ADAPTING SPIKING RECURRENT NEURAL NETWORKS WITH SYNAPTIC PLASTICITY FOR TIME-SERIES PATTERN RECOGNITION

Joseph Chrol-Cannon

A thesis submitted for the degree of Doctor of Philosophy

October 2015
Abstract

Self-organization in biological nervous systems during the lifetime is known to largely occur through a process of plasticity that is dependent upon the spike-timing activity in connected neurons. This thesis investigates how synaptic plasticity can be applied to recurrently connected spiking neural networks in order to improve time-series pattern recognition accuracy by learning the temporal structure of the input signal within the synaptic weights.

Reservoir computing is a recurrent neural network paradigm that can naturally process temporal signals in real-time. However, the learning in the model is limited to linear regression of a simple perceptron, not the recurrent part of the network, preventing the temporal structure of the input to be learned by the model. It is the intention of this work to allow the learning of temporal structure in the recurrent connections through application of synaptic plasticity models derived from biological observation. By allowing all parameters to adapt to the input signal, it is expected that a trained output will achieve higher accuracy in a pattern recognition task.

However, it is found that adapting the recurrent connections with a variety of plasticity models does not necessarily improve accuracy, the difference observed is largely negligible, and is mostly detrimental when significant. To establish why this is, several metrics used to quantify separability and information content within recurrent networks are employed to determine the computational effect that synaptic plasticity is having on the network structure. Surprisingly, it is found that plasticity can improve or degrade the metrics unpredictably depending on the plasticity model, initial connectivity and input data used. Furthermore, the metrics themselves are demonstrated to have only weak correlation with the pattern recognition accuracy of a trained readout, suggesting that these measures are not as indicative of performance as often claimed.

A more fundamental analysis of synaptic weight change is undertaken that compares the geometric distance of the synaptic changes caused by input patterns of different classes. These experiments show that plasticity leads to remarkable class-specificity in synaptic changes in data sets that include auditory speech samples and visual human motion behaviors. The limitation of input-specific learning actually originates in interference within synaptic parameters caused
by the incremental nature of weight updates that leads to continuous overwriting of previously learned parameters.

These insights are then applied to propose a plasticity sensitive readout neuron for temporal pattern recognition that uses the change in weight rather than the neural activity to directly form an output. This method utilizes the input sensitivity of the plasticity models while avoiding the limitations of structural adaptation that is curbed by interference. In this framework, the role of structural adaptation is to reach a critical state in which each input drives the parameters away from the sample mean in a unique direction.

The challenges and limitations faced of current spiking and reservoir computing models are explored. These include interference and forgetting in the synaptic parameters when applying plasticity to the reservoir computing framework, and the limitations regarding use of advanced spike-time neural coding in recurrent networks with continuous activity.

Further work is discussed regarding solutions to the aforementioned limitations as well as possible new directions for more advanced applications of synaptic plasticity based on developmental genetic principles that could enhance the self-organizing ability of neural networks.
Contents

1 Introduction ................................................................. 11
  1.1 Background ............................................................ 12
  1.2 Motivation ............................................................. 14
  1.3 Structure of Thesis .................................................. 15

2 Reservoir Computing with Synaptic Plasticity ...................... 17
  2.1 Introduction .......................................................... 17
  2.2 Neural Code .......................................................... 19
  2.3 Reservoir Computing Models ...................................... 21
      2.3.1 Echo State Networks .......................................... 23
      2.3.2 Liquid State Machines ....................................... 23
  2.4 Plasticity in Reservoir Computing ................................ 23
      2.4.1 Improving Reservoir Computing Performance .............. 24
      2.4.2 Learning Input Structure and Coding ....................... 25
      2.4.3 Correlate or Decorrelate Neural Activity ................. 25
      2.4.4 Increasing Sparsity and Information Maximization ....... 26
  2.5 Rate Based Plasticity Models ...................................... 27
      2.5.1 Types of Neural Plasticity ................................... 27
      2.5.2 Hebb’s Postulate .............................................. 28
      2.5.3 Homeostatic Regulation ...................................... 28
      2.5.4 Self-Organising Networks ................................... 29
      2.5.5 Anti-Hebbian Learning ...................................... 30
2.5.6 Oja’s Rule .................................................. 31
2.5.7 BCM Theory ............................................. 31
2.6 Spike Timing Based Plasticity Models .......................... 32
  2.6.1 Spike Timing Dependent Plasticity ......................... 33
  2.6.2 Voltage-Dependent Plasticity ............................ 37
  2.6.3 Calcium Controlled Plasticity ............................ 38
  2.6.4 Gene-Regulated Plasticity ................................ 38
2.7 Discussion .................................................... 41
2.8 Summary ...................................................... 42

3 Reservoir Computing Metrics 45
  3.1 Introduction ............................................... 46
  3.2 Reservoir Metrics ......................................... 47
    3.2.1 Class Separation ................................... 47
    3.2.2 Kernel Quality ....................................... 48
    3.2.3 Lyapunov’s Exponent ................................ 48
    3.2.4 Spectral Radius ...................................... 49
  3.3 Methods .................................................... 50
    3.3.1 Reservoir Network .................................. 50
    3.3.2 Connectivity ......................................... 51
    3.3.3 Plasticity ............................................ 52
    3.3.4 Time Series Tasks ................................... 54
  3.4 Results ..................................................... 55
    3.4.1 Effect of Plasticity and Connectivity on Metrics .... 55
    3.4.2 Metric Correlation to Performance .................... 59
  3.5 Discussion .................................................. 61
  3.6 Summary .................................................... 62

4 Input-Specific Synaptic Adaptation under Plasticity 65
  4.1 Introduction ............................................... 66
5.5.2 Synaptic Plasticity Convergence .............................................. 100
5.6 Biological Plausibility .............................................................. 101
5.7 Summary .............................................................................. 102

6 Summary and Future Work ......................................................... 105
6.1 Main Contributions ................................................................. 105
6.2 Further Extensions ................................................................. 107
6.3 New Directions ................................................................. 110
Acronyms

**LSM** Liquid State Machine

**ESN** Echo State Network

**STDP** Spike-Timing Dependent Plasticity

**TP-STDP** Tri-Phasic Spike-Timing Dependent Plasticity

**BCM** Beinenstock-Cooper-Munro (form of plasticity)

**LTP** Long-Term Potentiating

**LTD** Long-Term Depression
List of Publications


Chapter 1

Introduction

This thesis occupies a marginal research area that is situated between – and draws upon – two others that are far more established in their own rights: computational neuroscience and machine learning. These fields have settled into their separate strands of investigation with now seemingly distinct aims, yet both actually share a common origin: cybernetics.

Cybernetics was really the starting point of the academic study of the idea of the brain as a computational system and the simultaneous intention to replicate the abilities of the brain inside the newly invented programmable digital computer. This proved a far more challenging task than originally predicted. Due to lack of progress on these intractable problems, cybernetics split into a number of slightly less ambitious fields. Computational investigations into Neuroscience now only sought to explain aspects of cognition in lieu of modelling the whole thing. Artificial intelligence and then machine learning sought to replicate increasingly specific abilities of human intelligence that now are largely reduced to pattern recognition and prediction tasks.

While this split has led to a great deal of progress in both neuroscientific understanding and given powerful adaptive abilities to computer systems – that could not have been preprogrammed – we feel it has also detracted from the original goal of recreating truly intelligent systems that have the ability to learn new tasks like people do; in real time, without being programmed to. The recent advance in machine learning, popularly referred to under the umbrella title ‘Deep Learning’, has significantly improved pattern recognition abilities in a variety of applications. However, it does not work in the same way as brains do [Szegedy et al., 2014, Nguyen et al.,
This thesis aims to advance the niche corner of research that is starting to bridge the gap between the study of natural and artificial learning systems. We apply biologically observed synaptic plasticity to networks of recurrently connected nature inspired spiking neurons that are used in a machine learning framework of trained regression to perform pattern recognition on temporal sensory data sets.

1.1 Background

The general purpose of this thesis is to make advances in understanding and improving computational neuroscientific models that have some of the desirable properties of the natural systems by which they are inspired. More specifically, this work is very much in the trend of taking ideas from computational neuroscience plasticity models and applying them to the computational learning framework of reservoir computing. It does not deal with the related task of improving understanding of neuroscience through the use of tradition machine learning methods. Neither does it focus on working within a typical machine learning framework to make advances in feature engineering or classifier accuracy. Rather, the aim is to bring findings from natural neural and plasticity models into a learning framework that provides some insight into how the brain does learning.

A number of features that we aim to bring to our contribution to computational neural learning systems are explained as follows.

**Nature inspired learning models.** Sensory perception in humans and animals far exceeds the pattern recognition ability of artificial machine learning models. The underlying biological systems that enable these abilities must consist of developmental and regulatory mechanisms that surpass current artificial models and differ from the machine learning paradigm at a fundamental level.

Furthermore, brains develop and adapt to a range of arbitrary tasks and environments without external control. They are able to determine context and relevant goals without having these things pre-specified.
1.1. BACKGROUND

**Real-time processing.** Despite neurons being slow (a maximum of about 200 impulses per second) and imprecise (on the order of a millisecond precision ex vivo, with likely far lower precision in a functioning brain) compared to electronic circuits, biological neural systems can react very quickly to sensory stimuli and process streams of information in real time. This is achieved by relying on few interactions between vast numbers of neurons, rather than performing computation on individual sequences of neural impulses.

By connecting groups of naturally inspired spiking neurons, it is possible to produce a complex transformation of an input stimuli that continuously updates in response to a changing input signal. Ultimately, a predictive output or action selection will be made whenever required by the velocity of the input signal and reaction time required by the task. In a real-time processing paradigm, neural output activity and structural learning of model parameters should both happen incrementally and ideally be aspects of the same processes.

**Temporal signal processing.** The highly recurrent spiking neural networks found in nature are ideally suited to processing and recognizing patterns in time-series data. Sensory input mostly consists of a significant temporal component in addition to spatial patterns. Recurrent connectivity enables input signals to interact with an echo of previous inputs and thereby perform computations naturally on time-series patterns. This is often conceptualized by thinking of the internal activity of a recurrent network providing a context for future input stimuli.

Spiking neural membrane models produce output patterns of activity that are highly sensitive to the temporal order of action potentials that are integrated by its synapses. This means that even a single neuron, without any recurrence, is able to distinguish between different time-series patterns. This ability is further magnified by the wide range of axonal conduction delays with which a single neuron can be connected to others in the network.

**Unsupervised learning.** In natural systems, learning is unsupervised because all information enters through the senses, following a common pathway before it is separated by neural processing. Unless electrodes are inserted directly into cortical tissue and used to stimulate
specific neurons, there is no means for an external executive signal to exert a special influence on the learning of a given task. The closest prospect of supervised learning in nature is through conditioning, where experiences are delivered at specific times to trigger the pleasure and pain centers of the brain in order to reinforce certain behaviors. This is supervised learning at a behavioral and cognitive level, but does not consist of any special supervised neural learning mechanisms.

Target patterns of neural activity must form within brains from the processing of incoming sensory signals in combination with the current neural activity. Associative synaptic plasticity forms connections between co-activating sensory input neurons and internal neural representations that emerge. Thus, pathways emerge that can begin to separate different input patterns from the shared sensory field.

Truly unsupervised learning systems do not rely on the presence of a target signal \textit{a priori}, any such signal must be created through impartial synaptic plasticity mechanisms that organize structure directly from the raw sensory input.

1.2 Motivation

While the previous section has outlined the general thrust of this thesis, we will now define some more specific areas of investigation that will determine the main contributions the work.

\textbf{Measure the impact of plasticity on recurrent neural networks.} After evaluating the literature to discover the potential functions of synaptic plasticity, we will test these functions through experiments that monitor the impact they have on reservoir computing properties. Many properties of reservoir computing have associated metrics that indicate their computational ability and information content. By testing how a number of common models of plasticity change these metrics, it will be possible to make conclusions on which of the proposed functions of plasticity holds strongest.

\textbf{Analyze input-specificity of synaptic adaptation.} The most widely acknowledged role of synaptic plasticity is in memorizing input patterns. In order to properly achieve this
function, the synaptic adaptation caused by different kinds of input pattern must be unique to those particular data. Also, in an iterative learning framework inherent in time-series pattern recognition, input-specific adaptations are often overwritten in a process of interference. We will analyze to what extent synaptic changes are input-specific and also if those unique changes are retained in the network structure during the training process.

**Improve reservoir computing using synaptic plasticity.** Whether it be through general improvement of the reservoirs computational properties or by allowing unsupervised learning of input patterns, it is the aim of this work to use plasticity to improve temporal pattern recognition accuracy of the standard reservoir computing model. Using the findings of the previous two points, the model of reservoir computing will be adapted in order to achieve this aim.

### 1.3 Structure of Thesis

A guide to the thesis is given here by way of a short synopsis of each of the other chapters. Chapters 2, 3 and 4 are self-contained, while 5 and 6 rely extensively on work performed in earlier chapters.

**Chapter 2:** A literature review is conducted into experimental work that investigates the application of plasticity to computational models of neural processing. The review includes artificial and spiking neural network models and phenomenological plasticity models with continuous (rate-based) and discreet (spike-based) formulations. Focus is given to studies that aim to improve temporal pattern recognition accuracy through application of plasticity or studies that analyse information theoretic properties of neural networks to determine the effect of plasticity. A stance is taken on the state of the literature that advocates a synthesis between neural plasticity models and machine learning that is driven more strongly towards biological developmental systems.

**Chapter 3:** Formal information theoretic metrics are explored for recurrent neural networks that include information capacity and pattern separation. This chapter has a dual purpose
with regards to empirical investigation into the metrics; find the correlation between each of the metrics with pattern recognition accuracy to determine the extent that the former can indicate the latter, and to analyse how neural plasticity and network structures affect the metrics themselves as well as pattern recognition accuracy.

Chapter 4: A more detailed analysis is made of how plasticity changes the neural network structure. Synaptic weight change is calculated directly as a set of features vectors for each sample. Geometric distance is calculated for these vectors between inter- and intra-class sets. By plotting a correlation heat map of the results, we show that unsupervised STDP and BCM do learn class-specific features of input patterns. An analysis of interference in the synaptic parameters is also undertaken and found to be at a significant level.

Chapter 5: A novel modification of standard reservoir computing is introduced that computes the feature vector of the reservoir state based on synaptic adaptation rather than neural activity. Compelling improvements in performance in sensory pattern recognition tasks are observed that achieve a lower testing error rate as well as lower variance between different random initializations of the reservoir model. The convergence is analyzed for the supervised readout training and the weight change brought about by synaptic plasticity. The results support our hypothesis that during plasticity, the network structure reaches a critical state that is sensitive to different classes of features in the input signal.

Chapter 6: Main contributions described in the thesis are summarized here. Further experiments are suggested that logically follow from the ones undertaken here or that can improve the models and methods used therein. Finally, we speculate on fruitful new directions of research that might go beyond many of the limitations faced by our methodology and by the field of applied computational neuroscience in general.
Chapter 2

Reservoir Computing with Synaptic Plasticity

2.1 Introduction

Understanding the principles behind the self-organization of biological nervous systems is the key to understanding cognition. Generally speaking, neural self-organization can be studied from the evolutionary and developmental perspectives. There were a number of major transitions or divergences in the evolution of nervous systems, for example, from the diffused nervous structure in cnidaria to the bilaterally symmetric one in flatworm [Ghysen, 2003]. Computational models have been built up for co-evolving the development of the neural system and body plan of an animate based on primitive organisms such as hydra and flatworm and the results suggest that energy efficiency might be the most important constraint in neural self-organization [Jin et al., 2009, Jones et al., 2008]. In addition, a strong coupling between the evolution of neural systems and body plan is also revealed [Jones et al., 2010, Schramm et al., 2012].

Meanwhile, increasing evidence has shown that adult brains undergo intensive rewiring [Holtmaat1 and Svoboda, 2009], which involves neural plasticity including the strengthening or weakening of existing connections, or even formation of new synapses and elimination existing ones. Seminal studies by Merzenich and Kaas [Merzenich et al., 1983, Merzenich et al., 1984] demonstrated that once sensory nerves are severed, the cortical maps to which they projected are
subsequently reorganized to accept nerves from surrounding nerves. This topographic adaptation can only be realised through neural plasticity and indicates the experience-dependent nature of plasticity and its central role in forming the basis of continual learning.

There has been a number of trend changes in the investigation of plasticity models. Initially, the focus was to provide a stable, self-regulated formulation of Hebbian learning [Hebb, 1949, Oja, 1982, Bienenstock et al., 1982]. Then, a shift towards spiking neural networks had models of plasticity emerge that depended on the precise timing of spikes between connected neurons [Song et al., 2000]. More recently, all of these models have been recognized as phenomenological approaches [Shouval et al., 2010], and more biological, molecular bases are being sought [Bush and Jin, 2012, Graupner and Brunel, 2012]. Also, neuro-modulators are being included in spike-timing models that add reinforcement capabilities on top of the purely associative [Izhikevich, 2007, Legenstein et al., 2008].

While the high level functions of neuroplasticity – learning and memory – are taken for granted, the suggested roles of plasticity in formally defined neural network models are varied and often contradictory. In some cases, simply applying models of plasticity to existing paradigms, such as reservoir computing, has yielded improved results [Steil, 2007, Schrauwen et al., 2008, Joshi and Triesch, 2009, Xue et al., 2013]. Other studies [Toyoizumi et al., 2005, Toyoizumi et al., 2007, Bohte and Mozer, 2007, Hennequin et al., 2010, Joshi and Triesch, 2009, Li et al., 2013] link the role of plasticity with increasing the mutual information in the signals between connected neurons. Some claim that Hebbian plasticity thus increases the correlation between neurons in a reservoir [van Rossum and Turrigiano, 2001], while others suggest that the neural activity is decorrelated and that this is, in fact, a desirable property [Jaeger, 2005, Babinec and Pospichal, 2007]. All of this is in addition to the classically proposed purpose of Hebbian learning as associative. Of course, there could be multiple roles that plasticity has to play in actual Human learning, each emerging in certain situations. Here we do not argue for one functional role in particular, but present a number of viewpoints.

The increasingly complex and self-regulated biological models of plasticity present a qualitatively different approach to the statistical optimization methods in machine learning. However, the success of these machine learning methods, particularly the recent advances made in deep
2.2. NEURAL CODE

In the beginning of artificial neural networks, neural activation simply consisted of a real number that was bounded by a neurons activation function such as $\tanh()$ that would limit the value by its asymptotes. This made sense from the biological side, as the real value was taken to approximate a real neurons firing rate, thus the name 'rate based coding'. As standard computers are optimized to process real values, this scheme is, of course, highly efficient. It can process information with high density in artificial networks.

Things become more complicated when using a model of a naturally inspired spiking neural network in computer simulations. Here the activation function is replaced by a membrane model that processes a sequence of action potentials – spikes – that cause excitation that leads to a further sequence of spikes. Here, information is encoded and processed through timing, instead
of a rate. If a rate interpretation of the spiking neural activity were adopted, then the temporal information in the spike sequence would be lost. This poses a problem for encoding and decoding real-valued information to and from learning models based on spiking neural networks.

Temporal coding schemes are methods of coding values into neural spikes, or sequences of them. For example, a simple scheme is known as 'time-to-first-spike'. In this method, a neuron's value is equal to the inverse of the time it takes to spike after a given input stimulus. That could be written as $\alpha = 1/\tau$ with $\alpha$ as the coded real value and $\tau$ as the time delay from the stimulus or some other event. Another example of a temporal coding scheme is 'rank-order' coding. This method uses the order of firing in a neural population to encode values which means there are $n!$ different combinations given $n$ neurons.

Both spike-time encoding schemes just described depend on a baseline from which to measure a specific spike by. This is sometimes referred to as a reference spike. The selection of which action potential to take as this reference spike is somewhat arbitrary.

In feed-forward networks in which each neuron need only be active once, these spike encoding schemes make sense and are practical. However, in recurrent networks, neural activity is continuous and oscillatory which means that the encoded value will drastically change depending on which time is selected as a reference spike.

One cannot rely on the regularity of recurrent network activity to periodically reset the reference spike to $t = 0$, as the oscillations vary throughout network activity, even when input stimuli is uniform. Figure 2.1 illustrates this problem. Oscillations in the activity start at about 52ms apart but then drift together and apart again. Almost all neurons spike in each oscillation (vertical stripe of dots). In order to apply either time-to-first-spike or rank-order coding methods, it would be necessary to accurately calculate the starting point of each oscillation as it occurs in order to provide a reliable reference baseline for the encoding or decoding processes.

We avoid these difficult problems by falling back on using rate coded encoding and decoding of real numbers into and from spike trains. It loses much temporal information, yet is reliably and simple to implement.
2.3. **Reservoir Computing Models**

Two forms of recurrent neural network are described in this section, both types of model of the reservoir computing paradigm. They take their inspiration from neural structures observed in the mammalian cortex. However, while biologically motivated, both are also designed to work algorithmically with machine learning principles on data classification and prediction tasks. We propose in this review, that these models are prime candidates for being augmented with neural plasticity models in order to improve their performance.

Reservoir computing [Jaeger, 2001, Maass et al., 2002] is a random projection paradigm in which a randomly connected recurrent neural network transforms an input signal into temporal, dynamic activity from which states are periodically ’read-out’ and used with standard regression to learn and predict supervised outputs. The reservoir computing framework is illustrated in Figure 2.2.

There are two main flavors of the reservoir computing paradigm.
Figure 2.2. Structural depiction of the Reservoir Computing architecture. The *reservoir* consists of a randomly connected set of artificial or spiking neurons. The *input vector* is fully or partially connected to a typically random subset of the reservoir nodes. The *state vector* consists of scalar values and has the same dimension as the number of neurons in the reservoir. If the reservoir consists of spiking neurons, the spike-train activity is typically transformed to scalar values to produce the state vector. The *readout* is a single layer perceptron, trained using gradient descent to produce a desired output.
2.4 Plasticity in Reservoir Computing

2.3.1 Echo State Networks

Echo State Networks (ESNs) [Jaeger, 2001] consist of artificial neurons, typically with sigmoid or tanh activation functions. There are no variable time delays at the neurons connections and the reservoir state is simply taken as the population of neuron activation values within a single time-step.

2.3.2 Liquid State Machines

Liquid State Machines (LSMs) [Maass et al., 2002] consist of spiking neurons, in which an excitable membrane is modelled and produces a sequence of timed spike activations in response to input stimuli. When taking the reservoir state, the spike sequences must be converted into scalar values, typically by using a low-pass filtering method. The connections are given varying delays to incorporate fast and longer-term dynamics to the recurrent activity. Synapses often use a dynamic model of transmission between neurons, that further increases the long-term dynamics of past activity in the network.

The potential for incorporating plasticity is greater for LSMs than it is for ESNs due to the former include information in activation timings between neurons as well as the strength of activation that ESNs rely on. Also, the neural and synaptic models are far richer, with more parameters to affect plasticity in activity-dependent ways.

2.4 Plasticity in Reservoir Computing

There has been much work to date on implementing plasticity in the reservoir computing framework. Both LSM and ESN models have received attention in terms of adapting the reservoir structure according to the input-driven neural activity. However, there have been many different, and often conflicting ideas about what role plasticity is designed to fulfil, and this is reflected in some contradicting results between implementations. The following sections provide an outline of the existing studies of plasticity in reservoir computing, introduced by their intended function. This will serve as a basis to identify the open questions for investigation to be addressed in later chapters.
2.4.1 Improving Reservoir Computing Performance

Reservoir computing, introduced in Section 2.3, does not rely on structural learning for its basic operation. However, recent studies have applied plasticity to 'shape' the reservoir and thereby improve performance in machine learning tasks such as prediction, regression and classification.

A large body of research work [Schrauwen et al., 2008, Joshi and Triesch, 2009, Xue et al., 2013, Babinec and Pospichal, 2007, Lazar et al., 2007, Yin et al., 2012a] shows an improvement in the reservoirs predictive performance when plasticity is active within the reservoir in the form of an unsupervised pre-training phase. It is thought this improves the reservoir characteristics such as the fading memory of the reverberating dynamics or scaling the spectral radius of the weight matrix closer to 1, leading to an activity regime balanced on the 'edge of chaos', sensitive to changes in an input signal.

In [Lazar et al., 2007], a k-winner-takes-all (kWTA) model of reservoir is shown to only improve when both IP and STDP are active together. When either plasticity rule is enabled on its own, there is a degeneration in the reservoir activity that prevents effective learning by the readout. IP alone leads to chaotic activity while STDP alone leads to time-locked spiking of initially activated neurons. This result may be an effect unique to using the kWTA model, however.

The ESN form of reservoir computing has most notably been applied to time-series prediction data. [Schrauwen et al., 2008, Lazar et al., 2007] are notable works that have applied IP and STDP to ESNs and that have shown improvement in regression to time-series prediction problems. NARMA and a prediction task based on Markov processes are used as the benchmark tasks. [Babiniec and Pospichal, 2007] applies anti-Oja plasticity to an ESN and shows an improvement in predictive performance on a time-series sun spot data set.

LSMs tend to be applied to classification of temporal sequence data, in contrast to prediction. [Xue et al., 2013, Yin et al., 2012a, Norton and Ventura, 2006] are works in which BCM and STDP have been used to improve results in temporal classification using LSMs. Notably, spoken digit and human behavior recognition are the applied learning tasks that consist of temporal samples in the form of sensory input. Having plasticity applied as a pre-training phase, or by having it continuously active at a slower rate than supervised learning, classification of temporal samples
2.4. PLASTICITY IN RESERVOIR COMPUTING

has been shown to improve.

2.4.2 Learning Input Structure and Coding

It has been demonstrated [Nowotny et al., 2003] that temporal information could be encoded spatially in a population of spiking neurons connected through STDP. Such a representation formed an auto-associative memory, where partial presentation of the temporal input signal triggered the activation of the whole spatial representation. This generalizes spatial auto-associative memories to spatio-temporal ones, with STDP taking the role of learning the temporal order of patterns through reinforcement of causal chains of activations.

Polychronous groups [Izhikevich, 2006] have been used to analyse [Paugam-Moisy et al., 2008] recurrently connected networks trained to classify temporal spike patterns. These groups are spatio-temporal structures which were found to develop under STDP and activated in response to particular classes of input. In this way, it is evident that these groups are input specific to a certain degree and can therefore be seen as a representation of the input patterns.

The structural development of recurrent networks has been studied [Clopath et al., 2010] in terms of receptive field formation. It was shown that using a voltage based model of STDP led to input specificity and structures forming that reflected the neural code. Feed-forward structure emerged temporal coded input while bi-directional structure emerged under rate coded input. This reinforces the hypothesis that the structure that develops under plasticity shares structure that is present in the input signal.

Under a number of STDP models, cell assemblies are shown to develop that can be reliably activated using either rate-based or temporal spike codes [Knoblauch and Hauser, 2011]. It is argued that a temporal code would be more energy efficient as it requires far fewer spikes. Such a temporal code presented in the previously mentioned work depends on synchronized inputs in order for neurons within a cell assembly to become synchronized.

2.4.3 Correlate or Decorrelate Neural Activity

An interesting contradictory area in the theoretical role of plasticity is whether it is desirable to increase or decrease the correlations of spikes in a population of neurons. The standard Hebbian
CHAPTER 2. RESERVOIR COMPUTING WITH SYNAPTIC PLASTICITY

interpretation of associative learning would have plasticity lead to an increase in correlation for any learned associations between neurons. Work in this vein has been modelled and it was shown that STDP would select pre-synaptic activity that had higher correlation [van Rossum and Turrigiano, 2001], thus would increase the overall level in a neural population. Conversely, it has been proposed [Joshi and Triesch, 2009, Jaeger, 2005, Babinec and Pospichal, 2007] that decorrelation is more desirable because it would maximize the information content in the network and also improve supervised learning methods. With this aim, a form of anti-Hebbian learning is employed. It is shown in [Babinec and Pospichal, 2007] that an anti-Oja rule leads to improved time-series prediction in a reservoir computing model.

2.4.4 Increasing Sparsity and Information Maximization

A computational model of neurons would have them perform some transformation on input patterns, such as in the case of reservoir computing mentioned above. In contrast to this view, some view neurons as signal carriers in which the maximal possible amount of information is transmitted between cells. It has been demonstrated that both BCM [Toyoizumi et al., 2005] and STDP [Toyoizumi et al., 2007] lead to maximal mutual information being retained between pre- and post-synaptic spike trains. Close to optimal transmission is shown to result from a triplet model of STDP, while the pair based model does show less improvement for information transmission [Hennequin et al., 2010]. A study has suggested that the purpose of STDP is to reduce the variance in a neurons response when presented with a specific spike input pattern, thus tuning selective sensitivity [Bohte and Mozer, 2007]. More recently, IP has been incorporated with a supervised learning scheme – the error-entropy minimization algorithm – as a cooperative information maximization mechanism that improves the performance on time-series prediction [Li et al., 2013].
2.5 Rate Based Plasticity Models

2.5.1 Types of Neural Plasticity

There are various types of neural plasticity that have been observed in experiments and thus also investigated in computational modelling. Generally speaking, neural plasticity can be divided into the following three types.

- **Synaptic plasticity.** Synaptic plasticity means the strengthening or weakening of the synapses that connect neurons and facilitate transmission of electro-chemical signals [Citri and Malenka, 2007]. The change in synaptic strength can be caused by a change in the abundance of neuro-transmitter molecules at the synapse, or by a rise or fall in conduction of post-synaptic receptors. Any models that directly modify the connection strength between neurons are examples of synaptic plasticity.

- **Intrinsic Plasticity.** Intrinsic plasticity denotes modification of a neuron’s intrinsic ability to generate or propagate action potentials [Brown and Randall, 2009]. This process is neuron wide and not synapse specific between two neurons. Intrinsic plasticity is often taken to self-regulate a neuron’s activity and to be involved in a kind of homeostatic mechanism to keep the activity within a practical range.

- **Homeostatic plasticity.** Modelling studies have demonstrated that changes in synaptic weight based on correlations between pre- and post- synaptic activity can be inherently unstable, as increases or decreases in synaptic weights cause increases or decreases in post-synaptic firing rate that subsequently generate further increases or decreases in synaptic weights in a positive feedback loop. Hence, homeostatic processes that regulate the total synaptic drive to a neuron and / or the long-term average firing rate of a neuron are critical for the stable operation of neural networks.

Two principal homeostatic mechanisms are synaptic scaling and meta-plasticity. The former refers to a uniform, cell-wide modulation of synaptic efficacies [Shepherd et al., 2006]; and the latter refers to a uniform, cell-wide modulation of the modification threshold for synaptic plasticity, each controlled by a long-term average of pre- or post- synaptic firing
rate [Abraham and Bear, 1996].

### 2.5.2 Hebb’s Postulate

An important landmark in the basis of associative learning in the brain came from Hebb [Hebb, 1949] in the form of a highly influential postulate. Essentially, it states that if cells are active at the same time, a connection will develop between them to facilitate further co-activation driven in a causal manner. In this way, coincident neural activity becomes associated activity and provides a mechanistic basis for the idea of associative learning that, until Hebb’s postulation, had existed largely as an abstract notion.

There are numerous quotes from [Hebb, 1949] that describe Hebb’s postulate. The following quote is chosen for being succinct:

> “Any two cells or systems of cells that are repeatedly active at the same time will tend to become 'associated', so that activity in one facilitates activity in the other.”

This can be formulated in the following equation:

$$\Delta w_i = \eta x_i y$$  \hspace{1cm} (2.1)

$\Delta w_i$, the change in synaptic strength, is increased in proportion to the product of the pre- ($x_i$) and post- ($y$) synaptic activity, multiplied by a learning rate, $\eta$. As can be seen from both Hebb’s postulate and the formula, connection strength can only increase due to there being no depression term. This leads to an untenable model in practice, as the synapses will continue to increase indefinitely. If maximum bounds are put on the synaptic strength, then the weights will saturate to the maximum values, at which point no further learning can take place.

### 2.5.3 Homeostatic Regulation

In early simulations incorporating Hebbian learning, it became apparent that a mechanism to reduce synaptic weight was needed to allow stability in the adaptation. von der Malsburg used a technique of synaptic scaling [Malsburg, 1973] to always keep the sum total of the synaptic weights connected to a post-synaptic neuron, constant. This way, any increases in synaptic
weight will be automatically balanced out with a decrease in the others. It is therefore ensured that further changes to all synapses are possible and always subject to the neural activity.

Synaptic re-normalisation equation from [Malsburg, 1973]:

\[
w_{ik} = w'_{ik} \cdot N \cdot \frac{w_{\text{avg}}}{w'_k} \tag{2.2}
\]

The normalized synaptic weight from \(i\) to \(k\), is \(w_{ik}\). The previous, un-normalized weight is \(w'_{ik}\) and the sum total of all weights to a post neuron \(k\) is \(w'_k\). The number of driving inputs is given as \(N\) while \(w_{\text{avg}}\) is the average weight value that all of the synapses take.

2.5.4 Self-Organising Networks

von der Malburg [Malsburg, 1973] was the first to show that Hebb’s plasticity could lead the network to functionally self-organize into stimuli sensitive ‘feature detectors’. The simulations showed parallel with findings from biological experiments – neighboring cells responding to similar stimuli [Hubel and Wiesel, 1962] (simple and complex cells in Hubel and Weisel’s early work).

This work demonstrated that neural networks would organize according to activity dependent synaptic modification in addition to the genetically determined connectivity that was previously assumed.

Willshaw and Malsburg [Willshaw and von der Malsburg, 1976] show how a continuous topographic map between a layer of pre- and post-synaptic cells emerges with Hebbian plasticity acting laterally between the cell sheets. It is argued that the concept of neural mapping goes beyond the feature detecting neuron, and has a system-matching property where symmetries in one layer are preserved in another. Topographic maps are known to be particularly important in brain areas between the retina and visual cortex in order to preserve spatial patterns in images. For ordered maps to develop between cell sheets of varying dimensions, it was demonstrated that a marker model is needed to establish the initial synaptic contacts [Willshaw and von der Malsburg, 1979]. However, later work [Erdi and Barna, 1984] proposes a model based only on activity dependent synaptic modifications that can form topographic maps. It uses noise induced transitions to determine the ordered connections.
Amari [Amari, 1983] formulates a complete mathematical description of a self-organizing neural field that treats the population of individual cells as a continuum, divided spatially into a finite set of parameters. A time step is also assumed so the framework is coarse-grained in both space and time. Amari shows analytically how pattern formation in the model has the ability to produce feature detectors for categorisation and topographic maps. Stability conditions for these patterns are also derived. When it comes to the stability of self-organizing neural systems, some works have gone beyond the synaptic modification models. A morphogenetic model of synaptogenesis is presented in [Dammasch et al., 1986], that just considers the growth and removal of synapses. A number of free elements are quantified for each cell that allows it to form new connections or those of a different strength. The elements are re-distributed through the network and therefore so are the connections. In this regulated algorithm, it leads to a form of homeostasis. The stability conditions required for ongoing operation are analysed in another work [Dammasch et al., 1988].

2.5.5 Anti-Hebbian Learning

An inverted form of Hebbian adaptation – anti-Hebbian plasticity – is proposed [Barlow and Foldiak, 1989] to be active in decorrelating sensory inputs in taste and vision, between laterally connected ganglion cells. This is to ensure that the output signals of these cells represent changes in the input signal in the most efficient manner.

Due to overlap in the cell sensitivities to the inputs, the initial outputs are correlated. After a period of anti-Hebbian training on the lateral connections between the cells, the variables are decorrelated. This is shown to produce a larger spread of output values for correlated inputs and therefore increases the sensitivity for small changes in the input which leads to a more efficient representation of the input variables.

The equation for anti-Hebbian adaptation takes the same form as Equation 2.1 above, but now with a negative learning rate.
2.5. RATE BASED PLASTICITY MODELS

2.5.6 Oja’s Rule

Oja’s rule [Oja, 1982] is a modification of the plain Hebbian interpretation that aims to address the stability problem of an exclusively potentiating mechanism. The rule is given in the following formula:

\[
\Delta w_i = \eta (x_i y - y^2 w_i)
\]  \hspace{1cm} (2.3)

The pre-synaptic weight, \( w_i \), is the synaptic strength from neuron \( i \). \( \eta \) is a small learning rate. \( x_i \) is the pre-synaptic input and \( y \) is the post-synaptic neurons output.

This formula is similar to Equation 1, except that it includes a depressive term, \( -y^2 w_i \). The depression is proportional to both the current weight of the connection strength and to the square of the post synaptic activity. The higher the weight and the resultant post-synaptic activity is driven, the greater this depressive term will be. This constitutes a self-regulating system that balances against the purely potentiating Hebbian term.

In [Oja, 1982], it is shown that when the rule is applied to the incoming synapses of a single neuron, the post-synaptic response extracts the principle component of the pre-synaptic input vector. This is after the rule has been applied long enough for the synapses to converge to a stable state where the average weight change across the inputs is zero.

2.5.7 BCM Theory

Another regulated form of Hebbian plasticity was proposed at the same time as Oja’s work. Beinenstock, Cooper and Munro [Bienenstock et al., 1982] took a similar approach to regulating the post-synaptic activity in their model (named BCM), but this time using a sliding threshold to determine whether the weight change should be positive or negative.

The equation for the BCM weight update is as follows:

\[
\Delta w_i = y(y - \theta_M)x_i - \epsilon w_i
\]  \hspace{1cm} (2.4)

Here, \( \theta_M \), the sliding threshold, is given as a temporal average firing rate of the post-synaptic neuron. This is given in the following formula:
CHAPTER 2. RESERVOIR COMPUTING WITH SYNAPTIC PLASTICITY

Figure 2.3. The Bienenstock-Cooper-Munro plasticity rule illustrated with synaptic weight change on the y-scale and post-synaptic activity on the x-scale. $\theta_M$ is the sliding modification threshold that changes based on a temporal average of post-synaptic activity.

$$\theta_M = E^P[(y/y_0)]$$  \hspace{1cm} (2.5)

$E^P[...]$ is some function of the neural activity that constitutes a temporal averaging. $y$ is the post-synaptic output and $y_0$ is a desired value that the post-synaptic output will be regulated to. The sliding threshold increases as the post-synaptic output exceeds the desired activity level. This causes the weight change to be negative, thereby providing regulation to the weight adjustment.

Like Oja’s rule, the weight change is also regulated based on the current value of the connection weight. In BCM this is done in a uniform decay of all weights using a small, subtractively applied learning rate, $\epsilon$.

In its introduction, BCM theory is claimed to explain neural selectivity observed in the receptive fields of the visual cortex. It was also claimed to provide a competitive mechanism in the context of binocular competition.

2.6 Spike Timing Based Plasticity Models

With an increased focus on spike based neural models that model a biologically more realistic excitable membrane, new interpretations of plasticity were required to make use of the new parameters that went beyond a simple rate code. This trend, combined with increasingly detailed
biological experiments, has led to a number of new directions in the development of plasticity models that we present in this section.

2.6.1 Spike Timing Dependent Plasticity

The previously described plasticity mechanisms all work on rate based neuron activation models, in which the activity level of a neuron is assumed to be its instantaneous rate of generating action potentials. This means they are not directly applicable to more realistic spiking models which account for precise timing of action potentials rather than the average rate of firing.

Experimental studies in a number of works [Bell et al., 1997, Bi and Poo, 1998, Zhang et al., 1998, Feldman, 2000, Wittenberg and Wang, 2006] have shown that the amount and direction of synaptic plasticity is dependent on both the order of pre- and post- synaptic spikes and the time delay between them. After repeated stimulation of brain cells in culture at different spike timings, the increase in post-synaptic potential is plotted against the delay between synaptic transmission and post-synaptic action potential. The resulting patterns from these experiments are reproduced in Figure 2.4, and constitute the observed ‘learning windows’ of spike timing dependent plasticity (STDP). Theoretically, these learning windows present a temporal interpretation of Hebbian learning in which causal co-activation is reinforced and anti-causal co-activation is diminished. There is no explicit self-regulation in plain STDP, yet stability can be achieved through the presence of depressive regions in the learning window.

The following subsections describe mathematical formulations of two commonly modelled forms of STDP.

Bi-Phasic STDP

The original formulation of STDP as a mathematical model is made in [Song et al., 2000]. It consists of two phases: a depressive phase in which pre- follows post-synaptic spike, and a potentiating phase in which post- follows pre-synaptic spike. In both phases, the weight change decreases in magnitude as the delay between spikes increases.

The formula is as follows:
Figure 2.4. Results from a collection of STDP protocol experiments. A variety of patterns are visible, even for studies with the same experimental protocol. Experiments are performed on brain slices from creatures of different species, and from different brain regions such as cortex and hippocampus. Data was extracted from plots in: (a),(b) [Bell et al., 1997] (c) [Bi and Poo, 1998] (d),(e) [Buchanan and Mellor, 2007] (f) [Campanac and Debanne, 2008] (g) [Feldman, 2000] (h) [Fino et al., 2005] (i) [Froemke and Dan, 2002] (j) [Froemke et al., 2006] (k) [Nishiyama et al., 2000] (l) [Wang et al., 2005] (m),(n),(o) [Wittenberg and Wang, 2006] (p) [Zhang et al., 1998].
2.6. SPIKE TIMING BASED PLASTICITY MODELS

\[
\Delta w_i = \begin{cases} 
A_+ \exp(\Delta t_i / \tau_+) & \text{if } \Delta t_i < 0 \\
-A_- \exp(-\Delta t_i / \tau_-) & \text{if } \Delta t_i > 0
\end{cases} 
\]  

(2.6)

\(A_+\) and \(A_-\) are the learning rates for the potentiation and depression, respectively. \(\Delta t_i\) is the delay of the post-synaptic spike occurring after the transmission of the pre-synaptic spike. \(\tau_+\) and \(\tau_-\) control the rates of the exponential decrease in plasticity across the learning window.

Tri-Phasic STDP

In light of further STDP protocol experiments in the CA3-CA1 regions in the hippocampus [Wittenberg and Wang, 2006], a new pattern for the learning window emerged. A tri-phasic rule is observed in this case, with a short potentiating region surrounded on either side by two depressive regions. This can be observed in a number of sub-plots in Figure 2.4 and is illustrated as a plotted formula in Figure 2.5.

One equation to describe the tri-phasic learning window is given in [Waddington et al., 2012]:

\[
\Delta w_i = A \left[ 1 - \left( \frac{(\Delta t_i - \alpha)^2}{\alpha^2} \right) \right] \exp \left( \frac{-|\Delta t_i - \alpha|}{\alpha} \right) 
\]  

(2.7)

Another is given in a recent comparative study of plasticity rules [Chrol-Cannon et al., 2012]:

\[
\Delta w_i = A_+ \exp \left( -\frac{(\Delta t_i - 15)^2}{200} \right) - A_- \exp \left( -\frac{(\Delta t_i - 15)^2}{2000} \right)
\]  

(2.8)

This learning window is visualized in Figure 2.5, with \(A_+ = 0.25\) and \(A_- = 0.1\). This formula simply consists of two Gaussian functions, one narrow additive curve, within another wider subtractive Gaussian of a lower magnitude. The parameters were chosen to generally match the values observed in the data in Figure 2.4.

Reward-Modulated STDP

Plain, unsupervised STDP has been questioned as a plausible mechanism for meaningful learning to occur. Legenstein [Legenstein et al., 2008] proposes that a reward modulated form of STDP (RM-STDP) can provide a tenable model for supervised learning while maintaining a method
Figure 2.5. The two predominantly studied STDP learning windows. The bi-phasic rule is plotted from the formula in [Song et al., 2000] and the tri-phasic rule plotted from the formula in [Chrol-Cannon et al., 2012]. These curves have been derived by fitting formulas to the experimental data produced from the STDP protocol. The bi-phasic rule roughly matches the observed pattern in Figure 2.4c, and the tri-phasic rule roughly matches Figure 2.4g.

Reservations for Pure STDP

As a phenomenological model, STDP is particularly vulnerable to criticisms from biologically oriented studies. In particular, it has been suggested in [Lisman and Spruston, 2005] that contrary to a spike-timing model of plasticity, a post-synaptic spike is not required for a change in synaptic efficacy. The study goes further, to present experimental evidence that the back-propagating signals that are implicitly required by STDP, are neither necessary nor sufficient for plasticity in vivo. A follow-up work [Lisman and Spruston, 2010], reinforces this position by proposing that the protocol followed in STDP experiments is artificial. That is, the post-synaptic current injection leads to a phenomenon that is not observed when the Excitatory Post-Synaptic Potential (EPSP) causes a spike, as is the norm.
The current fixation that plasticity models have on the synapse has also recently been called into question in [Debanne and Poo, 2010]. There, it is brought to attention that many experiments show that the intrinsic excitability of neurons change, often in accordance with the synaptic efficacy. At the extremes of potentiation/depression, the relation is observed to reverse, with a decrease/increase in excitability, respectively.

It is easy to get immersed in precise formulations of plasticity that are either derived from a class of biological experiment or inspired by a cybernetic principle of self-organization and self-regulation. Of course, there is no guarantee that the selected models for plasticity have much relevance to the processes interacting in a living brain.

In a poster report on experimental findings [Graupner and Ostojic, 2013], it is stated that when under conditions of irregular, natural spiking patterns, popular models of plasticity – including STDP, voltage-dependent plasticity and a calcium controlled model – all have less influence than previously assumed. Irregular spiking reduces the level of potentiation and depression in all tested models. Furthermore, the calcium model becomes insensitive to spike correlations when there is a high average firing rate.

### 2.6.2 Voltage-Dependent Plasticity

Rather than updating a synapses weight based on the pre- and post- synaptic timing of action-potentials, a recent form of plasticity model uses the instantaneous voltage-level of the pre- and post-synaptic membrane [Clopath et al., 2010]. The following equation shows this model that shares much of its form with the equation for bi-phasic STDP:

\[
\Delta w_i = -A_{LTD} x_i [\bar{u} - \bar{\theta}] + A_{LTP} \bar{x}_i [u - \theta] [\bar{u} - \bar{\theta}]
\] (2.9)

In this formulation; \( x \) is the pre-synaptic spike train, \( u \) is the post-synaptic voltage, and \( \theta \) is a threshold to be exceeded for synaptic weight change to occur. Parameters \( \bar{x}, \bar{u} \) and \( \bar{\theta} \) are low-pass filtered values of the previous three parameters. \( A_{LTD} \) and \( A_{LTP} \) are learning rates for long-term depression and potentiation, respectively.

The notable aspect of this rule is that an action-potential is no longer required to trigger a change in synaptic efficacy, as has been observed in experiment. Under different stimulation
regimes, it has been shown to fit both bi-phasic STDP and BCM learning patterns. The introductory paper used the model to explore cell selectivity and receptive field emergence that also corresponds to experimental findings.

2.6.3 Calcium Controlled Plasticity

STDP, as a class of models, does not consist of underlying molecular mechanisms formalized in biological terms. In fact, some of the implicit assumptions of STDP have been called into question, as discussed later in this section.

Shouval has commented that the simple viewpoint of STDP neglects that actual mechanisms that modify synapses [Shouval et al., 2010]. He points out “that synaptic plasticity is induced by a variety of receptor-generated second messengers, which in turn activate kinases, phosphatases, and other downstream targets”. By assuming that the essential character of plasticity can be abstracted away from the biology, STDP will miss these fundamental mechanisms that may prove essential to learning and memory ability.

A new class of plasticity model that is governed by calcium concentration at the synapse is emerging as a possible underlying mechanism that is compatible with some of the empirical observations in STDP experiments. Two recent models are defined in [Bush and Jin, 2012] and [Graupner and Brunel, 2012]. The former has a more explicit biological grounding in that calcium concentration is regulated through kinase and phosphatase channels. The latter forgoes this detail with the benefit of having a simpler model to implement and analyse. Both models emphasize their ability in reproducing a set of commonly observed STDP learning windows. Figure 2.6 shows one such set produced by the model in [Graupner and Brunel, 2012].

2.6.4 Gene-Regulated Plasticity

Genetically driven neural development has been computationally modelled in a number of works [Kitano, 1995, Van Ooyen, 2003, van Ooyen, 2011] that demonstrate the activity-independent organization of neural networks. Of course, even after the development of the brain, the genetic factors will remain to also affect plasticity in response to activity driven from the environment.

It is becoming clear through neurobiology studies [Flavell and Greenberg, 2008] that neural
Figure 2.6. A variety of STDP learning windows can be predicted/explained by a unified plasticity model based on calcium concentration. If one look carefully at the experimental data presented in Figure 2.4, corresponding patterns can be observed for each of the learning windows. The calcium control model is [Graupner and Brunel, 2012] and these sub-plots are reproduced from there.
activity, plasticity and gene-regulatory networks are interlinked in a complex system of adaptation and regulation. A commentary [Le Novere, 2007] laments the lack of computational modelling that includes all the facets of neuronal systems biology. Some work has been done on analyzing gene-regulated neural development [Benuskova and Kasabov, 2008, Benuskova et al., 2006], but it has yet to be applied to learning models for data driven tasks, instead being used to approximate synthetic spectral patterns.

Gene regulated neural plasticity in spiking neural networks have been applied to machine learning tasks. An evolutionary approach to incorporating plasticity in reservoir computing is presented in [Yin et al., 2012a]. Here, a gene regulatory network (GRN) is evolved that adapts the parameters in BCM plasticity that is then applied to the reservoir. In speech and human behavior recognition tasks, this form of adaptation is shown to significantly improve classification accuracy, as well as improving regression on a time-series benchmark. The evolving process of the GRN-BCM reservoir model is illustrated in Figure 2.7. A similar GRN-regulated BCM rule has also been applied to adapt the weights of a feed-forward spiking neural network [Meng et al., 2011].
The models previously described in this literature review each consist of a relatively simplified set of formulae when compared to the overwhelming complexity found in the biological literature. Currently, calcium control theory provides one of the most comprehensive plasticity models from the systems biological perspective. In [Bush and Jin, 2012], calcium neuro-transmitter concentration is regulated by kinase and phosphatase channels which provides specific molecular mechanisms for a biological model of plasticity. However, there are many more ion channels, neuro-transmitter and neuro-modulating proteins that are not accounted for [Cortes-Mendoza et al., 2013]. Furthermore, the complex and varying network of interactions between these elements is not even well understood in the biological literature.

While these approaches have the advantages of faster computability and easier comprehensibility, the diversity and variation lost from the complete experimental accounts may prove to be a weakness at the same time. The high degree of complexity and diversity in gene regulatory networks may lead to much higher tunability and robustness in the resulting epi-genetic interactions [Hasty et al., 2001, Geard and Willadsen, 2009], thus leading to unified models of neural plasticity that account for synaptic, intrinsic and homeostatic plasticity [Bush and Jin, 2011].

2.7 Discussion

There is a current divide in the spiking network and plasticity literature. On one side, the neuroscientific literature analyses the structural adaptation and learning of networks under the influence of plasticity [Clopath et al., 2010, Gilson et al., 2009a, Gilson et al., 2009b, Gilson et al., 2009c, Gilson et al., 2009d]. In these, typically, a hierarchical network inspired by the layered structure and connectivity of the cortex is shown to develop input specific receptive fields that correspond to audio or visual stimuli. However, little/no application of these methods are made to machine learning algorithms. On the other side, some computational studies have been reported that do show that plasticity can improve applied neural network methods [Xue et al., 2013, Babiniec and Pospichal, 2007, Yin et al., 2012a]. However, these works tend to have an extremely limited analysis of the input-specific structural learning that takes place within the networks. It is very unclear how or why plasticity works in these circumstances.

For a complete and coherent understanding of neuro-plasticity, a precise formalization of how
structural adaptation contributes to specific learning tasks is required. This may be possible by applying the analysis of receptive field emergence to applied neural networks that use regression learning. However, in the process, the current network models used in machine learning tasks, like reservoirs, will have to be modified to allow the abstract, numerical feature-vector data sets to be converted into the sensory field format of input stimulus in neuroscience literature.

Different existing studies argue for and report different functions for synaptic plasticity in reservoir computing models. We intend to tackle this problem with a comprehensive comparison of reservoir computing metrics that reflect the main features of reservoir networks and show how plasticity impacts them. So far in the literature, cited earlier in this chapter, studies choose to employ only the metrics that agree with their particular assumptions of the function of plasticity. An empirical comparison of all the main widely used reservoir metrics is needed in order to gain an indication of which ideas are closer to reality.

Another open question raised in this literature review is whether plasticity learns input- or task-specific features in the adapted network structure or whether it just improves the general properties of the reservoir. If the latter is true, this might be evaluated by some metrics that will be introduced in the next chapter. We also intend to undertake an empirical study in Chapter 4 to answer this question: Does unsupervised plasticity adapt input-specific structure within a reservoir computing model? This is a question unanswered thus far.

Using the findings from the above two investigations, we intend to find an effective method of employing plasticity to improve the performance over a static reservoir structure.

2.8 Summary

This chapter presented the prominent computational models of plasticity and their current applicability to empirical improvements as neural network adaptation mechanisms. The existing examples of plasticity in use tend to apply it to randomly connected recurrent reservoirs to learn the structural information in the inputs, achieve sparsity in neural connectivity and enhance learning performance.

In the next chapter we implement neural plasticity in an LSM model to determine if pattern recognition performance is enhanced for temporal input stimuli. We hypothesize that pattern
recognition ability for a neural network will improve if the connections in the network adapt a structure that is based on the temporal statistics of the input. This is what plasticity is widely considered to do, as has been explored in this chapter.

In the coming experiments, emphasis will be given to evaluate how plasticity impacts a number of information theoretic network metrics, in addition to performance, in order to better establish the functional role that plasticity models have in the context of being applied to the reservoir computing framework. We hope that this will shed some light on the open questions reviewed in this chapter, particularly Section 2.4.
CHAPTER 2. RESERVOIR COMPUTING WITH SYNAPTIC PLASTICITY
Chapter 3

Reservoir Computing Metrics

Reservoir computing provides a simpler paradigm of training recurrent networks by initializing and adapting the recurrent connections separately to a supervised linear readout. This creates a problem, though. As the recurrent weights and topology are now separated from adapting to the task, there is a burden on the reservoir designer to construct an effective network that happens to produce state vectors that can be mapped linearly into the desired outputs. Guidance in forming a reservoir can be through the use of some established metrics which link a number of theoretical properties of the reservoir computing paradigm to quantitative measures that can be used to evaluate the effectiveness of a given design. We provide a comprehensive empirical study of four metrics; class separation, kernel quality, Lyapunov’s exponent and spectral radius. These metrics are each compared over a number of repeated runs, for different reservoir computing set-ups that include three types of network topology and three mechanisms of weight adaptation through synaptic plasticity. Each combination of these methods is tested on two time-series classification problems. We find that the two metrics that correlate most strongly with the classification performance are Lyapunov’s exponent and kernel quality. It is also evident in the comparisons that these two metrics both measure a similar property of the reservoir dynamics. We also find that class separation and spectral radius are both less reliable and less effective in predicting performance.
CHAPTER 3. RESERVOIR COMPUTING METRICS

3.1 Introduction

Reservoir computing has become a successfully applied recurrent neural network paradigm [Lukosevicius and Jaeger, 2009, Lukosevicius et al., 2012]. It was initially introduced from both biologically inspired [Maass et al., 2002] and signal processing [Jaeger, 2001] groundings, and has since been applied successfully to real-world time-series pattern recognition problems [Yin et al., 2012a, Xue et al., 2013].

While the reservoir method has simplified the training of recurrent networks, the visibility into the workings of the internal computation remain largely opaque. In fact, we suggest that reservoir computing is more of a black-box than traditional feed-forward networks, because of the inability to trace clear paths from input features to internal nodes due to the highly recurrent connections.

The difficulty in functionally analyzing reservoir networks has stifled attempts to improve the model parameters. Incorporating synaptic plasticity to adapt reservoir weights has been attempted [Jaeger, 2005, Norton and Ventura, 2006] and sometimes lead to improvements in performance [Joshi and Triesch, 2009, Yin et al., 2012a, Xue et al., 2013]. However, the principles by which plasticity improves the parameters are not understood. Reservoir adaptation is still essentially a trial and error affair.

Some metrics for measuring reservoir characteristics have been put forward. These tend to center around the concepts of separation [Goodman and Ventura, 2006], edge-of-chaos criticality [Bertschinger and Natschläger, 2004], and fading memory [Jaeger, 2002].

In this study, we will compare a selection of reservoir metrics on two time-series classification tasks, comparing the consistency between them. The stability of each metric will be studied by running each experiment over 10 random initializations. A comparison will also be made of how three widely used plasticity rules and three initial connectivity structures affect each of the metrics. Finally, we look at the correlation between the metrics and classification accuracy to determine the extent that the metrics can be used to indicate performance.

These empirical comparisons will provide experimental guidance to complement the theoretical claims made for these measures.
3.2 Reservoir Metrics

The metrics studied in this chapter serve a variety of purposes in their measurements. Two that we have selected; 'rank separation' and 'Lyapunov’s exponent' are to quantify the information content of the neural states in the reservoir. Information content is important as a higher quantity indicates greater computational power of the neural network. The 'spectral radius' of the weight matrix is used as a proxy measure for stability in temporal neural activity. Stability has an important role to play in recurrent networks to ensure that activity caused by past input patterns slowly fade over time as new stimuli are presented to the network. Pattern separability of the neural states is determined by a geometric 'class separation' metric on the reservoir state vectors. Separation is particularly useful to determine how well patterns can be distinguished from each other in classification tasks. Each of these metrics is now described in detail.

3.2.1 Class Separation

The class separation is a measure of the comparative distance between the reservoir states corresponding to different classes of stimuli. It was first introduced in [Goodman and Ventura, 2006] and further expanded in [Norton and Ventura, 2010] as a way to determine how well a reservoir can distinguish one class of inputs from another based on the geometric distance between the class centroids. The reservoir states are taken to define the multi-dimensional coordinates of each sample. Class separation is defined in Equations 3.1, 3.2, and 3.3 as follows:

\[
Sep_\psi(O(t)) = \frac{C_d(t)}{C_v(t) + 1} 
\]  

\[
C_d(t) = \sum_{l=1}^{n} \sum_{m=1}^{n} \frac{\| \mu(O_l(t)) - \mu(O_m(t)) \|_2}{n^2} 
\]  

\[
C_v(t) = \frac{1}{n} \sum_{l=1}^{n} var(O_l(t)) 
\]

The class separation \( Sep(\ldots) \), for a given reservoir \( \psi \) and set of state vectors \( O(t) \), is defined in Equation 3.1. It is the inter-class distance divided by the intra-class variance, with 1 added to the denominator to prevent dividing by zero. Inter-class distance is calculated according to
CHAPTER 3. RESERVOIR COMPUTING METRICS

Equation 3.2. The class centroids are calculated as the mean state vector for a given class, denoted by $\mu(O(t))$. There are $n$ classes in total. Intra-class variance is calculated according to Equation 3.3. The within-class variance is given as $\text{var}(O(t))$. It is calculated by summing the geometric distance between each state vector and its corresponding class average, $\mu(O(t))$.

The rationale behind this class separation measure is that if the distance between different classes of inputs is higher than the distance within the classes, it will be easier for a linear readout to learn a set of weights that distinguishes between the reservoir states of different classes.

3.2.2 Kernel Quality

The kernel quality, introduced in [Legenstein and Maass, 2007], is a class-agnostic measure of the reservoir’s ability to separate input patterns, in so far as it is independent of the target output. However, it is not quite a task independent measure of a reservoir, due to the dependence of the task-specific input patterns in forming the reservoir states. Like class separation, kernel quality is based on the complete set of $n$ reservoir states produced by input stimuli. Here, a matrix $M$ is formed from all of the collected reservoir state vectors, each of which forms one column of dimension $m$. The rank $r$ of $M$ is then taken to be a measure of the computational power of the reservoir, with the maximum rank, and highest computational power to be $r = m$, assuming that the number of state vectors is greater than the dimension, $n > m$. When this is the case, each column in $M$ cannot be computed from a linear combination of any other column and therefore it is possible for a linear readout to separate each one of the reservoir states to produce different outputs. Equation 3.4 states the rank of matrix $M$ as the smallest integer $k$ that satisfies a decomposition of $M_{mn}$ into two matrices of dimensions $m \times k$ and $k \times n$.

$$\text{rank}(M) = \min\{k\} \quad \text{s.t.} \quad M_{mn} = C_{mk}R_{kn}$$ (3.4)

This measure is also referred to as the linear separation property of a reservoir.

3.2.3 Lyapunov’s Exponent

Lyapunov’s exponent estimate is a method of calculating the amount of chaos in the dynamics of the reservoir activity. The principle is based on the assumption that internal activity, $x_j(t)$,
3.2. RESERVOIR METRICS

that is generated based on the input signal, \( u_j(t) \), should vary in accordance with that signal, in a system with orderly dynamics. We use the calculation method defined in [Gibbons, 2010] which was formulated based on theory described in [Bertschinger and Natschläger, 2004]. This method is defined in Equation 3.5. It is scaled by an undetermined constant \( k \) and so can be taken as proportional to the Lyapunov exponent. Therefore it can be compared only to other values using this method, not to other studies, unless the constant \( k \) were determined for both.

\[
\lambda(t) = k \sum_{n=1}^{N} \ln \left( \frac{\| x_j(t) - \hat{x}_j(t) \|}{\| u_j(t) - \hat{u}_j(t) \|} \right)
\] (3.5)

In this formulation, \( j \) and \( \hat{j} \) refer to consecutive input patterns of \( u \) and the corresponding reservoir states \( x \). The total number of samples in the sequence of inputs is \( N \).

3.2.4 Spectral Radius

The spectral radius [Jaeger, 2001] is a measure taken directly on the weight matrix of the reservoir, rather than the reservoir states as the others are. It is the largest absolute eigenvalue of the weight matrix that indicates the scale of the weight values. Having a spectral radius less than 1 implies that input driven activity will fade within the network over time. By having a spectral radius exceeding 1, the reservoir dynamics would reach an unstable regime where the activity continually perpetuates and interferes with future inputs. It is therefore suggested that this value be kept below 1, while being a high as possible to allow time-series samples to interact in sufficiently long time-scales. However, the concept of spectral radius assumes that the node activation has a unity output function. It is not clear what implications this metric has with spiking neuron models with connection delays. Equation 3.6 gives the spectral radius, \( \rho(\cdot) \) of matrix \( M \), as the largest absolute eigenvalue in \( \lambda_n \).

\[
\rho(M) = \max\{|\lambda_1|, \ldots, |\lambda_n|\}
\] (3.6)
3.3 Methods

3.3.1 Reservoir Network

The reservoir model that we use is illustrated in Figure 3.1. The reservoir nodes, indicated by $L$, are stimulated by the inputs directly as injected current, $I$, into the membrane potential modelled with Izhikevich’s simple model [Izhikevich, 2003]. The real-valued inputs are normalized between 0 and 1, which are multiplied by a scaling factor of 20 before being injected as current into $L$. Input connections number $0.2 \times \text{reservoir size}$, projected randomly to the reservoir nodes. The reservoir dynamics are then simulated for 150ms. Then, the resulting spike trains produced by each of the reservoir nodes is passed through a low-pass filter, $f$, to produce a real valued vector used to train a linear readout.

\[ f \rightarrow \sum w_i x_i \]

**Figure 3.1.** Depiction of the elements of our reservoir computing model. $I$ is a multi-dimensional input signal, $L$ nodes constitute the recurrent reservoir, the $x$ vector is the reservoir state, $f$ is the filtering of the spike trains and $y$ is the output after weight and sum.

Our reservoir consists of 135 spiking neurons with the ratio of excitatory to inhibitory as 4:1. Neurons are connected with static synapses (delta impulse function), according to connectivity described in the following subsection. weights are drawn from two Gaussian distributions; $\mathcal{N}(6, 0.5)$ for excitatory and $\mathcal{N}(-5, 0.5)$ for inhibitory. When plasticity adapts the reservoir weights, $w_{max}$ is clamped at 10 and $w_{min}$ at -10. All parameters for excitatory and inhibitory neuron membranes are taken from [Izhikevich, 2003].

To generate an output, the spike train from each reservoir node is low-pass filtered and a
weight-and-sum readout is applied according to the methods in [Maass et al., 2002]. This output is trained with the iterative, stochastic gradient descent method: Least Mean Squares, given in Equation 3.7.

\[ w_i \leftarrow w_i + \mu (d - y)x_i. \]  

Here, \( d \) is the desired output, \( y \) is the actual output, \( x_i \) is the input taken from a neuron’s filtered state, and \( \mu \) is a small learning rate. The weight from \( x_i \) to the output is \( w_i \). We find experimentally that setting initial values of \( w_i \) to 0 leads to more stable and consistent learning convergence than small random values.

### 3.3.2 Connectivity

The type of connectivity used determines the topology of the recurrent network structure in the reservoir. As the synaptic plasticity models used in this work only modify the weights, not the topology, different connectivities will maintain their characteristic structures throughout the simulations. The following connection models are used to probabilistically connect reservoir nodes:

1. **Uniform random:** The probability for any two neurons to be connected is a fixed value \( C \). To add a new connection, source and target neurons are both selected randomly with a uniform distribution. This leads to an Erdős-Rényi type network structure [Erdős and Rényi, 1959]. An illustration is provided in Figure 3.2.

2. **Scale-free:** In a network with the scale-free property, the degree distribution – the number of connections for each node – follows a power law: \( P(k) \sim Ck^{-\gamma} \) [Barabasi and Albert, 1999]. The probability \( P \) of a node having \( k \) connections, is scaled by some constant \( C \). For a growth model when adding new connections, we use the Barabasi-Albert model. Subscript \( i \) indicates connections to the node in question and subscript \( j \) indicates connections to all other nodes.

\[ p_i = C \frac{k_i}{\sum_j k_j} \]  

(3.8)
This leads to a structure with densely connected hubs. An illustration is provided in Figure 3.2.

3. **Distance based lattice:** The original model for LSM connectivity [Maass et al., 2002] arranged neurons in a 3D grid with the probability of a connection between two nodes, inversely proportional to the distance between them. The formula defining the probability of a connection between two neurons is as follows:

\[
p_{ab} = C \cdot \exp \left( -\frac{D^2(a,b)}{\lambda^2} \right)
\]

(3.9)

Where \(D(a,b)\) is the Euclidean distance between neurons \(a\) and \(b\). The parameter \(\lambda\) controls both the average number of connections and the average distance with which neurons are connected by.

![Figure 3.2](image.png)

**Figure 3.2.** Illustration of two types of connectivity model. A uniform connection policy produces variable length chains of connections with some groups of neurons disconnected from others. A scale-free connection policy leads to a structure of a few highly connected hubs and many sparsely connected leaves.

### 3.3.3 Plasticity

Three synaptic plasticity mechanisms are employed in this study, each of them based on the Hebbian postulate [Hebb, 1949] of “neurons that fire together, wire together”. The BCM rule regulates the spike rate of the post-synaptic neuron to match a desired rate of spiking. Spike
3.3. METHODS

Timing dependent plasticity (STDP) is also utilized with two forms of learning window that have been observed in biological experiments. Each mechanism is outlined as follows:

1. BCM rule: The BCM rule [Bienenstock et al., 1982] is a rate based Hebbian rule that also regulates the post-neuron firing rate to a desired level. It works on a temporal average of pre- and post-synaptic activity. The BCM rule is given in Equation 3.10. The regulating parameter is the dynamic threshold $\Theta_M$, which changes based on the post-synaptic activity $y$ and the desired level $y_0$ in the following relationship: $E[y/y_0]$, where $E[\cdot]$ denotes a temporal average. There is also a decay parameter $\epsilon w_i$ for additional stability, that slowly reduces connection strength and so provides a mechanism for uniform weight decay, irrespective of the level of activity or correlation. Subscript $i$ indexes the pre-synaptic activity and weights, $x$ and $w$, according to the pre-synaptic neurons. A plot of the BCM weight change is presented in Figure 2.3 in Chapter 2.

$$\frac{dw_i}{dt} = y(y - \theta_M)x_i - \epsilon w_i \quad (3.10)$$

2. Bi-phasic STDP: The STDP rule depends on the temporal correlation between pre- and post-synaptic spikes. The synaptic weight change is computed based on the delay between the firing times of the pre- and post-neuron. This is described in a fixed 'learning window' in which the y-axis is the level of weight change and the x-axis is the time delay between a pre- and post-synaptic spike occurrence. The bi-phasic STDP rule consists of two decaying exponential curves [Song et al., 2000], a positive one to potentiate in-order spikes, and a negative one to depress out-of-order spikes. This rule was derived from experimental work carried out on populations of neurons in vitro [Markram et al., 1997] [Bi and Poo, 1998]. Bi-phasic STDP is given in Equation 3.11.

$$\Delta w_i = \begin{cases} A_+ \exp(\Delta t_i / \tau_+) & \text{if } \Delta t_i < 0 \\ -A_- \exp(-\Delta t_i / \tau_-) & \text{if } \Delta t_i > 0 \end{cases} \quad (3.11)$$

$A_+$ and $A_-$ are the learning rates for the potentiation and depression, respectively. $\Delta t_i$ is the delay of the post-synaptic spike occurring after the transmission of the pre-synaptic spike. $\tau_+$ and $\tau_-$ control the rates of the exponential decrease in plasticity across the
learning window. For our experiments the learning window is symmetric with $A_+ = A_- = 0.15$ and $\tau_+ = \tau_- = 20\, ms$.

3. **Tri-phasic STDP**: A tri-phasic STDP learning window consists of a narrow potentiating region for closely correlated activity but depressing regions on either side: for recently uncorrelated activity, and for correlated but late activity. This learning window has been observed *in vitro*, most notably in the hippocampi, between areas CA3 and CA1 [Wittenberg and Wang, 2006]. The tri-phasic STDP is given in Equation 3.12 from [Chrol-Cannon et al., 2012].

$$\Delta w_i = A_+ \exp \left( -\frac{(\Delta t_i - 15)^2}{200} \right) - A_- \exp \left( -\frac{(\Delta t_i - 15)^2}{2000} \right)$$

(3.12)

The learning rates are set as $A_+ = 0.25$ and $A_- = 0.1$. Both STDP learning windows are plotted in Figure 2.5 in Chapter 2.

### 3.3.4 Time Series Tasks

1. **Tri-function generator**: A synthetic benchmark is taken from a study performed by Jaeger on ESNs [Jaeger, 2007]. The task is to predict which of three signal generating functions is currently active in producing a varying input signal. To generate a sample of the signal at a given timestep, one of the three following function types is used; 1) A sine function of a randomly selected period, 2) A chaotic iterated tent map, 3) A randomly chosen constant. The generator is given some low probability, 0.05, of switching to another function at each time-step. The full method of generating the data is described in [Jaeger, 2007]. Part of the generated signal is presented in Figure 3.3.

2. **Speaker recognition**: A speaker recognition task is a classification problem dealing with mapping time-series audio input data to target speaker labels. We use a data set taken from [Kudo et al., 1999] which consists of utterances of 9 male Japanese speakers pronouncing the vowel /ae/. The task is to correctly discriminate each speaker based on the speech samples. Each sample is comprised of a sequence of 12 feature audio frames. The features of each frame are the LPC cepstrum coefficients. The sample sequence ranges between 7-29 frames. The dataset is divided into training and testing sets of 270 and 370
samples each, respectively. Note that unlike the benchmark data used in this chapter, the samples are not in a time-series, yet each sample consists of a time-series of audio frames.

Figure 3.3. Plot of 500 of the 50,000 data samples generated according to Jaeger’s time-series benchmark [Jaeger, 2007].

3.4 Results

The results are divided into two parts. The first part varies the reservoir connectivity and adaptation mechanism to show the effect this has on each of the metrics and classification performance. The second part shows how the metrics correlate with performance by plotting the quantities against each other and calculating Pearson’s correlation.

3.4.1 Effect of Plasticity and Connectivity on Metrics

Each of the figures in this section are box and whisker diagrams in which each box represents 10 randomly initialized simulation runs with a given parameter set indicated by the x-axis labels. The metric/performance is indicated on the y-axis.

Performance: Figure 3.4 shows the results for performance, specifically the classification accuracy for each time-series task. In terms of reservoir adaptation, the Bienenstock, Cooper, Munro (BCM) rule produces slightly better results on the benchmark task while a static reservoir performs better on the speaker task. In both cases, tri-phasic STDP performs worse and has
more variable results. For connectivity there is no significant trend, with uniform random connection performing somewhat better than the other two for the speaker task and somewhat worse for the benchmark.

**Figure 3.4.** Classification accuracy results for 10 initializations for each combination of plasticity rule, connectivity method and time-series task.

**Figure 3.5.** Class separation results for 10 initializations for each combination of plasticity rule, connectivity method and time-series task.

**Class separation:** Figure 3.5 shows the results for the class separation metric. Considering a higher class separation leads to better chances of learning, in theory, the tri-phasic STDP plasticity rule tends to give slightly better values for the metric. However, this form of plasticity
3.4. RESULTS

is also the least stable and sensitive to initial connection/weight values, as can be seen by the significantly larger box size. The separation results vary drastically between the 2 time-series tasks tested. The speaker recognition task has much greater stability, indicated by smaller box size, apart from with the tri-phasic rule. Also interesting to note is that the class separation is higher on the speaker task than on the artificial benchmark data even though it has 9 classes compared with 3.

Figure 3.6. Kernel quality results for 10 initializations for each combination of plasticity rule, connectivity method and time-series task.

Kernel quality: Figure 3.6 shows the results for kernel quality. Kernel quality is also a measure to be maximized, with the greatest value in both tasks being 135, the dimension of the reservoir. Bi-phasic STDP and a static reservoir tend to give the best results for this measure. Tri-phasic STDP gives significantly lower, the opposite trend compared to class separation. Connectivity does not have a large effect, except scale-free producing better results for the benchmark task. The speaker recognition task again benefits from better values for this metric.

Lyapunov’s exponent: Figure 3.7 shows the results for Lyapunov’s exponent estimate. According to idea of desiring self-organized criticality, a value approaching 1, that represents the edge-of-chaos is ideal. Due to the dimension of the reservoir state, the results have been scaled by 135. Therefore 135 is the target value for these results. Strikingly, Lyapunov’s exponent results follow kernel quality very closely. The relationship between them is almost exactly the same for the different reservoir settings which suggests that both play a similar role in estimating
Figure 3.7. Lyapunov exponent estimate results for 10 initialisations for each combination of plasticity rule, connectivity method and time-series task.

how rich the dynamics are in terms of computational transformation of the input.

Figure 3.8. Spectral radius results for 10 initializations for each combination of plasticity rule, connectivity method and time-series task.

Spectral radius: Figure 3.8 shows the results for the spectral radius. Again, according to edge-of-chaos recurrent activity, this value is ideally approaching 1, at least claimed when dealing with non-spiking reservoirs [Verstraeten et al., 2007]. Greater values than 1 will lead to instability of a supervised readout, while low values will lead to low computational power. The BCM rule consistently adapts the weights to give spectral radius values less than, but
approaching 1. The other settings all lead to significantly higher values. Tri-phasic STDP always leads to weight matrices that are invalid for use with eigenvector detecting methods. This is also occasionally true with other plasticity rules when using scale-free connectivity. The numerical procedures to detect eigenvectors are approximate methods and not guaranteed to work with any arbitrary matrix.

3.4.2 Metric Correlation to Performance

For all experimental simulation runs, the metric results are plotted against performance in Figure 3.9. This gives a visual indication of how strongly each metric can predict performance. For class separation, in both tasks there is practically no correlated pattern. Kernel quality and Lyapunov’s exponent both show strong positive correlation for small values, but shortly level off and the pattern breaks down for large values. This could be due to the idea that it is only strictly necessary for the number of distinct reservoir states to exceed the number of input classes that require separation. This would explain why the 9-class speaker task has a shallower initial gradient than the 3-class benchmark. The spectral radius plots are distorted due to many failed calculations returning zero for the metric. Otherwise, there is significant negative correlation with the speaker task performance, but none for the benchmark.

![Figure 3.9](image)

**Figure 3.9.** Each of the metrics for all simulation results plotted against classification accuracy in both tasks. This indicates the extent that each metric can be used to predict performance.
Table 3.1. Pearson’s Correlation between Metrics and Performance. Pearson’s correlation coefficient between the results for each metrics and the corresponding classification accuracy for each task.

<table>
<thead>
<tr>
<th>Metric</th>
<th>PCC of Benchmark Task</th>
<th>PCC of Speech Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class Separation</td>
<td>-0.04</td>
<td>-0.2</td>
</tr>
<tr>
<td>Kernel Quality</td>
<td>0.22</td>
<td>0.29</td>
</tr>
<tr>
<td>Lyapunov’s Exponent</td>
<td>\textbf{0.26}</td>
<td>\textbf{0.31}</td>
</tr>
<tr>
<td>Spectral Radius</td>
<td>0.05</td>
<td>-0.16</td>
</tr>
</tbody>
</table>

To determine numerically how well each metric can be used to predict performance for a given reservoir, we look at Pearson’s correlation results for each metric against both tasks, shown in Table 3.1. For each task there is a total of 120 reservoir initializations from which the metric results are taken.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig3_10.png}
\caption{Lyapunov’s exponent results plotted against kernel quality in both tasks to show the similarity between the metrics.}
\end{figure}

The two metrics that can give the strongest indication of performance in these tasks are Lyapunov’s exponent followed closely by kernel quality. Their closeness in this aspect adds weight to the idea that they are measuring a similar property of a reservoir, in addition to the
3.5. **DISCUSSION**

similar pattern of results in Figures 3.6 and 3.7. Figure 3.10 highlights the striking correlation between these metrics in both tasks. When visualizing the link between Lyapunov’s exponent and performance, the correlated trend is not as well defined. Although there is a significant correlation, Figure 3.9 shows that the metric has a large effect on performance only when it is within a small value range. As it increases, it seems to have less effect in determining performance.

For the benchmark task, class separation and spectral radius show no correlated pattern. Therefore, in this case, they do not give any hint to the performance at all. In the speaker recognition task, these metrics both show a significant negative correlation. However, this is not as strong as the positive correlation shown for Lyapunov’s exponent and kernel quality. Surprisingly, class separation produces a negative correlation with performance where we would expect the opposite.

3.5 **Discussion**

We have tested four of the most well established reservoir computing metrics on 2 classification tasks and under a number of different conditions. Out of these, 2 metrics have emerged as being more stable under a variety of settings: Lyapunov’s exponent and kernel quality. Furthermore, both of these show remarkable similarity in the patterns they follow in their results. This leads to the conclusion that they are very likely measuring the same property of a reservoir. We suggest that this relates to the often used, but ill-defined phrase ‘rich reservoir dynamics’. In addition to their stability, it is these metrics that provide the best indication of performance, with Lyapunov’s exponent coming out slightly ahead.

The spectral radius has a sole dependence on the weight matrix and is activity – and simulation – independent. Therefore, it would be highly beneficial for this measure to be utilized effectively, as it would indicate a reservoir’s success before any simulation need commence. Unfortunately, in our case, the spectral radius does not provide a reliable indication of performance in the tasks we tested. Nor can it even be reliably computed, with tri-phasic STDP and scale-free connectivity producing weight matrices that were invalid for the metrics computational procedure.
Generally, tri-phasic STDP and scale-free connectivity led to worse values for each metric and a wider spread of results for each set of random initializations. There are a couple of exceptions to this; 1) tri-phasic STDP produces higher class separation, 2) scale-free connectivity leads to higher kernel quality and Lyapunov’s exponent for the benchmark task.

Class separation also fared poorly in that most set-ups gave a large spread within results for multiple initializations. Also, it failed to reliably predict performance, giving no correlation for the benchmark task and fairly weak in the speech task.

The two tasks that were selected in this comparison of reservoir metrics are characteristically very different. The benchmark data rapidly fluctuates in time while the spectrograph speech data changes much more slowly but has more features. Therefore, we expect that observations that hold true for both datasets will tend to generalize well. For example, Lyapunov’s Exponent is the strongest indicator of performance of the tested metrics for both datasets. Also, the strong positive link between Lyapunov’s exponent and kernel quality is evident in both tasks.

3.6 Summary

In this chapter, we found through experiments that implementing synaptic plasticity as a pre-training process did not improve pattern recognition accuracy on the tasks tested. We attempt to use some commonly used reservoir metrics to understand what impact plasticity has on the properties of the recurrent network. However, even the best of the metrics, rank separation and Lyapunov’s exponent are only weakly correlated with performance in pattern recognition. None of the metrics give insight into why plasticity does not improve accuracy in the tasks – neither in the empirical effect that plasticity has on the metrics, nor in the theoretical basis of the metrics themselves. It is therefore difficult to confirm what role that the synaptic learning rules are having with regards to a learning task.

We assumed that simply by being exposed to input patterns, synaptic plasticity would encode some input-specific structure into the connections which would be enough in itself to cause some neurons in the reservoir to become active for specific types of input pattern. At least part of this assumption seems to be ill-founded. Therefore, in the next chapter we will focus on testing the first basis of our assumption that plasticity learns input structure.
3.6. SUMMARY

In order to test if input class-specific structure is learned, we must go beyond the metrics explored here and make an analysis at the level of synaptic change during the process of plasticity.
Chapter 4

Input-Specific Synaptic Adaptation under Plasticity

Synaptic plasticity is often explored as a form of unsupervised adaptation in cortical microcircuits to learn the structure of complex sensory inputs and thereby improve performance of classification and prediction. The question of whether the specific structure of the input patterns is encoded in the structure of neural networks has been largely neglected. Existing studies [Gilson et al., 2009a, Gilson et al., 2009b, Gilson et al., 2009c, Gilson et al., 2009d, Gilson et al., 2010] that have analysed input-specific structural adaptation have used simplified, synthetic inputs in contrast to complex and noisy patterns found in real-world sensory data.

In this chapter, input-specific structural changes are analysed for three empirically derived models of plasticity applied to three temporal sensory classification tasks that include complex, real-world visual and auditory data. Two forms of spike-timing dependent plasticity (STDP) and the Bienenstock-Cooper-Munro (BCM) plasticity rule are used to adapt the recurrent network structure during the training process before performance is tested on the pattern recognition tasks.

It is shown that synaptic adaptation is highly sensitive to specific classes of input pattern. However, plasticity does not improve the performance on sensory pattern recognition tasks, partly due to synaptic interference between consecutively presented input samples. The changes in synaptic strength produced by one stimulus are reversed by the presentation of another, thus
largely preventing input-specific synaptic changes from being retained in the structure of the network.

To overcome the problem of interference, we suggest that models of plasticity be extended to restrict neural activity and synaptic modification to a subset of the neural circuit, which is increasingly found to be the case in experimental neuroscience [Karni and Sagi, 1991, Schwartz et al., 2002, Lepousez et al., 2014, Li et al., 2013].

4.1 Introduction

Recurrent neural networks consisting of biologically based spiking neuron models have only recently been applied to real-world learning tasks under a framework called reservoir computing [Buonomano and Maass, 2009, Maass et al., 2002]. The models of this framework use a recurrently connected set of neurons driven by an input signal to create a non-linear, high-dimensional temporal transformation of the input that is used by single layer perceptrons [Rosenblatt, 1958] to produce desired outputs. This restricts the training algorithms to a linear regression task, while still allowing the potential to work on temporal data in a non-linear fashion.

Given an initially generated static connectivity, reservoir computing is based on the principle of random projections of the input signal in which the network structure is completely independent of the input patterns. In these models, the only features learned by the trainable parameters of the perceptron readout are the correlations between the randomly projected features and the desired output signal.

We believe that learning in neural networks should go further than supervised training based on error from the output. All synapses should adapt to be able to encode the structure of the input signal and ideally, should not rely on the presence of a desired output signal from which to calculate an error with the actual output. The neural activity generated by the input signal should provide enough information for synapses to adapt and encode properties of the signal in the network structure. By applying unsupervised adaptation to the synapses in the form of biologically derived plasticity rules [Bi and Poo, 1998, Bienenstock et al., 1982, Wittenberg and Wang, 2006] it is hoped to provide the means for the recurrently connected neurons of the network to learn a structure that generates more effective features than a completely random
4.1. **INTRODUCTION**

projection that is not specific to the input data.

On a conceptual level, unsupervised learning is important in the understanding of how synaptic adaptation occurs because it is still unknown what the sources of supervised signals are in the brain, if any exist. From early work on synaptic self-organization [Hebb, 1949], the principle of learning has rested on correlations in neural activity becoming associated together and forming assemblies that activate simultaneously. These structures are thought to encode invariances in the sensory input that are key in developing the ability to recognize previously encountered patterns.

In this chapter we will explore the impact of applying several biologically derived plasticity mechanisms on three temporal sensory discrimination tasks. Two forms of spike-timing dependent plasticity (STDP) [Bi and Poo, 1998, Wittenberg and Wang, 2006] will be tested, along with the Beinenstock-Cooper-Munro (BCM) rule [Bienenstock et al., 1982]. The sensory tasks will include real-world speech and video data of human motion. Synaptic plasticity will be applied in an unsupervised pre-training phase, before the supervised regression of the perceptron readout occurs. We will compare the impact that plasticity has on the performance in these tasks and also analyze the specific structural adaptation of the weight matrices between each of the classes of input sample in each task. A method will be introduced to evaluate the extent to which the synaptic changes encode class-specific features in the network structure.

Interference between different samples is a well-established phenomenon in sequentially trained learning models [French, 1999, McCloskey and Cohen, 1989, Ratcliff, 1990]. When presented to a learning model, an input pattern will cause specific changes to be made in the models parameters – in the case of neural networks, the synapses. However, during this encoding process, existing structure in the synaptic values is interfered with. In this way, consecutive input patterns disrupt previously learned features, sometimes completely. This effect is known as forgetting. It is of direct concern to neural networks trained on sensory recognition tasks that consist of spatio-temporal patterns projected through a common neural processing pathway. We will quantify the level of interference between the synaptic parameters for each tested plasticity model being applied to each type of sensory data.

Existing studies report that adapting neural circuits with plasticity can improve their perfor-
mance on pattern recognition tasks [Xue et al., 2013, Yin et al., 2012b] but there is no analysis of how the adaptation of synaptic parameters leads to this result. On the other hand, work that does detailed analysis on the structural adaptation of the network does so using synthetic input patterns that are already linearly separable [Toutounji and Pipa, 2014] or Poisson inputs projecting to single and recurrently connected neurons [Gilson et al., 2010]. For a review of work applying plasticity models to improve the general properties of neural networks, the reader is referred to [Chrol-Cannon and Jin, 2014a].

The experiments undertaken in this chapter will be performed on a typical reservoir computing model with its recurrent connections adapted with plasticity. Two main angles of analysis are made; we determine the strength of input specific synaptic adaptation and the extent to which consecutive inputs interfere within the synapses. Both of these are achieved through analysis of the change in weight matrix in response to each pattern.

4.2 Methods

4.2.1 Recurrent Network

The neural network model used in this work is illustrated in Figure 3.1. Recurrently connected neurons, indicated by $L$ are stimulated by current $I$ that is the sum total of injected current from the input signal, $I_{inj}$ and stimulating current from the pre-synapses, $I_{rec}$. The total current $I$ perturbs the membrane potential that is modeled with a simple model that matches neuron spiking patterns observed in biology [Izhikevich, 2003]. This method for modeling the spiking activity of a neuron is shown to reproduce most naturally occurring patterns of activity [Izhikevich, 2004]. The real-valued inputs are normalized between 0 and 1, which are multiplied by a scaling factor of 20 before being injected as current into $L$. Input connections number $0.2 \cdot \text{network size}$, projected randomly to the network nodes. Weights are uniformly initialized at random between 0.1. The video data set used in this work consists of significantly higher dimension inputs – 768 features – than the other data sets. Therefore in this case each feature only projects to one neuron, initially selected at random (a neuron can have connections from multiple inputs). Also the synaptic weights are scaled by 0.25.
4.2. METHODS

The network activity dynamics are simulated for 30 ms for each frame of data in a time-series input sample. This value is chosen as it roughly approximates the actual millisecond delay between digital audio and video data frames. Then, the resulting spike trains produced by each of the neurons are passed through a low-pass filter, \( f \), to produce a real valued vector used to train a linear readout with the iterative, stochastic gradient descent method (each described in the next section).

In our experiments the network consists of 35 or 135 spiking neurons (weight matrix plots consist of 35, performance trials consist of 135) with the ratio of excitatory to inhibitory as 4:1. Neurons are connected with static synapses i.e. the delta impulse (step) function. Connectivity is formed by having \( N^2 \cdot C \) synapses that each have source and target neurons drawn according to uniform random distribution, where \( N \) is the number of neurons and \( C \) is 0.1, the probability of a connection between any two neurons. Weights are drawn from two Gaussian distributions; \( \mathcal{N}(6, 0.5) \) for excitatory and \( \mathcal{N}(-5, 0.5) \) for inhibitory. When plasticity adapts the reservoir weights, \( w_{\text{max}} \) is clamped at 10 and \( w_{\text{min}} \) at -10. All parameters for excitatory and inhibitory neuron membranes are taken from [Izhikevich, 2003]. The equations for the membrane model are as follows:

\[
\begin{align*}
    v' &= 0.04v^2 + 5v + 140 - u + I \\
    u' &= a(bv - u)
\end{align*}
\]

With the spike firing condition:

\[
\text{if } v > 30 \text{mV then } \begin{cases} 
    v \leftarrow c \\
    u \leftarrow u + d
\end{cases}
\]

Parameters for the above equations are; \( a = 0.2, b = 0.2, c = -65, d = 8 \) for excitatory neurons and; \( a = 0.1, b = 0.2, c = -65, d = 2 \) for inhibitory neurons.
4.2.2 Trained Readout

To generate a real-valued output from the discrete spiking activity, the spike train from each neuron is convolved with a decaying exponential according to Equation 4.4. The vector of values produced is then weighted with the readout weight matrix and summed to produce a single output value, shown in Equation 4.5.

$$x_i = f(S(t)) = \max\left(\sum_{t=1}^{T} \exp\left(-\frac{S(t)}{\tau}\right)\right)$$  \hspace{1cm} (4.4)

$$y = \sum_{i=1}^{n} x_i \cdot w_i$$  \hspace{1cm} (4.5)

The state vector for a neuron is denoted by $x_i$, the filter function is $f()$ and the spike train is $S(t)$. The maximum number of time-steps in $S(t)$ is $T$, in this case 50. The decay constant $\tau$ is $6ms$.

The maximum value is taken from the low-pass filtered values in Equation 4.4 in order to detect the highest level of burst activity in the given neuron. We take this approach under the assumption that burst activity is more representative of spiking neural computation than a sum total of the firing rate.

These output weights are updated according to the iterative, stochastic gradient descent method: Least Mean Squares, given in Equation 3.7.

$$w_i \leftarrow w_i + \mu(y_d - y_o)x_i$$  \hspace{1cm} (4.6)

Here, $y_d$ is the desired output, $y_o$ is the actual output, $x_i$ is the input taken from a neuron’s filtered state, and $\mu$ is a small learning rate of 0.005. The weight from $x_i$ to the output is $w_i$. For the classification tasks of pattern recognition, $y_d$ takes the values of 0 or 1 depending if the class corresponding to the readout is the label of the current input sample.

4.2.3 Synaptic Plasticity Models

Three synaptic plasticity mechanisms are employed in this study, each of them based on the Hebbian postulate [Hebb, 1949] of “neurons that fire together, wire together”. Each mechanism
4.2. METHODS

is outlined as follows:

**BCM Plasticity:** The BCM rule [Bienenstock et al., 1982] is a rate based Hebbian rule that also regulates the post-neuron firing rate to a desired level. It works on a temporal average of pre- and post-synaptic activity. The BCM rule is given in Equation 3.10. The regulating parameter is the dynamic threshold $\theta_M$, which changes based on the post-synaptic activity $y$ in the following function: $\theta_M = E[y]$, where $E[\cdot]$ denotes a temporal average. In our case, $E[\cdot]$ is calculated as an exponential moving average of the post-synaptic neurons membrane potential. The exponential decay coefficient used for this is 0.935. As the membrane potential is model-dependant, we normalize it between 0–1 in real-time by continuously updating max and min variables of previous values. There is also a uniform decay parameter $\epsilon w$ set as 0.0001 that slowly reduces connection strength and so provides a means for weight decay, irrespective of the level of activity or correlation between pre-synaptic inputs and post-synaptic potential. A plot of the BCM weight change is presented in Figure S9 in Supporting Information.

$$\Delta w = y(y - \theta_M)\epsilon w$$  \hspace{1cm} (4.7)

**Bi-phasic STDP:** The STDP rule depends on the temporal correlation between pre- and post-synaptic spikes. The synaptic weight change is computed based on the delay between the firing times of the pre- and post-neurons. This is described in a fixed 'learning window' in which the $y$-axis is the level of weight change and the $x$-axis is the time delay between a pre- and post-synaptic spike occurrence. The bi-phasic STDP rule consists of two decaying exponential curves [Song et al., 2000], a positive one to potentiate in-order spikes, and a negative one to depress out-of-order spikes. This rule was derived from experimental work carried out on populations of neurons *in vitro* [Markram et al., 1997] [Bi and Poo, 1998]. Bi-phasic STDP is given in Equation 3.11.

$$\Delta w(\Delta t) = \begin{cases} A_+ \cdot \exp \left( \frac{-\Delta t}{\tau_+} \right) & \text{if } t > 0 \\ -A_- \cdot \exp \left( \frac{\Delta t}{\tau_-} \right) & \text{if } t \leq 0 \end{cases}$$  \hspace{1cm} (4.8)

$A_+$ and $A_-$ are the learning rates for the potentiation and depression, respectively. $\Delta t$ is the delay of the post-synaptic spike occurring after the transmission of the pre-synaptic spike. $\tau_+$
and $\tau_-$ control the rates of the exponential decrease in plasticity across the learning window. For our experiments the learning window is symmetric with $A_+ = A_- = 0.15$ and $\tau_+ = \tau_- = 20\text{ms}$.

**Tri-phasic STDP:** A tri-phasic STDP learning window consists of a narrow potentiating region for closely correlated activity but depressing regions on either side: for recently uncorrelated activity, and for correlated but late activity. This learning window has been observed *in vitro*, most notably in the hippocampi, between areas CA3 and CA1 [Wittenberg and Wang, 2006]. The tri-phasic STDP is given in Equation 3.12.

$$\Delta w(\Delta t) = A_+ \exp \left(\frac{-(\Delta t - 15)^2}{200}\right) - A_- \exp \left(\frac{-(\Delta t - 15)^2}{2000}\right)$$

(4.9)

The learning rates are set as $A_+ = 0.25$ and $A_- = 0.1$. Both STDP learning windows are plotted in Figure S10 in Supporting Information.

### 4.2.4 Synaptic Interference Measure

We wish to quantify interference directly between synaptic adaptations of plasticity. Our formulation of synaptic interference is based on the synaptic changes from sequentially presented samples. Synaptic adaptation for a given class of sample is called $\Delta W_t$ and average adaptation for all others are $\Delta W_o$. Interference must be calculated individually for each class of sample, $I_t^\text{class}$, and averaged together to get the overall interference, $I_t^\text{total}$. The equations are as follows:

$$I_t^\text{class} = \frac{1}{N} \sum_{i=1}^{N} \left[ \Delta W_{ti} \cdot \Delta W_{oi} < 0 \right] \left[ |\Delta W_{ti}| < |\Delta W_{oi}| \cdot C_n \right]$$

(4.10)

$$I_t^\text{total} = \frac{C_n}{\sum_{t=1}^{C_n} I_t^\text{class}}$$

(4.11)

Where $I$ is interference, $N$ is the number of synapses, $C_n$ is the number of competing sample classes and $\Delta W$ is a vector of synaptic changes. Subscript $i$ denotes the parameter index, subscript $t$ denotes samples of a given class 'this' and subscript $o$ denotes samples of all 'other' classes.

In Equation 4.10, the first set of Iverson brackets returns 1 if synaptic adaptation of a given class is of a different sign than that of the average adaptation of other class samples. The second set of Iverson Brackets returns 1 only if the magnitude of the synaptic adaptation of a
class is less than the average weight adaptation of other classes multiplied by the total number. This leads to us taking a conservative measure of synaptic interference where we will only flag interference within a synapse for a class of pattern if the weight change is in a different direction to the average as well as being lower in magnitude than the total weight adaptation of other inputs.

4.2.5 Synthetic Signal Data

A synthetic benchmark task is taken from a study performed with Echo State Networks [Jaeger, 2007], a similar type of network model to the one we employ, but using continuous rate-based neurons instead. The task is to predict which of three signal generating functions is currently active in producing a time-varying input signal. To generate a sample of the signal at a given time step, one of the three following function types is used; 1) A sine function of a randomly selected period, 2) A chaotic iterated tent map, 3) A randomly chosen constant. The generator is given some low probability, 0.05, of switching to another function at each time-step. The full method of generating the data is described in [Jaeger, 2007].

4.2.6 Speaker Recognition Data

A speaker recognition task is a classification problem dealing with mapping time-series audio input data to target speaker labels. We use a data set taken from [Kudo et al., 1999] which consists of utterances of 9 male Japanese speakers pronouncing the vowel /ae/. The task is to correctly discriminate each speaker based on the speech samples. Each sample is comprised of a sequence of 12 feature audio frames. The features of each frame are the LPC cepstrum coefficients. The sample sequence ranges between 7-29 frames. The dataset is divided into training and testing sets of 270 and 370 samples each, respectively. Note that unlike the benchmark data used in this report, the samples are not in a consecutive time-series, yet each sample consists of a time-series sequence of audio frames.
4.2.7 Pre-processing of the Human Motion Data

A visual task is selected to test high dimensional spatial-temporal input data. The KTH data set [Schuldt et al., 2004] consists of 2391 video files of people performing one of six actions; boxing, clapping, waving, walking and jogging. There are 25 different subjects and the samples cover a range of conditions that are described in more detail in [Schuldt et al., 2004]. Each video sample is taken at 25 frames per second and down sampled to a resolution of 160x120 pixels. We process the raw video sequences according to a formula shown in the following equations:

\[
M(t) = \| \Delta(I_1, I_2, ..., \Delta(I_{N-1}, I_N)) \| \\
\]

\[
M(t, i) = \begin{cases} 
1 & \text{if } M(t, i) \geq 0.2 \cdot \max(M) \\
0 & \text{else} 
\end{cases} \\
\]

The final input matrix \(M\) is indexed by time-frames, \(t\) and spatial samples \(i\). Column vectors \(I_n\) are individual frames, re-shaped into one dimension. Each sample contains up to a total of \(N\) frames. In plain language, this process essentially further down samples by a factor of 0.2 and calculates the difference between pixels in consecutive frames, which are then used as the new input features. Each frame is then re-shaped into a single dimensional column vector then appended together to form an input matrix in which each column is used as the neural network input at consecutive time steps. Figure 4.2 shows frames extracted from an example of each type on motion along with the corresponding processed features.

4.3 Results

4.3.1 Description of Sensory Inputs

Complex sensory signals are projected through a common set of nerve fibers to cortical regions that must learn to distinguish between them based on differences in their spatial-temporal features.

Three sensory recognition tasks are selected, among which two of them consist of real audio and video signals of human speech and motion. For all tasks, the neural network output is
4.3. RESULTS

trained to respond uniquely to each of the different types of input sample and therefore be able to perform effective recognition between them. Also, sample specific synaptic adaptations are analyzed to determine if unique structure is learned within the network due to synaptic plasticity.

The auditory task is to distinguish between nine different speakers based on short utterances of the vowel /ae/. Each of the 640 samples consists of a frequency 'spectrogram' that plots frequency intensity over a sequence of audio time frames. Figure 4.1 plots an example sample from each of the nine speakers.

![Figure 4.1](image)

**Figure 4.1.** Vowel samples from the nine speakers in the speaker recognition task. The audio signals in the data set are pre-processed into 12 Mel-frequency cepstrum coefficients (MFCC) features. Samples from each speaker have variable time duration in the number of audio frames they consist of.

The visual task is to distinguish between six types of human behavior; boxing, clapping, waving, walking, running and jogging. The 2391 samples are video sequences of many different subjects performing those six motions. There is a simple pre-processing stage that converts the video data into a sparse representation before being used as input to the neural network. Extracted still frames and processed features are plotted in Figure 4.2 for one subject performing each of the six behaviors.

A synthetic data set is generated to model a low spatial dimension but very high frequency temporal structure, in contrast to the previous two sensory tasks. Three functions generate time-varying single dimensional signals that the network learns to distinguish between. A complete
Figure 4.2. Human motion samples for the six types of behavior in the KTH visual discrimination task. This illustration consists of different behaviors from a single person, while the whole data set contains 25 persons. **Top row:** Still frames from example video samples; boxing, clapping, waving, walking, running and jogging. **Bottom row:** Features extracted corresponding to the samples above. Features are the raw time-series activity used as input to the neural network.

description and method for generating the data is described in [Jaeger, 2007].

The auditory and visual tasks are described in [Kudo et al., 1999] and [Schuldt et al., 2004], respectively, with data availability also provided.

### 4.3.2 Analysis of Synaptic Adaptation

Synaptic weight adaptation matrices form the basis of the analysis in this work. Figure 4.3 depicts the process of these matrices being collected and used for analysis of class-specific synaptic plasticity. Firstly, synaptic plasticity is applied to the network to adapt a baseline weight matrix that reflects the general statistics of the input patterns in the data set. Secondly, each the
weight adaptation matrix is collected for each sample and these are grouped by class and also into two sets based on the training and testing data division. Finally, the Euclidean distance is calculated between each weight matrix, with the average distance between each set plotted in a type of 'confusion matrix' in which a low distance indicates high similarity between the adaptation of synaptic parameters.

**Figure 4.3.** Three step process describing the analysis of input-specific synaptic adaptations. Firstly, the recurrent connections are adapted under plasticity in the same way as in Figure 5.1. Secondly, each input sample is presented and plasticity adapts the synapses. The change in the weight matrix is stored for each sample and grouped by the input class label, $C_x$ and into two sets, *train* and *test*. Finally, the Euclidean distance between the matrices in *train* and *test* is calculated and the average for each class label is plotted in a confusion matrix.

In the confusion matrix just described, if the diagonal values are lower than the others it means that synaptic plasticity is sensitive to the structural differences in input samples that are labelled as different classes. The stronger the diagonal trend, the more sensitive plasticity is to features of the input. It means that plasticity learns to distinguish class labels, such as different speakers or human actions, without ever being exposed to the labels themselves *a priori*.

The weight adaptation matrices are also used to estimate the amount of interference between different input samples within the synaptic parameters. This is described further later in the Results section.
4.3.3 Learning Input-Specific Structure using Plasticity

We wish to test the hypothesis that synaptic plasticity is encoding a distinct structure for input samples of different labels. For the speech task, these labels consist of different speakers and for the video recognition task the labels consist of different human behaviors.

The data sets are divided evenly into two. Each subset is used to train a recurrently connected network for 10000 iterations, selecting a sample at random on each iteration. The changes to the weight matrix due to plasticity are recorded for each sample presentation. This is then used to create a class-specific average weight change for each of the class labels in both of the sample subsets. Finally, we calculate the Euclidean distance between each class in one set and each class in the other according to the following formula:

$$\text{Dist}(C^X, C^Y) = \sum_{i=1}^{N} |\Delta W_i(C^X) - \Delta W_i(C^Y)|$$

Where $C$ denote class labels, $X$ and $Y$ distinguish the separated sets of samples, $\Delta W$ is the change in weight matrix for a presented sample, $N$ is the number of synapses, and $i$ the synapse index.

This effectively produces a confusion matrix of similarity in the synaptic weight change for different classes of input. Having lower values on the descending diagonal means that there is structural adaptation that is specific to the class of that column compared with the similarity between structural adaptations of two different classes.

Figure 4.4 shows the 'weight change confusion matrices' described above, for each plasticity model applied to all sensory tasks (nine experiments in total). All of the experiments show at least some stronger similarity in the descending diagonals and most are stark in this manner. It is certainly a strong enough pattern to show that through the many iterations of training, each of the plasticity models have become sensitive to the particular structure of the sensory input signals so that each different class of sample will give rise to changes in synaptic strength that are distinct from other classes compared with the similarity to themselves. We re-iterate that the class labels were not used in any way in the plasticity models themselves and so the differences in the weight change arise from the input signals alone.

There are a few exceptions to the strong diagonal patterns in Figure 4.4. This means that
4.3. RESULTS

Figure 4.4. Class correlation of structural synaptic adaptation. Heat map plots indicate the structure learned on each class for the three tasks under each of the plasticity rules. Essentially, it is a confusion matrix of the geometric distance between the weight matrix adaptation of each class of sample. The training data for each task is divided into two sets. Class-average adaptation is found for each set. There is then a distance calculated between each class of the two sets. Lower values on the descending diagonal indicate higher correlation within a class adaptation and therefore strong class-specific structure learned.

some classes are not effectively distinguished from each other; speakers 8/9 with bi-phasic STDP, behaviors 1/2 with BCM, behaviors 1/2/3 and 4/5/6 with tri-phasic STDP. The latter confusion corresponds to the behaviors of boxing/clapping/waving and walking/running/jogging. From
the similarity of those input features shown in the lower panes of Figure 4.2, it is evident why this confusion might occur.

4.3.4 Classification Performance with Plasticity

Perhaps the ultimate goal of neural network methods when applied to sensory tasks is the ability to accurately distinguish different types of input sample by their patterns. We compare the error rates achieved by our neural network on the three sensory tasks, with and without the different forms of plasticity used in this work. Table 4.1 lists the error rates achieved for each of the learning tasks with the different plasticity rules active in a pre-training phase in addition to a static network with fixed internal synapses.

Table 4.1. Classification Error Rates. Classification testing error rates given for each of the learning tasks with a static network, as well as each of the three forms of plasticity used for synaptic adaptation.

<table>
<thead>
<tr>
<th></th>
<th>Static</th>
<th>BCM</th>
<th>STDP</th>
<th>TP-STDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tri-func</td>
<td>0.153</td>
<td>0.157</td>
<td>0.204</td>
<td>0.138</td>
</tr>
<tr>
<td>KTH</td>
<td>0.283</td>
<td>0.3</td>
<td>0.333</td>
<td>0.383</td>
</tr>
<tr>
<td>Vowels</td>
<td>0.089</td>
<td><strong>0.086</strong></td>
<td>0.092</td>
<td><strong>0.086</strong></td>
</tr>
</tbody>
</table>

Values averaged over ten trials with random seed based on system clock. SD did not exceed 0.03 for all values.

From the error rates in Table 4.1 it is evident that pre-training the network with synaptic plasticity can make slight improvements in lowering the error rate. However, the results here indicate that it can have a greater negative impact than a positive one. In the KTH human behavior data set, all three plasticity models increase the error rate by between 1.7% and 10%. Conversely, the best improvement was found on the tri-function signal recognition task with tri-phasic STDP at only 1.5%.

It is evident from the network output that pre-training with synaptic plasticity is not a
suitable method for this class of model. This does not contradict with the result that plastic synapses are learning useful, input-specific structure. However, it does suggest that the structure being learned is not effectively utilized in the generation of a network output. We next investigate interference between synaptic changes to determine if the structural learning is retained in the network or if interference is a barrier for effective application of synaptic plasticity.

### 4.3.5 Synaptic Interference

When a model adapts incrementally to sequentially presented input, existing patterns that have been learned by the model parameters are prone to be overwritten by learning new patterns. This is known as interference. The work that has studied this effect [French, 1999, McCloskey and Cohen, 1989, Ratcliff, 1990], test the ability to recognize previously presented input after the model has been trained on new ones in order to estimate how much learning has been undone. When new training leaves the model unable to recognize old patterns, it is said there has been catastrophic interference and forgetting.

We introduce a method of measuring interference directly in synaptic parameters instead of the model output. Our measure is described in detail in the Methods section. $I_{total}$ directly quantifies all synaptic changes that are overwritten.

The interference for each of our experiments is listed in Table 4.2. In all but one of the experiments the interference level is between 82\% and 96\%. Most of the learned structure for each class of input is forgotten as consecutive samples overwrite each other’s previous changes. Bi-phasic STDP applied to speaker recognition has the lowest level of interference at 58\%.

To further explore interference and visualize the impact of plasticity, synaptic changes will be analyzed directly. Figure 4.5 is an illustrative example for the speaker recognition task with BCM plasticity (similar figures for the other experiments are given in S1–S8). It shows the adaptation of the synaptic weight matrix produced by each speaker in the voice recognition task. This is plotted against the activity level for each neuron, $S$, and the readout weights, $R$, that are trained to generate an output that is sensitive to that given speaker. Each of these subplots is the average response taken over all sample presentations from that speaker. This makes a whole chain of effect visible: from the synaptic change of an internal network connection, to
Table 4.2. Synaptic Interference. The total level of synaptic interference for each of the tasks under each plasticity model. Calculated according to Equation 4.11. The minimum level of interference is highlighted for each task. Smaller values indicate the experiments with less overwriting of synaptic adaptations.

<table>
<thead>
<tr>
<th></th>
<th>BCM</th>
<th>STDP</th>
<th>TP-STDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tri-func</td>
<td>0.82</td>
<td><strong>0.8</strong></td>
<td>0.88</td>
</tr>
<tr>
<td>KTH</td>
<td><strong>0.92</strong></td>
<td>0.93</td>
<td>0.96</td>
</tr>
<tr>
<td>Vowels</td>
<td>0.96</td>
<td><strong>0.58</strong></td>
<td>0.9</td>
</tr>
</tbody>
</table>

Values averaged over ten trials with random seed based on system clock. SD did not exceed 0.07 for all values.

the average neuron state for a given speaker, to the selective weights of the readout for that speaker. For all to be working well in a cohesive system, we expect that a positive weight change should correspond with a neuron activation unique to the class which would in turn improve the recognition ability of the readout to identify that class.

The sections of the class weight matrix highlighted in green in Figure 4.5, highlight an example where synaptic interference is occurring between different types of pattern. Directly opposing features in the weight matrix adaptations show the samples negating each other’s changes. However, the same features are also most distinctively class specific.

Any synapse can only change in two directions: positively or negatively, which means that a single synapse can only adapt to distinguish between two mutually exclusive kinds of input pattern. If \( n \) synapses are considered in combination, then the number of input patterns that can be discriminated becomes \( 2^n \) in ideal theoretical conditions. Figure 4.5 illustrates this principle in practice with regards to the nine speaker recognition tasks. The adapted synapses labeled (a) can clearly distinguish speaker \{#1\} from speakers \{#2, #3\} but cannot distinguish \{#2\} from \{#3\}. Similarly, the adapted synapses labeled (b) can distinguish speakers \{#1, #6, #8\} from speakers \{#3, #4, #9\} but cannot distinguish speakers within either of those sets. However,
4.3. RESULTS

**Figure 4.5.** The class-specific synaptic adaptation for the 9 class speaker recognition task under BCM plasticity. The main heat maps in each subplot show the adaptation of the weight matrix (synapses) after the presentation of voice input data from each speaker. Blue values show a reduction in synaptic strength and red values show an increase. The bar-chart, $S$, shows the average neuron activation for each class. The bar-chart, $R$, shows the learned readout weights. Labeled synapses $a, b, c, d$ indicate key structural changes that are selective between different speakers. Each label alone can distinguish between two sets of speaker. Taken all together, the labeled synapses adapt specifically to each speaker in a unique pattern, learning a distinct network structure for each one.

If the synapses $(a), (b), (c)$ and $(d)$ are considered in combination, then all speakers can be distinguished by synaptic plasticity changes alone.

Figure 4.5 also shows the weight changes are not correlated with the neural activity or readout
weights. For plasticity to improve the accuracy of sensory discrimination, it would be expected that synapses would strengthen for class specific neural activity and weaken for common neural activity. This is not the case in our results.

4.4 Discussion

4.4.1 Unsupervised Plasticity Learns Label Specific Structure

Both STDP and BCM models adapt the synapses of a network in distinctive patterns according to which type of sample is being presented to the network. We can conclude that presenting a training signal with the sample label is not required for plasticity to learn specific information for complex sensory inputs from different sources. This result holds for the speech, visual and benchmark pattern recognition tasks. To achieve this feat, we hypothesize that plasticity drives the synaptic parameters to a structure that represents an average between all input samples. Once converged, any further input stimulus will drive the synaptic parameters in a unique direction away from this average structure. On balance, scrambled presentation of random inputs keeps the network in this sensitive state.

4.4.2 Uniformly Applied Plasticity Leads to Synaptic Interference

We show synaptic plasticity spends most of its action counter-acting previous changes and overwriting learned patterns. The same patterns of synaptic adaptation that distinctly characterizes each class of input are the same ones that reverse adaptations made by other inputs.

Plasticity is applied uniformly to all synapses. All neurons in a recurrent network produce activity when given input stimulus. Combined, these factors mean that any input sample will cause the same synapses to change. This leads to synaptic competition, interference and ultimately, forgetting.

4.4.3 Locality of Plasticity Required to Overcome Interference

To overcome the problem of interference, the mechanisms of plasticity need to be restricted to adapt only a subset of the synapses for any given input stimulus. There is much existing research
that supports this conclusion and a number of possible mechanisms that can restrict the locality of plasticity.

It has been shown \textit{in vivo} (using fMRI and neurological experiment) that synaptic plasticity learns highly specific adaptations early in the visual perceptual pathway [Karni and Sagi, 1991, Schwartz et al., 2002]. Simulated models of sensory systems have demonstrated that sparsity of activity is essential for sensitivity to input-specific features [Barranca et al., 2014, Finelli et al., 2008]. In fact, in a single-layer, non-recurrent structure, STDP is shown to promote sparsity in a model olfactory system [Finelli et al., 2008]. Conversely, in recurrent networks, STDP alone is unable to learn input specific structure because it ‘over-associates’ [Bourjaily and Miller, 2011]. Strengthened inhibition was used to overcome this problem and combined with reinforcement learning to produce selectivity in the output [Bourjaily and Miller, 2011]. By promoting sparsity, the lack of activity in most of the network will prevent activity-dependent models of plasticity in adapting those connections.

Reward modulated plasticity has also been widely explored in simulated [Darshan et al., 2014, Gavornik et al., 2009] and biological experiment [Lepousez et al., 2014, Li et al., 2013]. Input-specific synaptic changes are shown to be strongest in the presence of a reward signal [Gavornik et al., 2009, Lepousez et al., 2014]. Lasting memories (synaptic changes not subject to interference), are also seen to rely on a process of re-consolidation consisting of fear conditioning [Li et al., 2013]. A reinforcement signal based on either reward or fear conditioning can be effectively used to restrict synaptic changes in a task dependent context such as sensory pattern recognition.

Another way to restrict synaptic changes in a task dependent way is to rely on a back-propagated error signal that has well established use in artificial neural networks. This might be achieved in a biologically plausible way through axonal propagation [Kempter et al., 2001] or top-down cortical projections sending signals backwards through the sensory pathways [Schfer et al., 2007]. Top-down neural function in general is thought to be essential in determining structure in neural networks [Sharp, 2014], providing a context for any adaptations.

Neural cascades [Polat and Sagi, 1994] and synchronicity [Hoppensteadt, 1989] have also been suggested as factors that enable input-specific plasticity.
4.4.4 Learning Input Structure does not Necessarily Improve Performance

Structural adaptation with plasticity in the pre-training phase, while specific, may not be utilized by the output produced by the network readout. This could be due to the following reasons. Firstly, there is a disparity in the neural code. The output from a recurrent spiking network model is currently decoded as a rate code. In contrast, synaptic plasticity updates structure in a way that depends on the precise temporal activity of neural spikes. Secondly, information content is reduced. While creating associations between co-activating neurons, Hebbian forms of plasticity may also increase correlations and reduce information and separation. These can determine the computational capacity of a recurrent network model [Chrol-Cannon and Jin, 2014b]. Both discrepancies could be barriers for the effective application of plasticity to improve pattern recognition. Therefore new frameworks of neural processing should be based directly on the adapting synapses. This will lead to functional models of neural computing that are not merely improved by synaptic plasticity, but that rely on it as an integral element.

4.5 Summary

Circuits in the brain are able to learn to recognize complex sensory patterns without a supervised signal providing labels to guide the learning process. We have shown that current models of unsupervised synaptic plasticity are able to learn label specific structure of temporal patterns of audio and video data without ever being exposed to the class labels during training. This learned structure is short lived. Interference between consecutively presented sensory inputs causes synaptic changes to be overwritten and forgetting to occur. This prevents specific patterns from being learned within the structure – the synaptic weight can only converge to an average of all input patterns.

It is likely that once class average synaptic weights are reached, a pattern of a specific class will cause plasticity changes in a unique direction away from this average. The sum total of all class unique changes maintain the average structure over time.

Our result suggests that in brains, plasticity is either not operating uniformly over all synapses in the network, or is not a continuous process happening evenly at each moment.
in time. To overcome the severe forgetting factor we observe, it may be necessary to restrict synaptic changes by relying on sparse neural activity or making long term changes only in the presence of a reward signal.
Chapter 5

Plasticity Sensitive Readout Neuron

5.1 Introduction

Since we show in the previous chapter that unsupervised plasticity had significant sensitivity to structural differences in input signal, it would surely follow that it is possible to exploit that sensitivity in a pattern recognition system. As we also show, the inter-class separability is not transferred into the synaptic parameters due to the problem of interference. The neural activity used to form future outputs is prevented from improvement, as activity depends on the structure, which itself is prevented from development. Therefore, it would appear that the biological models of spiking reservoirs with synaptic plasticity are untenable in their current form. In order to make an adjustment that could utilize the structure learned by plasticity, we suggest the following change to the reservoir model: to use the change in synaptic strength directly in the calculation of an output, ignoring the neural activity. Of course, neural activity still plays a central role in determining plasticity, the only difference in the proposed approach is the method of generating an output for pattern recognition. This requires a readout neuron that is sensitive to synaptic adaptation, no only pre-synaptic neural activity. It is uncertain whether there are chemical processes in the brain that convert rates of change in synaptic formation into instantaneous bursts of activity. There is no reported evidence of such a specific mechanism, which would be required for this idea to plausibly exist as a feature of neural processing. However, there is biological observation of a related feature of lateral spread of LTP/LTD, that could provide a
plausible basis to this mechanism. In any case, for this chapter, we mainly intend to determine how effectively the information generated by unsupervised plasticity can be applied to pattern recognition compared to the standard reservoir computing method.

5.2 Synaptic Adaptation as Feature Vector

In conceptual terms, the change in architecture of this proposed pattern recognition system is simple: replace the feature vector of the reservoir state with a feature vector generated by a plasticity method such as STDP or BCM rule.

The equations that describe such a system are as follows:

\[ x^M(t) = (L^M u)(t) \]  \hspace{1cm} (5.1)

\[ \Delta L^M(t) = S(x^M(t)) \]  \hspace{1cm} (5.2)

\[ y(t) = f^M(\Delta L^M(t)) \]  \hspace{1cm} (5.3)

Equation 5.1 shows the instantaneous liquid state \( x^M(t) \) computed from the input signal \( u \) convolved with the recurrently connected spiking network \( L^M \). Equation 5.2 shows the change in reservoir structure \( \Delta L^M(t) \) calculated with the plasticity function \( S \) on the liquid state. Equation 5.3 shows the instantaneous output \( y(t) \) computed from the readout map \( f^M \) on the change in reservoir structure.

Equations 5.1 and 5.3 are taken from the work introducing the LSM model [Maass et al., 2002], with the latter having had the liquid state \( x^M(t) \) replaced with the change in reservoir structure \( \Delta L^M(t) \).

The conceptual change to the standard LSM model is simple. However, the implementation of this change has a number of significant consequences that impact on various aspects of the training process.

At some point, for each input sample, a difference between the original weight matrix and the weight matrix after the sample has been presented must be computed. Two considerations
are important in this process: The initial state of the network structure (i.e. the samples that have immediately preceded the current one) and how many time steps should be taken before measuring the changed state of the weights.

The initial weight matrix of the reservoir will be highly dependent on the samples that immediately preceded the current sample. They would have the 'last word' on how the weights are adapted according to the interference problem that we observe in the previous chapter. To minimize the effect of this, a period of pre-training is performed with a scrambled order of samples to reduce the bias that any single class can introduce into the initial weight matrix. Subsequently, for training and testing phases, the initial structure derived through pre-training is loaded back to reset the network before the collection of reservoir feature vectors occurs. This means that the order of presentation of samples in either training or testing phase is of no concern, it is only critical in the pre-training.

The question of when to measure the change in value of the synaptic weights might seem a straightforward one – at the end of each sample. This policy does make sense when all samples are of equal duration. However, in the real-world data sets used in this work, samples are of varying durations. The effect this has is that longer samples would tend to naturally produce greater weight change than shorter samples regardless of the class of each. This clearly introduces a source of bias that we wish to avoid. Therefore, in our case, a heuristic is used for calculating the weight change at the length of the shortest sample in the dataset, thus truncating the rest of the samples.

Another factor the proposed method faces is an increased dimension to the feature vector. As the features are the synaptic changes instead of activity, the dimension has risen from \( N \), the number of neurons, to potentially \( N^2 \) the dimension of the weight matrix, although in our simulations each neuron is only connected to a small fraction of the total, making the actual dimension \( N^2 \cdot 0.1 \). Functionally, a dimension increase has no effect on the architecture – a linear readout can still be trained in the same manner. It may, however, negatively impact the convergence of the trained readout, a possibility that will be explored further shortly.
5.3 Methods

The basic LSM model and experimental set-up follows the exact same format as in Chapter 4. The only differences are parameter values which are listed below, and replacement of reservoir activity state with synaptic weight change which is described as follows:

1) Pre-training with STDP for 1000 iterations is performed to allow the network to reach a structure that is sensitive to input samples of different classes. It is found empirically that unless this stage is performed, using the weight change as a feature vector is not suitable for classification.

2) When collecting the reservoir state for a particular sample; initially the synapses are reset to their values immediately following pre-training. This is done to ensure consistency. The input sample is used to stimulate the reservoir, neural activity is simulated, along with STDP plasticity. After 160 ms, the change in synaptic weights from their values at the start of the sample are calculated and saved as the feature vector for the sample. The value of 160 ms is selected to remove the factor of samples of variable length. In order to derive synaptic changes from each sample that are of comparable magnitude, the longest length is determined that does not exceed the bounds of the shortest sample which is the aforementioned value.

Note that only the STDP plasticity rule was used for our proposed classifier due to each rule requiring parameter tuning of all other reservoir settings in order to make each one work. We decided to try to keep things simple by working with just one plasticity model.

5.3.1 Algorithm Description

Instead of re-defining the models used for our methods, the reader is referred to Chapter 4 Methods section. Here, we will take the opportunity to describe the simulation model algorithmically in pseudocode with some expressions referring to equations in previous chapters where relevant. We hope that this complements the models that have so far been described analytically in equations but not yet procedurally.

The overall training process for our plasticity based classifier is illustrated in Figure 5.1. It delineates a three step process, each of which we now describe in a section of pseudocode. Referenced equations are noted in-line as comments following the applicable expression.
5.3. METHODS

**Figure 5.1.** Three step process describing a reservoir computing model extended by having the recurrent connections adapted with unsupervised plasticity in a pre-training phase. Firstly, input samples $I$ are presented in random order while the resulting neural activity drives synaptic adaptation under plasticity. Secondly, each input sample is presented in sequence with the resulting neural activity causes synaptic adaptation under plasticity. The synaptic adaptation vectors $S$ are collected for each sample. Finally, the adaptation vectors are used as the input to train a set of perceptron readouts, one to recognize each class of sample, $C_x$.

Firstly, the pre-training phase is used to adapt the initial, random structure of the synapses to match the statistics of the input according to synaptic plasticity.

```markdown
// pre-train recurrent neurons with plasticity
for each iteration $I$ in preTrainIterations
    select random sample $S$ from trainingSamples
    for each frame $f$ in $S$
        for each attribute $x$ in $f$
            for each connection $c$ in $C_{in}$
                $c$.input($W_{in}[x][c] \cdot S[f][x] \cdot inputScale$)
            for each timestep $t$ in frameDuration
                neurons.simulateActivity() // Eq. 4.1 - 4.3
                synapses.applyPlasticity() // Eq. 4.7 - 4.9
        neurons.resetActivity()
```
The synaptic weights adapted by pre-training, now provide a basis to collect the sample-specific synaptic adaptations. The weight matrix is reset to the pre-trained weights between each sample in order to simplify the implementation with calculating the synaptic adaptation vectors.

// collect neural activation state vectors
baseWeights.value ← synapses.value
for each sample S in trainingSamples
    for each frame f in S
        for each attribute x in f
            for each connection c in C_in
                c.input(W_in[x][c] · S[f][x] · inputScale)
            for each timestep t in frameDuration
                neurons.simulateActivity() // Eq. 4.1 - 4.3
                synapses.applyPlasticity() // Eq. 4.7 - 4.9
                S.fv ← synapses.value − baseWeights.value
        neurons.resetActivity()
        synapses.value ← baseWeights.value

Finally, the plasticity feature vectors that have been stored for each sample are now used to train a set of readout perceptrons, one to predict each class, using least mean squares regression. For a given sample, the readout with a matching class label is given a desired target of 1 and the rest are given 0. The class label of the readout with the maximum value is taken as the prediction for classification.
5.3. METHODS

// train readouts with linear regression
for each iteration $I$ in $readoutTrainIterations$

select random feature vector $fv$ from $trainingSamples.fv$

for each class readout $R$ in $nClass$

if $R.classLabel = fv.classLabel$

// boost readout for matching class
$R.output \leftarrow R.lms(fv, 1)$ // Eq. 4.5, 4.6

else

// suppress other readouts
$R.output \leftarrow R.lms(fv, 0)$ // Eq. 4.5, 4.6

prediction $P \leftarrow \max(R.output)$

if $P.classLabel \neq fv.classLabel$

$errorSum \leftarrow errorSum + 1$

$errorCummulative \leftarrow errorSum \div I$

5.3.2 Parameter Settings

Table 5.1 lists all of the parameters used in the method described in the previous subsection. Injected current for an input is equal to $||W_{in}|| \cdot ||x|| \cdot inputScale$ where $||W_{in}||$ are normalized input weights, $||x||$ are normalized input values, and $inputScale$ scales the input to the magnitude of the neuron model. The time taken to simulate each audio/video frames in the sequence is indicated by $frameDuration$. The iterations of STDP used in pre-training is $preTrainIterations$ and LMS iterations by the readout as $readoutTrainIterations$. The other parameters relate to parts of the reservoir computing model and STDP that are defined in Chapter 4.
Table 5.1. List of Parameter Settings. Parameter settings are listed for the LSM model using STDP adaptation feature vectors.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>preTrainIterations</td>
<td>1000</td>
</tr>
<tr>
<td>readoutTrainIterations</td>
<td>100000</td>
</tr>
<tr>
<td>frameDuration</td>
<td>5ms</td>
</tr>
<tr>
<td>inputScale</td>
<td>20</td>
</tr>
<tr>
<td>trainingSamples</td>
<td>70%</td>
</tr>
<tr>
<td>testingSamples</td>
<td>30%</td>
</tr>
<tr>
<td>LMS learning rate $\mu$</td>
<td>0.005</td>
</tr>
<tr>
<td>reservoirSize (speaker task)</td>
<td>200</td>
</tr>
<tr>
<td>reservoirSize (video task)</td>
<td>768</td>
</tr>
<tr>
<td>synapseCount (speaker task)</td>
<td>4000</td>
</tr>
<tr>
<td>synapseCount (video task)</td>
<td>5376</td>
</tr>
<tr>
<td>$C_{in}/W_{in}$ dimension (speaker task)</td>
<td>240</td>
</tr>
<tr>
<td>$C_{in}/W_{in}$ dimension (video task)</td>
<td>768</td>
</tr>
<tr>
<td>STDP $A^+$</td>
<td>0.12</td>
</tr>
<tr>
<td>STDP $A^-$</td>
<td>-0.12</td>
</tr>
<tr>
<td>STDP $\tau^+$</td>
<td>30</td>
</tr>
<tr>
<td>STDP $\tau^-$</td>
<td>30</td>
</tr>
</tbody>
</table>

5.4 Performance

The best accuracy achieved with a plain LSM for the speaker recognition task is 0.92 and for the human behavior task, 0.77. The results with our proposed plasticity based LSM are 0.98 and 0.82, respectively – an absolute improvement of 6% and 5% in each case. These figures are for the best performance achieved for a single random parameter initialization for each model.

More complete performance results can be seen in Figure 5.2, in which ten repeated runs
with different random seeds were made for each data point. Learning convergence is plotted against the number of least mean squares iterations. As can be seen from the graph, the typical improvement brought by our method is between 5% and 10%. Also of note, is the wider error bars for the plain LSM experiments. Our proposed method therefore has less variation in performance for different random initializations of the model, especially regarding the speaker recognition task. As well as reaching a lower testing error rate, the plasticity based LSM also converges more rapidly in the training of the readout, again, particularly for the speaker recognition task.

These are encouraging results that indicate that the unique structure shown to be learned by plasticity in Chapter 4 is indeed more representative of the input patterns than the neural activity within the reservoir.

**Figure 5.2.** The classification error plotted for the testing data for the human motion and speaker recognition tasks. The error is plotted against iterations of least mean squares regression. 10 runs are performed for each data point with the error bars as standard deviation. Blue indicates performance with a standard liquid state machine architecture while the red line indicates performance with our proposed plasticity based classifier.
5.5 Learning Convergence

For the assessment of the learning convergence, there are two phases that need to be measured for the plasticity based classifier: the pre-training of the recurrent connections, and the stochastic gradient descent of the linear readout. The convergence in the pre-training phase is not based on an error signal, because the synaptic weight adaptation is unsupervised. The convergence, therefore, is solely based on the magnitude of sum weight change for each time step. If the sum absolute weight change decreases over the course of simulated plasticity, then we can say that convergence to some structure is occurring. The convergence of linear regression of the readout, is task specific and dependent on the error signal with respect to the desired output. Linear stochastic gradient descent methods such as the least mean squares rule employed in this work are essentially guaranteed to converge to the global optimum. The factors that we look to measure are the rate and stability of the convergence as well as the converged value, which corresponds with the training accuracy that can be achieved on the task.

5.5.1 Readout Regression Convergence

In all cases, least mean squares regression converges mostly smoothly at least when looking at increments of 1000x iteration time-steps. The 100,000 total iterations of training may seem many, however, due to the simplicity of the learning rule this only takes about two seconds on a contemporary computer CPU.

Figure 5.2 shows that some level of over-fitting does occur in the training of the readouts using neural activity with negligible or no over-fitting for the readouts using plasticity features. Over-fitting is indicated when the testing error begins to rise again as training proceeds forward after the period of convergence to a minimum error rate. It means that a few parameters are being exploited in the training data that are not present in the testing data, and therefore the generalization ability of the model will suffer. Our plasticity based classifier only suffers from over-fitting on the human motion data set, and even then it is subtle. There is somewhat more over-fitting in both experiments with the plain LSM. For the human motion task, average testing error rises after convergence by 1% for plasticity features and 3% for activity features. For the speaker recognition task, average testing error rises by less than a percentage point for plasticity
features and by 5% for activity features. As well as having larger variance for different model initializations, the plain LSM also fluctuates in testing error more erratically in the speaker recognition task.

We now take a formal approach to quantify the convergence of the readout learning curves observed in Figure 5.2.

\[
\lim_{k \to \infty} \frac{|x_{k+1} - L|}{|x_k - L|^q} = \mu
\]

Equation 5.5, taken from [Schatzman, 2002] defines the rate of convergence, \( \mu \). An iterative sequence, \( x_k \) is said to converge with order \( q \) to a limit, \( L \), for \( q > 1 \) as long as \( \mu \in (0,1) \). Values of \( \mu \) closer to 0 indicate that the sequence converges strongly according to order \( q \) while values greater than 1 indicate that the sequence converges with a lower order. When \( q = 1 \), if \( \mu = 0 \), the sequence is said to converge superlinearly while if \( \mu \geq 1 \), it is said to converge sublinearly. Values closer to zero indicate quicker convergence.

Table 5.2 lists the convergence rate, \( \mu \), for both of the sensory tasks with activity and plasticity feature vectors. It also lists the convergence rates for different orders, \( q \). In all cases, the rates of convergence for \( q = 3 \) are greater than 1 but the values for \( q = 2 \) are within \( (0,1) \). Therefore, we can see that according to the linear and quadratic orders of convergence, there is some convergence towards the limits. However, as the convergence rates are much closer to 0 for \( q = 1 \), the overall LMS regression has linear convergence. This is to be expected for stochastic gradient descent methods like LMS. When comparing the convergence for the two types of feature vector, we can see that the rate of convergence is consistently faster (smaller values) for the plasticity feature vector over the standard activity driven readout.

Table 5.3 lists the lower limits of the LMS learning curves, that are equal to their average lowest testing error rate (the learning curves studied here are an average of ten runs). Table 5.3 also lists the number of iterations which LMS regression took to reach these lower limits. It is evident from these results that the readouts driven by plasticity reached lower error rates and within fewer iterations of LMS.
Table 5.2. LMS Convergence Rates. Convergence rates, $\mu$ from Equation 5.5, of the LMS output for activity and plasticity feature vectors with different orders of convergence $q$.

<table>
<thead>
<tr>
<th>Feature Vector</th>
<th>$q = 1$</th>
<th>$q = 2$</th>
<th>$q = 3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity FV (speaker)</td>
<td>0.129</td>
<td>0.651</td>
<td>2.55</td>
</tr>
<tr>
<td>Plasticity FV (speaker)</td>
<td>0.023</td>
<td>0.129</td>
<td>1.89</td>
</tr>
<tr>
<td>Activity FV (video)</td>
<td>0.045</td>
<td>0.184</td>
<td>3.67</td>
</tr>
<tr>
<td>Plasticity FV (video)</td>
<td>0.028</td>
<td>0.161</td>
<td>3.08</td>
</tr>
</tbody>
</table>

Convergence rates are calculated from sequence $x_k$ that is an average of ten runs.

5.5.2 Synaptic Plasticity Convergence

The sum total absolute weight change for each iteration of synaptic plasticity is plotted for the presentation of ten input randomly selected input samples for each of the experimental set-ups. This means that each of the plasticity rules is tested with both sensory data tasks. Figure 5.3 shows the results. It should be noted that each of the selected input samples is of varying duration.

Even though the patterns of synaptic weight change are visually very different in each experiment, one trend is most clearly visible in five out of six of the experiments. There is a sharp pulse of increased weight change marked at the beginning of each of the patterns presented.

Another common property that holds for all experiments is a very large magnitude of weight change in the very first time-step of plasticity. In fact, in all graphs in Figure 5.3, the initial reading was off the chart – it was plotted in that way to prevent the remainder of the graph being scaled to small in comparison.

These features of synaptic weight change, taken together count as strong evidence to support our hypothesis in Chapter 4. Once the network structure has converged to a critically sensitive state that is the average between the class samples, presentation of any sample will drive the
5.6. **BIOLOGICAL PLAUSIBILITY**

Table 5.3. LMS Convergence Limits. Convergence limits of LMS, $L$ from Equation 5.5, are stated in addition to the number of steps to reach it.

<table>
<thead>
<tr>
<th></th>
<th>Limit $L$</th>
<th>Steps to Reach $L$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity FV (speaker)</td>
<td>0.115</td>
<td>26k</td>
</tr>
<tr>
<td>Plasticity FV (speaker)</td>
<td>0.026</td>
<td>16k</td>
</tr>
<tr>
<td>Activity FV (video)</td>
<td>0.3</td>
<td>13k</td>
</tr>
<tr>
<td>Plasticity FV (video)</td>
<td>0.21</td>
<td>10k</td>
</tr>
</tbody>
</table>

Limits are calculated from sequence $x_k$ that is an average of ten runs.

weights in a unique direction away from the average state. The graphs here indicate that this happens in the first few iterations of a sample being presented, and then reduce quickly as the network adapts to the specific sample.

The speaker recognition task produces slower and smoother convergence in all synaptic plasticity rules than the human behavior recognition task.

### 5.6 Biological Plausibility

At first impression, the idea that adapting readout neurons should be sensitive to plasticity in pre-synaptic neurons seems biologically implausible. According to conventional theory, neurons should react to pre-synaptic activity but not to plasticity. However, upon investigating the neuroscientific literature, one observed phenomenon does seem capable of providing a biological mechanism to support this idea: lateral spread of long-term potentiation (LTP) and long-term depression (LTD). Lateral spread pertains to a change in one synapse leading to a similar change in another synapse with a common pre-neuron or post-neuron. Assume internal reservoir neurons are connected $A \rightarrow B$ and that $A$ is also connected to a readout neuron $A \rightarrow Y$. If STDP changes the strength between $A$ and $B$ and assuming that there is post-lateral spread occurring, then the strength would correspondingly change between $A$ and $Y$. This effectively makes the
adaptation – and therefore activity and stimulation – of $Y$ sensitive to any plasticity from $A$, or any other neuron from which it receives pre-synaptic input. There are both studies that observe pre-lateral [Fitzsimonds et al., 1997, Tao et al., 2000, Harris, 2008] and post-lateral [Frey and Morris, 1997, Fitzsimonds et al., 1997, Harris, 2008] spread of LTP and LTD. These can provide an opportunity for a purely biological mechanism to be derived for a plasticity based neural readout.

5.7 Summary

We have proposed an alternative method of decoding the spiking neural activity for use within reservoir computing. Models of spike-time dependent plasticity are designed to be particularly sensitive to the spiking activity between connected neurons and therefore it should not be a big surprise that the features produced by plasticity are more effective than the low-pass filtering that was traditionally used to decode discreet spikes for use in reservoir computing. More information about the temporal spiking activity must be conveyed by a plasticity based code to the readout, as borne out by the improved results attained. We still take for granted, of course, that neural activity is the predominant medium of computation in the brain and that changes in synaptic strength perform only the functions of learning and memory. However, there is some neuroscientific literature briefly discussed in this chapter that indicates that an adapting neural output could be sensitive to neighbouring synaptic plasticity. This might be biologically justified by the phenomenon of lateral spread of LTP and LTD.

However, in introducing this new method, we have not only improved accuracy over the default LSM in pattern recognitions tasks, but also strengthened the finding in Chapter 4: that plasticity is capable of adapting neural structure in a way that is sensitive to classes of input patterns. Plasticity feature vectors thus seem to have more class specific information than the neural activation states of the LSM. Considering that plasticity is determined purely by neural activity, this suggests that the traditional method of calculating the reservoir states from the neural spike trains is losing critical information that plasticity manages to capture in synaptic adaptations.

The fact that the total synaptic weight change seems to peak immediately following the
presentation of an input suggests that it is possible to further refine the plasticity based LSM. Instead of taking the change in weight over the whole sample, as is currently done, the change could just be calculated when the total value rises or falls suddenly. We predict this because we associate rapid weight change with the class-unique adaptations that were mentioned in this chapter as well as Chapter 4.
Figure 5.3. The total absolute level of synaptic weight change is plotted over the presentation of 10 samples, for six experiments including both the visual and auditory tasks with each of the plasticity models active in the network: BCM, STDP and TP-STDP. All of the samples presented are of varying duration but can often be visually distinguished by the peaks of synaptic weight change.
Chapter 6

Summary and Future Work

This thesis has taken a quite diverse range of paths in investigation, from experiments focused on analysis of self-organizing neural models to a newly suggested type of neural readout that is sensitive to the change in synaptic strength. It spans the gap between biological neural and plasticity models and more abstract learning frameworks. Much of the work undertaken in the creation of this thesis was not planned, but led naturally from questions raised in earlier experiments as well as unexpected results from experiments that were assumed to behave predictably. It is hoped that the reasoning process that leads from chapter to chapter has been made apparent and that the journey taken in this research makes sense.

6.1 Main Contributions

The main contributions that have been made throughout the course of this thesis are summarized in the following items:

Review of Plasticity in Neural Networks. An investigation is made into currently existing research that uses models of synaptic plasticity to improve computational abilities of neural processing methods. Particular emphasis is provided to biologically inspired neural models that include spiking neurons and recurrent connectivity. We present a diverse range of findings that have analyzed a number of information theoretic computational features that are affected when synapses are adapted with plasticity in a pre-training phase. Some
trends are drawn for biologically inspired self-organized neural models.

**Pre-training with Plasticity.** Comparing the performance of a static reservoir computing model with one pre-trained using biological synaptic plasticity. Three models of synaptic plasticity – two forms of STDP and the BCM model – have been incorporated into the liquid state machine framework as an unsupervised pre-training process. The impact of these additions on temporal sensory pattern recognition tasks was evaluated and found not to produce reliable improvements in performance. Therefore, we find that unsupervised self-organization of synaptic parameters does not guarantee improved ability in a specific supervised task in reservoir computing.

**Reservoir Metrics Analysis.** An empirical investigation into computational properties of reservoir computing with spiking neurons. The information content of neural states is quantified with rank separation and Lyapunov’s exponent metrics. The spectral radius of the weight matrix is used as a proxy measure for stability in temporal neural activity. Pattern separability of the neural states is determined by a geometric separation metric on the reservoir state vectors. Each of these metrics is correlated with performance in pattern recognition and is shown to vary under the influence of synaptic plasticity and for different network connectivities. We conclude that the metrics commonly used for reservoir analysis only correlate weakly with performance on the real-world data sets that we have tested. Also, synaptic plasticity changes them in various, non-predictable ways depending on the starting conditions of the simulations.

**Input-specific Plasticity.** Analysis of input-specific synaptic weight changes. A new form of analysis is introduced that compares synaptic adaptation for specific classes of input patterns. This method is suitable for analyzing the structural adaptation of specific input features, in contrast to the metrics used in the previous item that deals with general properties of the neural network. We show that STDP and BCM plasticity models do learn features of specific input patterns. However, we also show that these features are not encoded in the structure of the network due to continual over-writing of iteratively presented input samples – catastrophic forgetting.
Plasticity Sensitive Readout. Introduction of a novel temporal pattern recognition approach that uses synaptic weight change in place of neural activity. The finding that self-organized synaptic plasticity is sensitive to specific inputs leads to the possibility of replacing the current activity based classifier used in neural models with a plasticity based one. It is found that the testing error is significantly lower (between 5%–10%) on the visual and auditory pattern recognition tasks that were tested when using this newly proposed method. The synaptic changes are analyzed over the course of multiple sample presentations to determine the periods of learning and convergence. From most of these experiments, it is indicated that synaptic weight change is greatest at the beginning of a new sample and will subsequently reduce over time until a different sample is presented. This indicates that the weight matrix settles quickly into a sensitive state for a group of samples but can adapt quickly to specific samples when new ones are presented.

6.2 Further Extensions

There are a number of open research questions that remain which follow quite naturally from the existing experimental work performed in the creation of this thesis. The main extensions to the work that clearly stand out are described as follows.

Improved Plasticity Classifier. Improving the plasticity based classifier by including a method to determine the precise time to take the feature vector when a sudden peak in total synaptic weight change occurs. As it stands, our plasticity based classifier uses the synaptic weight change over the course of a whole sample as a feature vector to classify that sample. We show in Chapter 5, that the synaptic weight change is the strongest as soon as a new sample begins to stimulate the network. This may be an indication that the initial weight change is more characteristic of the sample class than any continuing adaptation during the rest of the sample presentation. Therefore, we suggest a technical improvement to the plasticity based classifier introduced in Chapter 5: calculate the classification feature vector in the first few time-steps of the pattern presentation, immediately following a sharp drop in the sum of absolute weight change. Ideally, this will not be a pre-defined
number of time-steps, as the number of time-steps can vary depending on the network parameters and the plasticity model. A threshold might be used instead, to determine the moment that the sum of weight change drops by a significant margin. We speculate that this extension might remove the need to present whole input samples, thus saving much simulation time and reducing the computational requirements of the spiking network in applications.

**Locally Regulated Plasticity.** A plasticity model with locally regulated parameters for heterogeneous structural adaptation. Part of the motivation to use self-organizing systems is so the structure of those systems can adapt to reflect features of the input/environment. However, if all of the parameters in our network models adapt together according to the same rules, there is a danger that any structure encoded in the network is a homogeneous average of the patterns in the input. We suggest that regulating plasticity according to localized network parameters can enhance the role of plasticity in generating a heterogeneous structure that can more accurately reflect the input. In Chapter 4 we showed how STDP and BCM plasticity models adapted all synapses in similar pattern, regardless of the class of input pattern. Notably, the STDP plasticity model contains no explicit regulation and BCM plasticity only regulates on a temporal average of post synaptic activity. We advocate further work in locally regulated synaptic plasticity in addition to neural development, from within a framework of genetic regulated networks (GRN), as we describe in Chapter 2. The local parameters of a GRN have the potential to dynamically restrict synaptic weight change to specific parts of the neural network which will lead to; more characteristic structure learned of each input, less interference between consecutively presented patterns and increased modularity in network structure which is thought to be an important feature of biological neural networks [Clune et al., 2013, Ellefsen et al., 2015].

**Temporal Coding in Recurrent Networks.** New methods to make use of temporal coding in recurrent neural networks. In Chapter 2, it was briefly described why we had selected to use rate-based encoding and decoding for input and output signals to recurrent neural networks. Current temporal coding schemes have strict requirements for a $t = 0$ baseline. This is incompatible with recurrent neural activity which continuously propagates through
the connections with repeated activity and no external indication of when the $t = 0$ baseline can be reset for further encoding or decoding in a time-series signal. On the other hand, resorting to a rate-based code, as we have done, loses a large amount of information in the neural processing of spiking recurrent networks. We suggest there could be an alternative, at least for classification problems, in using a neural code derived from the theory of polychronization [Izhikevich, 2006]. Polychronous groups describe neural structures consisting of time-locked neural activity – repeating patterns of activation – that can be detected at any point in time during simulation. At least for the problem of decoding, polychronous group activation could be used in place of rate-decoded neural activity to associate precise spike timing patterns with desired outputs. This avoids the issues of temporal codes that require an arbitrary reference time while retaining the ability to make use of spike-timing information in the model output. The main reservation we have regarding the use of polychronous groups is the high computational complexity of detecting them in unsorted neural activity data.

New Reservoir Metrics. Investigation into computational properties of recurrent networks using newly introduced measures of reservoir distance. New work [Chen et al., 2014] in applying reservoir computing models to system fault detection has introduced a new method for calculating distance between reservoir states by using the difference of two consecutively trained readouts to determine if there has been a system change, or anomaly, in the reservoir activity. We propose that this method can be tested as a more functional alternative to the class separation metric used in Chapter 3. All samples would be presented as a time sequence and increases in the distance metric would be correlated with the change in sample in order to measure the sensitivity or the reservoir activity to samples of different classes of input. One aspect of this metric is that its computation is not directly performed on the network parameters but on those of the readout functions. An advantage is that the metric does not require class labels \textit{a priori} in order to calculate the model distance.
6.3 New Directions

In the course of performing the work described in this thesis, we always aspire to improve the self-organizing properties of the learning models that we work with. Often, self-organization of learning models is taken to mean that the system parameters are adaptive to the task at hand, and to the input to the system. This does not mean that the learning model is self-improving, as its behavior is fixed at the outset. Ideally, we desire models that adapt themselves, and can therefore improve themselves – only then, could the goal of constructing systems that learn new tasks without being explicitly programmed or tuned manually, be achieved. Our current model is adaptive in the sense that the parameters undergo adaptation according to pre-specified rules and equations derived from biological experiment and taken from numerical optimization. However, this means the system itself is not adaptive as the implemented equations stay fixed throughout the runtime of the model. We now suggest a new direction that can transform a neural model adapted with plasticity into a self-modifying and self-improving system.

**Self-improving Neural Development.** Fixed synaptic plasticity models lead to inevitable limits on neural development in artificial neural networks. Specifically enabling the plasticity rules to adapt by modulating their parameters with a gene regulatory network would push these limits somewhat further back, but eventually the definition of this additional system would be the new limiting factor.

The only way apparent to us to remove theoretical limitation of the neural developmental model is to evolve neural growth and plasticity models with a general computational representation that allows the algorithms to change their form as well as their parameters.

We suggest a neuro-genetic programming system with an evolve-able language containing operators for the construction and modification of networks of neurons and their synapses. In this approach an evolutionary algorithm would adapt the programs used to grow and adapt a neural network in the presence of input-stimuli. The fitness function would evaluate the performance of the objective of the neural network – high accuracy in one or more classification or regression tasks.

Such a set of self-defined neural organization algorithms would more accurately replicate
the intrinsic ability of biology to adapt its own mechanisms at a fundamental level. This stands in contrast to the plasticity models constructed by people in an attempt to fit biological data. Figure 2.4 in Chapter 2 shows the futility of that approach, no curve, or set of curves could fit such a wide variety of patterns. Given that approximation of data from biological experiment has failed, we advocate the value of following fundamental biological principals above an irrational attention to detail.

We hope that our discussed new direction gives some inspiration and together with the suggested extensions to our existing experiments in the previous section, provides some grounds for fruitful future research.
Bibliography


Echo State Networks and Liquid State Machines.


Appendix

**Figure 6.1.** The class-specific weight adaptation for the 3 class time-series benchmark task under BCM plasticity. Description of each sub-plot follows Figure 4.5.

**Figure 6.2.** The class-specific weight adaptation for the 3 class time-series benchmark task under bi-phasic STDP. Description of each sub-plot follows Figure 4.5.
**Figure 6.3.** The class-specific weight adaptation for the 3 class time-series benchmark task under tri-phasic STDP. Description of each sub-plot follows Figure 4.5.

**Figure 6.4.** The class-specific weight adaptation for the 6 class time-series human behavior recognition task under BCM plasticity. Description of each sub-plot follows Figure 4.5.
Figure 6.5. The class-specific weight adaptation for the 6 class time-series human behavior recognition task under bi-phasic STDP. Description of each sub-plot follows Figure 4.5.
Figure 6.6. The class-specific weight adaptation for the 6 class time-series human behavior recognition task under tri-phasic STDP. Description of each sub-plot follows Figure 4.5.
Figure 6.7. The class-specific weight adaptation for the 9 class speaker recognition task under STDP. Description of each sub-plot follows Figure 4.5.
Figure 6.8. The class-specific weight adaptation for the 9 class speaker recognition task under Tri-phasic STDP. Description of each sub-plot follows Figure 4.5.