matrix is a measure of the average distance between the class means and the global mean of the dataset.

(b) The within-class scatter matrix:

\[
S_w = \sum_{i=1}^{M} P_i \Sigma_i
\]

where \( \Sigma_i \) is the covariance matrix of the data that belong to class \( \omega_i \), and \( P_i, M \) is as in Eq. (1). Consequently, the sum of the elements of the main diagonal in matrix \( S_w \) corresponds to the average variance of all the features in the dataset.

(c) The covariance matrix \( S_m \) with respect to the \( \mu_0 \) global mean:

\[
S_m = S_w + S_0
\]

As Eq. (3) shows, the \( S_m \) matrix can be used as a measure of the sum of variances of the features around the global mean. Consequently, the separation of different classes can be addressed by considering a quantity that is proportional to the trace of the \( S_m \) matrix and inversely proportional to the trace of the \( S_w \) matrix, as described in Eq. (4):

\[
FDR = \frac{\text{trace}(S_0^{-1}S_m)}{\text{trace}(S_w)}
\]

which is the generalization of the Fisher Discriminant Ratio (FDR) [13] to more than two classes. For a one dimensional, two-class problem, it is evident that for equiprobable classes, the matrix \( S_w \) is proportional to \( \sigma_1^2 + \sigma_2^2 \) while \( S_0 \) is proportional to \( (\mu_1 - \mu_2)^2 \), where \( \mu_1, \sigma_1 \) and \( \mu_2, \sigma_2 \) are the class means and variances. In the next sub-section we will outline how Eq. (4) can be used to quantify the separation of the liquid states for different classes.

![Fig. 1](image-url) Graphical illustration of the two measures used to quantify the separation of class data. Class 1 (blue circles) is well separated from class 2 (red circles) if (a) the class means are as far away as possible from each other and (b) the class variances are small. (c) An example of how data points from classes with large variance can overlap. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)
2.2. Integrating the FDR measure with the liquid states

To compute the FDR measure (Eq. (4)) one must obtain a representation that contains discrete class vectors for each dataset. For an LSM, this means that the continually changing dynamics of the liquid states must be transformed into a set of discrete values.

To accomplish this, in the current paper, we represent all liquid states as a set of discrete geometric locations in a \( n \) dimensional input space, where \( n \) is the number of neurons in the liquid. To create this representation, we excite the LSM with a task stimulus, and simulate its neurons for a certain period of time. The discrete spike-trains outputted in each liquid state are filtered using the exponential function in order to obtain a continuous signal that preserves the intensity of the spike train in the temporal domain. The continuous, filtered signal is then sampled with a resolution \( dt \) steps.

The discrete spike trains that are output during simulation are converted to continuous signals using an exponential filter. The output of this filter is then subsequently sampled with a resolution \( dt \), in order to obtain the data for the separation measure.

\[ s_{ij} = \exp(o_{jn}) \]  

where \( o_{jn} \) is the spike train outputted by neuron \( n \). To apply the FDR measure, we sample the spatio-temporal patterns of the liquid’s filtered action potentials with respect to each input, and construct a state matrix \( S \) for each respective class \( i \), as shown in Eq. (6):

\[ S_i = \begin{bmatrix} s_{11} & \cdots & s_{1j} \\ \vdots & \ddots & \vdots \\ s_{i1} & \cdots & s_{ij} \end{bmatrix} \]

where \( s_{ij} \) 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 can then be calculated by obtaining each class’s mean and covariance matrix, and applying Eqs. (1)–(4).

Graphically, the concept can be illustrated in the following figure (Fig. 2). The discrete spike trains that are output during simulation are converted to continuous signals using an exponential filter. The output of this filter is then subsequently sampled with a resolution \( dt \), in order to obtain the data for the separation measure.

In the next section, we describe how the steps of sampling and computing the FDR measure, outlined above, can be integrated in an evolutionary framework that will optimize the performance of an LSM.

3. Genetic algorithm based liquid evolution

As mentioned in the introduction, the separation of the liquid is positively correlated with the performance of the LSM [7]. 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 [14]. 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 Ref. [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. Each of these properties is encoded into a different section of the chromosome that will be employed by the Genetic Algorithm to fine tune the LSM. Below we discuss how these parameters affect the LSM performance.

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. 3). In architecture 1, all input neurons are
connected to all neurons within the liquid. Thus all task informa-
tion is integrated in overlapping liquid locations. This is the
original setup suggested in Ref. [11]. 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.

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 [2], the last parameter affects the chaotic
dynamics of the liquid, i.e. the period in which a certain input
affects the liquid state. Each of these values were encoded using
a real-code representation, as shown in Fig. 4.

To evolve the chromosomes we use 3 different operators,
mutation, two-point crossover and selection. Mutation was imple-
mented 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.
This means that on every new population, half the chromosomes
change based on the cross-over operator, so that the algorithm can
exploit the solution space of the different parameters more thoro-
guously. Selection was implemented using a roulette wheel function.

3.1. LSM implementation

To implement the neurons in the liquid we use the Leaky
Integrate and Fire (LIF) model [16], because it is a computa-
tional convenient way to simulate spike dynamics. In the LIF model,
the evolution of the membrane potential is governed by the following
differential equation:

$$\tau_m \frac{dV_m}{dt} = -(V_m - V_{rest}) + R_m \times (I_{syn}(t) + I_{inject} + I_{noise})$$  \hspace{1cm} (7)

where $V_m$ is the membrane voltage, $\tau_m = C_m \times R_m$ is the membrane
time constant, $R_m$ is the membrane resistance, $C_m$ is the resistor
capacitance, $I_{inject}$ is a constant current injected to the neuron and $I_{noise}$
a Gaussian random variable with zero mean and a small variance noise. After the emission of a spike, the membrane potential is reset to its resting value $V_{rest}$. $I_{syn}(t)$ is the incoming
current from the presynaptic neurons, and is calculated according to
the following equation:

$$I_{syn}(t) = \begin{cases} \sum_{j} w_{ij} EPSP_j(t) & \text{if } t-t(f) > ref \\ 0 & \text{if } t-t(f) \leq ref \end{cases}$$  \hspace{1cm} (8)

The absolute refractoriness of each neuron, i.e. the time period
after the emission of a spike where the input current has no effect
on the membrane potential, is modeled by setting the input current to zero for a short time period (ref) after a spike emission at $t(f)$. $EPSP_j$ is the output of the $j$th pre-synaptic neuron, $t$ is the current simulation time, while $w_{ij}$ is the weight connecting the
presynaptic neuron $i$ and the postsynaptic neuron $j$.

Neurons within the liquid are connected with the dynamic
synapse model suggested in Ref. [17]. In this model, a synapse’s $n$
post synaptic potential ($EPSP_n$) changes dynamically due to the
arrival of new spikes. It is governed by the following equations:

$$EPSP_n = K \times R_n \times u_n$$  \hspace{1cm} (9)

$$u_{n+1} = u_n \exp \left( \frac{-\Delta t}{\tau_{facil}} \right) + U \left( 1 - u_n \exp \left( \frac{-\Delta t}{\tau_{facil}} \right) \right)$$  \hspace{1cm} (10)

$$R_{n+1} = R_n (1 - u_{n+1}) \exp \left( \frac{-\Delta t}{\tau_{rec}} \right) + 1 - \exp \left( \frac{-\Delta t}{\tau_{rec}} \right)$$  \hspace{1cm} (11)

The maximum output of the synapse is governed by the
absolute synaptic efficacy $K$. The change of the efficacy is
determined using the variables $u_n$ and $R_n$, which are calculated
using Eqs. (10), (11) respectively. $\tau_{facil}$ and $\tau_{rec}$ are constant
parameters, experimentally specified. $u_n$ defines the utilization
of the synaptic efficacy which decays exponentially based on the
$\tau_{facil}$ parameter to its resting value $U$. $R_n$ is the fraction of available
synaptic efficacy and defines the strength of the $EPSP_n$ at a given
spike. It reduces due to the arrival of new spikes and recovers
exponentially according to the $\tau_{rec}$ parameter. At $t=0$, the
following initializations occur: $u_n=U$ and $R_n=1$. $\Delta t$ is the time
difference between the nth and (n+1)th spike.

At the initialization of a simulation all neurons are placed in a
3-dimensional grid and are assigned a triplet of $x,y,z$ coordinates.
In the liquid, the probability that two neurons are connected is
governed by the following equation:

$$p_c(a,b) = C(a,b) \times e^{-\frac{\Delta x^2}{\lambda^2}}$$  \hspace{1cm} (12)

Eq. (12) defines the connection probability of two neurons being
connected, according to their distance in this grid. $a$ denotes
the presynaptic neuron and $b$ the postsynaptic neuron. The constant $C(a,b)$ takes different values according to the excitation
status of the pre- and post-synaptic neurons (i.e. whether the
neurons are inhibitory ($I$) or excitatory ($E$)). These are set to 0.2 for
EE, 0.3 for EI, 0.4 for II and 0.2 for IE. $D(a,b)$ is the Euclidean
distance between the two neurons, while $\lambda$ scales the average
length of each connection.

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.
the two state vectors inputted to the 4 readout units. This purpose consists of a pool of 125 spiking neurons, arranged in o classes.

Diverse input encodings can have a different effect on the liquid. Different methods for encoding the input 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 Ref. [1]. 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 [19]. The second readout implements linear regression [20]. The third readout implements a classifier that uses least squares to find the regression coefficients [20], while the fourth the p-Delta rule on a parallel perceptron layer [21].

4.1. Comparison with popular measures in the literature

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 1000 ms to a pool of LIF [16] 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 10 ms for 1 s and inputted 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. (6) that correspond to classes \( \omega_1 \) and \( \omega_2 \), and (b) compared against the performance of a linear regression readout for 16 different simulations (Fig. 5).

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. 5 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. 5 show that the proposed measure outperforms the Centroids measure and, at the same time, performs better than the Rank measure. By comparing Fig. 5a and b, 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).

4.2. Measure Evaluation

In the current section we consider three additional classification tasks, in order to evaluate whether the FDR measure can predict the performance of an LSM while solving them. Each task incorporates a different method for encoding the input. This is important since diverse input encodings can have a different effect on the liquid dynamics. Population codes [18] 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 [18] 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 Ref. [1].

4.2.1. Classifying different behaviors

The first task requires the LSM to classify two different motions of an object, based on the projection of its image on a 9 \( \times \) 9 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 30 Hz when their corresponding cell in the retina field is occupied, and at a rate of 5 Hz otherwise. This output is then projected to a liquid with 63 neurons, where we record the post-synaptic potentials of the neurons for 3 s (3000 ms).

The LSM is used to classify two different motions of an object, when it is projected on a 9 \( \times \) 9 grid of receptive field neurons. Fig. 6 illustrates a sample motion of 5 s duration. Information from the liquid response is classified using four readouts: (i) linear regression, (ii) feedforward neural network, (iii) linear classification and (iv) p-Delta rule.

The liquid must learn 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 performed for each step of the simulation, and normalized to 1. Results for this task are presented as graphs of the errors of the readouts against the FDR (Fig. 9a).

As Fig. 9a illustrates, FDR follows quite closely the corresponding error in all cases. The correlation between the FDR measure...
The readout error was in all cases above 0.8, indicating a close relationship between the two.

4.2.2. Classifying different objects types based on their shape

For the second task we use an LSM that must classify the type of three different objects, a circle, a square and a hexagon, based on their images. To encode the input we first sharpen each image using a Laplacian filter and consequently convolve it with 4 different Gabor filters with orientations $\pi$, $\pi/2$, $2\pi$ and $-\pi/2$ respectively (Fig. 7).

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.

\[
\begin{bmatrix}
1 & 1 & 1 & 1 \\
-a & a-1 & a+1 & 1 \\
-a & a+1 & a-1 & -a
\end{bmatrix}
\]

Fig. 7. The Laplacian filter used to convolve each image (left) and the result of the convolution of the 4 Gabor filters with the polygon image (right). In each row on the right image we show two pairs of the filter in different directions (left subplot for each pair) and the result of the convolution (right subplot for each pair).

Fig. 8. The representation of the retina field of the input and the encoded stimulus representation of this field. The plot demonstrates two different positions. (a) retina field activations for position 15 (left) and spike trains generated for the LSM (right), and (b) retina field activations for position 9 (left) and spike trains generated for the LSM (right).

Fig. 9. Graphs of the FDR measure (blue line) against the readout error (red line) of linear regression, backpropagation, linear classification and p-Delta 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. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)
As the four plots in Fig. 9b show, the FDR measure (blue line) is able to predict with high accuracy the classification error for each one of the four readouts used (red line in each plot).

4.2.3. Classifying the location of an object on a retina field

For the third task, we use an LSM that must classify whether an object is on a certain location upon a grid. To encode the spatial representation of the environment we use 25 Poisson neurons that become active only when the object’s position is upon their respective location in the grid. Neurons fire with an intensity of 50 Hz if they are located at the center of the object’s position, while at 20 Hz if they are placed in one of the neighboring cells. Fig. 8 illustrates the representation and encoding of the stimulus used for the LSM, for two different positions.

The output of the retina field is projected onto an LSM, which is simulated for 1 s (1000 ms). The liquid states, after being filtered and sampled, are used to calculate the FDR measure for each different position, i.e. the classification contains 25 classes, each corresponding to a different location in the grid. In Fig. 9c we illustrate how the FDR measure was able to predict the performance of the LSM for the current task, for all four readouts.

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. Having established an accurate measure of the quality of performance of the LSM, in the following section we examine whether a GA framework can optimize the performance of the classification performed by the liquid.

4.3. 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. 10a), three speed profiles (Fig. 10b) and five random ending positions for the train and test sets (Fig. 10c and d). The training set consists of the trajectories between the initial position (Fig. 10a) and a random end position (Fig. 10c). The test set is generated using a different set of ending positions (Fig. 10d).

To determine the trajectory between a starting and ending position, a random speed profile is chosen from the templates in Fig. 10b. 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.

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 \(xy\) 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 conducted 100 simulations for the train set kinematics and 30 simulations for the test set kinematics. To learn the classification task, these liquid states were inputted 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 set to terminate when the cumulative change of the fitness functions over 5 generations was below the value of 0.01. This resulted in an evolutionary process of 18 generations, during which the FDR criterion reached a very low value (Fig. 11). As the four rightmost plots in Fig. 11 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. 11 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. 11, right four subplots), simply by optimizing the FDR measure on the liquid (Fig. 11, 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. 12 shows, all the readouts are able to classify different movements with a high degree of accuracy.

In the first example (Fig. 12, plot a), the robot’s arm never reaches the target location in a distance closer than required. In the

---

**Fig. 10.** (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, and (d) the five different configurations of the arm for the test set.
second (Fig. 12, 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 100 ms intervals for all graphs.

4.4. GA framework evolved LSM parameters

In the current section we present the statistics of the genetic evolution with respect to the chromosome parameters for the 30 optimal individuals in the final generation. These are important since they can describe the strategies employed by the GA to solve the classification task. We also measure the effect that each parameter has on the liquid performance, by calculating its correlation with the error of the linear regression readout.

As Table 1 shows, the architecture of a liquid has a significant effect on the performance of the readouts, since its correlation with the error is very high. In the current task, the first architecture produced fitter individuals than architectures 2 and 3. The locality of the connections (l parameter) and the noise added to each neuron (I noise) are also positively correlated. The medium value of 3 for the l parameter in the optimal chromosomes shows that local synapses are better suited for connecting the neurons in a liquid. Finally, the threshold of the neurons and the size of the liquid are negatively correlated with the liquid’s performance. The latter result has also been pointed out in Ref. [1].

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 on 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 considered 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 are suited to the specific dynamics of a general classification task. Due to the high correlation of the FDR measure with the performance of an LSM (Figs. 5, 9 and 11) it can be applied in any optimization method in order to improve the performance of an LSM.

Moreover, one important issue about the FDR measure is that it does not make any assumptions about the structure of the data. For example, other methods of class separability, such as the divergence or Bhattacharyya distance [20] must make a Gaussian assumption in order to be computed. In contrast, the FDR measure is constructed from simple low-level criteria, that describe the geometrical scatter of feature vectors in the problem space, and therefore does not make any assumption about the data.

In the current paper we have used Genetic Algorithms, because they are a stochastic optimization method, and therefore require the measure to make consistently good predictions regarding the performance of the liquid. The small number of generations required to reach convergence suggests that additional methods could be employed in order to exploit the solution space of the motor problem more accurately.

In the future we plan to use additional optimization methods with the FDR measure, in order to investigate how the classification accuracy of an LSM could be increased further. For example we plan to employ different methods (e.g. Monte Carlo) in order to derive more elaborate conclusions about the effect of different parameters on the liquid’s performance. Furthermore, we will investigate new ways of reducing the computational load of LSMS, by exploiting additional methods for increasing the separation between different classes. In this context, we will evaluate whether transforming the liquid states, by projecting them along the eigenvectors of the argument of the FDR measure, can increase their separation. This addition has the potential to complement the proposed methodology and offer a concrete set of mathematical tools that can evaluate and further advance the computational performance of LSMS. Moreover, the application of the proposed framework in relevant tasks seems very promising. Candidate tasks are the ones that can readily benefit from the enhanced computational abilities of the proposed LSM structure, such as (a) the development of large-scale computational models that are composed of multiple components with distinct functionalities and (b) scene segmentation in video sequences.

References


Emmanouil Houdakis received his Ph.D. in Computer Science from the University of Crete, Greece (March 2012). Currently he is a Postdoctoral researcher at the Computational Vision and Robotics laboratory of the Foundation for Research and Technology — Hellas (FORTH). He received his Ph.D. in Computer Science from the National Technical University of Athens, Greece, in 1988. Following that, he had positions as Research Associate at the Institute of Informatics and Telecommunications, National Center for Scientific Research “Demokritos”, Athens, Greece (1989–1991), and at the Department of Electrical and Computer Engineering, University of Toronto, Toronto, Canada (1991–1993). He has participated in many research projects in image processing and analysis at the University of Toronto and has been a consultant to SPAR Aerospace Ltd., Toronto. Since 1993, he is with the University of Crete and FORTH. He has held the position of Director of Graduate Studies at the Department of Computer Science, University of Crete and currently he chairs the same Department. At FORTH he heads the Computational Vision and Robotics Laboratory, where he coordinates research and development activities in human–robot visual interaction, robot navigation, visual tracking, and brain-inspired robotic control. He has coordinated and participated in many research projects funded by the European Commission and Greek funding agencies. He has participated in the Programme Committees of numerous International Conferences and has been General Chair of Eurographics 2008 (EGC08) and the European Conference of Computer Vision 2010 (ECCV 10). He has published over 110 papers in technical journals and conference proceedings.

Panos Trahanias is a Professor with the Department of Computer Science, University of Crete, Greece and the Institute of Computer Science, Foundation for Research and Technology — Hellas (FORTH). He received his Ph.D. in Computer Science from the National Technical University of Athens, Greece, in 1988. Following that, he had positions as Research Associate at the Institute of Informatics and Telecommunications, National Center for Scientific Research “Demokritos”, Athens, Greece (1989–1991), and at the Department of Electrical and Computer Engineering, University of Toronto, Toronto, Canada (1991–1993). He has participated in many research projects in image processing and analysis at the University of Toronto and has been a consultant to SPAR Aerospace Ltd., Toronto. Since 1993, he is with the University of Crete and FORTH. He has held the position of Director of Graduate Studies at the Department of Computer Science, University of Crete and currently he chairs the same Department. At FORTH he heads the Computational Vision and Robotics Laboratory, where he coordinates research and development activities in human–robot visual interaction, robot navigation, visual tracking, and brain-inspired robotic control. He has coordinated and participated in many research projects funded by the European Commission and Greek funding agencies. He has participated in the Programme Committees of numerous International Conferences and has been General Chair of Eurographics 2008 (EGC08) and the European Conference of Computer Vision 2010 (ECCV 10). He has published over 110 papers in technical journals and conference proceedings.