Are Probabilistic Spiking Neural Networks Suitable for Reservoir Computing?

Stefan Schliebs, Ammar Mohemmed and Nikola Kasabov

***Abstract*— This study employs networks of stochastic spiking neurons as reservoirs for liquid state machines (LSM). We experimentally investigate the separation property of these reservoirs and show their ability to generalize classes of input signals. Similar to traditional LSM, probabilistic LSM (pLSM) have the separation property enabling them to distinguish between different classes of input stimuli. Furthermore, our results indicate some potential advantages of non-deterministic LSM by improving upon the separation ability of the liquid. Three non-deterministic neural models are considered and for each of them several parameter configurations are explored. We demonstrate some of the characteristics of pLSM and compare them to their deterministic counterparts. pLSM offer more flexibility due to the probabilistic parameters resulting in a better performance for some values of these parameters.**

I. Introduction

THE desire to better understand the remarkable information processing capabilities of the mammalian brain has recently led to the development of more complex and biologically plausible connectionist models, namely spiking neural networks (SNN). See e.g. [1] for a comprehensive standard text on the material. These models use trains of spikes as internal information representation rather than continuous variables. By explicitly including time into the neural model, especially *recurrent* networks of spiking neurons are believed to be suitable methods for processing temporal information. However, training algorithms for such networks have proved to be very difficult to develop; see the excellent review on supervised learning algorithms for SNN by [2].

Liquid State Machines (LSM) [3] represent an elegant way to exploit the computational capabilities of recurrent SNN without the need to directly train the network itself. LSM employ concepts of the reservoir computing (RC) paradigm [4]; see [5] for a review on recent trends in this research field. The reservoir approach was shown to be very suitable to process spatio-temporal data [6], [7].

A LSM consists of two main components, a "liquid" (also called reservoir) in the form of a recurrent SNN and a trainable readout function. The liquid is stimulated by spatio-temporal input signals causing neural activity in the SNN that is further propagated through the network due to its recurrent topology. A snapshot of the reservoir contains information about the current and past inputs to the system. The function of the liquid is to accumulate the temporal and spatial information of all input signals into a single high-dimensional intermediate state in order to enhance the separability between network inputs. The readout function is then trained to transform this intermediate state into a desired system output. Since the integration of inputs over time provides the reservoir with a fading memory of previous input events, the read-out function can be memory-less and comparatively simple. In fact, a linear learning algorithm such as linear regression or single layer perceptron have shown to be sufficient to learn the mapping of the reservoir response to a desired network output [3].

Earlier studies have investigated the suitability of different neural models in the context of LSM. In particular, the well-known integrate-and-fire [3] and the Hodgkin-Huxley [8] model were considered, but also Resonate-and-Fire, FitzHugh-Nagamo, Morris-Lecar, Hindmarsh-Rose and Izhikevich neurons have been investigated [9]. Although the LSM are clearly inspired by the biological micro-circuit of the brain, all of these neural models are deterministic. In this study, we address the question whether recurrent networks of probabilistic neurons are principally suitable to be employed as reservoirs. More specifically, we focus on the demonstration of the separation property of a probabilistic LSM (pLSM), i.e. the distance of two liquid states obtained after the separate injection of two input stimuli A and B is roughly proportional to the distance between A and B. A potential advantage of employing non-deterministic neurons in a reservoir has been alluded already in some initial experiments [10]. Here a non-deterministic LSM was constructed using some simple extensions of the Leaky Integrate-and-Fire (LIF) model. Designed as a small-scale experimental study, it was concluded that a pLSM may have the potential to increase the ability of the reservoir to separate between input classes. In this study, we provide a significantly larger experimental analysis for demonstrating the characteristics of pLSM.

II. Probabilistic Neural Models

In this section, we describe the probabilistic neural models we have used to replace the deterministic LIF neurons of a traditional LSM. The probabilistic approach is motivated by the fact that also biological neurons exhibit significant stochastic characteristics. Including non-deterministic elements into the neural model may reveal a benefit for the resulting brain-like information processing system. Models of probabilistic neurons have been proposed in many studies, e.g. in the form of dynamic synapses [11], the stochastic integration of the post-synaptic potential [1] and stochastic firing thresholds [12], but also in [13] where the spike propagation and generation are defined as stochastic processes.

The authors are with the Knowledge Engineering and Discovery Research Institute, Auckland University of Technology, New Zealand (email: {sschlieb,amohemme,nkasabov}@aut.ac.nz).

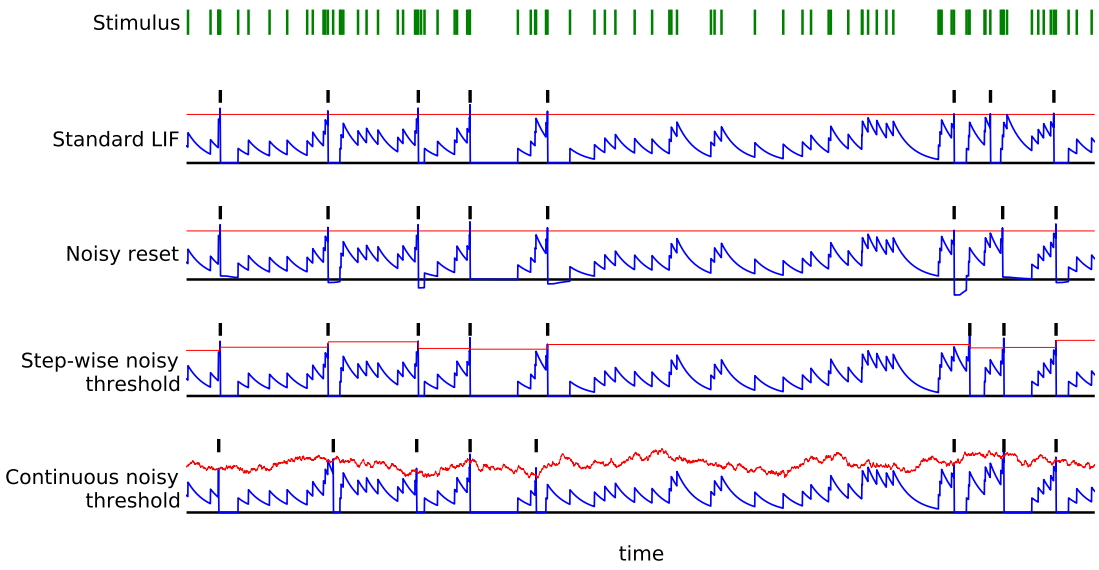


Fig. 1. Evolution of the post-synaptic potential $u(t)$ and the firing threshold $\vartheta(t)$ over time (blue (dark) and yellow (light) curves respectively) recorded from a single neuron of each neural model. The input stimulus for each neuron is shown at the top of the diagram. The output spikes of each neuron are shown as thick vertical lines above the corresponding threshold curve.

In this study, we employ some very simple probabilistic extensions of the LIF model. These stochastic models are well-known and are comprehensively described in [1]. Based on a brief summary of the LIF neural model, we explain the probabilistic extensions in the next paragraphs.

The LIF neuron is arguably the best known model for simulating spiking networks. It is based on the idea of an electrical circuit containing a capacitor with capacitance C and a resistor with resistance R, where both C and R are assumed to be constant. The model dynamics are then described by the following differential equation:

$$\tau_m \frac{du}{dt} = -u(t) + R\ I(t) \tag{1}$$

The constant τ_m is called the membrane time constant of the neuron. Whenever the membrane potential u crosses a threshold ϑ from below, the neuron fires a spike and its potential is reset to a resting potential u_r. It is noteworthy that the shape of the spike itself is not explicitly described in the traditional LIF model. Only the firing times are considered to be relevant.

We define a *stochastic reset* (SR) model that replaces the deterministic reset of the potential after spike generation with a stochastic one. Let $t^{(f)} : u(t^{(f)}) = \vartheta$ be the firing time of a LIF neuron, then

$$\lim_{t \to t^{(f)}, t > t^{(f)}} u(t) = \mathcal{N}(u_r, \sigma_{SR}) \tag{2}$$

defines the reset of the post-synaptic potential. $\mathcal{N}(\mu, \sigma)$ is a Gaussian distributed random variable with mean μ and standard deviation σ. Variable σ_{SR} represents a parameter of the model.

We define two stochastic threshold models that replace the constant firing threshold ϑ of the LIF model with a stochastic one. Once more, let $t^{(f)}$ be the firing time of a LIF neuron. In the *step-wise stochastic threshold* (ST) model, the dynamics of the threshold update are defined as

$$\lim_{t \to t^{(f)}, t > t^{(f)}} \vartheta(t) = \mathcal{N}(\vartheta_0, \sigma_{ST}) \tag{3}$$

Variable σ_{ST} represents the standard deviation of the Gaussian distribution $\mathcal{N}$ and is a parameter of the model. According to Eq. 3, the threshold is the outcome of a ϑ_0-centered Gaussian random variable which is sampled whenever the neuron fires. We note that this model does not allow spontaneous spike activity. More specifically, the neuron can only spike at time $t^{(f)}$ when also receiving a pre-synaptic input spike at $t^{(f)}$. Without such a stimulus a spike output is not possible.

The *continuous stochastic threshold* (CT) model updates the threshold $\vartheta(t)$ continuously over time. Consequently, this model allows spontaneous spike activity, i.e. a neuron may spike at time $t^{(f)}$ even in the absence of a pre-synaptic input spike at $t^{(f)}$. The threshold is defined as an Ornstein-Uhlenbeck process [14]:

$$\tau_\vartheta \frac{d\vartheta}{dt} = \vartheta_0 - \vartheta(t) + \sigma_{CT}\sqrt{2\tau_\vartheta}\xi(t) \tag{4}$$

where the noise term ξ corresponds to Gaussian white noise with zero mean and unit standard deviation. Variable σ_{CT} represents the standard deviation of the fluctuations of $\vartheta(t)$ and is a parameter of the model. We note that $\vartheta(t)$ has an overall drift to a mean value ϑ_0, i.e. $\vartheta(t)$ reverts to ϑ_0 exponentially with rate τ_ϑ, the magnitude being in direct proportion to the distance $\vartheta_0 - \vartheta(t)$.

The dynamics of the four models are presented in Figure 1. For each model a single neuron is shown that is stimulated by a random spike train generated by a Poisson process with

mean rate 150Hz. Both the evolution of the post-synaptic potential $u(t)$ and the evolution of the firing threshold $\vartheta(t)$ are recorded and shown in the figure. We note the step-wise and the continuous update of the two threshold models and the stochastic reset of the reset model. Due to the stochastic dynamics each probabilistic model displays a different spike output pattern compared to the deterministic LIF neuron.

III. Experiments

The experimental setup of the presented study is illustrated in Figure 2. Four recurrent SNN are generated each employing one of the neural models described above. All networks have the same network topology and the same connection weight matrix. A detailed description of the network generation and parametrisation is given in the next section. The networks are stimulated by two input spike trains A and B. In our experiments we leave A constant, but choose spike trains B of varying similarity to A. The exact definition of the synthetic data is explained in detail in the next section. For different pairs $\{A, B\}$, the response of the reservoir, i.e. the liquid state, to the inputs is recorded. We are interested in the separation capability of the constructed LSM (using different stochastic neural models) regarding the two presented input spike trains. For suitable liquids, the similarity of the obtained liquid states should be roughly proportional to the similarity of A and B.

For comparing the separation properties of different randomly generated reservoirs, we adopt an interesting metric that was recently introduced in [15]. For this procedure the responses of a reservoir to input stimuli are recorded. Each input stimulus is labelled and belongs to one of n classes. The recorded liquid states O are divided into subsets O_l, one subset for each class. The idea of the separation metric is to determine the ratio between the inter-class distance c_d and the intra-class variance c_v. The inter-class distance is defined as:

$$c_d = \sum_{l=1}^{n} \sum_{m=1}^{n} \frac{\|\mu(O_l) - \mu(O_m)\|_2}{n^2} \tag{5}$$

where $\mu(O_l)$ is the center of mass for each class l:

$$\mu(O_l) = \frac{\sum_{\mathbf{o} \in O_l} \mathbf{o}}{|O_l|} \tag{6}$$

The notation $|\cdot|$ is used for set cardinality and $\|\cdot\|_k$ corresponds to the L_k-norm.

The intra-class variance is defined as the mean variance of a set of state vectors O_l:

$$c_v = \frac{1}{n} \sum_{l=1}^{n} \rho(O_l) \tag{7}$$

where

$$\rho(O_l) = \frac{\sum_{\mathbf{o} \in O_l} \|\mu(O_l) - \mathbf{o}\|_2}{n} \tag{8}$$

The separation of a liquid Ψ that produces the response $O = \{O_l | l = 1, \ldots, n\}$ is then defined as

$$\mathrm{Sep}_\Psi(O) = \frac{c_d}{c_v + 1} \tag{9}$$

We refer to [15] for a more detailed discussion of the separation metric.

A. Synthetic data

For our investigations we have created a synthetic data set which was inspired by the study presented in [16]. The data resembles a binary classification problem in which the clusters represented by the two classes may overlap. By controlling the extent of the overlapping, we investigate how well a LSM can differentiate between the two classes. For suitable reservoirs, the extent of overlap should be proportional to the separation capabilities of the reservoir.

We created the data in the following manner. First, a large set of random spike trains was generated by a Poisson process with a mean rate of 150Hz. From this set, we randomly selected two spike trains A and B having the same number m of spikes in their sequence. Consequently, A and B differ only regarding their spike times. Using these two sequences, some additional inputs were derived by generating copies of A and shifting these copies by a step s towards B. Formally, we computed the difference d between the spike times of A and B and divided it by the number k of additional spike trains to be generated.

$$d(a, b) = \frac{b - a}{k} \tag{10}$$

Here $a, b \in \mathbb{R}^m$ refer to vectors of ordered spike times observed in the spike trains A and B respectively. Now a copy a' of the spike times a that is shifted by a step $s \in \{1, \ldots, k\}$ towards b can be obtained:

$$a'(s) = a + s \times d \tag{11}$$

Using this procedure, we generated $k = 9$ shifted and jittered copies of A.

For the spike trains A, B and all generated copies A', 50 jittered samples are created by subjecting the individual spike times of each train to a Gaussian noise having a standard deviation of 1ms. The created data set is shown in Figure 3. Each row of spike trains represents samples belonging to the same class. In the experiments, we created binary classification problems by selecting a set of two classes $\{a, a'(i)\} \forall 1 \leq i \leq k$ from the data and passing them to the reservoir for investigating its separability.

B. Setup

We construct a reservoir having a small-world interconnectivity pattern as described in [3]. A recurrent SNN is generated by aligning 1000 neurons in a three-dimensional grid of size $10 \times 10 \times 10$. In this grid, two neurons A and B are connected with a connection probability

$$P(A, B) = C \times e^{\frac{-d(A,B)}{\lambda^2}} \tag{12}$$

where $d(A, B)$ denotes the Euclidean distance between two neurons and λ corresponds to the density of connections which was set to $\lambda = 2$ in all simulations. Parameter C depends on the type of the neurons. We discriminate into excitatory (ex) and inhibitory (inh) neural types resulting in

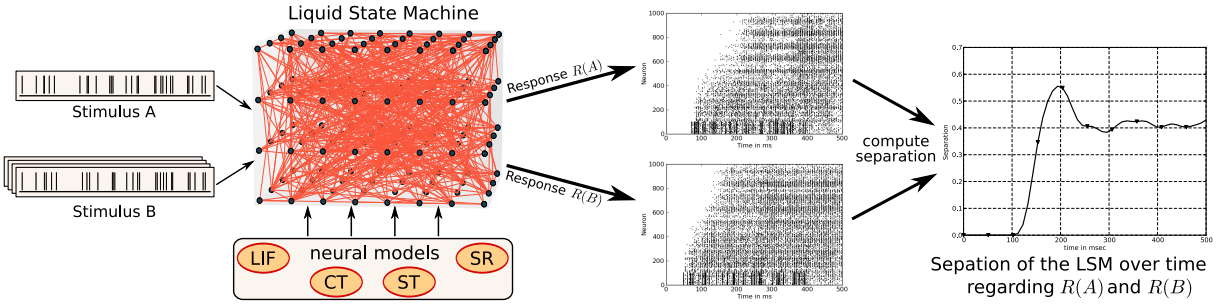


Fig. 2. Experimental setup of the study.

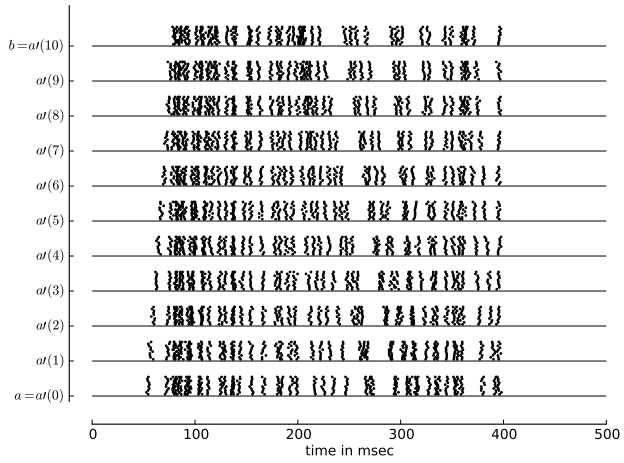


Fig. 3. Synthetic data used in the experiments. Samples of spike trains are generated by successively shifting the spike times a towards the spike times b and applying a Gaussian jitter.

Parameter name	Value
Membrane time constant in ms	$\tau_m = 30$
Resting potential in mV	$u_r = 0$
Firing threshold in mV	$\vartheta_0 = 5$
Refractory period in ms	$\Delta^{\text{abs}} = 3$
Synaptic delay in ms	5
Standard deviation of reset fluctuations in mV	$\sigma_{SR} \in \{0.1, 0.5, 1, 1.5, 2\}$
Standard deviation of step-wise firing threshold in mV	$\sigma_{ST} \in \{0.01, 0.05, 0.1, 0.5, 1\}$
Standard deviation of continuous firing threshold in mV	$\sigma_{CT} \in \{0.01, 0.05, 0.1, 0.5, 1\}$

TABLE I
PARAMETERS OF THE NEURAL MODELS

the following parameters for C: $C_{ex-ex} = 0.3$, $C_{ex-inh} = 0.2$, $C_{inh-ex} = 0.4$ and $C_{inh-inh} = 0.1$. The network contained 80% excitatory and 20% inhibitory neurons. All parameter values are directly adopted from [9]. The membrane potential of the neurons is randomly initialized in the interval $[0mV, 4mV]$.

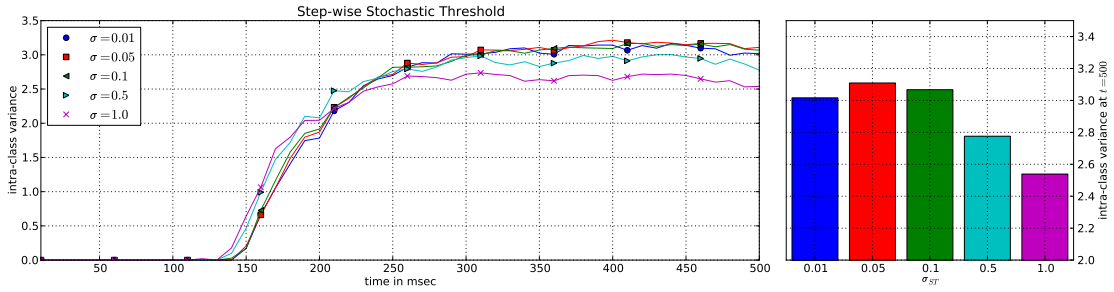
We define the 10×10 neurons located in the first layer of the grid to be the input neurons. These neurons are stimulated by the input spike trains of our synthetic dataset. The last layer of neurons in the grid (consisting 10×10 neurons as well) corresponds to the output layer. The spiking activity of these neurons occurring in a time window of 10ms is recorded and referred to as the liquid state of the reservoir at a given time. More specifically, the liquid state at time t is a binary vector indicating whether a certain neuron has fired or not in the time window $[t, t + 10\text{ms}]$.

It is noteworthy that the generation of a suitable liquid, i.e. a liquid that can satisfyingly separate between different classes of inputs, is not an easy task. Suitable liquids are commonly identified by generating numerous random networks and selecting the one that maximizes the classification accuracy of the readout function. Some interesting alternative approaches have suggested to use Hebbian learning strategies for improving the characteristics of the liquid [15], [17]. In this study, we restrict ourselves to explore a single reservoir only and leave the optimization of the liquid for future directions. In other words, we keep the topology of the network along with its connection weight matrix constant for all the experiments. Consequently, the tested reservoirs differ only in the employed neural model from each other.

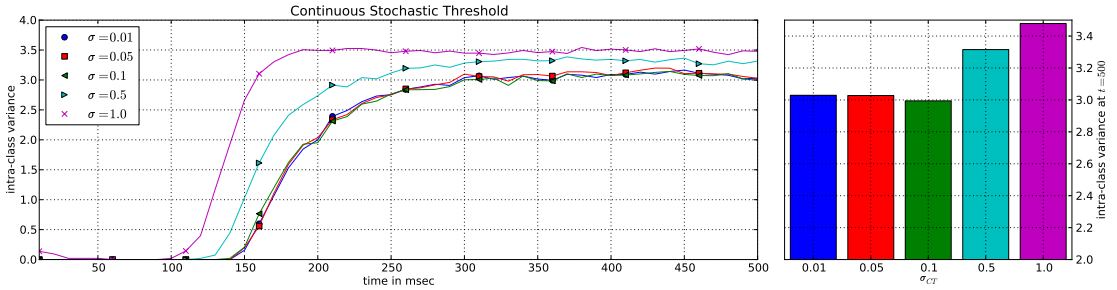
For the deterministic LIF model we chose a "typical" parameter configuration that is similar to the one described in [3]. Using this configuration, the recorded liquid states did not exhibit the undesired behavior of *over-stratification* and *pathological synchrony* – effects that are common for randomly generated liquids. Pathological synchrony occurs when the neurons of the liquid are caught in an infinite loop of maximum activity, while over-stratification describes the effect of a too low spiking activity in the network. Both behaviors decrease the separation abilities of the liquid [17]. Two typical liquid states for the LIF neurons is shown Figure 2. The parameters of the stochastic neural models were varied and are summarized in Table I. All simulations were performed using the SNN simulator Brian [18].

C. Impact of stochastic parameters

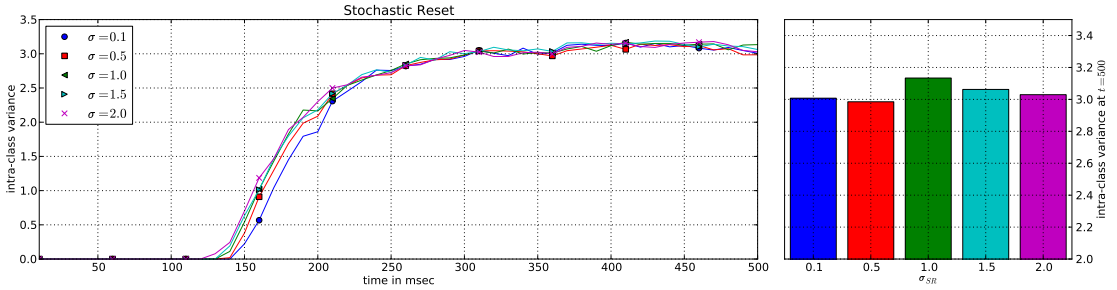
In order to demonstrate the impact of the stochastic parameters on the behavior of the reservoir, we have recorded the liquid states of a probabilistic reservoir that was stimulated by the same input spike train in 100 runs. From the recorded responses, the intra-class variance was computed for each time window $[t, t + 10\text{ms}]$ using Eq. 7. The evolution of the



(a)



(b)



(c)

Fig. 4. Evolution of the intra-class variance over time for reservoirs employing different classes of neural models (left diagrams). For each neural model, several configurations are shown demonstrating the impact of these settings on the intra-class variance of the reservoir. The bar charts on the right show the intra-class variance of the last time window for different configuration of the model. This variance is generally used for determining the separation ability of the reservoir.

intra-class variance over time is shown in Figure 4 for each of the stochastic models. On the right hand-side, the variance occurring in the last time window is shown as a bar for each configuration. The variance at the end of the simulation is generally important for the separation of output classes and is exploited by the readout function of the LSM, as for example done in [9], [15].

The most obvious impact on the recorded liquid states has parameter σ_{CT} of the continuous stochastic threshold model, cf. Figure 4b. The intra-class variance clearly increases with increasing σ_{CT}. Interesting to note is the intra-class variance between time $t = 0$ms and $t = 25$ms for $\sigma_{CT} = 1$. This behavior is caused by a spontaneous neural activity after the random initialization of the membrane potential. Due to its variations, the firing threshold can fall below the current membrane potential of the neuron triggering a spontaneous spike. The variations of the other tested values of $\sigma_{CT} \leq 0.5$ appear to be too small to activate neurons immediately after initialization.

For the step-wise stochastic threshold model and the stochastic reset model, the effects of the stochastic parameters are less obvious. However, especially at the middle stages of the simulation ($t \approx 150$ms), they follow a pattern similar to the one observed in continuous stochastic threshold model, cf. Figure 4a and 4c. The lower variance for $\sigma_{ST} = 1$ in Figure 4a is caused by a decreased overall neural activity of the reservoir. Since the threshold is only updated when a neuron fires, the sampling of a too high threshold can effectively prevent a neuron from firing during a simulation. The loss of the neural activity of some neurons prevents the stimulation of connected neighboring neurons which in turn will further decrease the overall activity of the network. This chain reaction is more likely to occur for larger σ_{ST} and the effect is visible for $\sigma_{ST} \geq 0.5$ in the diagram.

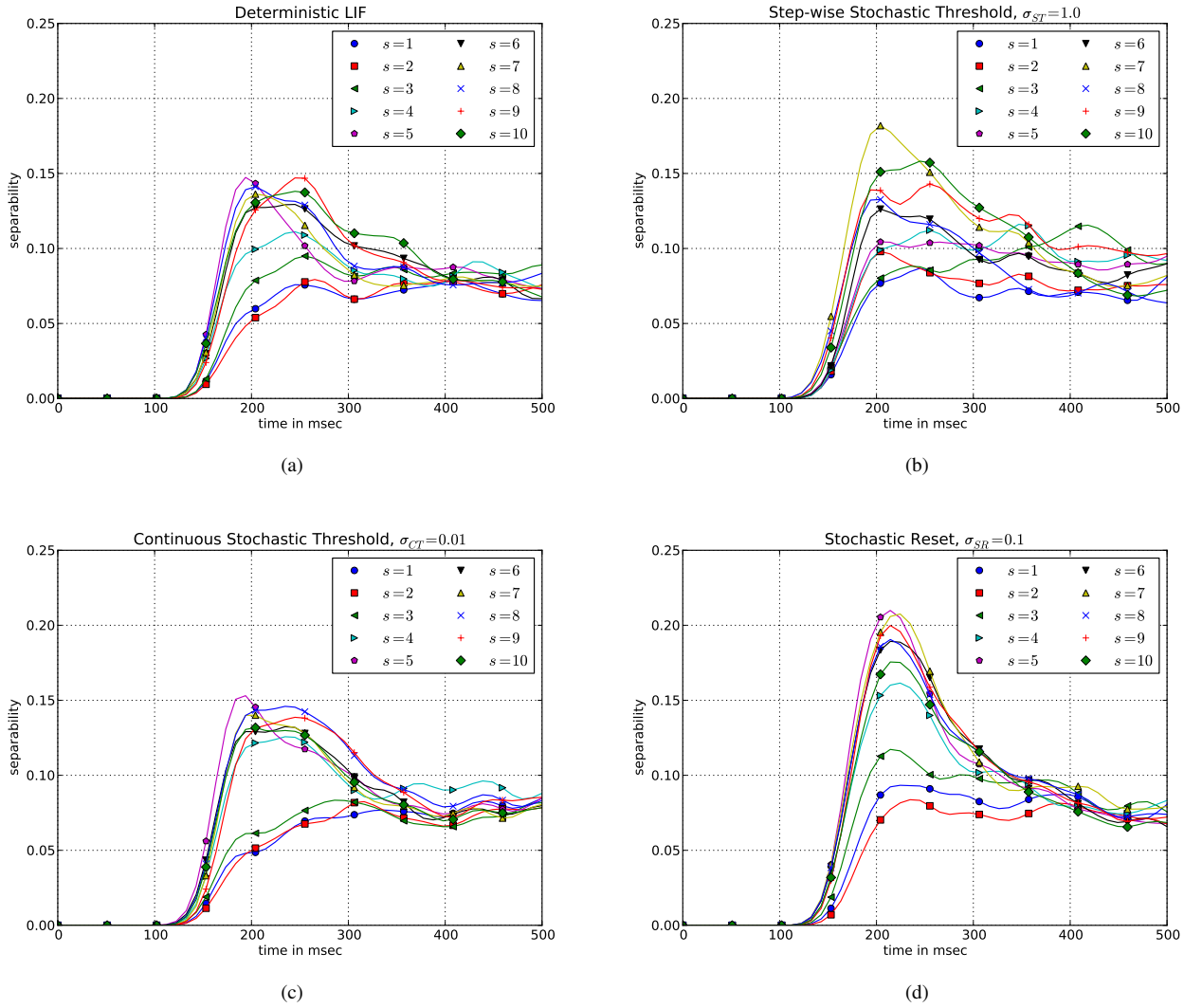


Fig. 5. The evolution of the separation of a reservoir over time for the tested neural models. Each diagram shows a reservoir employing a different neural model. The curves in the diagrams represent the separation of the reservoir regarding the difference between input classes (shifting step s controls the difference/overlap between the samples of two input classes).

D. Results

Figure 5 shows the evolution of the separation Sep_Ψ as defined in Eq. 9 over time for the tested neural models. While each diagram shows the behavior of a particular neural model, each curve in a diagram reflects the separation of the reservoir regarding the stimulus A a jittered and shifted copy A' for a certain shifting step s. Independent of the employed neural model, a maximal separation is achieved at $t \approx 200$ms. As expected, all of the reservoirs can separate A and A' better if A' is closer to B. This characteristic is clearly a desired behavior, since it enables the reservoir to separate between different input classes.

Furthermore, we observe a general order in the separation curves. The smaller the shifting step s and thus the smaller the difference between the input spike trains A and A', the smaller the separation of the reservoir. This behavior is very prominent especially for the stochastic reset model at $t = 200$ms. However, a similar order is also observed in the reservoirs using the other neural models.

After time $t = 300$, most tested reservoirs suffer a decrease of separation Sep_Ψ. At the end of the simulation these reservoirs show a separation capability that is independent from the difference of the input stimuli. An exception of this observation is given by the step-wise stochastic model with $\sigma_{ST} = 1$. Its capability to separate A from some of the more distant spike trains (e.g. $A'(9)$) can still be detected until the end of the simulation. In this respect this reservoir exhibits a more desirable behavior than the deterministic model.

The shape of the separation curves may appear surprising at first. However, similar shapes have been reported in [3]. Due to the applied inner-class jitter (and the stochastic effects occurring in the pLSM), differences in the input stimuli can rapidly amplify in the liquid over time. The initial differences between the liquid states around $t = 200$ are decreasing over

time due to the chaotic effects of the reservoir.

It is noteworthy that the separation for the stochastic reservoir is generally equal or higher compared to the deterministic reservoirs. Since the separability is proportional to the classification accuracy of the readout neurons [7], [15], this observation may indicate an important advantage of probabilistic reservoirs over deterministic ones.

In order to further investigate this finding, we have computed the average separation for all neural models in the time window $[100\text{ms}, 300\text{ms}]$. In this time interval the separation is maximal for all tested reservoirs. The average separation regarding the shifting step s is presented in Figure 6. We have applied a smoothing with a small window length on the curves in order to improve the clarity of the figure. The correlation between the difference of the input classes and the separation capability of the reservoir is clearly demonstrated. For all stochastic neural models we can identify a configuration that behaves very similar to the deterministic LIF model. Some of the reservoirs employing stochastic models, especially the stochastic reset model, show superior separation compared to the deterministic reservoir.

IV. Conclusion and Future Directions

In this study, we have addressed the question whether non-deterministic neural models are principally suitable liquids in the context of reservoir computing. We have experimentally shown that, similar to traditional LSM, also probabilistic LSM have the separation property enabling them to distinguish between different classes of input stimuli. Our results have indicated some potential advantages of non-deterministic LSM, since they may improve upon the separation ability of the liquid. However, additional analysis is necessary to support this hypothesis. A future study should consider the application of Hebbian learning techniques that allow a significant improvement of the quality of the liquid. This includes dynamic learning as it can reduce the variability of the neural response to noisy input stimuli [19] and have been proposed in the context of LSM in [15], [17] and in the context of Echo State Machines in [20]. Furthermore, we are interested in the performance of pLSM when a large number of input classes have to be separated from each other utilising some principles of the evolving neural network classification framework [21].

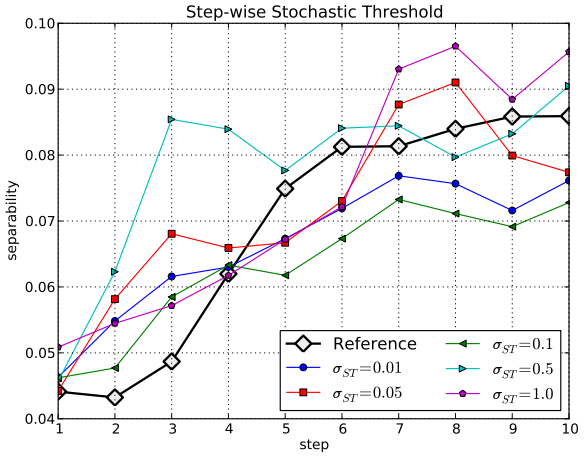
References

[1] W. Gerstner and W. M. Kistler, *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge, MA: Cambridge University Press, 2002.

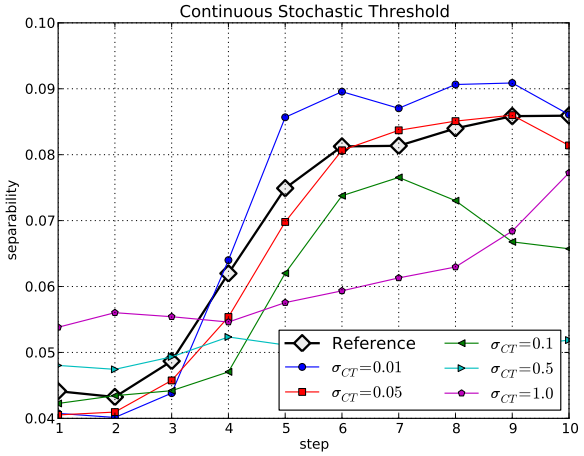
[2] A. J. Kasinski and F. Ponulak, "Comparison of supervised learning methods for spike time coding in spiking neural networks," *Int. J.of Applied Mathematics and Computer Science*, vol. 16, pp. 101–113, 2006.

[3] W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," *Neural Computation*, vol. 14, no. 11, pp. 2531–2560, 2002.

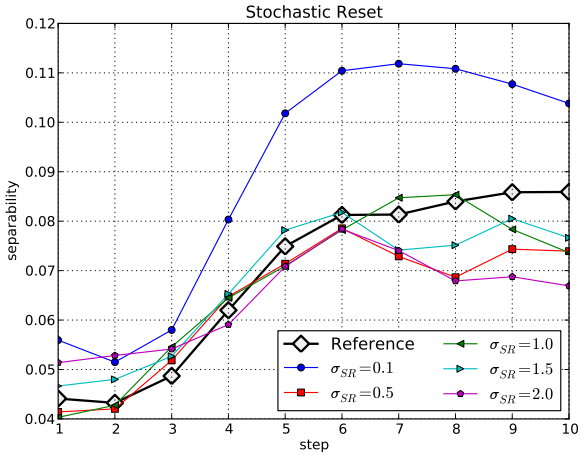
[4] D. Verstraeten, B. Schrauwen, M. D'Haene, and D. Stroobandt, "An experimental unification of reservoir computing methods," *Neural Networks*, vol. 20, no. 3, pp. 391 – 403, 2007.



(a)



(b)



(c)

Fig. 6. Average separation of the tested reservoirs in the time interval $[100\text{ms}, 300\text{ms}]$ in dependence on the shifting step s. The larger the shifting step, the larger the difference between the samples of the considered binary classification problem becomes. The correlation between the difference of the input classes and the separation capability of the reservoir is clearly demonstrated.

[5] B. Schrauwen, D. Verstraeten, and J. V. Campenhout, "An overview of reservoir computing: theory, applications and implementations," in *Proceedings of the 15th European Symposium on Artificial Neural Networks*, 2007, pp. 471–482.

[6] D. Verstraeten, B. Schrauwen, and D. Stroobandt, "Isolated word recognition using a liquid state machine," in *ESANN*, 2005, pp. 435–440.

[7] E. Goodman and D. Ventura, "Spatiotemporal pattern recognition via liquid state machines," in *Neural Networks, 2006. IJCNN '06. International Joint Conference on*, Vancouver, BC, 2006, pp. 3848–3853.

[8] G. M. Wojcik and W. A. Kaminski, "Liquid state machine built of hodgkin-huxley neurons and pattern recognition," *Neurocomputing*, vol. 58-60, pp. 245 – 251, 2004, computational Neuroscience: Trends in Research 2004.

[9] B. J. Grzyb, E. Chinellato, G. M. Wojcik, and W. A. Kaminski, "Which model to use for the liquid state machine?" in *IJCNN'09: Proceedings of the 2009 international joint conference on Neural Networks*. Piscataway, NJ, USA: IEEE Press, 2009, pp. 1692–1698.

[10] S. Schliebs, N. Nuntalid, and N. Kasabov, "Towards spatio-temporal pattern recognition using evolving spiking neural network," in *ICONIP'10*, vol. 0. Sydney, Australia: IEEE Computer Society, 2010.

[11] W. Maass and A. Zador, "Dynamic stochastic synapses as computational units," in *Advances in Neural Information Processing Systems*. MIT Press, 1999, pp. 903–917.

[12] C. Clopath, R. Jolivet, A. Rauch, H.-R. Lüscher, and W. Gerstner, "Predicting neuronal activity with simple models of the threshold type: Adaptive exponential integrate-and-fire model with two compartments," *Neurocomput.*, vol. 70, no. 10-12, pp. 1668–1673, 2007.

[13] N. Kasabov, "To spike or not to spike: A probabilistic spiking neuron model," *Neural Networks*, vol. 23, no. 1, pp. 16–19, 2010.

[14] N. G. van Kampen, *Stochastic Processes in Physics and Chemistry*. North-Holland, 2007.

[15] D. Norton and D. Ventura, "Improving liquid state machines through iterative refinement of the reservoir," *Neurocomputing*, vol. 73, no. 16-18, pp. 2893 – 2904, 2010, 10th Brazilian Symposium on Neural Networks (SBRN2008).

[16] S. Schreiber, J. Fellous, D. Whitmer, P. Tiesinga, and T. Sejnowski, "A new correlation-based measure of spike timing reliability," *Neurocomputing*, vol. 52-54, pp. 925 – 931, 2003, computational Neuroscience: Trends in Research 2003.

[17] D. Norton and D. Ventura, "Preparing more effective liquid state machines using hebbian learning," in *International Joint Conference on Neural Networks, IJCNN 2006*. Vancouver, BC: IEEE, 2006, pp. 4243–4248.

[18] D. Goodman and R. Brette, "Brian: a simulator for spiking neural networks in python," *BMC Neuroscience*, vol. 9, no. Suppl 1, p. P92, 2008.

[19] S. M. Bohte and M. C. Mozer, "Reducing the variability of neural responses: A computational theory of spike-timing-dependent plasticity," *Neural Comput.*, vol. 19, pp. 371–403, February 2007.

[20] Š. Babinec and J. Pospíchal, "Improving the prediction accuracy of echo state neural networks by anti-oja's learning," in *Artificial Neural Networks - ICANN 2007*, ser. Lecture Notes in Computer Science, J. de Sá, L. Alexandre, W. Duch, and D. Mandic, Eds. Springer Berlin / Heidelberg, 2007, vol. 4668, pp. 19–28.

[21] S. G. Wysoski, L. Benuskova, and N. Kasabov, "Evolving spiking neural networks for audiovisual information processing," *Neural Networks*, vol. 23, no. 7, pp. 819 – 835, 2010.