

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/221908982>

Neuro-Inspired Speech Recognition Based on Reservoir Computing

Chapter · August 2010

DOI: 10.5772/10186 · Source: InTech

CITATIONS

8

READS

148

1 author:



Arfan Ghani

Coventry University

33 PUBLICATIONS 184 CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:



Digital steganography [View project](#)



Ultra low power sub/near-threshold CMOS based implantable wireless medical device and interface with electrode arrays [View project](#)

Neuro-Inspired Speech Recognition Based on Reservoir Computing

A Ghani, T.M. McGinnity, L Maguire, L McDaid and A Belatreche
University of Ulster
N. Ireland, United Kingdom

1. Introduction

This chapter investigates the potential of recurrent spiking neurons for classification problems. It presents a hybrid approach based on the paradigm of *Reservoir Computing*. The practical applications based on recurrent spiking neurons are limited due to the lack of learning algorithms. Most of the previous work in the literature has focused on feed forward networks because computation in these networks is comparatively easy to analyse. The details of such networks have been reported in detail in (Haykin, 1999) (Pavlidis et al., 2005) (Bohte et al., 2000). Recently, a strategy proposed by Maass (Maass et al., 2002) and Jaeger (Jaeger, 2001) offers to overcome the burden of recurrent neural networks training. In this paradigm, instead of training the whole recurrent network only the output layer (known as readout neuron) is trained.

This chapter investigates the potential of recurrent spiking neurons as the basic building blocks for the liquid or so called *reservoir*. These recurrent neural networks are termed as microcircuits which are viewed as basic computational units in cortical computation (Maass et al., 2002). These microcircuits are connected as columns which are linked with other neighboring columns in cortical areas. These columns read out information from each other and serve both as reservoir and readout. The reservoir is modeled as a dynamical system perturbed by the input stream where only readouts are trained to extract information from the reservoir. The basic motivation behind investigating recurrent neurons is their potential to memorise relevant events over short periods of time (Maass et al., 2002). The use of feedback enables recurrent networks to acquire state representation which makes them suitable for temporal based applications such as speech recognition. It is challenging to solve such problems with recurrent networks due to the burden of training. The paradigm of *reservoir computing* also referred to as *liquid computing* relaxes the burden of training because only an output layer is trained instead of training the whole network. The work presented in this chapter analyses the theoretical framework of *Reservoir Computing* and demonstrates results in terms of classification accuracy through the application of speech recognition. The design space for this paradigm is split into three domains; front end, reservoir, and back end. This work contributes to the identification of suitable front and back end processing techniques along with stable reservoir dynamics, which provides a reliable framework for classification related problems.

The work presented in this chapter suggests a simple and efficient biologically plausible approach based on a hybrid implementation of recurrent spiking neurons and classical feed

forward networks for an application of isolated digit recognition. The structure of this chapter is as follows: section 2 elaborates the motivation, related work, theoretical review and description of the paradigm of reservoir computing. Section 3 contains details about the experimental setup and investigates front-end pre-processing techniques and reservoir dynamics. A baseline feed forward classifier is described in section 4 and results are presented. Results based on reservoir recognition are presented in section 5. Section 6 discusses results obtained through Poisson spike encoding. A thorough discussion and conclusion of the chapter is provided in section 7.

2. Computing with recurrent neurons

The paradigm of reservoir or liquid computing is promising because it offers an alternative to the computational power of recurrent neural networks, however analytical study of such networks is not trivial (Legenstein et al., 2003) (Joshi & Maass, 2005) (Jaeger & Haas, 2004). It facilitates training in a recurrent neural network where a linearly non separable low dimensional data is projected on a high dimensional space. The readout of the reservoir can be trained with partial information extracted from the reservoir which suffices to solve complex problems such as speech recognition. In this approach, readout only observes the membrane potential of the spiking neurons at particular time steps which is far more efficient than fully quantifying the reservoir dynamics. It is due to this property that relatively simple readout can be trained with meaningful internal dynamics of the reservoir. The framework of reservoir computing is more suitable for hardware implementation because network connections remains fixed in the network and there is no need to implement weight adaptation for recurrent reservoirs. This paradigm is inherently noise robust therefore more suitable for digital hardware implementation on reconfigurable platforms such as FPGAs. FPGA implementation of recurrent spiking neurons gives the flexibility to develop such networks with simple real-world interface and offers other desirable features such as noise robustness. This paradigm appears to have great potential for engineering applications such as robotics, speech recognition, and wireless communication (Joshi & Maass, 2004) due to the computationally inexpensive training of readout neurons. This paradigm can also be used for channel equalization of high speed data streams in wireless communication as suggested by (Jaeger & Haas, 2004).

Since the inception of the theoretical foundation by Jaeger and Maass, various groups have focused on investigating different aspects of the paradigm for engineering applications e.g., Skowronski et al., investigated the paradigm of echo state networks for speech recognition applications where the HFCC (Human Factor Cepstral Coefficient) technique was investigated for front end processing and HMM (Hidden Markov Model) classifier was used for back end processing. The overall performance was compared with the baseline HMM classifier. The main focus of the work was to investigate the noise robustness of the system based on echo state networks (Skowronski & Harris, 2007). Verstraeten et al., analysed the classification accuracy of a reservoir with different benchmarks. In their study, both sigmoidal and LIF (Leaky Integrate-and-Fire) based reservoirs were tested for evaluating the memory capacity and overall classification accuracy was calculated based on different sizes of the reservoir. The memory capacity was analysed by evaluating the maximum number of patterns that could be stored for short period of time and memory is analysed by different circuit connections in the reservoir. Moreover, different speech pre processing techniques were also elaborated and their robustness is measured against overall system performance (Verstraeten et al., 2007).

In previous studies, different solutions have been proposed in order to improve the reservoir dynamics to get better accuracies. Maass and Jaeger stated that it is possible to obtain a stable reservoir if topology and weights are drawn randomly (Maass et al., 2002). Jaeger emphasised to control the scaling of the weights while Maass emphasised that stable dynamics can be obtained with proper connection topologies. The objective in both cases was to ensure the property of *fading memory* or *echo state* in the network. It appeared from these studies that the paradigm does not depend on a specific connectivity of a reservoir rather more on a distributed, stable and redundant representation of the neurons. In a recent study, Ismail Uysal investigated a noise robust technique by using phase synchrony coding (Uysal et al., 2007). However, in all of these studies, there are no specific guidelines regarding implementation and stability of reservoirs and all reported techniques that significantly vary from each other based on the authors' experiences of reservoir computing. Implementing stable reservoir is a challenging task, however the stability of the reservoir is not the only criteria which will guarantee a solution to the problems at hand. Two important factors are the proper investigation of front and back end techniques. Regardless of the size of the reservoir or processing nodes, it is rather difficult to solve a problem without investigating a robust front end technique. This work will investigate three main areas: a robust front end, a stable and compact reservoir, and an efficient back end engine for the task of recognition. The overall accuracy of the reservoir based classification technique will be compared with the baseline feedforward network.

2.1 Theoretical background

In contrast to feedforward networks, where inputs are propagated to the output layer in a feedforward manner, feedback loops are built into the design of recurrent networks in order to incorporate dynamical properties. One of the simple and well known architecture was introduced by John J Hopfield (Hopfield, 1982) and Elman in 1990 (Elman, 1990). Other modified architectures based on recurrent neural networks (RNN) have been proposed by Jordan (Jordan, 1996) and Bengio (Bengio, 1996) (see Fig. 2). The Hopfield networks consists of a set of neurons and corresponding unit delays with no hidden units as illustrated in Fig. 1. In this network, the total numbers of feedback loops are equal to the total number of neurons where the output of each neuron is fed back to each of the other neurons in the network with no self feedback loop (Haykin, 1999). Hopfield neural networks are promising but they require large number of neurons compared to the number of classes and take considerably more time to compute compared to feedforward networks (Looney, 1997). Elman network commonly is a two layer network where output from the first layer is fed back to the input of the same layer. A short term memory can be implemented by including delay in the connection which stores values from the previous time step that can be used in the current time step. Because of this short term memory capability, Elman networks can be trained to respond to the spatio-temporal patterns. Jordan type artificial neural networks are recurrent networks with delayed loopback connections between a context output and input layer. The context layer allows the network to produce different values with the same input based on the history. In Bengio networks, the network response is fed back to the input through a context layer and a delayed output from the previous time steps.

Recurrent neural networks have also been investigated by other researchers such as (Doya, 1995) (Atiya, 2000) and (Pearlmutter, 1995). Recurrent neural networks can be represented as a Mealy state machine which receives inputs and produces outputs that are dependent both on its internal state and the input (see Fig. 3).

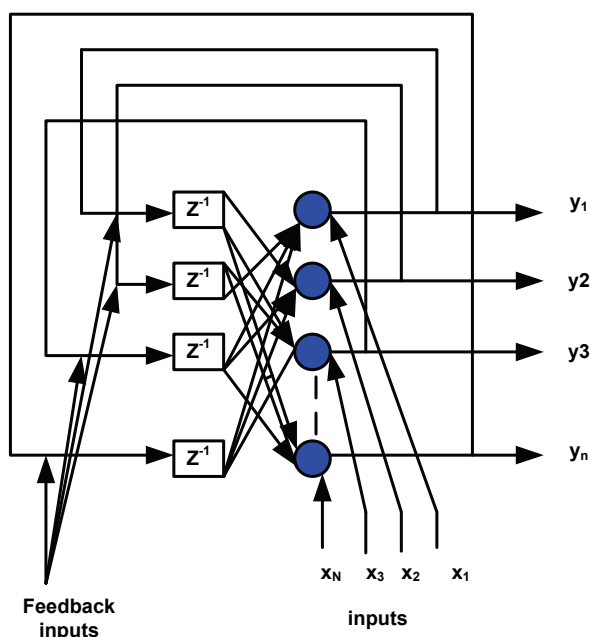


Fig. 1. This figure shows Hopfield network model which consists of a set of neurons and corresponding set of unit delays. This makes this model a recurrent multiple loop feedback system.

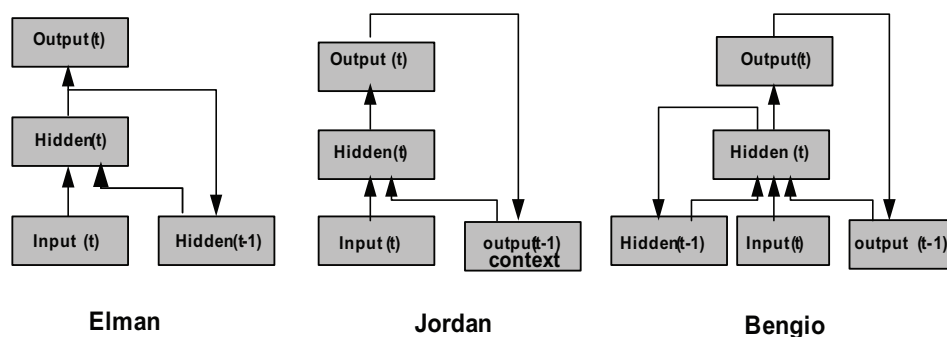


Fig. 2. Recurrent neural networks

A major advantage of recurrent neural networks such as Elman, Bengio, and Jordan is their capability to store information for a short period of time. In feedforward networks, memory can be augmented with tapped delay lines while recurrent networks are provided with build-in memory by recurrent loops (Haykin, 1999). There is no straight forward way to construct a recurrent neural network which will work as a finite state machine, therefore RNNs have to be trained to simulate a specific problem. There has been limited success in this regard and there is no standard algorithm for training RNNs (Jaeger, 2001). There have

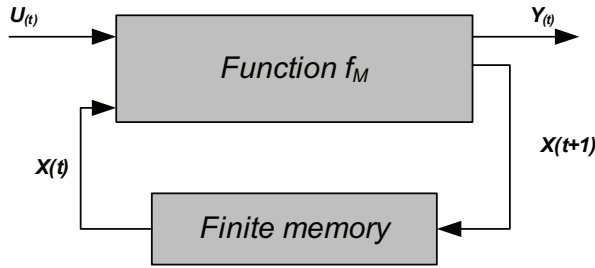


Fig. 3. Mealy type finite state machine where output $Y(t)$ is dependent on the input $U(t)$, internal state, $X(t)$ and the function f^M .

been some attempts to develop learning algorithms for recurrent networks but they are computationally much more expensive and non-trivial to converge (Haykin, 1991) (Jaeger, 2002) (Williams & Zipser, 1989) (Singhal & Wu, 1989) (Atiya et al., 2000). Some of these algorithms are based on approximating the gradient, others are based on approaches such as extended Kalman filter, EM (expectation-maximisation) based algorithms and novel architectures such as focused backpropagation and the approximated Levenberg-Marquardt algorithm. A detailed discussion about training of recurrent neural networks is provided by (Atiya et al., 2000). There has been significant research in the temporal phenomena at the synapse level (Abbott & Nelson, 2000) but less emphasis on the learning dynamics at the network level (Jaeger, 2001).

Recently, in order to overcome the burden of training in recurrent networks, the paradigm of liquid computing was introduced by Maass and Jaeger. This paradigm covers three main techniques in classification related problems: Echo State Machine (Jaeger, 2001), Backpropagation Decorrelation (Steil, 2004), and Liquid State Machine (Maass et al., 2002). The fundamental motivation behind all these techniques is to overcome the computational burden of the recurrent neural network training. In the paradigm of liquid computing, the partial response of a recurrent reservoir is observed from outside by any suitable classification algorithm such as back propagation. It is much easier and more computationally efficient to train the output layer (feedforward network) or so called 'readout' neurons, instead of the complete network of recurrent neurons. An abstract overview of the Liquid State Machine is shown in Fig. 4.

The mathematical theory of liquid computing is based on the observation that if a complex recurrent neural circuit is excited by an input stream $u(t)$ and after some time s , such that when $t > s$ a liquid state $x(t)$ is captured, then it is very likely that this state will contain most of the information about recent inputs. According to the theory, it is not possible to understand the neural code but it is not important because liquid by itself serves as short term memory and the major task of learning depends on the state vector $x(t)$ which is exclusively used by the *readout* neurons. The liquid reservoir transforms the input stream $u(t)$ to a high dimensional spatial state $x(t)$ (Maass et al., 2002). The paradigm is shown in Fig. 5.

The liquid state machine (LSM) is somewhat similar to the finite state machine (FSM) but the major difference is that LSM is viewed as a state machine with no limited states in contrast to the FSM where all transitions are custom designed and pre determined. According to the theory, if the reservoir state $x(t)$ is high dimensional and if its dynamics are sufficiently complex then many concrete finite state machines are embedded in it. Mathematically, a

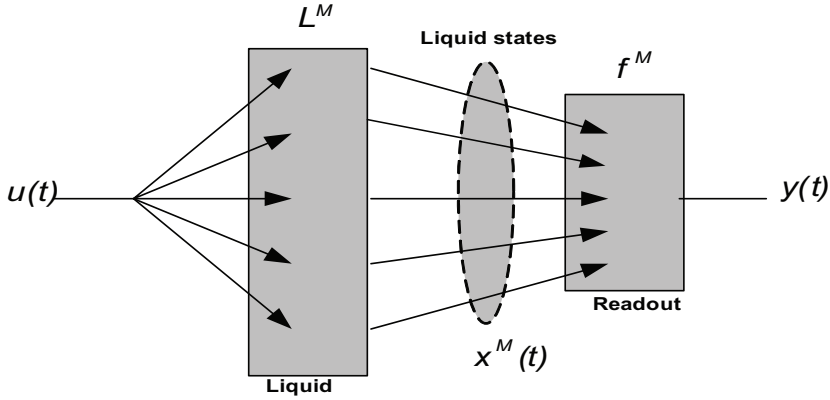


Fig. 4. An abstract overview of a liquid state machine where an input stream $u(t)$ is mapped to a target function $y(t)$. An input is injected to the liquid filter L^M and state vector $x^M(t)$ is captured at each time step t . The state vector is applied to the readout neurons through mapping f^M in order to approximate the target function $y(t)$ (figure annotated from Maass et al., 2002).

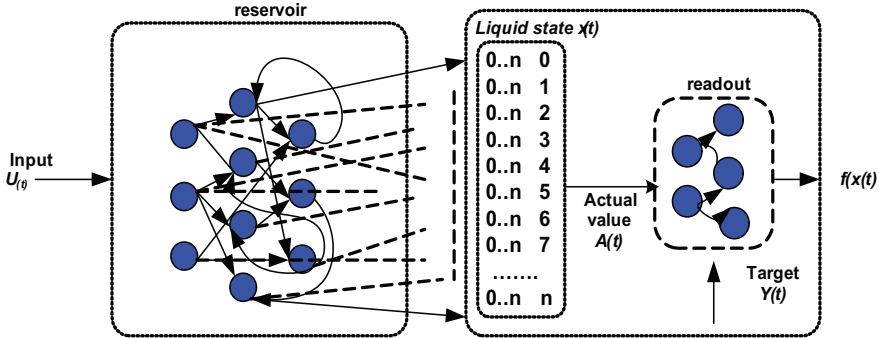


Fig. 5. An abstract overview of a neural reservoir which shows an input $U(t)$ is fed into the spiking recurrent reservoir where different states are captured in the block 'liquid state' which are used as training vectors for readout neurons.

liquid state machine M consists of a filter L^M that maps input stream $u(t)$ onto reservoir state $x(t)$, where $x(t)$ not only depends on $u(t)$ but also on previous input $u(s)$ (Maass et al., 2002). Mathematically this can be written as:

$$x(t) = (L^M u)(t) \quad (1)$$

While a readout function f^M maps the state of the liquid $x(t)$ into a target output $y(t)$.

$$y(t) = f^M(x^M(t)) \quad (2)$$

The advantage of the neural reservoir is that it does not require a task specific connectivity and it does not require any specific code by which information is represented in the neural

reservoir because only the readout neurons are trained (Maass et al., 2002). Theoretical results imply that this is a universal state machine which has no limitation on the power of neural microcircuit as long as the *reservoir* and readouts fulfill the separation and approximation properties. In order to construct a neural microcircuit, the following three steps are required:

The structure of a neural reservoir is defined in terms of processing node types (LIF, HH or Izhikevich), total number of recurrent neurons, their connectivity and parameters. Whereas, the state vector $x(t)$ of the neural reservoir is recorded at different time steps for different inputs $u(t)$. A supervised learning algorithm is applied to train a readout function f such that an actual output $f(x(t))$ is as close as possible to the target value $y(t)$. The simulation experiments performed by Jaeger and Maass showed that a simple readout would be sufficient to extract information from the recurrent neural reservoir. The major difference between ESN and LSM are the node types where sigmoid neurons are used for ESN and LIF neurons for LSM. The results are demonstrated with classification problems as reported in (Jaeger, 2001) (Maass et al., 2002) (Maass et al., 2004b). Maass et al., examined the recurrent LIF neurons in a bench-mark task proposed by (Hopfield & Brody, 2001) (Hopfield & Brody, 2000). The robustness of a neural reservoir is justified by Cover's separability theorem, which states that if a pattern classification problem is projected non-linearly on a high dimensional space, it is more likely to be linearly separable in comparison to the low dimensional space (Cover, 1965) (Skrowski et al., 2007).

It was stated by Maass and Jaeger that temporal integration can be achieved by randomly created recurrent neural reservoirs and various readouts can be trained with the same reservoir. A classification can be guaranteed by this paradigm if the dynamics of a reservoir exhibit a property of *fading memory* or *echo state property*. The concept of echo state property or fading memory is introduced because different states disappear over time. Theoretically, this paradigm appears to have no limits on the power of the neural microcircuit but there are no specific guidelines as how to construct a stable or ordered recurrent neural reservoir, an appropriate front end and classification algorithm for backend readout. In the following section, an experimental framework is proposed inspired by the paradigm of reservoir computing where the design space is split into three main areas: front end, back end and reservoir. Each one of these areas were analysed individually and then integrated for their performance evaluation (Ghani et al., 2006) (Ghani et al., 2008).

3. Experimental setup

The task of isolated spoken digit recognition is significantly complex and different techniques for solving this problem have been reported in the literature (Sivakumar et al., 2000) (Zhao, 1991) (Kim et al., 1999). In this section, a biologically plausible hybrid engine is proposed inspired by the framework of reservoir computing to solve isolated digit recognition. A rather simple feed forward multilayer perceptron is proposed and used as *readout* for classification of 10 isolated spoken digits from the TI46 speech corpus (Doddington & Schalk, 1981). The readout neurons were trained with standard back propagation algorithms and with only partial information extracted from recurrent reservoir being used in the training. It is computationally expensive to process all the states recorded from the recurrent reservoir and therefore inefficient for backend classification. In this experiment, only five states were recorded sampled at every 25 ms from start to the end of a 1 second simulation of the reservoir. A detailed design flow of the reservoir classification engine is shown in Fig. 6.

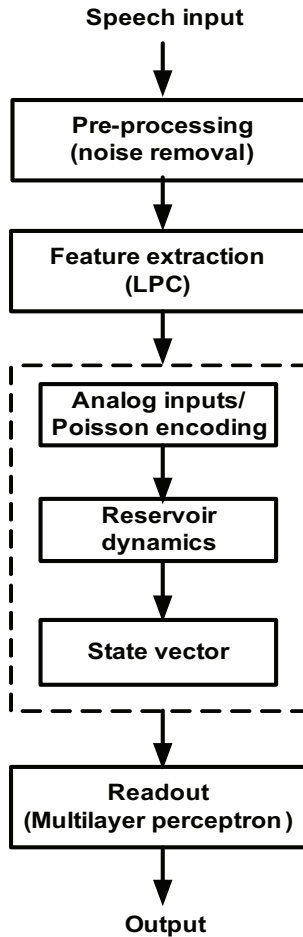


Fig. 6. This figure shows different steps involved in the investigation of the speech recognition application with a recurrent neural reservoir. The input speech samples are pre-processed and noise is removed by an end-point detection technique. Features are extracted and processed as inputs for the neural reservoir. In order to reduce the computational burden of readout, only partial information is extracted and simple feedforward neural network is used for classification.

In the following sections, the approach is investigated and analysed in three stages: feature extraction through linear predictive coding, investigation of stable neural reservoir dynamics and back end processing through simple gradient based learning. The experiment is based on the subset of isolated digit recognition (digits 0-9) dataset from the TI46 speech corpus. The dataset provides samples spoken by five different speakers with each digit being uttered four times by each speaker. This provides a total dataset of 200 speech samples (10 digits x 5 speakers x 4 utterances).

3.1 Pre-processing

In the application of speech recognition, it is very important to detect the signal in the presence of background noise in order to improve the accuracy of a system. A speech signal can be divided into three states: silence, unvoiced and voice (Luh et al., 2004). It is very important to remove the silence state in order to save the overall processing time and hence to improve the accuracy. In order to detect the silence part, an end-point detection technique is used where signal energy is calculated and a threshold value is determined. The total amount of data processing is minimised by accurately detecting the start and stop points in a sample speech signal (see Fig. 7). As shown in the figure, a spoken word 'five' is sampled at 12 KHz for 8260 samples or a duration of 0.69 seconds. Total silence time before and after voice is around 0.37 seconds or 4453 samples. By reducing this silence time, the overall signal pre processing time can be improved to 53%.

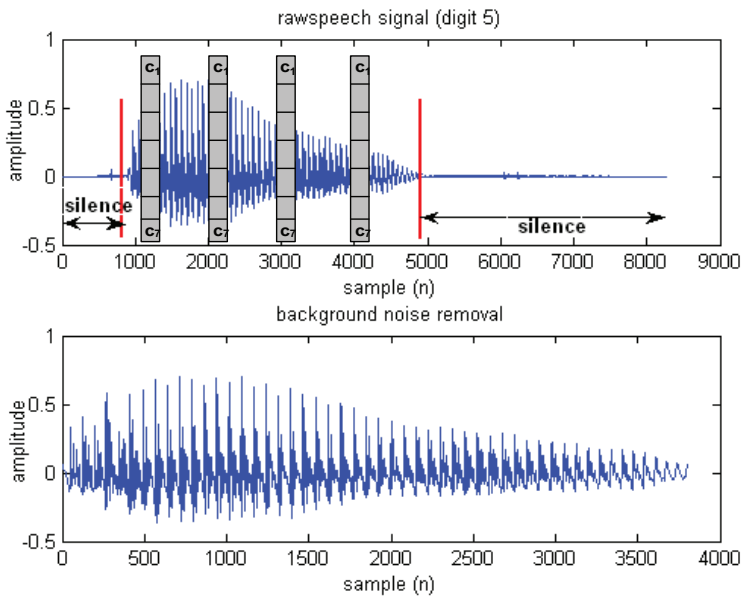


Fig. 7. This figure shows a raw speech signal sampled at 12 KHz for 8260 samples. The utterance can be divided in three clearly differentiable parts: silence, speech and then silence. The waveform has quite a significant part of silence in the beginning and the end of a signal. End-point detection technique removes the silence part from raw speech signal and only voiced portion of the signal as shown in the bottom plot is processed. By using an end point detection technique, overall processing time can be improved to 53%.

3.2 Feature extraction

Once the noise is filtered out from the speech signal, an appropriate speech coding technique is applied for feature selection. Different biologically plausible and signal processing based techniques such as frequential based MFCC (Mel Frequency Cepstral Coefficient), Lyon Passive Ear and Inner Hair Cell models have been reported in the literature and a detailed comparison is provided by (Verstraeten et al., 2005). These

techniques provide a good analysis but none of them offers an optimal solution. In this experiment, a temporal based LPC (Linear Predictive Coding) technique is applied which is one of the most useful methods for encoding a speech signal. In this method, present samples of the speech are predicted by the past p speech samples. Mathematically, this can be written as:

$$\tilde{x}(n) = a_1x(n-1) + a_2x(n-2) + \dots + a_px(n-p) \quad (3)$$

Where $\tilde{x}(n)$ is the predicted signal value, $x(n-p)$ the previous observed value and a_p the predictor coefficient. The coefficients, a_1, \dots, a_p remain constant while the objective is to estimate the next sample by linearly combining the most recent samples. Another important consideration is to minimise the mean square error between the actual sample and the estimated one. The error generated by this estimate can be calculated as:

$$e(n) = x(n) - \tilde{x}(n) \quad (4)$$

Where $e(n)$ is the calculated error and $x(n)$ is the true signal value.

Speech is sampled at the rate of 12 KHz and noise is removed by the end-point detection technique. The frame size is chosen as 30 *ms* and a frame rate 20 *ms*. Autocorrelation coefficients were computed from the windowed frame and Hamming window is used to minimise the signal discontinuities at the beginning and end of the frame. An efficient Levinson-Durbin's algorithm is used to estimate the coefficients from a given speech signal. It is computationally expensive and not feasible to process all frames in the signal. It also leads to few problems because due to the various signal lengths the total numbers of frames are different. For this experiment, total four frames were selected for each spoken digit in linear distance from the start and end point of the signal, 7 coefficients per time frame over four frames and hence total 28 features per sample were processed. These feature vectors were found to be a good compromise between computational complexity and robustness (Shiraki & Honda, 1988). These frames were used for training and testing the baseline feed forward classifier. For reservoir based approach, same coefficients were used as inputs, further details are reported in the following section.

3.3 Reservoir dynamics

In order to model a stable reservoir, it is very important that it should have two qualities: separation and approximation (Maass et al., 2002). The approximation property refers to the capability of a readout function to classify the state vectors sampled from the reservoir. The separation property refers to the ability to separate two different input sequences from each other. This is important, because the readout network needs to be able to distinguish between two different input patterns to have a good classification accuracy. If the output responses of a reservoir for two different inputs are the same then the readout network will not be able to differentiate between the two patterns and thus will not be able to classify which pattern belongs to a particular class.

In this study, reservoirs are generated in a stochastic manner where a 3D column is constructed (see Fig. 8a) which is a biologically plausible way to imitate microcolumnar structures in a neocortex (Maass et al., 2002), other configurations are also possible as shown in Fig. 8 b, 9a and 9b. There have been other strategies proposed in the literature for their satisfactory performance based on empirical data (Legenstein & Maass, 2005) (Skowronski &

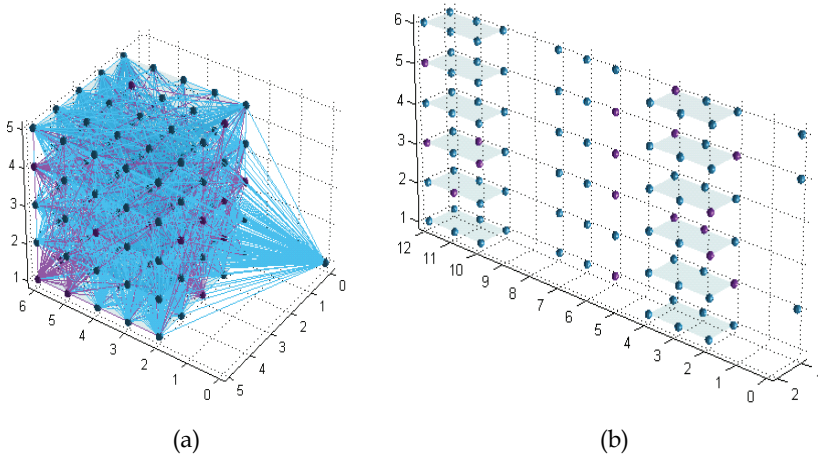


Fig. 8. (a) A 5x5x5 3D grid constructed with a single input neuron
 (b) Three microcolumns of size (3x2x6) (3x1x6) and (3x2x6) with three input neurons.

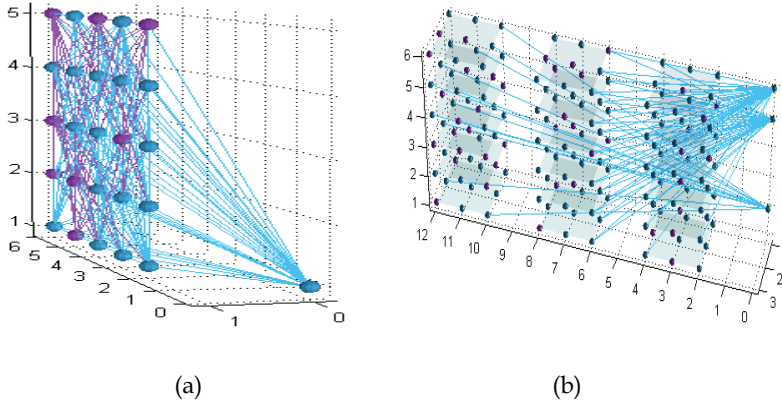


Fig. 9. (a) A 5x1x5 cortical column fully connected with a single input neuron
 (b) A 3x3x6 grid with three input neurons

Harris, 2007) (Jaeger, 2002) (Verstraeten et al., 2007) (Uysal et al., 2007). The design space for constructing a stable reservoir is huge and depends on various important factors such as node type, probability of local and global connections and the size of the reservoir. Apart from the reservoir intrinsic dynamics, there are other factors which contribute to the overall performance of a reservoir such as input features which are used to perturb the reservoir. It is very important that a reliable front end is investigated so that the reservoir can effectively separate different input streams. Once a stable reservoir is constructed then different snapshots are recorded from the reservoir at different time steps and processed for approximating a readout function (Legenstein & Maass, 2005). If two reservoir states $x_u(t)=(R_u)(t)$ and $x_v(t)=(R_v)(t)$ for two different histories $x(\cdot)$ and $v(\cdot)$ are different then the reservoir dynamics are stable, otherwise they will be considered as chaotic. This property is

desirable from practical point of view because different input signals separated by the reservoir can more easily be classified by the readout neurons.

Additionally, the key factor behind a stable reservoir is its short term memory capability which depends on a number of parameters such as membrane threshold, reset voltage and leaky integration. The advantage of bigger reservoirs is that they increase the dimensionality of the reservoir states and data becomes more visible to the readout neurons. One of the criteria in evaluating the computational capability of a recurrent reservoir is to analyse its separation property. It was observed from experimentation that a task which is solved by a large reservoir can also be solved by a much smaller recurrent reservoir provided that the network is capable of differentiating between two different input streams. It is important to observe the reservoir states $x_u(t) = (R_u)(t)$ and $x_v(t) = (R_v)(t)$ for two different input histories $u(t)$ and $v(t)$. In order to fulfill the separation property, two different histories have to be captured by the reservoir to prove that the reservoir dynamics are not chaotic. This is also important for a readout function to distinguish between two different inputs in order to approximate the output function. It is demonstrated in this experiment that simple readout neurons such as an MLP could classify different input streams if properly separated by the reservoir. A major advantage of this approach is the improved classification accuracy with a few neurons which overcomes the burden of bigger reservoirs.

The reservoir is constructed as a three dimensional grid (see Fig. 8a) and the probability of connecting two neurons with each other is determined by calculating the Euclidean distance D between the nodes i and j :

$$i = (i_x, i_y, i_z) \text{ and } j = (j_x, j_y, j_z) \quad (5)$$

Where distance between nodes i and j is calculated as:

$$D(i, j) = \sqrt{(i_x - j_x)^2 + (i_y - j_y)^2 + (i_z - j_z)^2} \quad (6)$$

In equation 5, i and j are input neurons with three coordinates, x , y and z . The probability to connect two neurons i and j is calculated by using the following equation:

$$P_{conn(i,j)} = C.e^{\frac{-D(i,j)}{\lambda}} \quad (7)$$

In equation 7, λ is used to control the average amount of connections between neurons. Depending on whether neurons i and j were excitatory or inhibitory, the value of C was used as suggested by (Maass et al., 2002). This is an important factor in controlling the reservoir dynamics. Different reservoirs were investigated in order to analyse their short term memory and results are reported. In Fig. 10, an architecture of a cortical column is shown where an input stimulus is fully connected to the reservoir of 27 spiking neurons and states were recorded. The reservoir states $x(t)$ refer to the states of all the neurons in the reservoir in terms of membrane voltages and spike firing times at particular time steps ($t, t+1 \dots t+n$), further details regarding reservoir states are provided in section 5.

As shown in Fig. 11, most of the neurons in the reservoir are active which shows an ordered activity in response to the input stimulus. These reservoir states can be used as short-term memory for readout neurons. The membrane time constant is set to 30 ms, absolute refractory period to 3 ms and threshold voltage to 15 mV. The reservoir neurons i , j and

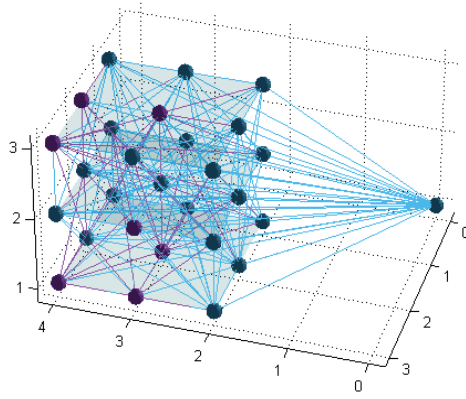


Fig. 10. This figure shows a reservoir of 27 (3x3x3) LIF neurons fully connected with an input neuron. Some neurons are marked with magenta balls which denote inhibitory neurons while others are excitatory neurons.

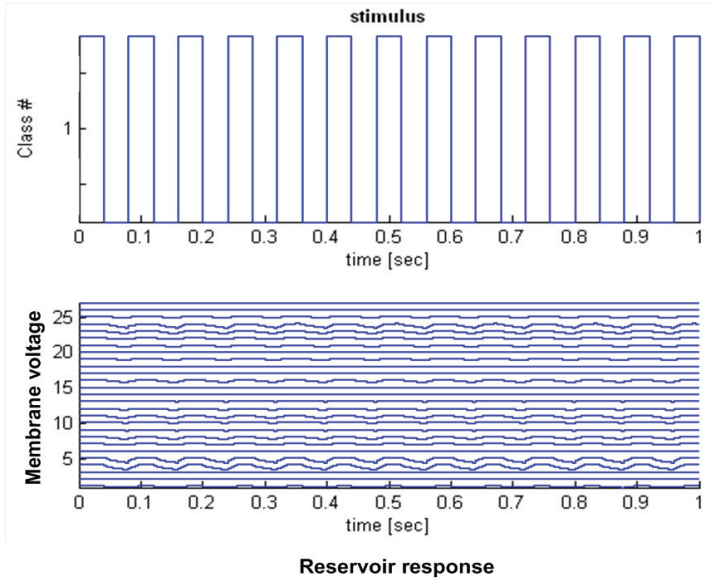


Fig. 11. This figure shows a response of an input square wave in terms of membrane voltages. The top plot shows an input stimulus and bottom plot shows the membrane voltages. The vertical axes show the neurons' membrane potential and the horizontal axis shows the simulation time.

their synaptic connectivity is defined by equation 7 where average amount of connections were controlled by parameter λ . The λ value of 2 is used in these simulations. The parameters selection is based on the biological data obtained from Henry Markram's Lab in Lausanne (Gupta et al., 2003). The data is obtained through experiments on rat somatosensory cortex and suggested by Maass (Maass et al., 2002).

In the paradigm of reservoir computing, readout neurons have an exclusive access to the liquid or reservoir states $x(t)$. For stable reservoir dynamics, it is required that two different inputs $u(s)$ and $v(s)$ should produce two significantly different states $x_u(t)$ and $x_v(t)$ which will hold information about preceding inputs. If the reservoir dynamics are stable (ordered) then a simple memory-less readout can produce the desired output (Natschlaeger et al., 2002). In order to analyse the separation property of reservoirs, various reservoir architectures were simulated and their responses were observed in terms of membrane voltages and spike times. In these simulations, 3D columns were used with different reservoir sizes. A standard leaky integrate and fire model was simulated where the membrane potential of a neuron was calculated as follows:

$$\tau_m \frac{dV_m}{dt} = -(V_m - V_{resting}) + R_m \cdot (I_{syn}(t) + I_{noise}) \quad (8)$$

Where τ_m is the membrane time constant, V_m the membrane voltage, $V_{resting}$ is the membrane resting potential which is 0 V, $I_{syn}(t)$ is the synaptic input current, I_{noise} is a Gaussian random noise with zero mean and a given variance. The membrane potential is set to the value of V_{init} (0.013 V). If the membrane voltage V_m exceeds a certain threshold V_{th} (0.015 V) it is reset to the V_{reset} which is similar to the value of V_{init} .

In order to observe more realistic responses of the reservoir, the features extracted through LPC technique were fed into the reservoir and the reservoir states were recorded. The inputs can be fed into the reservoir in two different ways: analog currents and spike trains. In these simulations analog currents were used as inputs. In section 7, the input values were encoded in spike trains and results were analysed. In Fig. 12, the reservoir architecture is shown with a column of 8 neurons fully connected with an input neuron. The membrane voltages and spike times in response to input digits 1 and 7 in Fig. 13 and 15. For these simulations, the membrane threshold voltages were set to 15 mV and reset voltages to 13.5 mV. An ordered activity is observed in all these responses. It can be seen from these simulations that most of the neuron's membrane voltages and spike firing times were in order which shows a stable reservoir dynamics.

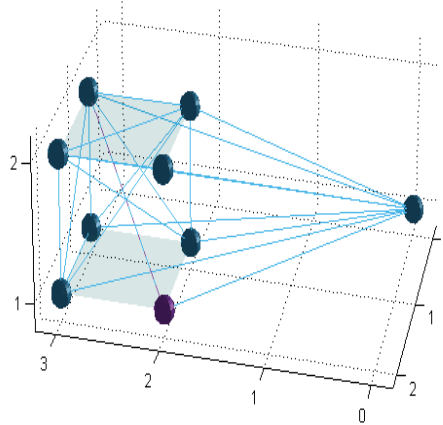


Fig. 12. This figure shows a column of 8 neurons (2x2x2) fully connected with an input neuron. The input to this reservoir is digit 0.

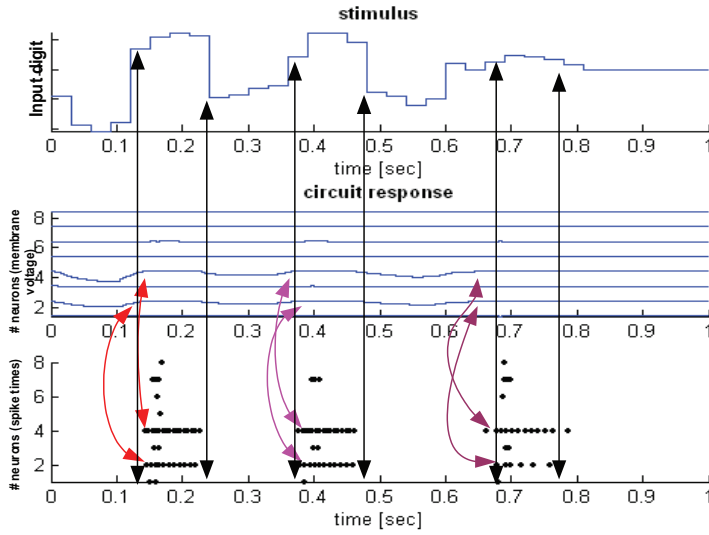


Fig. 13. This figure shows an input digit one and its response in terms of membrane voltage and spike times. The simulation is run for 1 second and spikes were recorded when membrane potential exceeded a threshold value of 15 mV. The reset voltage was set to 13.5 mV. The weights were randomly drawn between -0.1 and 0.1. Total 200 states were recorded by setting the recorder's time step of 5 ms. The middle plot shows the membrane potential and bottom plot shows the spike times. The membrane activity is shown in comparison with the spike firing times.

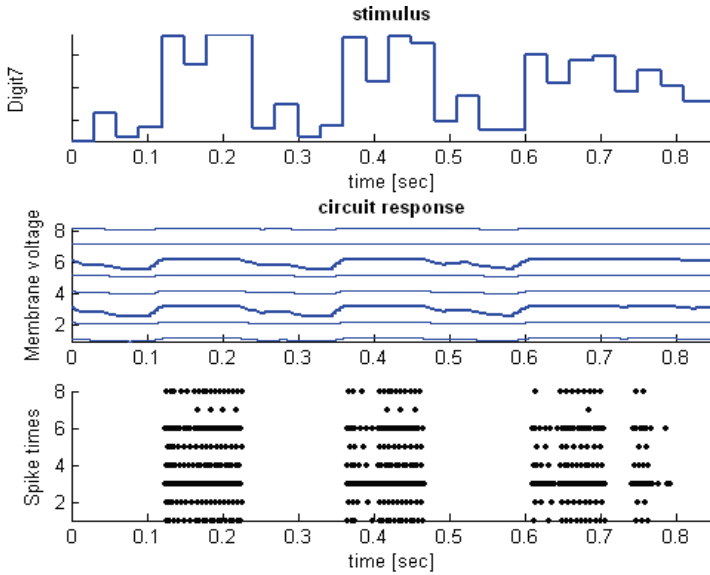


Fig. 14. This figure shows the internal dynamics of a reservoir in response to digit 7.

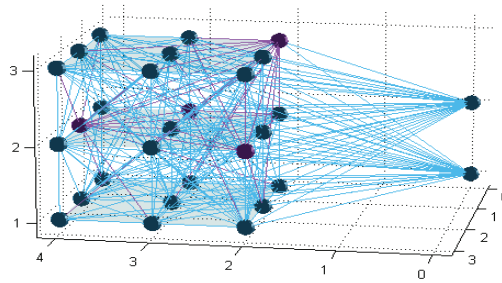


Fig. 15. This figure shows an architecture of 27 ($3 \times 3 \times 3$) recurrent LIF neurons fully connected with two input neurons.

In these simulations the reservoir architecture and the number of neurons were remained fixed and internal states were analysed in response to different input stimuli. It is important to observe that reservoir internal states correspond to the input stimulus and could separate two different inputs. This is an important property which has to be verified for successful classification because readout neurons will have an exclusive access to the membrane voltages of the neurons in the reservoir, if the reservoir states were not significantly different from each other then the readout neurons will not be able to classify different inputs.

In order to analyse the robustness of a reservoir, two inputs were simultaneously applied to the fully connected reservoir and states were recorded. The state differences were analysed as shown in Fig. 18 and 19. Fig. 17 shows the architecture used for these simulations with 27 LIF neurons fully connected with two input neurons. The inputs to the reservoir are two sine waves with different frequencies.

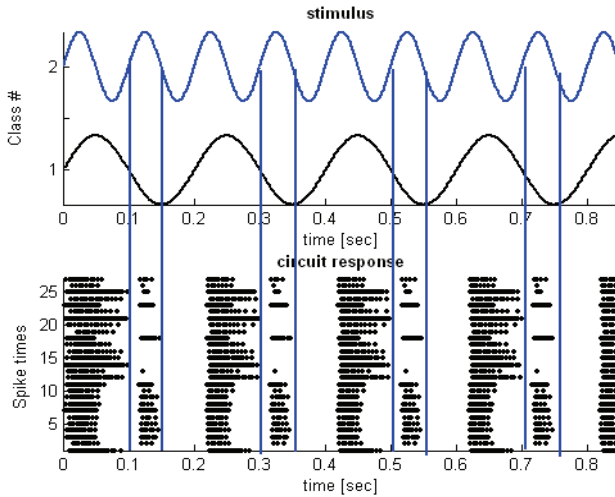


Fig. 16. This figure shows spike times in the bottom plot in response to two different inputs (top plot). The vertical blue lines show the neurons firing times in response to input 2 which is clearly separable from input 1.

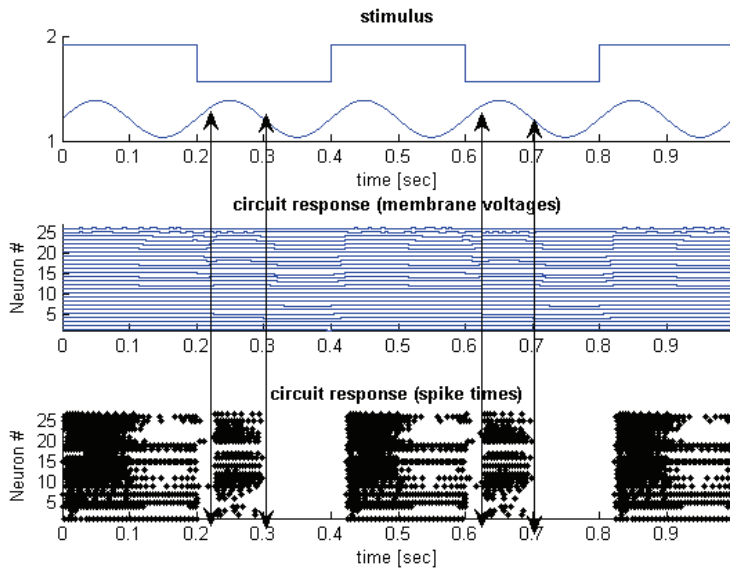


Fig. 17. This figure shows the membrane voltages and spike times in response to two different inputs. The reservoir activity is in order with regard to the input stimuli where both spike times and membrane voltages are separable.

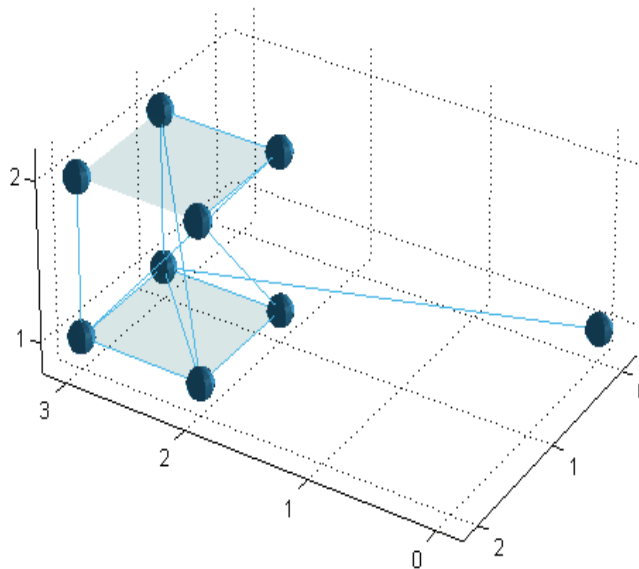


Fig. 18. This figure shows an architecture of 8 neurons ($2 \times 2 \times 2$) partially connected with an input neuron.

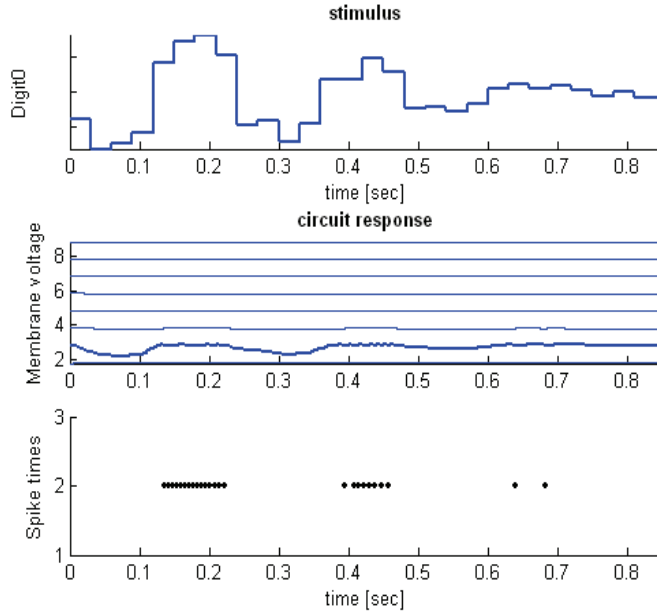


Fig. 19. This figure shows a reservoir response in terms of membrane voltages and spike times. Due to the limited connectivity, only one neuron triggered to fire.

Information processing in cortical neurons crucially depends on their local circuit connectivity. Many efforts have been made to investigate the neuronal wiring of cortical neurons and is still an active area of research (Braitenberg & Schuz, 1998) (Holmgren et al., 2003) (Gupta et al., 2000) (Foldy et al., 2005) (Yoshimura et al., 2005). The overall state of a reservoir very much depends on the connectivity of input neurons with the reservoir. For partially connected inputs, it is less likely that most of the neurons in the reservoir will have short term memory, however, for fully connected inputs, the probability of neurons having the short term memory increases many folds. A series of experiments carried out with different input connectivity and results are reported. In Fig. 18, an architecture is shown with a reservoir size of 8 neurons where input is connected with the reservoir with just one connection. The stimulus to the reservoir is digit 0 and states were recorded in terms of membrane voltages and spike times. An extremely low activity is observed as shown in Fig. 19, however, the short term memory of a reservoir increases with increased input connectivity as shown in Figs. 21 and 23. The reservoir architectures are shown in Figs. 20 and 22.

Input connectivity is an important design decision and affects the overall accuracy of the classifier because reservoir states will be used as training vectors for readout neurons. If the reservoir states were not properly recorded then regardless of the size of the reservoir, the readouts will not be able to classify.

This section investigated different reservoir topologies and dynamics with the help of routines from the CSIM toolbox (Natschlaeger et al., 2002). In section 4, state vectors recorded from the reservoir will be used for training the neural classifier (backend) in order to check the accuracy and robustness of the classifier. The rationale behind this investigation was to analyse the reservoir dynamics which is crucial for readout neurons. It is far from

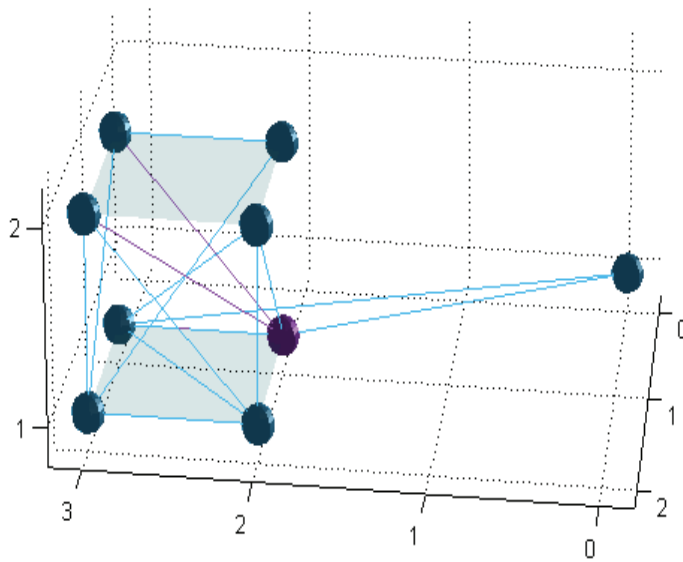


Fig. 20. This figure shows an improved connectivity where input neuron is connected with reservoir with two connections.

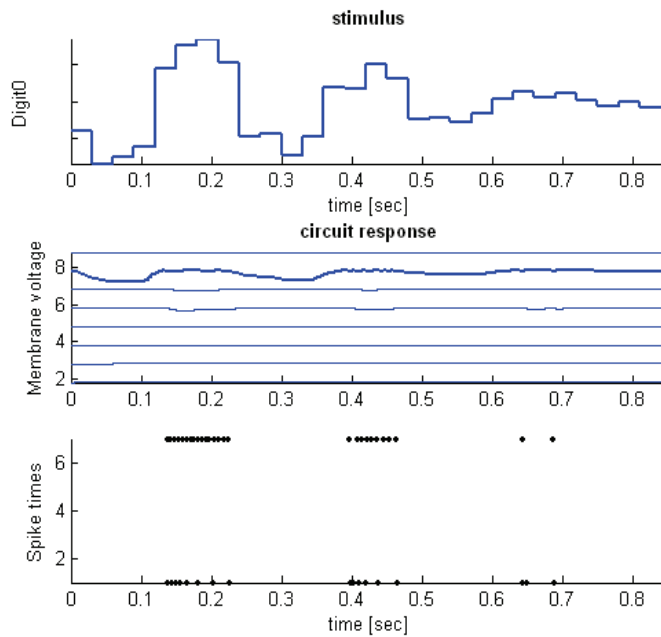


Fig. 21. This figure shows a response of reservoir with an improved neuron activity.

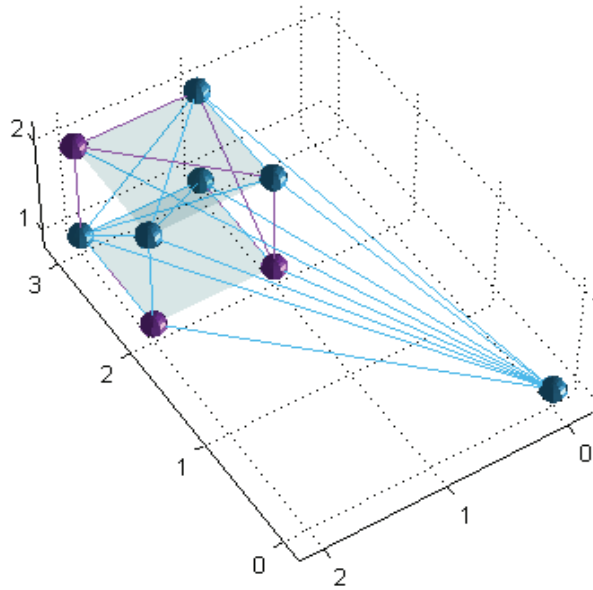


Fig. 22. This figure shows a fully connected network where input neuron is fully connected with all the neurons in the reservoir.

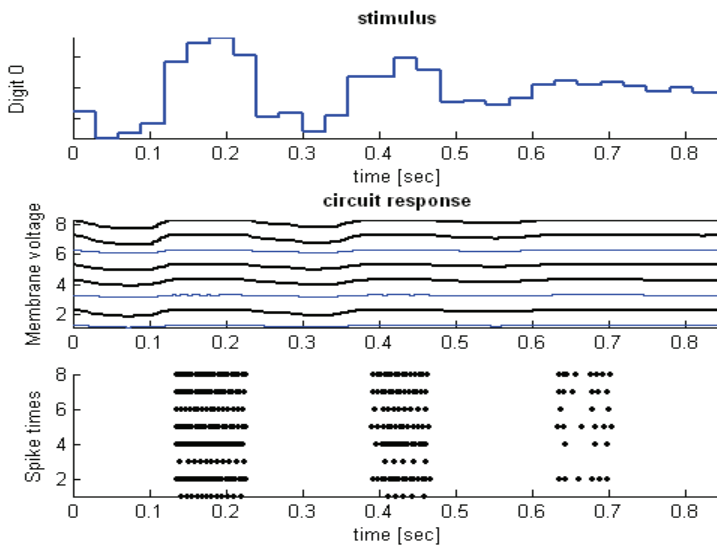


Fig. 23. This figure shows that almost all the neurons are active in the reservoir and the reservoir dynamics are ordered and correspond to the input stimulus.

trivial to model a stable reservoir and very much depends on the experience with this paradigm. The parameters selected for these simulations were empirical, however, once a stable reservoir is modelled, it can successfully be used for different input stimuli and the memory capacity (maximum number of patterns that could be stored for a short period of time) can easily be improved by increasing the size of the reservoir without any significant degradation. In the following section, a baseline feed forward MLP classifier is implemented and features extracted from speech front end are used for training and testing.

The neocortex constitutes almost 80% of the brain and appears to be the fundamental unit of information processing in mammals. The neural microcircuit plays an important role in functions such as adaptability, memory and higher cognitive functions (Natschlager et al., 2003). This study emphasises the importance of connectivity and compact stable reservoirs for microcircuit design. The most interesting characteristic of the neocortical microcircuit is their outstanding computational power where temporal aspect is not necessary for the training of readout neurons because temporal processing is done only at the reservoir level. The biggest advantage of neural microcircuit is that it doesn't require any task specific connectivity, rather the same circuit can be used for different computational tasks and only readouts are required to be trained to produce desired outputs. It is theoretically analysed and predicted by Mass (Maass et al., 2002) that there are no limitations to the power of this model and it can be used as a universal computational model, however, one needs to investigate a reservoir which will work as a fading memory and fulfill the separation property.

4. MLP baseline classifier

The paradigm of reservoir computing very much depends on suitable techniques for front and back end processing. In the previous experiments, front end and reservoir dynamics were investigated; this section investigates a feed forward MLP classifier for backend processing. In order to evaluate classification accuracy, different network sizes and standard backpropagation learning algorithms (backpropagation, resilient backpropagation and Levenberg-Marquardt) were evaluated.

In order to pre-process input data, four frames were selected to extract coefficients from an input speech signal with 7 coefficients per frame and 28 features per sample. The features were extracted through an LPC technique as stated earlier. The pre processing is done in order to remove the noise from a signal and then coefficients were extracted at the frame rate of 20 ms and analysis is done by windowing the speech data with a window size of 30 ms. The noise removal is performed with a threshold technique where threshold is compared with the standard deviation of the signal power.

For this experiment, the total dataset consisted of 200 samples, divided into two sets (training and testing), 20 samples (5 speakers x 4 utterances) for each digit with 28 LPC features per sample. In order to analyse the classification accuracy, different training sets and hidden layer neurons were investigated. The results in terms of classification accuracy are shown in Tabs. 1 and 2. In a series of experiments, the best results obtained in those trials are shown in the following tables. For performance evaluation, the Matlab gradient descent with adaptation (traingda) training algorithm was used where learning rate was adjusted to the value of 0.1 and the goal was set to 0.01.

It is observed from these experiments that by increasing the number of training samples and hidden neurons the overall accuracy improves except the decrease in performance when numbers of hidden neurons were increased from 20 to 25 in Table 1 and from 25 to 30 in

Hidden neurons	Digit 1 (%)	Digit 2 (%)	Digit 3 (%)	Digit 4 (%)	Digit 5 (%)	Digit 6 (%)	Digit 7 (%)	Digit 8 (%)	Digit 9 (%)	Digit 0 (%)	Mean (%)
10	100	60	60	60	40	40	20	0	40	80	50
20	100	40	80	60	80	60	20	20	60	100	62
25	80	60	60	60	80	40	20	20	60	60	54
30	100	60	60	60	80	60	40	20	40	100	62
35	80	60	80	80	100	80	0	40	80	80	68

Table 1. Total samples = 150, training samples = 100, test samples = 50

Hidden neurons	Digit 1 (%)	Digit 2 (%)	Digit 3 (%)	Digit 4 (%)	Digit 5 (%)	Digit 6 (%)	Digit 7 (%)	Digit 8 (%)	Digit 9 (%)	Digit 0 (%)	Mean (%)
6	80	20	100	100	60	60	20	20	80	100	58
20	80	40	100	40	60	80	20	20	80	100	62
25	100	40	100	80	80	80	40	20	80	100	72
30	100	40	100	80	80	80	20	20	80	80	68
30	100	40	100	80	100	80	20	60	80	100	76
35	100	40	100	80	80	80	40	20	80	100	72

Table 2. Total samples = 200, training samples = 150, test samples = 50

Table 2, however the maximum performance obtained was limited to 76%. By increasing the hidden neurons more than 35 the overall performance starts decreasing. The investigation of feed forward network was motivated due to two reasons: first to evaluate the performance of feed forward networks as a standalone classifier, and second to observe the bottlenecks because the same classifier will be used as readout for the reservoir where features extracted from the reservoir will be used as training sets.

5. Reservoir based recognition

The previous sections investigated the front end (speech pre processing), backend (feed forward network) and neural reservoir dynamics. This section will investigate the realization of the complete paradigm of reservoir computing by integrating the overall components investigated in previous sections (see Fig. 24). The paradigm operates by feeding input features extracted through LPC technique into the recurrent neural reservoir as the post synaptic currents. The reservoir is constructed from a random recurrent spiking neural network which projects linearly non separable low dimensional inputs to a high dimensional space. The recurrent neural network by itself is not trained, rather, different reservoir states sampled every 25 ms were recorded for backend classification by the MLP (see Fig. 25). Theoretically, there is no limit to the computing power of this paradigm if the reservoir dynamics are stable and fulfill the requirements of short term memory and state separation. If the reservoir dynamics are carefully analysed then a simple gradient based algorithm can successfully classify a complex problem such as speech recognition. In this experiment, standard supervised algorithms were investigated to quantify their classification accuracies based on the input features sampled through the reservoir. In order to analyse the separation property, different reservoir sizes were chosen and classification accuracies were calculated. The LIF neurons were used for reservoir construction. Both static and dynamic

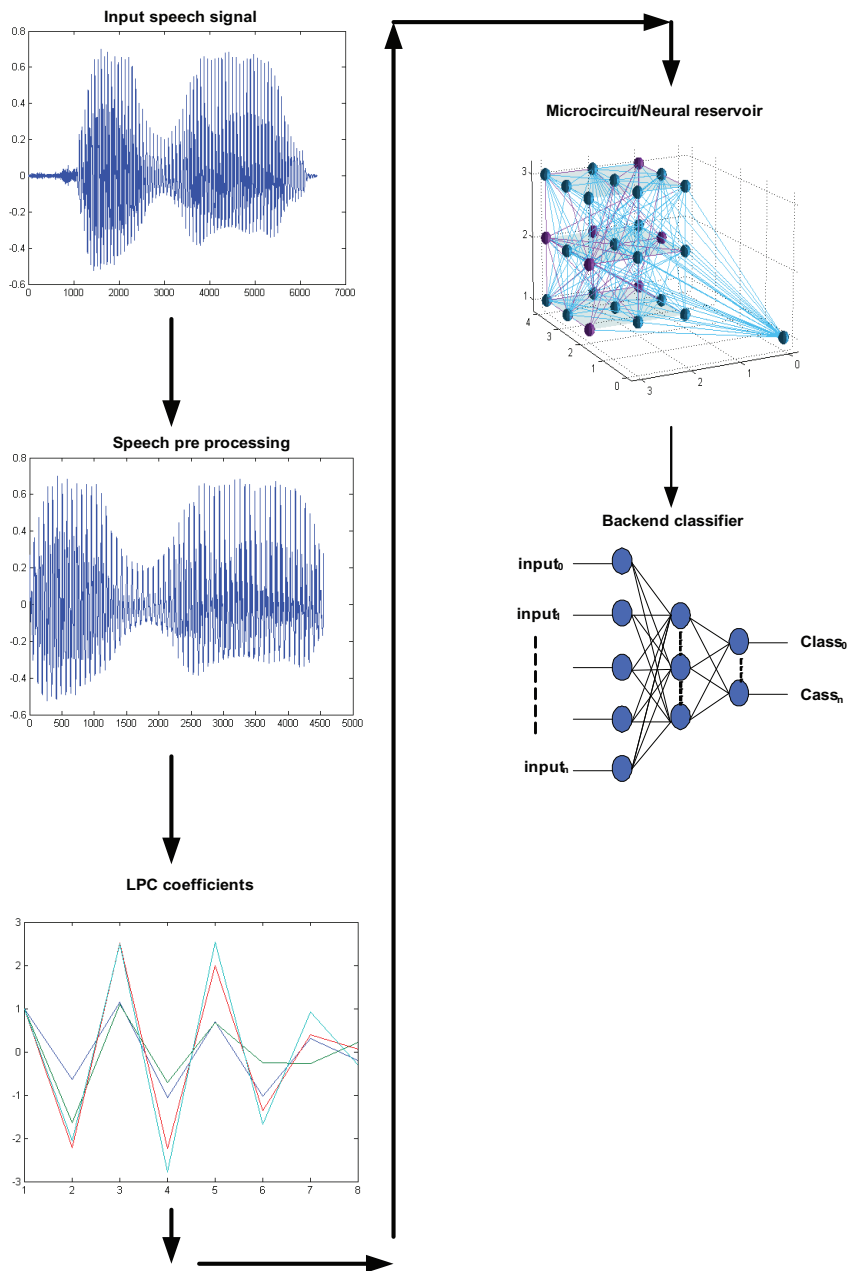


Fig. 24. This figure shows a proposed reservoir based speech recognition approach where input speech signals are pre processed and data was fed into the recurrent neural reservoir. The learning is performed by the feed forward readout neurons.

synapses were used amongst which 20% were chosen to be inhibitory as suggested by (Maass et al., 2002) based on the data from Henry Markram’s Lab in Lausanne (Markram, Lausanne). Once the reservoir is perturbed with the input stimuli, the state vectors were recorded from the reservoir sampled at 25 ms and used for training the feed forward MLP classifier with a single hidden layer and 10 output neurons. The recorder time was set to 5 ms and for 1 second simulation of a reservoir total 200 states were recorded. This state vector contained the membrane voltages of all the neurons in the reservoir at each recording time step. It is neither required nor feasible to process all these states for backend classification, therefore these states were sampled at 25 ms in linear scale from start to the end of simulations and used as training vectors for the classifier. In all these experiments, only the readout neurons were trained whereas the reservoir connectivity remained fixed for generating the reservoir states. The performance of backend feedforward classifier was evaluated with test samples and the best results obtained in different trials are shown in Table 3, 4 and 5.

In Table 3, a reservoir size of 8 neurons successfully classified the input data and the results shown in Tables 4 and 5 also correspond to the theoretical framework of reservoir computing. It is evident from these experiments that for better classification accuracies, stable reservoirs are more important than merely the size of the reservoir. These results also justify the Cover’s separability theorem (Cover, 1965). Using backpropagation algorithms

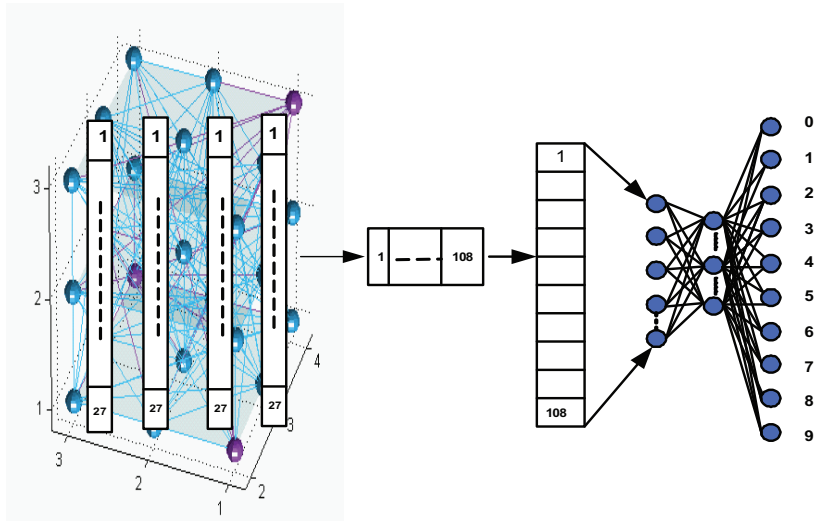


Fig. 25. Processing all frames in the reservoir are computationally expensive therefore specific frames are selected in linear distance with reference to the start point and the end point of the simulations. Each frame consists of the total number of neurons in the reservoir sampled at the rate of 25 ms.

Readout	Reservoir size	Network structure	Test accuracy (%)
Matlab RP	8	32-20-10	94.8
Matlab LM	8	32-30-10	100
Matlab BP	8	32-50-10	96

Table 3. Test performance with reservoir size = 8

Readout	Reservoir size	Network structure	Test accuracy (%)
Matlab RP	15	60-20-10	80.4
Matlab LM	15	60-30-10	98.8
Matlab BP	15	60-50-10	92.8

Table 4. Test performance with reservoir size = 15

Readout	Reservoir size	Network structure	Test accuracy (%)
Matlab RP	27	108-20-10	100
Matlab LM	27	108-30-10	96
Matlab BP	27	108-50-10	100

Table 5. Test performance with reservoir size = 27

for readout neurons are advantageous because they can approximate complicated target functions if appropriate network architectures were selected. In this study, different architectures were investigated with different hidden layers and number of neurons and results are reported. A thorough discussion about these algorithms is out of the scope of this chapter, reader is refer to (Hertz et al., 1991) (Duda et al., 2001) for further details. There is no specific learning algorithm which will guarantee a good accuracy of the readout neurons. A logical approach is to start with standard backpropagation algorithms and test the accuracy with different hidden layers and number of neurons. The good classification accuracy will not only infer to the suitable architecture of a feed forward neural network but will also show that the reservoir has significantly separated different inputs and have projected data sufficiently on a high dimensional space. These series of experiments have thoroughly investigated the theoretical framework of reservoir computing and results are demonstrated with an speech recognition application. In order to quantify results, a baseline feedforward classifier is implemented and results are compared with the reservoir based technique. This section has investigated the framework with analog inputs extracted through speech front end, in the following section, an experiment is carried out where input stimuli is converted into Poisson spike trains and results were analysed.

6. Spike based coding

Biologically plausible neurons communicate through spike trains, this section will investigate the framework by encoding the analog input values into spike trains and results are reported. As stated by Squire and Kosslyn that the timing of successive action potentials is irregular in the cortex (Squire & Kosslyn, 1998), therefore Poisson spike coding technique is used where the generation of each spike was dependent only on an underlying analog driving signal and each spike was considered to be independent of all the other spikes (Heeger, 2000).

In order to generate a spike train, an interspike interval is randomly drawn from an exponential distribution and each successive spiketime is calculated by the previous spiketime plus a randomly drawn interspike interval. In order to convert input features into Poisson spike trains, first negative analog values are converted into positive values and spike times were calculated (see Fig. 26). The spike times were sampled at 100 ms for maximum time in order to generate spike trains (see Fig. 27).

The spiketimes calculated through Poisson encoding were fed into the reservoir and states were recorded. The total numbers of recorded states were kept the same for fair comparison with analog inputs as stated in section 5. The results are shown in Table 6, 7 and 8. The reservoir is tested with the size of 8 and 15 neurons and the overall accuracy didn't improve by increasing the size of reservoir. The best accuracy achieved with Poisson encoding was limited to 98%.

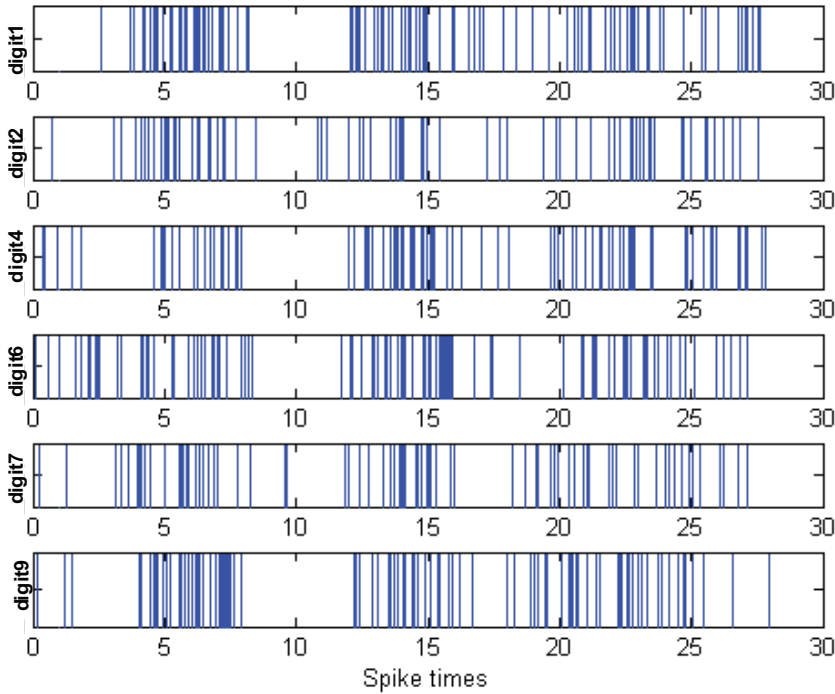


Fig. 26. Spike times with Poisson encoding for digit1, 2, 4, 6, 7, and 9

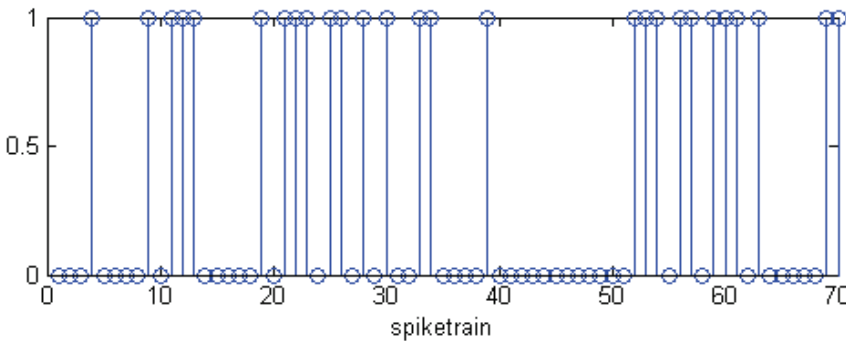


Fig. 27. Spike times sampled at rate 0.1 for maximum time in order to generate spike trains

Readout	Reservoir size	Network structure	Test accuracy (%)
Matlab RP	8	32-20-10	62
Matlab LM	8	32-30-10	98
Matlab BP	8	32-50-10	80.4

Table 6. Test performance with reservoir size 8

Readout	Reservoir size	Network structure	Test accuracy (%)
Matlab RP	15	60-20-10	70.3
Matlab LM	15	60-30-10	78
Matlab BP	15	60-50-10	77.5

Table 7. Test performance with reservoir size 15

Readout	Reservoir size	Network structure	Test accuracy (%)
Matlab RP	27	108-20-10	72
Matlab LM	27	108-30-10	71.5
Matlab BP	27	108-50-10	75.3

Table 8. Test performance with reservoir size 15

In this experiment, Poisson spike trains were used as input for reservoir but no significant improvement is achieved by increasing the reservoir size and accuracy found to be inferior than previous experiments where analog values were used. The possible reason is due to the highly random Poisson process, the spike trains were randomly generated and reservoir couldn't differentiate between different spike trains. The results vary from one trial to another and best results achieved from those trials are reported in Table 6, 7 and 8.

7. Summary

One of the key properties of the reservoir computing is their short term memory and ability to separate different inputs. This property is called fading or echo state property which is very much dependent on the internal dynamics of the reservoir. The memory capacity of bigger reservoirs can only be useful if reservoir dynamics are not chaotic. If the reservoirs are not stable then regardless of the size, memory characteristics can not be achieved. The minicolumns in the cortex appear to be the basic unit of computing where each microcolumn consists of few neurons (Mountcastle, 1997). The columns are distributed therefore these experiments suggest that small reservoirs with stable dynamics are more reliable than bigger chaotic reservoirs. Once an optimal reservoir is investigated, backend processing can further improve the overall performance.

The rationale behind the experiments conducted in this work was not to exhaustively check the readouts or suitable front ends, rather to investigate the theoretical framework and its viability as a universal classifier. Recently, different aspects of the reservoir computing such as mean field theory, edge of chaos, biologically plausible front ends, computational nodes and memory capacity have been investigated and reported in (Maass et al., 2002) (Skowinski et al., 2007) (Jaeger, 2001) (Verstraeten et al., 2007) and (Uysal et al., 2007). The

hybrid implementation presented in this chapter is more suitable for the framework of reservoir computing and the results provided support the theoretical framework.

Given the complexity of the speech recognition problem, the paradigm was split into three sub sections, front end, back end and optimal reservoir. The components were implemented and analyzed individually and integrated for several final experiments. Most of the parameter selection in these experiments were empirical and depends on the experience related to the reservoir computing. It is obvious from these experiments that pre and post processing are important factors because reservoir computing can not guarantee to perform well if either the front or backend are not properly selected. Despite the promising results obtained through this investigation, a fundamental question remains open regarding the way data is pre processed in this study and other related work. In SNNs, pre processing may not be the best way to communicate with spiking neurons and this is the fundamental area that needs further investigation and which is outside the scope of this book chapter.

This chapter thoroughly investigated the theoretical framework of reservoir computing and extended by analysing the compact reservoir dynamics, front end pre processing and back end classification technique. The reservoir based recurrent neural architectures have proven to perform better on classification related tasks such as speech recognition, however, their performance can be increased if combined with feed forward networks. An alternative approach is proposed by utilizing the idea of reservoir computing with efficient feature extraction technique and learning by rather simple feed forward network. This framework revealed a powerful alternative for recognition task and provides a significant improvement in terms of their performance and robustness. To the best of authors' knowledge, none of the existing reservoir based techniques successfully classify the speech recognition problem with an extremely compact reservoir. This study emphasized the modelling of a compact dynamic reservoir and empirically investigated the short term memory capacity and separation property for stable reservoirs. The results show that a stable reservoir and efficient front end technique can solve significantly complex recognition task with simple readouts.

8. References

- Haykin, S (1999), *Neural Networks, A comprehensive foundation*, 2 ed, Prentice hall, Inc, New Jersey.
- Pavlidis N.G, Tasoulis D.K, Plagianakos V.P, Nikiforidis G and Vrahatis M.N. (2005), Spiking neural network training using evolutionary algorithms, *International Joint Conference on Neural Networks, IJCNN'05*, pp. 1-5.
- Bohte S., Kok J and La Poutre H (2000), Spike-prop: Error backpropagation in multi-layer networks of spiking neurons, *Euro Symposium on Artificial Neural Networks ESANN'2000*, pp. 419-425.
- Maass, W, Natschläger, T and Markram, H (2002), Real-time computing without stable states: A new framework for neural computation based on perturbations, *Neural Computation*, 14(11), pp. 2531-2560.
- Jaeger, H. (2001), The "echo state" approach to analysing and training recurrent neural networks, Tech. Rep, Fraunhofer Institute for Autonomous Intelligent Systems, *German National Research Center for Information Technology*, (GMD Report 148)
- Joshi, P and Maass, W (2005), Movement generation with circuits of spiking neurons, *Neural Computation*, 17(8), pp. 1715-1738.

- Jaeger, H., Haas, H.: Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication, *Science* 304(5667), pp. 78–80 (2004).
- Skowronski, M.D., Harris, J.G (2007).: Automatic speech recognition using a predictive echo state network classifier, *Neural Networks* 20(3), pp. 414–423.
- Verstraeten, D., Schrauwen, B., D’Haene, M., Stroobandt, D. (2007), An experimental unification of reservoir computing methods, *Neural Networks* 20, pp. 391–403.
- Uysal, I., Sathyendra, H., Harris, J.G. (2007), Spike based feature extraction for noise robust speech recognition using phase synchrony coding, *International Symposium on Circuits and Systems*, pp. 1529–1532.
- Hopfield, J. J. (1982) Neural networks and physical systems with emergent collective computational properties, *Proc. Nat. Acad. Sci. (USA)* 79, pp. 2554–2558.
- Elman, J.J (1990), Finding structure in time, *Cognitive Science*, 14, pp. 179 – 211.
- Jordan, MI (1996), Neural networks, A Tucker editors, *CRC handbook of computer science*, CRC press.
- Bengio, Y (1996), Neural networks for speech and sequence recognition, *Book published by International Thomson Computer Press*.
- Looney, C G (1997), Pattern recognition using neural networks, theory and algorithms for engineers and scientists, *book published by OUP USA*.
- K. Doya (1995), Recurrent Neural Networks, supervised learning in M.A Arbib Editors, *The Handbook of Brain Theory and Neural Networks*, MIT press.
- Atiya, A. F and Parlog, A G (2000), New results on recurrent network training: unifying the algorithms and accelerating convergence, *IEEE Transactions in Neural Networks*: 11(3), pp. 697-709.
- Pearlmutter, B.A (1995) Gradient calculation for dynamic recurrent neural networks: a survey, *IEEE Transactions on Neural Networks*: 6(5), pp. 1212–1228.
- Jaeger, H. (2002), Adaptive nonlinear system identification with echo state Networks, *Advances in neural information processing systems*, pp. 593–600.
- Williams, RJ and Zipser, D (1989), A learning algorithm for continually running fully recurrent neural networks, *Neural Computations*, 1, pp. 270 - 280.
- S. Singhal and L. Wu. Training feed-forward networks with the extended Kalman filter, *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 1187.1190, 1989
- Atiya, A. F and Parlog, A G (2000), New results on recurrent network training: unifying the algorithms and accelerating convergence, *IEEE Transactions on Neural Networks*: 11(3), pp. 697-709.
- Abbott, L.F and Nelson S.B (2000), Synaptic plasticity, taming the beast, *Nature Neuroscience*, 3(supplement), 1178–1183.
- Steil, J.J (2004), Backpropagation-Decorrelation: online recurrent learning with $O(N)$ Complexity, *International Joint Conference on Neural Networks*, vol. 1, pp.843–848 (2004).
- Maass, W., Natschläger, T and Markram, H (2004), Computational models for generic cortical microcircuits, *Computational Neuroscience: A Comprehensive Approach*, Chapter 18, pages 575–605
- Hopfield, JJ and Brody, CD (2000), What is a moment? Transient synchrony as a collective mechanism for spatiotemporal integration, *PNAS* (98), pp. 1282–1287.
- Cover, T.M (1965), “Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition”, *IEEE Transactions on Electronic Computers*, EC -14(3), pp. 326 – 334.

- Sivakumar, SC., Phillip, WJ and Robertson, W (2000), Isolated digit recognition using a block diagonal recurrent neural network, *Canadian conference on electrical and computer engineering*, pp. 726 – 729
- Zhao, Z (1991), Connectionist training of non-linear hidden Markov models for speech recognition, *IJCNN*, pp. 1647 – 1652.
- Kim, SH., Koh, SY., Ahn, JY and Hur, KI (1999), A study on the recognition of the isolated digits using recurrent neural predictive HMMs, *TENCON*, pp. 593 – 596
- Luh, C and Jantan, A (2004), Digit recognition using neural networks, *Malaysian Journal Of Computer Science*, Vol. 17, pp. 40-54.
- Verstraeten, D, Schrauwen, B., Stroobandt, D and Campenhout, JV (2005), Isolated word recognition with the liquid state machine: a case study, *Information Processing Letters* 95, pp. 521 – 528.
- Shiraki, Y and Honda, M (1988), LPC speech coding based on variable-length segment quantization, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(9), pp. 1437 -1444.
- Legenstein, R and Maass, W (2005), What Makes a Dynamically System Computationally Powerful? New Directions in Statistical Signal Processing: From Systems to Brain, *MIT press*.
- Gupta A, Wang Y, Markram H (2003) Organizing principles for a diversity of GABAergic Inter-neurons and synapses, *Neocortex Science* (287), pp. 273–278.
- Thomas Natschläger, Wolfgang Maass (2002), Spiking neurons and the induction of Finite State Machines, *Theor. Comput. Sci.* 287(1): pp. 251-265.
- Squire, L.R., Kosslyn, S.M. (1998), Findings and current opinion in cognitive neuroscience, *The MIT Press*, USA
- Braitenberg, V., Schuz, A. (19989): Anatomy of the Cortex: Statistics and Geometry, *Springer-Verlag*.
- Mountcastle, V B(1978), An organizing principle for cerebral function: the unit model and the distributed system, in the mindful brain, *MIT press*, Cambridge, MA.
- Doddington, G.R., Schalk, T.B. (1981), Speech recognition: Turning theory to practice, *IEEE Spectrum* 18(9).
- Foldy, C., Dyhrfeld-Johnsen, J., Soltesz, I. (2005), Structure of cortical microcircuit theory, *J. Physiol.* 562, 47–54.
- Yoshimura, Y., Dantzker, J.L.M. (2005), Callaway, E.M.: Excitatory cortical neurons form fine-scale functional networks, *Nature* 433, 868–873.
- Gupta, A., Wang, Y., Markram, H. (2000), Organizing principles for a diversity of GABAergic interneuron and synapses, *Neocortex. Science* 287, pp. 273–278.
- Holmgren, C., Harkany, T., Svennenfors, B., Zilberter, Y. (2003), Pyramidal cell communication within local networks in layer 2/3 of rat neocortex, *J. Physiol.* 551, pp. 139–153.
- Braitenberg V, Schuz A (1998) Cortex: Statics and Geometry of Neuronal Connectivity, book published by *Springer-Verlag*, Berlin.
- Ghani, A., McGinnity, T.M., Maguire, L.P., Harkin, J.G. (2006), Analyzing the framework of ‘Reservoir Computing’ for hardware implementation, *NIPS workshop on Echo State Networks*, pp. 1–2.
- Arfan Ghani, T. Martin McGinnity, Liam P. Maguire, Jim Harkin (2008), Neuro-inspired Speech Recognition with Recurrent Spiking Neurons, *ICANN (1)*, pp. 513-522.