# Spiking Neural Network for Speech Intonation Modeling

Loïc Jeanningros
Idiap Research Institute, Martigny, Switzerland

October 18, 2019

**Abstract**

Understanding the mechanisms behind the production of speech intonation requires a physiological model of the pitch production process. Intonation is fully characterized by the fundamental frequency ($f0$) of the speech signal that relies on laryngeal muscles. The Generalized Command-Response (GCR) model that is used approximates muscle dynamics in response to neural spikes. For this reason, a Spiking Neural Network is proposed to learn the sequences of spikes driving muscle responses. It is a spike-based reservoir network inspired from neocortical neural microcircuits called the Liquid State Machine (LSM). It allows to process continuous input streams into a high-dimensional state performing diverse operation and integration of both present and past inputs. Motor commands are extracted from the LSM through Remote Supervison (ReSuMe and SPAN learning rules) in an online manner. Such online computations that transform time-varying input streams into time-varying output streams have never been applied to intonation before. An analysis of the LSM's separation property as well as a comparison of ReSuMe and SPAN algorithm are provided. The model can accurately reproduce single speech utterances. However, the prediction of unseen utterances is not achieved. We hypothesize that the generalization failure is due to both motor commands encoding scheme and bad generalization-capability of the presented LSM.

## 1 Introduction

Intonation is a prosodic feature of speech that carries non-linguistic information such as emphasis and emotion. As a distorted pitch can change the meaning of a sentence, or can reveal the speaker's emotional state, a good model of intonation is crucial for speech-to-speech translation systems that intend to transfer para-linguistics between languages.

In previous work with colleagues [7], a physiologically plausible intonation ($f0$) model based on the Command-Response (CR) model of Fujisaki [4] has been investigated. Then, a Generalized CR (GCR) [31] that extracts atoms from an intonation contour using a matching pursuit algorithm [18] has been presented. Because GCR atoms approximate muscle responses to incoming neural spikes,

we propose to use a Spiking Neural Network (SNN) for learning sequences of spikes that generate intonation contours through muscle responses.

Learning in traditional Artificial Neural Network is usually performed by gradient techniques. However, no explicit evaluation of the gradient in SNNs can be made due to the discontinuous-in-time nature of spikes. Nevertheless, indirect approaches have been attempted : SpikeProp algorithm [2] employed difference between desirable and actual time of output spike as error. A supervised spike-based learning algorithm based on a probabilistic approach [25] optimizes the likelihood of post-synaptic firing by gradient ascent. Nevertheless, these algorithms do not enable learning patterns composed of more than a single or few spikes, respectively. Another algorithm [9] allows to learn spike trains, but break the all-or-none nature of synaptic currents.

Error back-propagation seems to be inefficient for learning in SNN. Hence, we turned to an alternative to paradigm based on attractor neural networks, called reservoir computing (RC). RC is attractive for non-experts developers due to the simplicity of the training method, it meets the demands for low training cost and real-time processing of many applications [33]. The basic idea behind this new trend of understanding training is that, as long as a recurrent neural network possesses certain generic properties, supervised adaptation of all interconnection weights is not necessary. Only training a memory-less supervised readout from it is enough to obtain excellent performance in many tasks [14].

The Liquid State Machine (LSM) [17] is the first spike-based version of reservoir neural networks. It is a high-dimensional network based on neocortex microcircuits observation. Such network can be seen as a kernel on which input signals are projected. Recurrent connections are not trained, thus LSM's activity is task-independent. Under certain conditions, the liquid circuit approximates the computation of any linear and nonlinear mathematical operation and also provides all needed temporal integration of information [15]. Since the liquid state has only one attractor: the resting state, inputs can be seen as perturbations liquid state. Thus, readout neurons can learn to extract in real-time from the current perturbed liquid state about current and past inputs. The LSM is a model for real-time computations on continuous streams of data such as spike trains [15], hence it is appropriate for the intonation production task.

Reservoir computing has recently shown promising performance in various speech processing tasks [34, 35, 10, 40], in particular LSM with a spike-based learning algorithm for speech recognition [39]. However, as far as we know, it has never been applied to speech intonation modelling, and speech synthesis in general. The closest study seems to be an FORCE-trained spike-based reservoir network [23] that successfully reproduces songbird singing.

Readout neurons learn to extract motor commands from neural circuit information through the

plasticity of connections from the reservoir. A couple of Remote Supervision learning rules allow online local synaptic adaptation: SPAN [21, 22], that has been adapted in previous work [31] and ReSuMe [27, 28] that has proven itself adequate for learning motor commands. Indeed, SNN trained with ReSuMe become efficient neurocontrollers for movement generation and control due to the ability of the method to operate online and due to its fast convergence [28].

We suppose that readout neurons trained through Remote Supervision can learn to extract the motor commands that generate intonation contours from an LSM receiving phoneme input features. In the following sections, an analysis of the LSM's separation property in response to specific phoneme input features is presented. The learning capabilities of SPAN and ReSuMe are compared and tested on speech intonation modelling. Finally, potential improvements of the model are discussed.

## 2 Separation Property

The first component of a LSM $M$ is an operator or filter $L^M$, called liquid filter or liquid circuit. The liquid state $x^M(t)$ is simply the current output of the liquid filter $L^M$ that maps input functions $u(\cdot)$ onto functions $x^M(t)$ [17]:

$$x^M(t) = (L^M u)(t) \tag{1}$$

The liquid state $x^M(t)$ is defined as the vector of output values at time $t$ of linear filters with exponential decay ($\tau_m = 30$ ms) applied to the spike trains emitted by the liquid neurons. It represents all information that a readout neuron could extract at time $t$ from the circuit. The input state can be defined similarly.

The Separation Property (SP) addresses the amount of separation between the trajectories of internal states of the system that are caused by two different input streams. SP is evaluated by injecting pairs of input spike trains $u(\cdot)$ and $v(\cdot)$ and recording resulting trajectories $x_u^M(t)$ and $x_v^M(t)$.

The distance $d(u, v)$ between two input states is defined as the Euclidean norm of their difference averaged over time and neurons. The state distance $D_{u,v}(t) = \parallel x_u^M(t) - x_v^M(t) \parallel$ is the Euclidean norm of the liquid state difference averaged over neurons.

In order to evaluate LSM's SP, 599 feature combinations that are observable in data are selected as input samples. The LSM is initialized once to its initial state, it is run during 800 ms with unvoiced state feature active. The selected input sample is active in the interval from 300 to 600 ms. The network is restored to its initial state and the simulation is repeated for all input samples.

Pairs of input samples are selected such that their input distance is precisely $d(u, v) = 1.21$, 2.1, 3.2 or 4.3. 406 pairs are randomly chosen for each input distance and the corresponding liquid state distances are measured. The simulations are repeated three times with the same network structure,

but different initial values of the liquid neurons membrane potential, such that the state distance for $d(u, v) = 0$ can also be measured. This experiment is repeated three times with different generated networks. A linear regression allows to extract the slop $\alpha$ and the residuals $r$ of the relation between state and input distances as well as the standard deviation $\sigma$, all of them averaged across time in the interval [300 ms, 700 ms] and experiment repetitions.

SP is often used to optimize neuronal model [8, 38, 5, 37], liquid architecture [8], liquid connectivity density ($\lambda$) [8, 5, 17] or synaptic parameters [26]. SP has even been observed from *in vitro* cultured cortical networks [3]

Whereas, in these studies, inputs are often modeled as spike trains generated from a Poisson distribution of frequency in the order of 20 Hz (max. 50 Hz) from a limited number of neurons (a few up to 81), our phoneme inputs are composed of 374 input channels (max. 15 simultaneously active) regularly spiking at 200 Hz. Due to the specificity of our inputs, SP analysis is performed in order to optimize parameters of connections from inputs to LSM (probability of connection $p_i$ and $p_{vuv}$ and synaptic weights $w_i = w_{vuv}$) and intra-reservoir connections (density of connections $\lambda$). Simulations have been performed with a LSM of dimensions $8 \times 8 \times 16$.

As the probability of connection $p_i$ increases, $\alpha$, $r$ and $\sigma$ increase (not shown). The value $p_i = 0.015$ is chosen such that $r$ and $\sigma$ remain acceptable. When $p_{vuv}$ increases $\alpha$ decreases, but $r$ and $\sigma$ are large for small values of $p_{vuv}$. The rise of liquid state activity for the very first input stimulation is a typical observation in LSM studies. It is certainly due to the short-term memory of liquid synapses that release a larger amount of current when the presynaptic neuron did not fire recently. VUV state inputs allow to start the network activity before presenting input samples and to constantly maintain a minimal network activity during the simulations. $p_{vuv} = 0.1$ is chosen in that purpose. Difference between SP experiments where LSM is stimulated by VUV state inputs or not can be observed on Figure 1. The liquid state shows a chaotic behavior when the stimulation starts, at $t = 0$ ms if VUV state inputs are connected and at $t = 300$ ms if they are not, distorting the liquid response to phoneme inputs in the latter case.

As the synaptic weights $w_i$ increase, $\alpha$ increases and below $w_i = 60$ nA both $r$ and $\sigma$ are reasonable. However, following experiments show that it is more difficult to extract information from the liquid network stimulated with such strength, then $w_i = 5$ nA is chosen. It points out that the Euclidean distance or the SP itself could be insufficient measures. Some other studies perform entropy measures or auto-correlation functions. Also, the generalization property as to be considered in addition to SP for good learning performances (see Discussion).

The effect of $\lambda$ on the SP is very interesting (Figure 2). $\lambda$ has very little effect on $\alpha$, but both $r$ and $\sigma$ show a peak around $\lambda = 1.7$. After the peak ($\lambda > 1.9$), their values remain superior to what they

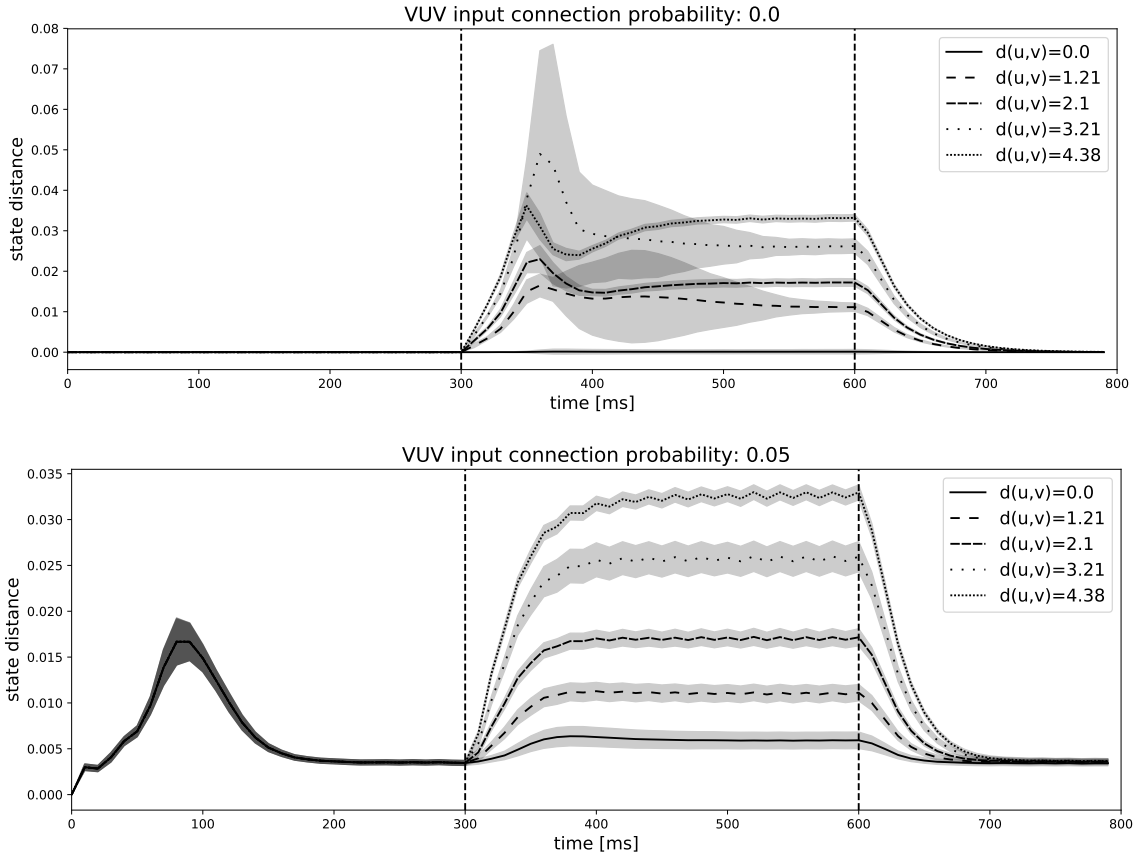


Figure 1: Liquid state distance $D_{u,v}$ as a function of time for different input distances $d(u, v)$. Liquid state distances are not proportional to input distances when VUV state inputs are not connected (top). Good proportionality is achieved with VUV state inputs connected (bottom) due to an earlier chaotic activation of the LSM (0-200 ms)

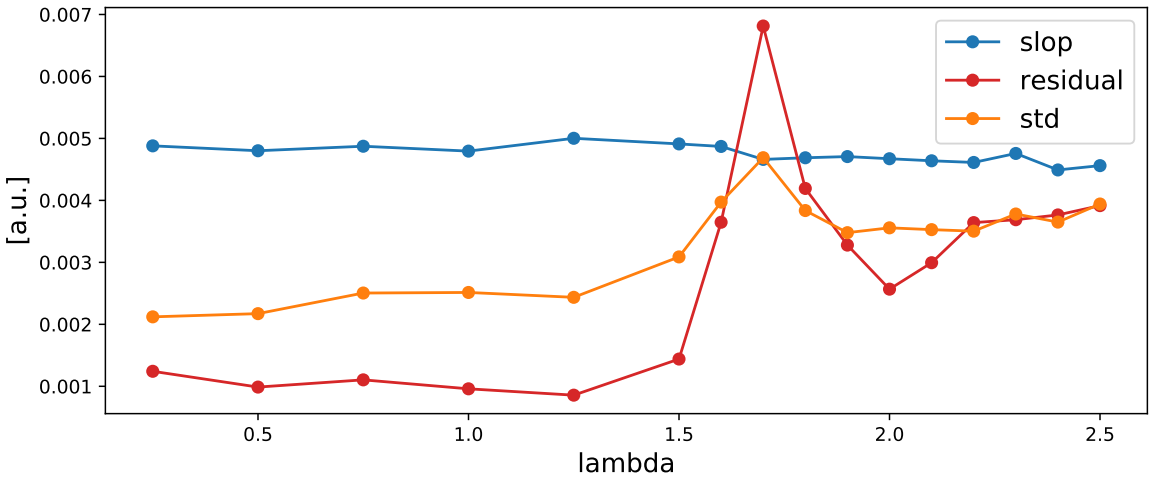


Figure 2: Slope $\alpha$ and residuals $r$ of the linear regression between liquid state distances $D_{u,v}$ and input distances $d(u, v)$, as well as standard deviation $\sigma$ averaged over the time interval [300 700] ms of SP experiment depending on the value of the LSM's density of connections $\lambda$.

were before the peak ($\lambda < 1.5$). This could be the transition from a non-chaotic regime to a chaotic regime. However no measure discriminating between non-chaotic and chaotic regimes is performed to confirm such assumption. The value $\lambda = 1.25$ is chosen because when a system parameter variation causes a transition between non-chaotic and chaotic regimes, it is often recommended that the parameter is set close to the transition point [1, 13].

## 3 Approximation property

The second component of an LSM $M$ is a memoryless readout map $f^M$ that transforms, at every time $t$, the current liquid state $x^M(t)$ into the output $y(t)$ [17]:

$$y(t) = f^M(x^M(t)) \tag{2}$$

The set of connections from the LSM network to readout neurons constitute the readout map. These connections are the only ones to be plastic, i.e. trained to reproduce sequences of spikes that generate intonation contours. Two similar learning rules, both derived from Windrow-Hoff rule, are investigated: ReSuMe [27, 28] and SPAN [21, 22]. Both learning rules belong to Remote Supervision, meaning that an instructive signal (the teacher) induces synaptic weight adaptation without contributing to the postsynaptic neuron dynamics. They are efficient supervised algorithm enabling readout neurons to reproduce the target sequences of spikes with high-precision. They are two main difference between SPAN and ReSuMe. The first one is that: while presynaptic spike trains are filtered by a kernel in both algorithms, ReSuMe uses precise spike times of postsynaptic and

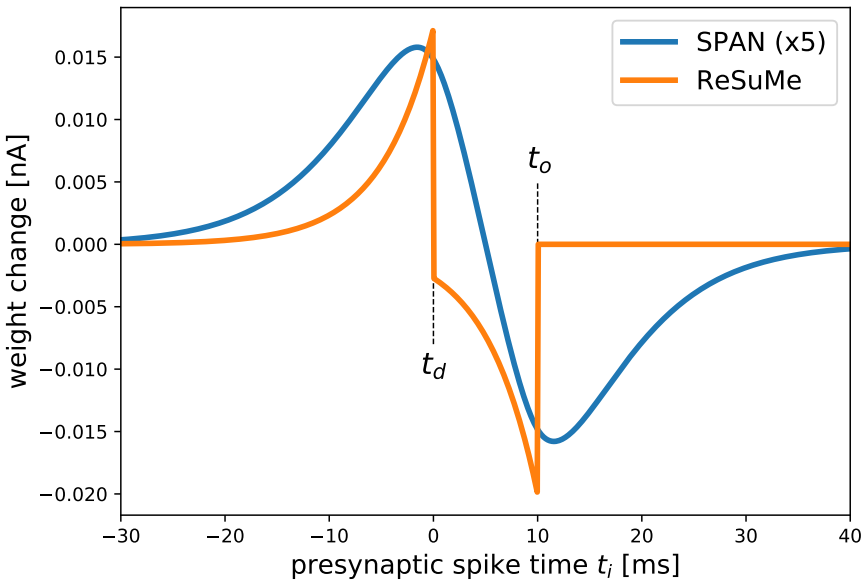


Figure 3: Weight update as a function of the presynaptic spike time for SPAN and ReSuMe learning rules. Postsynaptic spike time is $t_o = 10$ ms and teacher spike time is $t_d = 0$ ms

teacher neurons whereas SPAN also kernelizes postsynaptic and teacher spike trains. The second difference is the shape of kernels used to filter spike trains: ReSuMe applies an exponential kernel and SPAN an $\alpha$ kernel function. It has been shown that $\alpha$ kernel function achieves better results than the exponential kernel function. However, online implementation of $\alpha$ kernels implies resolving a second-order differential equation, while a a first-order one describes exponential kernels dynamics. Taking in account that postsynaptic and teacher spike trains have to be kernelized in addition, it leads to a significantly higher computational cost for learning with SPAN.

An other difference that ensued from filtered postsynaptic and teacher spike trains is that a presynaptic spike occuring after both postsynaptic and teacher spikes induce a weight change whereas this presynaptic spike does not contribute to the firing of the postsynaptic neuron at the precise teacher spike time. This allows some flexibility that could be beneficial for generalization, but it violates Hebb's principle [6]. Weight updates of both learning rules are plotted as a function of the presynaptic spike time in Figure 3.

Naively, the expected advantage of using SPAN learning rule was that $\alpha$-kernel shape is identical to muscle responses triggered by readout neurons. Thus, the time constant of SPAN kernels could have been in the order of muscle's time constant. However, long kernel shapes cannot lead to good learning performances with an online implementation of SPAN. This is due to the summation of presynaptic kernels since multiple presynaptic spikes occur within the kernel length. Better performance is obtained with short kernel's time constant ($\tau = 15$ ms is chosen), in the order of membrane potential dynamics rather than muscle dynamics, as presented in Figure 4. Thus, SPAN

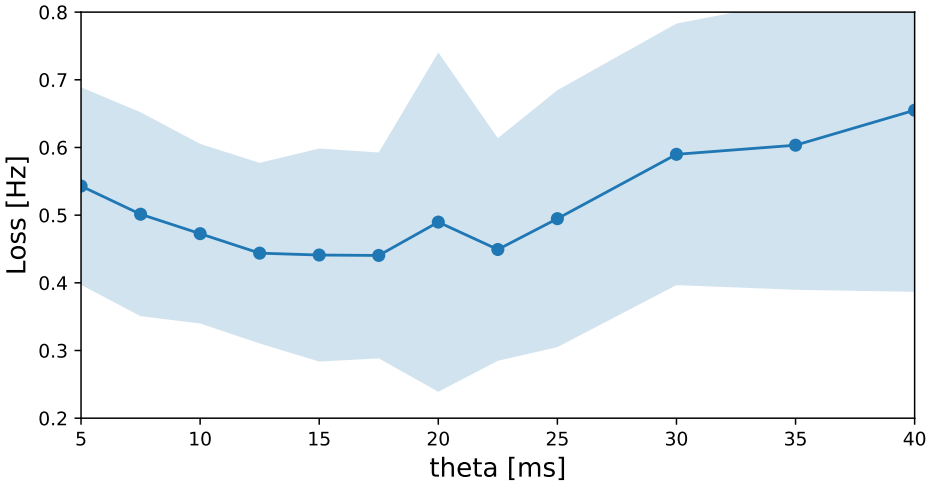


Figure 4: RMS loss between teacher and output LF0 curves as a function of SPAN time constant $\tau$

learning behavior becomes very similar to ReSuMe with a significantly higher computational cost.

ReSuMe (Figure 5) and SPAN (Figure 6) are both accurately reproducing a single sample after learning (ReSuMe performs slightly better than SPAN). However, none of them converges to the exact solution. As a consequence, synaptic weights magnitude continues to increase after the best performance is reached. From these results, it appears that the learning task is very complex to solve, otherwise the optimal solution would have been reached easily. It has been supposed that the difficulty could come from the fact that readout neurons have to be repeated several times in order to approximate the amplitude of motor commands. Then, multiple readout neurons have been trained to trigger the dynamic of the same muscle, such that a motor command represents maximum two spikes per readout neuron. This alternative encoding scheme did not lead to any performance improvement.

Figure 7 shows that the model does not generalize, i.e. training on multiple samples does not allow to predict unseen samples. The training error increases significantly with the number of learned samples, confirming that the learning task is very complex. The testing error remains large independently of the number of learned samples. This generalization failure is discussed in the following section. It has been supposed that the size of the neural network could improve performance, it effectively slightly decreases the training error, but no significant improvement for generalization.

## 4 Discussion

The Liquid State Machine allows to process continuous streams of multiple input. Readout neurons can learn to extract in an online manner from the current state of such recurrent circuit information about current and past inputs. Such online processing and extraction of information has never been

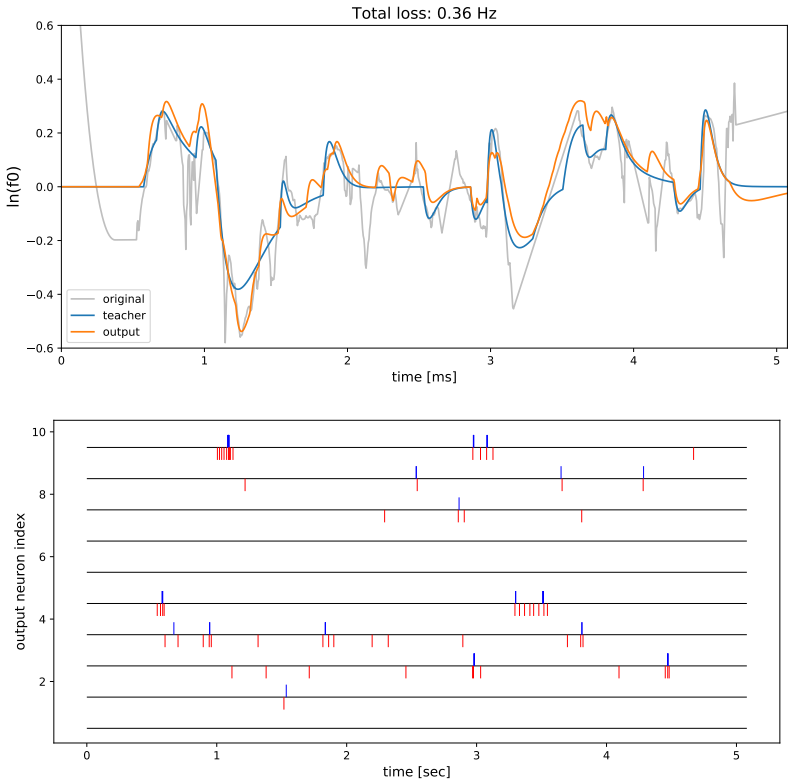


Figure 5: Top: Reproduction of LF0 teacher signal (after subtraction of a phrase component) with ReSuMe learning rule. Bottom: Reproduction of teacher spike trains (motor commands). Output spikes are red and teacher spikes blue for ten muscles

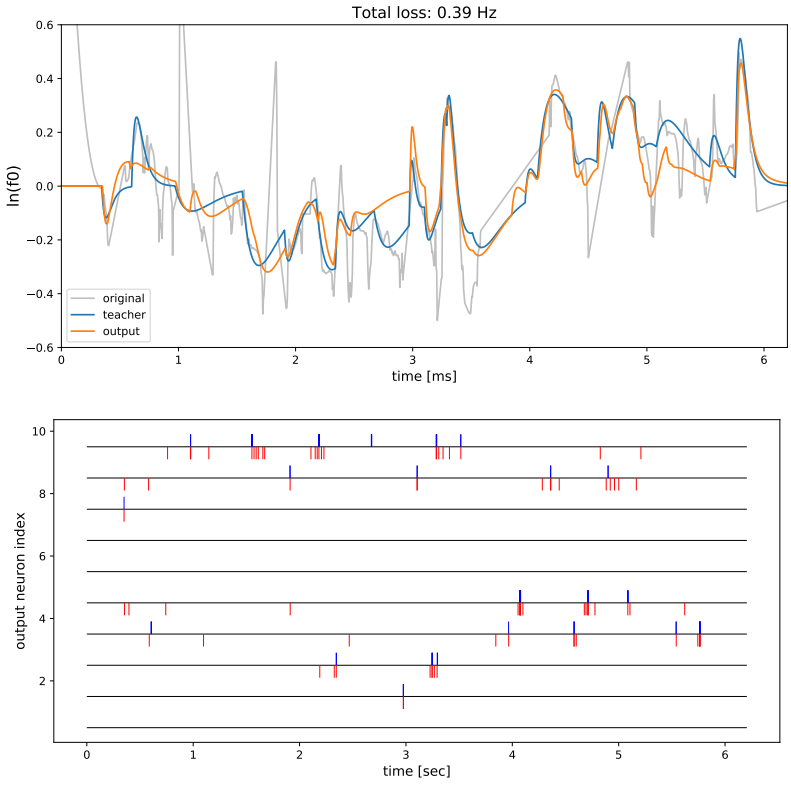


Figure 6: Same than Figure 5 on a different sample with SPAN learning rule

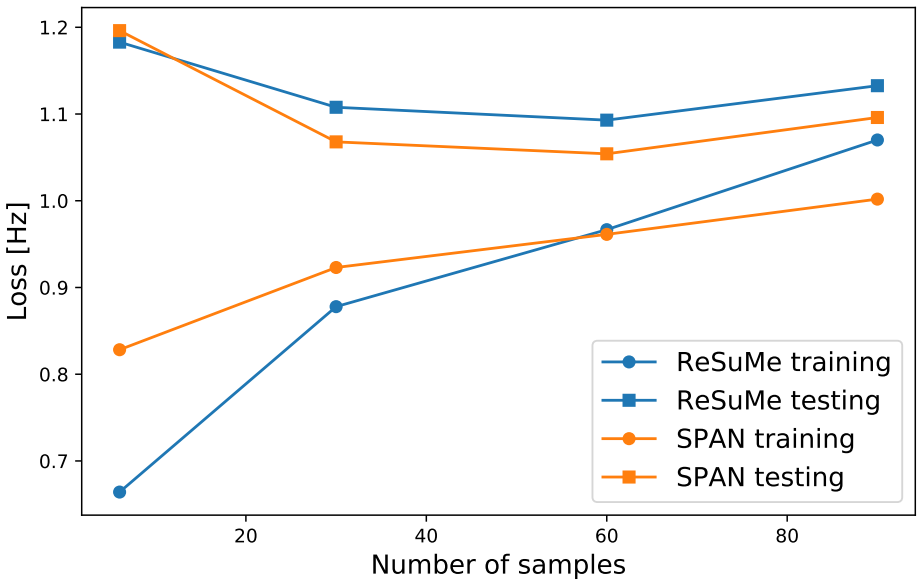


Figure 7: Prediction performance of ReSuMe and SPAN learning rules as a function of the total number of samples used for the 6-fold cross-validation

used in Text-To-Speech before. This spiking reservoir network model can learn to reproduce intonation contours of utterances with high accuracy. However, our model is not capable of generalization, meaning that the intonation's prediction of unseen utterances has not been achieved.

It has been demonstrated that spiking neurons trained with ReSuMe are able to transform inputs to the desired output spike trains for the patterns not used during the training [29]. In that sense, neurons trained with ReSuMe demonstrate the generalization property, then the learning rule is not responsible for the generalization failure. The inability of readout neurons to extract motor commands from the liquid state means that either the liquid state does not provide underlying information needed for generalization, or desired motor commands (i.e. teacher spike trains) does not represent a generalizable signal of intonation contours, or both.

**Motor commands** Motor commands, which are the spike trains triggering muscle responses could be unsuitable for generalization. Indeed, output spike trains are encoded such that several spikes (max. 16) confined in a very short time window (one spike every 2.2 ms) trigger a long muscle response (tens to hundreds of milliseconds). Then, output spike trains may be too sparse to be a generalizable signal. One solution would be to encode motor commands with a larger number of spikes distributed on a longer time period. In this way, output neurons would extract motor commands from periodic/regular activity of the liquid state rather than information available at a specific time. The periodic nature of the liquid state in response to phoneme inputs promotes such an alternative output encoding scheme. In addition, it makes sens that generalization emerges from features

of the liquid state that are periodic/permanent rather than spontaneous. This alternative encoding scheme corresponds to the approximation of a continuous muscle stimulation by a repetition of spikes on a given time period. It is supported by the original version of the Command-Response model [24] where accent commands are continuous step functions of time.

This alternative encoding scheme raises a particular problem: the pursuit algorithm used for extracting output spike times from intonation contours cannot generate such repetitions of spikes. However, the superposition of gamma kernels of order two from repeated spikes can probably approximate a gamma kernel of higher order. It would give the opportunity to extract muscle responses as higher-order gamma kernels with a matching pursuit algorithm and encoding output spike trains by the related superposition of order-2 gamma kernels for the learning task. Such relations between gamma-kernels orders should be investigated first.

**LSM generalization property** Although separation property is useful for characterizing the reservoir's network ability to separate different inputs, enhancing that property only allows to decrease the training error, in other words, to reproduce samples used for training. It would have been judicious to measure the reservoir's ability to predict unseen samples, named the generalization property. A study on prediction of computational performance for spike-based reservoir networks [13] not only measure the generalization-capability of the LSM, but present a more accurate measure of the separation property. The generalization-capability can be quantified in terms of the VC-dimension [36]. It is demonstrated and explained why computational performances are maximal at the "edge of chaos", by measuring both kernel-quality (SP) and generalization-capability.

**Feedback** Recent studies often resolve learning tasks with reservoir networks working in a different regime [11, 32, 23]. They generate reservoir networks that show spontaneous chaotic activity in absence of input stimulation. It has been shown in the original proposition of FORCE learning [32], that there are significant advantages using a network that exhibits chaotic activity prior to training: smaller number of required training periods, better performance, and smaller magnitude of the readout weight vector. However, chaotic activity during training must be avoided. The critical element for suppressing chaos is a feedback loop that carries the output back into the reservoir network. In the presence of feedback loop, output errors must be small enough to be fed back without introducing significant delayed effects that disrupt learning. To this end, synaptic modifications must be strong and rapid during the initial phases of training. As a result, the goal of training is not to significantly reduce error, but rather reducing the amount of modification needed to keep the errors small. ReSuMe appears ideal for such training paradigm due to its fast convergence. In particular, it

appeared that the parameter $a_d$ enables rapid synaptic changes allowing for low-error training. That parameter could be reduced as training progresses.

In a first attempt on computing with feedback, it would be pertinent to start with a rate-based reservoir network model, the Echo-State Network (ESN). This would simplify network dynamics while a spike-based neuron model can still be used for readout neurons. Then, a study of a FORCE trained reservoir [23] provides a deep analysis on spike-based reservoir networks with feedback.

The fact that feedback would help for better performance and in particular generalization is based on the following intuition: it is very difficult for a human to learn how to speak without hearing its own voice. However, cited studies which used feedback with a reservoir network have reproduced complex signals, but none of them performed a prediction task.

# 5 Methods

## 5.1 Inputs

Input data are from the speech database released for the Blizzard Challenge [12] on a subset (carroll, arctic, theherald 1,2,3) of the native English Voice A (Roger) of about $17.5$ minutes (169 utterances). 374 binary features are extracted from it (see Experimental Setup in previous work [31]) which represent past, current and following phonemes. These features are converted into spike trains by setting a spike generator for each feature. A spike is emitted at every frame (5 ms) when the corresponding feature is equal to one. A spike generator has a probability $p_i$ to be connected with each neuron of the LSM and $w_i = 5$ nA is the amount of current transmitted by a spike to the postsynaptic neuron.

The voiced/unvoiced flag is also taken as an input signal represented by a couple of spike generators that emit spikes regularly at $200$ Hz either during voiced or unvoiced periods. $p_{vuv}$ is the probability of connection to liquid neurons and $w_{vuv} = 5$ nA their synaptic weights.

The synaptic delays of inputs are randomly generated from a uniform distribution in the interval [0 ms, 5 ms].

## 5.2 Liquid State Machine

The LSM's neural components and topology are identical to its original description [17] except for the background current $I_b$. The network is a column in space of $N = 1024$ Integrate-and-Fire neurons, 20% of which is chosen to be inhibitory. Some parameters depends on whether the neuron is excitatory (E) or inhibitory (I).

**Neuronal parameters** The membrane potential dynamic is driven by the equation :

$$\tau_m \dot{v}(t) = -(v(t) - v_{rest}) + R_m(I_e(t) + I_i(t) + I_b) \tag{3}$$

where the membrane time constant $\tau_m = 30$ ms, the resting potential $v_{rest} = 0$ mV, the membrane resistance $R_m = 1$ MΩ and the background current $I_b$ is randomly chosen from uniform distribution between 13.5 and 14.5 nA. The postsynaptic currents coming from excitatory $I_e(t)$ and inhibitory $I_i(t)$ presynaptic neurons are modeled as exponential decay with respect to time constant $\tau_e = 3$ ms and $\tau_i = 6$ ms:

$$\tau_{e,i} \dot{I}_{e,i}(t) = -I_{e,i}(t) \tag{4}$$

When the membrane potential crosses the threshold potential $v_{th} = 15$ mV, the neuron emits a spike. Its membrane potential is reset to a reset potential $v_{reset} = 13.5$ mV and is insensitive to incoming currents ($v = v_{reset}$) during a refractory period $t_{ref}$ of 3 ms (E) or 2 ms (I).

**Connectivity structure** The probability of connection from neuron $a$ to neuron $b$ is defined as $Ce^{-(D(a,b)/\lambda)^2}$, where $\lambda$ is a parameter that controls both the average number of connections and the average distance between neurons that are synaptically connected. $N$ neurons are located on the integer points of a $8 \times 8 \times 16$ column in space, where $D(a, b)$ is the Euclidean distance between neurons $a$ and $b$. The value of $C$ is 0.3 (EE), 0.2 (EI), 0.4 (IE), 0.1 (II).

**Synaptic connections** The synaptic dynamics are driven by Tsodyks model [16] (which corrects a small error in [19]). The dynamics of the usage of releasable neurotransmitter per single action potential $u$ and the fraction of synaptic neurotransmitter resources available $x$ have been adapted from discrete time formulation to differential formulation as it follows :

$$F\dot{u} = -u \quad \text{and} \quad D\dot{x} = 1 - x \tag{5}$$

with following actions triggered by presynaptic spikes:

$$r = u \cdot x, \quad x \mathrel{-}= r \quad \text{and} \quad u \mathrel{+}= U(1 - u) \tag{6}$$

The synaptic parameters $U$ (use), $D$ (time constant for depression) and $F$ (time constant for facilitation) are randomly chosen from Gaussian distribution. The mean values of these three parameters (with $D$ and $F$ expressed in second) are .5, 1.1, .05 (EE), .05, .125, 1.2 (EI), .25, .7, .02 (IE), .32, .144 and .06 (II). The standard deviation of each parameter is 50% of its mean value (with nega-

tive values replaced by values chosen from an appropriate uniform distribution). The mean of the scaling parameter A (in nA) is 30 (EE), 60 (EI), -19 (IE) and -19 (II). The standard deviation of the paramater A is 100% of its mean and is drawn from a Gamma distribution. The transmission delays between liquid neurons are 1.5 ms (EE) and 0.8 ms for other connections. For each simulation, the membrane voltage at $t = 0$ is randomly chosen from a uniform distribution from interval [13.5 mV, 15 mV].

### 5.3 Learning rules

Synaptic weights of both learning rules are initialized from a Gaussian distribution of mean 0 nA and width 0.01 nA and learning sessions are always 100 repetitions of the training set.

**ReSuMe** ReSuMe learning rule is derived from Windrow-Hoff algorithm proposed for rate based neuron models. It is adapted to spiking neuron model using the interaction between two spike-timing dependent plasticity processes [28]. The final formulation describes the adaptation of synaptic weight between presynaptic neuron $i$ and postsynaptic neuron $o$ in real-time :

$$\dot{w}_{oi}(t) = [S_d(t) - S_o(t)] \left[ a_d + \int_0^\infty a_{di}(s) S_i(t-s) ds \right] \tag{7}$$

where $S_i(t)$, $S_o(t)$ and $S_d(t)$ are respectively presynaptic, postsynaptic and teacher spike trains. The role of the non-correlative factor (non-Hebbian term) $a_d$ is to adjust the average strength of the synaptic inputs so to impose on a neuron a desired level of activity such that the actual mean firing rate of $S_o(t)$ approaches the mean firing rate of signal $S_d(t)$. Thus, the task of setting up the precise timing of spikes, is attributed mainly to the Hebbian term $a_{id}(s)$. The kernel $a_{id}(s)$ defines the shape of a learning window:

$$a_{di}(s) = A_{di} \exp(-\frac{s}{\tau_{di}}) \tag{8}$$

The time constant $\tau_{di} = 5$ ms, the "learning rate" $A_{di} = 0.01$ nA and the parameter $a_d = 0$ nA are set for all simulations.

**SPAN** SPAN algorithm has been adapted from batch learning [21, 22] to online learning by writing $\alpha$ kernel in its differential form. $\alpha$-kernel is defined as :

$$x(t) = A \frac{t}{\tau} e^{-\frac{t}{\tau}} H(t) \tag{9}$$

where $A$ is the amplitude, $\tau$ the time constant and $H(t)$ the Heaviside function. It corresponds to the time course of a critically damped oscillator which differential equation can be written:

$$\ddot{x} + \frac{2}{\tau}\dot{x} + \frac{1}{\tau^2}x = 0 \quad \text{with} \quad x(0) = 0 \quad \text{and} \quad \dot{x}(0) = A \tag{10}$$

When $A = \tau^{-1}$, $x(t)$ is normed on $\mathbb{R}^+$ ($\int_0^\infty x(t)dt = 1$) and describes the Gamma distribution of shape 2 and scale $\tau$.

SPAN implementation implies that teacher, pre- or postsynaptic spike $f$ triggers $\alpha$-kernel $x_d(t-t_d^f)$, $x_i(t-t_i^f)$ or $x_o(t-t_o^f)$ respectively. The action $\dot{x_j}(t_j^f)$ += $A$ for $j = d, i, o$ induces such triggering process. Filtered version of spike train $S_j(t)$ becomes $\widetilde{S}_j(t) = \sum_f x_j(t-t_j^f)$. Synaptic weight $w_{oi}$ is updated every millisecond with respect to the learning rate $\eta$ :

$$\dot{w}_{oi}(t) = \eta\widetilde{S}_i(t)(\widetilde{S}_d(t)) - \widetilde{S}_o(t)) \tag{11}$$

The learning rate $\eta = 0.5$ is set for all simulations.

### 5.4 Readout neurons

Readout neurons are Integrate-and-Fire neurons driven by equation (3). Neuronal parameters are: time constant $\tau_m = 10$ ms, resting potential $v_{rest} = -60$ mV, threshold potential $v_{th} = -55$ mV, reset potential $v_{reset} = -65$ mV, membrane resistance $R_m = 10$ MΩ, synaptic current time constants $\tau_e = \tau_i = 1$ ms, background current $I_b = 1$ nA and refractory period $t_{ref} = 0$ ms. Membrane potential $v(t = 0)$ are initialized from uniform distribution in the interval [-62, -60] mV.

### 5.5 Muscles

Muscles dynamics are modelled by the Gamma distribution $x(t) = \frac{t}{\tau^2}e^{-\frac{t}{\tau}}H(t)$. It is identical to an $\alpha$-kernel with amplitude $A = \tau^{-1}$. The same differential equation (10) described in the precedent section is used for implementation. Five pairs of muscles are used for simulations. Within a pair, one muscle describes negative dynamics and the other describes positive dynamics. The time constant $\tau$ is the same within a pair and different for each pair: $\tau = [10, 20, 37, 42, 145]$ ms. A set of muscle responses (atoms) is extracted from intonation contours by a pursuit algorithm. A muscle response (of given length $\tau$) can be fully described by the precise timing and the amplitude of the command spike. Due to the stereotypical amplitude of neural spikes, the amplitude of the command spike is approximated by a repetition of neural spikes. Neural spikes are repeated every 2.2 ms, with each one corresponding to an amplitude of 0.3.

## Aknowledge

The present work has been done during an internship at Idiap Research Institute of Martigny from November 2018 to April 2019 under the supervision of Dr. P.N.Garner. It has been built upon previous work from B.Schnell and F.Marelli.

## References

[1] Nils Bertschinger and Thomas Natschläger. Real-time computation at the edge of chaos in recurrent neural networks. *Neural computation*, 16(7):1413–1436, 2004.

[2] Sander M Bohte, Joost N Kok, and Han La Poutre. Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing*, 48(1-4):17–37, 2002.

[3] Karl P Dockendorf, Il Park, Ping He, José C Príncipe, and Thomas B DeMarse. Liquid state machines and cultured cortical networks: The separation property. *Biosystems*, 95(2):90–97, 2009.

[4] Hiroya Fujisaki, Sumio Ohno, and Changfu Wang. A command-response model for f0 contour generation in multilingual speech synthesis. In *The Third ESCA/COCOSDA Workshop (ETRW) on Speech Synthesis*, 1998.

[5] Beata J Grzyb, Eris Chinellato, Grzegorz M Wojcik, and Wieslaw A Kaminski. Which model to use for the liquid state machine? In *2009 International Joint Conference on Neural Networks*, pages 1018–1024. IEEE, 2009.

[6] Donald O Hebb. The organization of behavior; a neuropsycholocigal theory. *A Wiley Book in Clinical Psychology.*, pages 62–78, 1949.

[7] Pierre-Edouard Honnet, Branislav Gerazov, and Philip N Garner. Atom decomposition-based intonation modelling. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4744–4748. IEEE, 2015.

[8] Emmanouil Hourdakis and Panos Trahanias. Use of the separation property to derive liquid state machines with enhanced classification performance. *Neurocomputing*, 107:40–48, 2013.

[9] Dongsung Huh and Terrence J Sejnowski. Gradient descent for spiking neural networks. In *Advances in Neural Information Processing Systems*, pages 1440–1450, 2018.

[10] Narumitsu Ikeda, Yoshinao Sato, and Hirokazu Takahashi. Short utterance speaker recognition by reservoir with self-organized mapping. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 1073–1077. IEEE, 2018.

[11] Herbert Jaeger and Harald Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *science*, 304(5667):78–80, 2004.

[12] Vasilis Karaiskos, Simon King, Robert AJ Clark, and Catherine Mayo. The blizzard challenge 2008. In *Proc. Blizzard Challenge Workshop, Brisbane, Australia*, 2008.

[13] Robert Legenstein and Wolfgang Maass. Edge of chaos and prediction of computational performance for neural circuit models. *Neural Networks*, 20(3):323–334, 2007.

[14] Mantas Lukoševičius, Herbert Jaeger, and Benjamin Schrauwen. Reservoir computing trends. *KI-Künstliche Intelligenz*, 26(4):365–371, 2012.

[15] Wolfgang Maass. Liquid state machines: motivation, theory, and applications. In *Computability in context: computation and logic in the real world*, pages 275–296. World Scientific, 2011.

[16] Wolfgang Maass and Henry Markram. Synapses as dynamic memory buffers. *Neural Networks*, 15(2):155–161, 2002.

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

[18] Stéphane G Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on signal processing*, 41(12):3397–3415, 1993.

[19] Henry Markram, Yun Wang, and Misha Tsodyks. Differential signaling via the same axon of neocortical pyramidal neurons. *Proceedings of the National Academy of Sciences*, 95(9):5323–5328, 1998.

[20] Raoul-Martin Memmesheimer, Ran Rubin, Bence P Ölveczky, and Haim Sompolinsky. Learning precisely timed spikes. *Neuron*, 82(4):925–938, 2014.

[21] Ammar Mohemmed, Stefan Schliebs, Satoshi Matsuda, and Nikola Kasabov. Span: Spike pattern association neuron for learning spatio-temporal spike patterns. *International journal of neural systems*, 22(04):1250012, 2012.

[22] Ammar Mohemmed, Stefan Schliebs, Satoshi Matsuda, and Nikola Kasabov. Training spiking neural networks to associate spatio-temporal input–output spike patterns. *Neurocomputing*, 107:3–10, 2013.

[23] Wilten Nicola and Claudia Clopath. Supervised learning in spiking neural networks with force training. *arXiv preprint arXiv:1609.02545*, 2016.

[24] Sumio Ohno, Hiroya Fujisaki, and Yoshikazu Hara. On the effects of speech rate upon parameters of the command-response model for the fundamental frequency contours of speech. In *Fifth International Conference on Spoken Language Processing*, 1998.

[25] Jean-Pascal Pfister, Taro Toyoizumi, David Barber, and Wulfram Gerstner. Optimal spike-timing-dependent plasticity for precise action potential firing in supervised learning. *Neural computation*, 18(6):1318–1348, 2006.

[26] Gordon Pipa and Robin Cao. Extended liquid computing in networks of spiking neurons. 2010.

[27] Filip Ponulak. Supervised learning in spiking neural networks with resume method. *Phd, Poznan University of Technology*, 46:47, 2006.

[28] Filip Ponulak and Andrzej Kasiński. Supervised learning in spiking neural networks with resume: sequence learning, classification, and spike shifting. *Neural computation*, 22(2):467–510, 2010.

[29] Filip Ponulak and Andrzej J Kasinski. Generalization properties of spiking neurons trained with resume method. In *ESANN*, pages 629–634. Citeseer, 2006.

[30] Ryan Pyle and Robert Rosenbaum. A model of reward-modulated motor learning with parallelcortical and basal ganglia pathways. *arXiv preprint arXiv:1803.03304*, 2018.

[31] Bastian Schnell and Philip N Garner. A neural model to predict parameters for a generalized command response model of intonation. *Proc. Interspeech 2018*, pages 3147–3151, 2018.

[32] David Sussillo and Larry F Abbott. Generating coherent patterns of activity from chaotic neural networks. *Neuron*, 63(4):544–557, 2009.

[33] Gouhei Tanaka, Toshiyuki Yamane, Jean Benoit Héroux, Ryosho Nakane, Naoki Kanazawa, Seiji Takeda, Hidetoshi Numata, Daiju Nakano, and Akira Hirose. Recent advances in physical reservoir computing: a review. *Neural Networks*, 2019.

[34] Fabian Triefenbach, Kris Demuynck, and Jean-Pierre Martens. Large vocabulary continuous speech recognition with reservoir-based acoustic models. *IEEE Signal Processing Letters*, 21(3):311–315, 2014.

[35] Fabian Triefenbach, Azarakhsh Jalalvand, Benjamin Schrauwen, and Jean-Pierre Martens. Phoneme recognition with large hierarchical reservoirs. In *Advances in neural information processing systems*, pages 2307–2315, 2010.

[36] V Vapnik. N.(1998) statistical learning theory.

[37] Grzegorz M Wojcik. Electrical parameters influence on the dynamics of the hodgkin–huxley liquid state machine. *Neurocomputing*, 79:68–74, 2012.

[38] Grzegorz M Wojcik and Wieslaw A Kaminski. Liquid state machine and its separation ability as function of electrical parameters of cell. *Neurocomputing*, 70(13-15):2593–2597, 2007.

[39] Yong Zhang, Peng Li, Yingyezhe Jin, and Yoonsuck Choe. A digital liquid state machine with biologically inspired learning and its application to speech recognition. *IEEE transactions on neural networks and learning systems*, 26(11):2635–2649, 2015.

[40] Ziyue Zhao, Huijun Liu, and Tim Fingscheidt. Nonlinear prediction of speech by echo state networks. In *2018 26th European Signal Processing Conference (EUSIPCO)*, pages 2085–2089. IEEE, 2018.