Spike-Based Classification on Accelerated Neuromorphic Hardware

Nathan Holford, Universität Heidelberg

13. Juli 2011

Spike-Based Classification on Accelerated Neuromorphic Hardware
A complete Liquid State Machine (LSM) was implemented and analysed on mixed-signal, neuromorphic hardware composed of 192 analogue Leaky Integrate-and-Fire (LI&F) neurons with approximately 100.000 synapses. The underlying liquid substrate is based on a self-stabilising neural architecture and connected to a dedicated LI&F neuron acting as a Tempotron. This binary classifier's original learning rule has been altered such that it can now learn without requiring a voltage trace, exclusively by means of spike-times. Despite this limitation, the LSM was demonstrated to achieve results comparable with those of an LSM utilising the original Tempotron. This modification enables us to massively reduce the amount of necessary information, thus overcoming hardware-related communication bottlenecks and ultimately performing classifications utilising the full potential of hardware which operates 10^4 times faster than biological real-time.

Klassifizierung mittels Aktionspotentialen auf beschleunigter neuromorpher Hardware

Eine vollständige Liquid State Machine (LSM) wurde auf mixed-signal neuromorpher Hardware mit 192 analogen Leaky Integrate-and-Fire (LI&F) Neuronen und ungefähr 100 000 Synapsen implementiert und untersucht. Das zugrundeliegende Liquid beruht auf einem selbststabilisierenden neuronalen Netzwerk und ist mit einem LI&F Neuron verbunden, welches ein Tempotron realisiert. Die ursprüngliche Lernregel dieses binären Klassifizierers wurde dahingehend verändert, dass es nun möglich ist ohne Membranspannungsmessung ausschließlich unter Zuhilfenahme von Aktionspotentialen zu lernen. Ungeachtet dieser Einschänkung wurden Ergebnisse erzielt die zu denen einer LSM mit unverändertem Tempotron vergleichbar sind. Diese Abwandlung erlaubt es die Menge an notwendigen Informationen drastisch zu reduzieren um so hardwarespezifische Bandbreitenbeschränkungen zu überwinden. Auf diese Weise lässt sich die volle Beschleunigung von 10^4 gegenüber biologischer Echtzeit auf der Hardware nutzen.

Inhaltsverzeichnis

Preface

This paper builds from the work of Jeltsch (2010) of constructing a Liquid State Machine using the FACETS Stage 1 Hardware. Throughout this project I have been experimenting with neural networks on this novel, neuromorphic hardware, both independently and in close cooperation with Dimitri Probst, who generated the handwriting recognition and 'difficult' tasks in Section 3.3. I have also received support and guidance others in the Electronic Vision(s) group of the Kirchhoff Institut für Physik, Ruprecht-Karls-Universität Heidelberg, and supervision from Dr. Daniel Brüderle, Eric Müller and Sebastian Jeltsch. During my time in the group I have also studied the variations of Postsynaptic Potentials generated by the synapse drivers on the hardware in collaboration with Ioannis Kokkinos, with a view to their calibration, as well as performing an extensive, although inconclusive, study to find optimal hardware parameters for Short-Term Plasticity.

1. Introduction

What is consciousness? The answers to this simple, yet pertinent, question are numerous and varied. Whether one chooses to take a purely materialist approach (thinking of the mind as merely the product of a series of electrical impulses) or a more Cartesian line (that consciousness is metaphysical but harnessed somehow within the brain), neither philosophers nor scientists have yet proposed a theory which fully explains the complex phenomenon.

Cartesian dualism fails to adequately explain the effects of chemical and electrical stimuli on consciousness, as well as the more extreme cases. One of the most famous is that of Phineas Gage, a responsible well-liked man who after severe damage to his frontal lobe became impulsive and thoughtless. This seriously undermines the notion that the brain and mind are separate entities. On the other hand, advocates of more material approaches have yet to fully explain the plethora of results indicating that consciousness is somehow more than the sum of the physical parts.

Although a definitive philosophical solution to this question is beyond the scope of neuroscience, illuminating the intimate processes of the brain and wider nervous system is at the heart of this increasingly dynamic field. Thanks to newly-developed techniques in biological measurement and imaging, alongside the continued development of computers with exponentially-increasing computational power, the discipline is experiencing something of a renaissance.

Since the advent of computing, new developments have often been heralded as quickening the era of machines capable of human-like thought, from the mid-20th Century code-busting Collosi, through the early personal computers of Apple and Commodore in the 1970's, to the very recent developments such as memristors and graphene. Concerted efforts have also been made to reverse engineer cortical neural networks in the brain, perhaps the best known of which is the Blue Brain Project (Markram) of the Brain Mind Institute at the EPFL in Lausanne. In this project, a mammalian neocortex with 10,000 biologically detailed neurons is simulated on the 16,384 processor IBM Blue Gene/P supercomputer (IBM, 2010). Models such as these however can require quadratically increasing processing power as more neurons are added to the model. A different approach to modelling biological neural networks is to construct physical models of neurons as silicon, hybrid digital-analogue integrated circuits. There are disadvantages to a physical emulation approach, namely that the model requires significant initial resources to design and build compared to a numerical simulation, and, once constructed, the model can be difficult to alter. However this approach also comes with compelling advantages, including the potential for great acceleration compared to the biological real-time, greatly reduced power consumption and the possibility of building without a significant slowdown in the computational speed due to its intrinsic parallelism (Mead and Mahowald, 1988; Mead, 1989, 1990) leading to applications in

robotics for self-organisation, visual processing and motor (Renaud et al., 2007; Indiveri et al., 2009; Gomez-Rodriguez et al., 2010; Lewis et al., 2000; Häfliger, 2007; Vogelstein et al., 2007)

The FP7 funded Fast Analog Computing with Emergent Transient States (FACETS, 2010) project has brought together theoretical and experimental expertise from a broad range of disciplines with the aim of applying their collective knowledge towards the construction of novel, biologically inspired computational paradigms. Models have been developed based on *in vivo* and *in vitro* measurements of biological systems spanning scales from individual neurons and synapses through to the behaviours of cortical networks. From these models, electronic emulations are being developed focussing on either biologically accurate neurons or large, versatile networks. The latter is being realised by Electronic Vision(s) group of the Universität Heidelberg, together with TU Dresden, as a single wafer-scale device built from blocks of highly interconnected and configurable synapses and neurons. Although still a work in progress, several chip prototypes are currently available for experimentation.

Outline Section 2.1 and Section 2.3.1 outline the fundamental biology of neurons and how models can be formulated in order to mimic some of the electrophysical behaviour of neurons. Sections 2.5 and 2.6 provide an introduction to methods of computation using models of individual neurons and networks of neurons. This paper goes forward in Section 3.1 with a proposed realisation of a binary classifier run on prototype neuromorphic hardware. The classifier was implemented and presented with a range of tasks described in Section 3.3, achieving results as presented in Section 4.

2. Background of the Methods

2.1. Biological Neurons

The human brain is estimated to have 100 billion neurons (Williams and Herrup, 1988), connected through around 100 trillion synapses. There are many different types of neurons, each adapted according to its function. Some can transmit a signal over a distance greater than 1 metre, whilst others transmit for only 10 µm. Some are connected to thousands of other neurons, whilst others receive signals from only a handful. All these neurons share similar features, however this section will focus on describing the properties of a pyramidal-type neuron, a neuron-type common in the cerebral cortex and thought to be linked to cognitive ability.

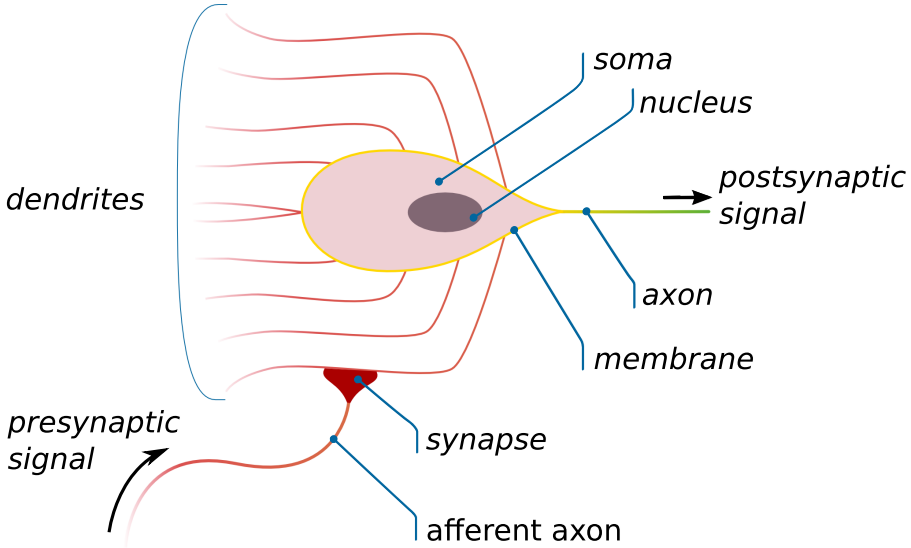


Abbildung 1: A schematic diagram of a biological neuron. Note: the dendrites branch out further than shown here into so-called "spines". A cell in the cortex will commonly be connected to afferent neurons through synapses on around 10^5 spines. The axon itself also branches out, connecting to approximately 10^5 postsynaptic neurons

.

The neuron has three main constituent parts, the *soma*, *dendrites* and *axon* (see Figure 1). The *soma* is the main body of the cell from which the other components extend. Its membrane can hold an electric potential. Signals from presynaptic neurons (the input signals) arrive at synapses which clasp on to spines along the arboreal branches of the *dendrites*. There is a small protrusion on the soma, called the *axon hillock*, from which the postsynaptic (output) signal of the neuron is propagated onwards, along the long, thin *axon*, which itself branches out hundreds of times to connect to other neurons before terminating.

The membrane of an unstimulated neuron remains at a certain *resting potential* until it receives some stimulation from its presynaptic neurons. This potential (typically -70 mV) is caused by a concentration gradient of ions found in the plasma, both inside and outside of the cell. When at rest, ions diffuse very slowly in and out of the cell through ion-specific channels embedded in the

membrane. However the resting potential is primarily maintained by sodium-potassium pumps, each of which expel sodium ions from within the cell whilst drawing in potassium ions from outside the cell in a single movement. These pumps maintain the potassium ion concentration raised at approximately 20-fold and the sodium ion concentration is depleted to 9-fold compared to the exterior concentrations, thus causing a potential to exist across the membrane.

The potential across the cell's membrane is briefly increased when the neuron is stimulated by a presynaptic signal. If excitatory stimuli are applied frequently enough to the membrane, then their cumulative charge increases and will pull the potential upwards until a specific *threshold potential* has been exceeded, typically around -55 mV, causing the neuron to "fire".

Firing is the occurrence of an action potential – a short spike in the membrane potential – being triggered as certain voltage gated ion channels in the cell membrane are briefly opened and closed (see Figure 2). Such spikes are said to form the fundamental language of neural communication: they are assumed to be the biological equivalents of binary 1's and 0's.

Synapses As previously mentioned, neurons communicate with one another through connections called synapses. Synapses come in two varieties, *electrical* and *chemical*. Electrical synapses are a direct linking of two cells' cytoplasm by way of ion channels, which allows for rapid propagation of signals, such as for defensive reflexes, but is of little use in neural computation since the postsynaptic signal cannot have any gain. Chemical synapses are much more versatile. In these synapses, a branch from an axon of one cell approaches a spine on the arbor of another cell's dendrites until they are close enough that a chemical neurotransmitter is able to diffuse from the former to the latter in approximately 0.5 ms (Katz and Miledi, 1965).

At a typical chemical synapse (Figure 3), the axon of the presynaptic neuron comes within 20 nm (this gap is called the *synaptic cleft*) of the postsynaptic neuron's dendrite. Tiny vesicles of neurotransmitter are formed within the axonal terminal and are held there until a presynaptic spike causes some vesicles to fuse with the membrane, thereby quickly releasing the neurotransmitters they were holding and diffusing them across the cleft to receptors on the opposite side. The receptors are attached to sodium channels, which, on receiving the neurotransmitter, briefly open to allow ions to diffuse into the postsynaptic cell, thus raising its membrane potential for a short time (see Figure 2. Recent research suggests that dendritic spines "promiscuously" grow out to synapse with other neurons in order to test as many connections as possible for compatibility (Kalisman et al., 2005).

A synapse is typically weak (i.e. only has a small effect on the postsynaptic neuron) when formed, and it is presumed that it will only survive, or become stronger, through use in a process called *synaptic plasticity*. This learning was first described by Canadian psychologist Donald Hebb (Hebb, 1949):

> "When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased".

This remarkably simple, yet accurate, insight into synaptic learning is an essential rule which leads to many of the emergent computational properties of neural networks, in particular their ability to learn to perform complex tasks.

2.2. Development of the Brain

At birth, the healthy infant's nervous system can already support the essential functions for life, its heart beats, its lungs breathe and its gut digests. The newborn's brain, however, is too complex to be defined by genes alone and it must learn how to think (Gierer, 1988). When we are born, most of our cerebral cortex - the thin layer covering the surface of our brain, which controls the higher functions such as language, memory, consciousness - is at its most plastic. The neurons of the child's minds are connected to one another through around 10^{15} (1 quadrillion) synapses, which need to be swiftly and meticulously pruned if they are to become useful. It is

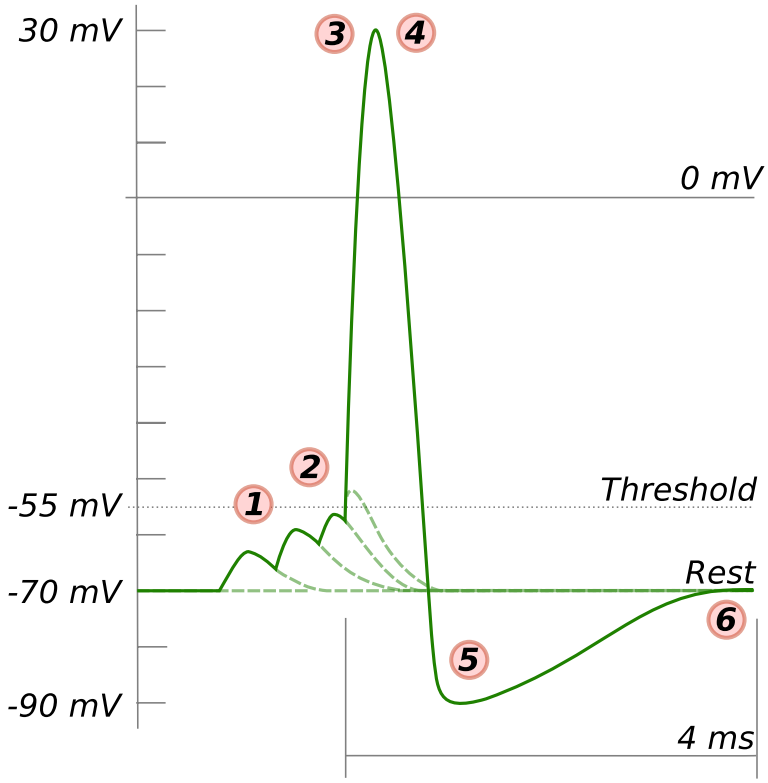


Abbildung 2: *An idealised action potential* **1.** An unstimulated neuron receives spikes from afferent neurons at excitatory synapses. After the presynaptic spike peaks, the membrane potential begins to decay towards the resting potential, but each quickly succeeding spike pulls the potential to a higher peak than the previous until **2.** when the fourth spike lifts the potential over the threshold, triggering the sodium channels to open, thus *depolarising* the membrane through the rapid diffusion of Na^+ ions into the cell. This continues until **3.** where the high potential triggers the closing of the sodium channels, and at **4.**, the opening of the potassium channels, leading to a *repolarisation* as the concentration of K^+ decreases rapidly until **5.** where the potential is said to be *hyperpolarised* as the potassium channels begin to close. The period after hyperpolarisation is called the *refractory period* where membrane potential rises slowly towards the *resting potential*, thereby reducing the possibility another action potential occurring immediately afterwards.

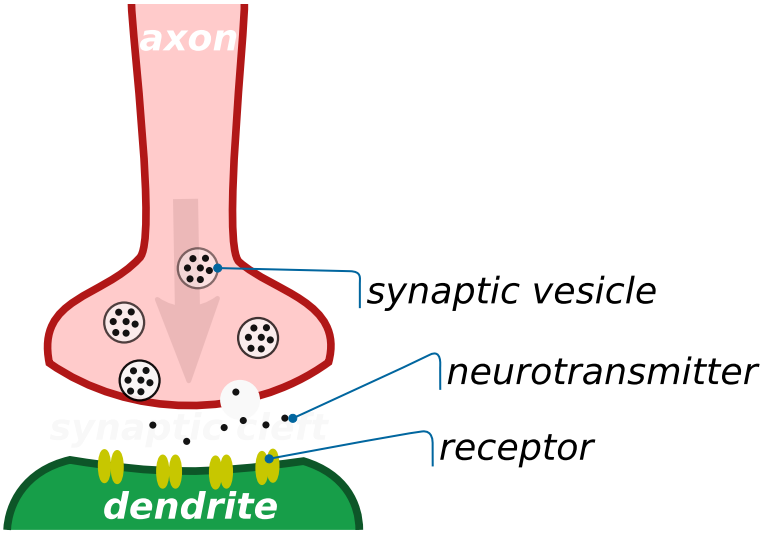


Abbildung 3: A schematic diagram of a chemical synapse. Here, the axon of an afferent neuron grows to within 20 nm of the dendrite of the postsynaptic neuron. When an action potential is triggered in the afferent neuron, a signal is transmitted down the axon and causes the synaptic vesicles, filled with molecules of neurotransmitter, to move towards the membrane at the synaptic cleft and release their contents. The molecules rapidly diffuse across the short distance and and some of them bind with receptors at the dendrite of the postsynaptic neuron. The neurotransmitter activates the receptor in some way, often opening specific ion gates in order to induce a PSP in the postsynaptic neuron.

suggested that strong synapses are maintained by chemicals released when an action potential occurs, whilst the weak synapses, if unused, will eventually starve and wilt (Sanes and Lichtman, 1999). This could be seen as the basic biochemical process behind Hebbian-style learning. By maturity, the number connections can fall as low as 10% of its original (Drachman, 2005).

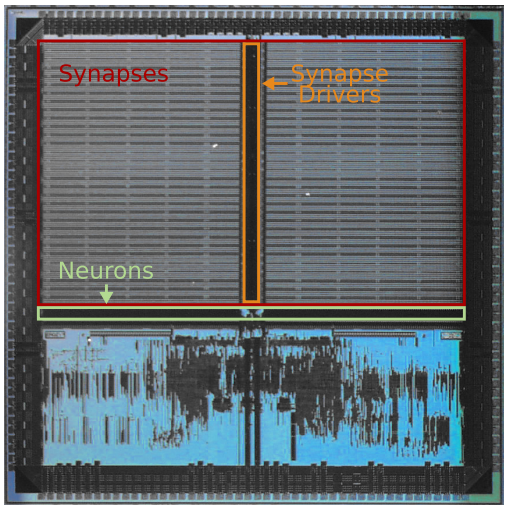
Artificial Neural Networks In classical computing, as pioneered by John von Neumann, a set of rigid instructions are fed-in to a computer in the form of algorithms, which are performed sequentially, one after another. In this traditional paradigm, upon which our laptops, mobile phones and calculators rely, microchips containing over a billion transistors, each capable of switching on nanosecond timescales, enable our electronic devices to perform most tasks on imperceptibly short timescales. However, when charged with a task such as facial recognition, classical computing can run into great difficulty.

100-step Rule When a human sees an image of a familiar person or object, it takes approximately $0.1s$ to discern who or what they are seeing. When one considers that a biological neuron has a switching time in the order of $10^{-3}s$, this accounts for around 100 computational steps being carried out for a successful recognition (Feldman and Ballard, 1982).In comparison, a traditional silicon chip can perform almost no meaningful computation after 100 sequential steps. The key to this speed lies in the intrinsic parallelism of neural networks, which are able to compute on many different inputs simultaneously rather than follow a strictly imposed regime.

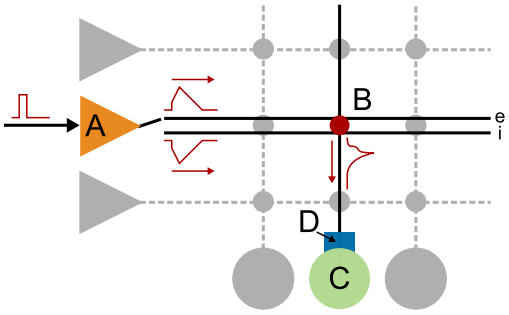
Artificial Neural Networks An artificial neural network is commonly described as a "black box" of neurons, linked to one another through directional, weighted connections. The weights (or efficacies) of these connections must be plastic (adjustable), as in a biological network of neurons. Contrary to the operation of a sequential computer, the neural network is not given a script to perform, but rather a scheme, a simple weight-tuning algorithm, under which it must learn. There are three main learning schemes for neural networks known as *unsupervised learning*, *reinforcement learning* and *supervised learning*. In *unsupervised learning*, the network is only given input patterns, which it learns to differentiate between and classify into categories as it sees fit. *Reinforcement learning* is another possible learning scheme that takes a "carrot or stick" approach (Chialvo and Bak, 1999). As well as being given inputs, the network is told, after every sequence, whether its classification was right or wrong, so that it can tweak its weights in the right direction. This scheme is clearly more effective than unsupervised learning in the context of artificial neural networks. *Supervised learning* schemes are yet more viable for computing tasks, as they not only tell the network whether is right or wrong, but also the exact response expected of the output neurons.

2.3. Neuromorphic Hardware

Out of the demand for an alternative to modelling neural networks on traditional, von Neumann style computers, the Electronic Vision(s) Group at the University of Heidelberg developed a fully-configurable, neuromorphic microchip "Spikey" (Schemmel et al., 2006, see Figure 4a), with 384 leaky Leaky Integrate-and-Fire (LI&F) neurons and 100k programmable conductance-based synapses (see Figure 4b). This chip is a proof-of-concept for a future, wafer-scale neuromorphic device, which will be capable of simulating a highly-interconnected network of around 200,000 neurons connected through $5\text{x}10^7$ synapses (Schemmel et al., 2008, 2010; Brüderle et al., 2011).



(a) A photograph of a "Spikey" neural network chip. The two chip blocks are mirrored along the synapse drivers (inputs), each consisting of 192 LI&F-type neurons and 256×192 synapses. The irregular structure in the lower part is due to auto-routing in digital support circuitry.



(b) Schematic of the synapse blocks. A digital spike (square pulse) arrives at a synapse driver (A), gets converted to a current ramp, multiplied with an 4-bit weight (B), converted to a conductance pulling the membrane voltage (C) to the corresponding reversal potential.

Abbildung 4: Two diagrams illustrating the layout of the "Spikey" chip.

2.3.1. The Leaky Integrate-and-Fire Model

The Integrate-and-Fire model is a spiking neuron model developed in 1907 by French neuroscientist, Louis Lapicque, which has since become one of the best-known models of a spiking neuron. An Integrate-and-Fire (I&F) neuron's membrane is modelled as a capacitor which charges until a threshold voltage is reached, at which point it discharges (see Figure 5). The model has been refined from its original construction to the more useful Leaky Integrate-and-Fire, which rejects the assumption that the neuron's membrane is a perfect insulator and introduces a more realistic "leak", causing the charge in the capacitor to wane over time. This mimics the biologically-observed diffusion of ions through the membrane, which occurs if no equilibrium between influx and outflux is reached.

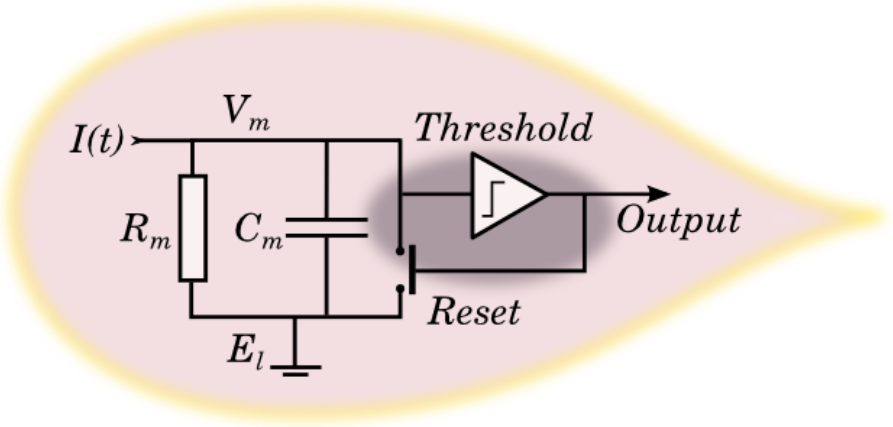


Abbildung 5: Circuit diagram of a LI&F neuron. A RC-circuit with capacitance C_m and resistance R_m is mimicking the membrane of a neuron and is loaded by an external current $I(t)$ which causes an increase of the membrane potential V_m in relation to the resting potential E_l. If the membrane potential crosses a threshold V_{thresh}, V_m is reset.

A circuit diagram of an LI&F neuron is shown in Figure 5 and its mathematical description is given by

$$C_m \frac{\mathrm{d}V_m(t)}{\mathrm{dt}} = I(t) - \frac{V_m(t) - E_l}{R_m}, \tag{1}$$

and its update condition

$$V_m(t) = E_l, \quad if \quad V_m(t) > V_{thresh}. \tag{2}$$

The firing frequency is

$$f(I) = \begin{cases} 0, & I \leq I_{th} \\ (t_{ref} - R_m C_m log(1 - \frac{V_{th}}{I R_m}))^{-1} & I > I_{th} \end{cases}$$

2.4. The Perceptron

One of the very first, and one of the simplest types of neural networks, called the "Perceptron", was invented over fifty years ago. It consists of some set of inputs (which were in the original instance

connected to an output "neuron" through motor-driven potentiometers) acting analogously as plastic synapses (Rosenblatt, 1958). The perceptron is made up of a simple feed-forward network, with a single weight layer (see Figure 6) that can learn to perform binary linear separation tasks. Classification of an arbitrary number of sets can be made possible by combining perceptrons in a multi-layer system. A perceptron classifies its input based on a linear combination of input characteristics, separating the input space into two sets by a hyperplane, drawn orthogonally to the weight vector $\vec{\omega}$ (see Figure 7).

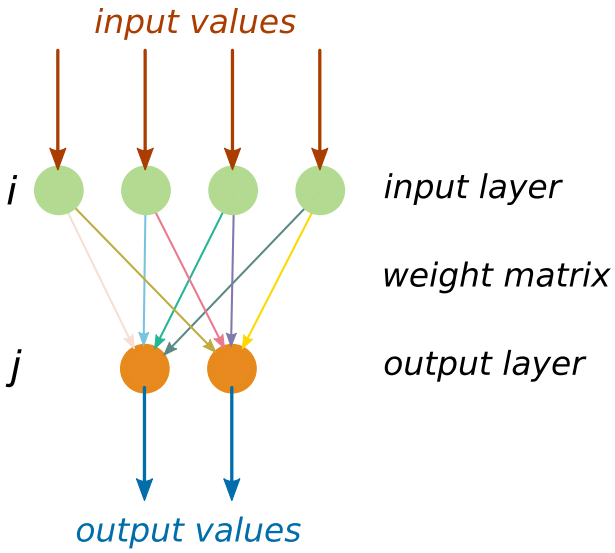


Abbildung 6: Diagram of a perceptron. A input layer of neurons i is connected to an output layer of neurons j via synapses. The strengths of the synaptic connections are defined by the weight matrix.

The output of the perceptron can be described as a weighted sum over the input vector $\vec{x}$ compared to some threshold value γ_{thresh}:

$$f(\vec{x}) = \begin{cases} 1 & \text{if} \quad \vec{w} \cdot \vec{x} + \gamma_{\text{thresh}} > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

i.e. if the input vector falls above the hyperplane, it is classified as input 1 and when below the hyperplane as input 0.

A common learning rule applied to train the weights of a perceptron is the *delta learning rule* that has been proven in Novikoff (1963) to converge to the correct classification provided that the input is linearly separable. The weight ω_{ij}^n of the connection from neuron i to neuron j (see Figure 6) after the nth learn step is updated by:

$$\omega_{ij}^{n+1} = \omega_{ij}^n + \Delta\omega_{ij}^n \tag{4}$$

where

$$\Delta\omega_{ij}^n = \alpha(n) \cdot (T_j - O_j) \cdot x_i \qquad . \tag{5}$$

$\alpha(n)$ is the learn rate and T_j is the target output whilst O_j is the observed output of the perceptron. When the output is correctly classified $(T_j - O_j) = 0$.

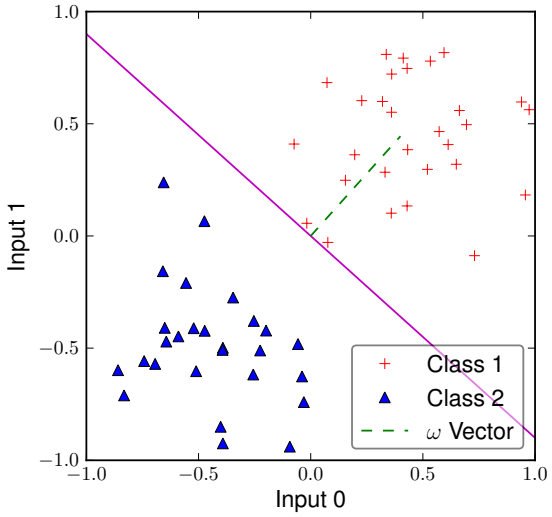


Abbildung 7: Example input space of a linear classifier. The space is divided by a hyperplane drawn orthogonally to the weight vector $\vec{\omega}$ and passes through the origin of the coordinate system (since $\gamma_{\text{thresh}} = 0$). For any point $\vec{x_p}$ in class 1, it holds that $\vec{w} \cdot \vec{x_p} > 0$ and for any point in class 2 it holds that $\vec{w} \cdot \vec{x_p} < 0$, respectively. Used with generous permission of Sebastian Jeltsch.

The perceptron, although a useful model, was not designed to use on spiking neurons and therefore cannot be run directly on the Spikey chip. However, the *Tempotron*, as described in the following chapter, can be successfully modelled on the neuromorphic hardware.

2.5. The Tempotron

The Tempotron is a single LI&F neuron, taught with a supervised learning rule to configure its synaptic weights to discern between two different input patterns (Gütig and Sompolinsky, 2006). For example, we connect a population of neurons to the Tempotron, and stimulate its neurons with two different patterns, X and Y. Before learning, with randomly assigned weights, the Tempotron will achieve an average classification rate equivalent to a coin toss, around 50%. During learning, the Tempotron is taught to emit exactly one spike in a certain timeframe when it observes pattern X, and no spike whilst observing pattern Y. A Tempotron trial has four possible outcomes (see Table 1), in the case of X_1 and Y_0 where the Tempotron has classified correctly, the weights of its connections to the inputs remain unaltered. In the cases of X_0 and Y_1 the Tempotron must be trained by adjusting the weights of its inputs.

The Tempotron, in its originally proposed form, is an LI&F neuron with current-based synapses (where the weights are positive for excitatory connections and negative for inhibitory). The subthreshold membrane potential $V_m(t)$ of a Tempotron with i afferent synapses of weight $\omega_i$can be written as a weighted sum of the postsynaptic potentials (PSPs) of incoming spikes:

$$V_m(t) = \sum_i \omega_i \sum_{t_i} K(t - t_i) + E_l \qquad . \tag{6}$$

Possible outcomes of a Tempotron trial

Outcome	Input	Output	Correct
X_1	X	1	Yes
X_0	X	0	No
Y_1	Y	1	No
Y_0	Y	0	Yes

Tabelle 1: Possible outcomes from the one trial of the Tempotron where pattern X is identified as eliciting a spike in the Tempotron. The weights are only updated in the instances of outcomes X_0 and Y_1

where $K(t-t_i)$, the PSP arriving at synapse i due to a spike time t_i is:

$$K(t-t_i) = V_0 \left(\exp\left(-\frac{t-t_i}{\tau}\right) - \exp\left(-\frac{t-t_i}{\tau_s}\right) \right) \cdot \Theta(t-t_i) \qquad . \tag{7}$$

The constants $\tau_m = R_m C_m$ and τ_s denote the time constant of membrane integration and synapse integration respectively, thereby determining the shape of the PSP. Θ is the Heaviside step function, defined as

$$\Theta(x) = \begin{cases} 0, & x < 0 \\ 1 & x \geq 0 \end{cases} \tag{8}$$

hence the Tempotron only learns in the case of mistaken classifications. In the aforementioned cases of X_0 and Y_1, the weights of the afferent synapses need to be updated. To do this, we start by defining a cost function

$$E_\pm = \pm(V_{\text{thresh}} - V_m(t_{\max}) \cdot \Theta\left(\pm(V_{\text{thresh}} - V_(t_{\max}))\right) \tag{9}$$

where $t_{\max}$ is the time at which the Tempotron's highest potential was recorded. The $(+)$ applies to instances when the Tempotron ought to have spiked, but did not and therefore needs to raise certain weights, whilst the $(-)$ is for the opposite case where the Tempotron has fired in error and the afferent weights must be reduced. The cost function measures how wrong the classification was as the difference between the V_{thresh} and $V_m(t_{\max})$. We want to minimize this cost function through learning in order to maximize the correct classification. As a first order approximation, the weight of the ith synapse needs to be shifted by $\Delta\omega_i$ in the opposite direction to the gradient of $E_\pm$:

$$-\frac{\mathrm{d}\, E_\pm}{\mathrm{d}\omega_\mathrm{i}} = \pm \sum_{t_i < t_{\max}} K(t_{\max} - t_i) \pm \frac{\partial V_m(t_{\max})}{\partial t_{\max}} \frac{\mathrm{d}\, t_{\max}}{\mathrm{d}\omega_\mathrm{i}} \qquad . \tag{10}$$

since $\partial V_m(t_{\max})/\partial t_{\max} = 0$ is always true by definition of $t_{\max}$, the second term of equation 10 can be discarded. With $\Delta\omega_i \sim -\,\mathrm{d}\, E_\pm / \,\mathrm{d}\omega_\mathrm{i}$ we find that weight change for the nth learning run can be calculated by

$$\Delta\omega_i^n = \alpha(n) \sum_{t_i < t_{\max}} K(t_{\max} - t_\mathrm{i}) \qquad . \tag{11}$$

where $\alpha(n)$ represents a learn rate, which can be defined to change (normally decay) over the learning period. For case X_0, when X was suppose to induce a spike but didn't, $\Delta\omega_i$ is added to the ith weight, whereas for case Y_1, it is subtracted.

2.6. Liquid Computing Paradigm

Finite-State Machines (FSMs) are a useful mathematical abstraction used as a tool to design algorithms involving transitions between a fixed number of stable states. A simple FSM could be a lamp with two *states*, on and off, which moves between states according two *transitions* "switch on" and "switch off". In contrast, a Liquid State Machine (LSM), rather than operating on stable states, is designed to function using transient states, thereby providing a method to approximate arbitrary, time-continuous filters.

2.6.1. Motivation and Biological Relevance

First proposed in Maass et al. (2002), the *liquid computing paradigm* proposes non-task-specific, recurrent neural network which projects applied inputs to a high-dimensional space before passing them to a classifier (see Figure 8). Important features of the liquid are that it has a fading memory, it acts on time-continuous inputs and that it is a "black box" *i.e.* the computation processes inside are unknown. A *liquid* can enable classifiers to perform above and beyond their normal limits.

A neat property of the concept of LSMs is that neither a certain substrate nor a particular structure must be enforced, meaning that any sufficiently complex system can be used as an LSM as long as it satisfies the following to properties:

Separation Property A requirement imposed to the underlying substrate itself, which encourages it towards a non-chaotic regime in the way that the differences between inputs (their "distances") are preserved. In this way input pairs that are close together for a given metric do not mix in the output with inputs, providing the inputs are sufficiently different.

Approximation Property A requirement imposed more on the utilised readout rather than the liquid, namely it dictates that the readout needs to be capable enough to carry out the actual computation based upon the high-dimensional projection of the input into the liquid. However, the computational power required from the readout also depends on the size and quality of the liquid. In other words, the more effective the underlying liquid is, the less computation needs to be carried out by the readout.

Since an LSM can compute on time-varying inputs in real-time, it has the potential to shed light on the behaviour of biological neural networks, neural coding in particular. As it stands, there are several incongruities between artificial networks of spiking neurons and their biological counterparts. Although it is possible to build a recurrent network of spiking neurons, which can perform Turing-style tasks, they are overly sensitive to realistic levels of noise, and also require a central clock to control the neurons - something non-existent in the human cortex. If one were rather to look at attractor network models, the need for a clock would be eliminated whilst the resilience to noise would also be increased. Such networks, however, take time to converge to attractors (and are hence of no use for computation on a time-continuous input) whilst if one wishes to store information in an attractor network, one would need 1024 attractors for just 10-bits.

2.6.2. Composing the Hardware LSM

In the utilised hardware setup, input patterns are projected onto the higher-dimensional space stretched by the liquid (see Section 3.1.1) which is then read-out by a Tempotron classifier, both of which share the same chip. Projecting an input onto a high-dimensional liquid can improve the classification abilities of the Tempotron (see Section 3.3.2).

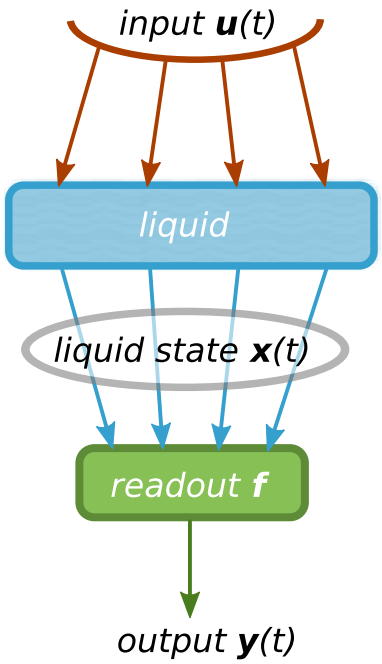


Abbildung 8: A mathematical model of a Liquid State Machine. An input $u(t)$ is transformed by the liquid, the state $x(t)$ of which is mapped onto the readout f which generates the output $y(t)$

Object Classification This is applied increasingly applied in areas of research and industry, from self-driving cars voice recognition. When carried out by a neuromorphic device, instead of through a conventional computer, the classification is often both quicker and requires less power.

3. Experimental Methods

This section will outline the methods applied in the experiments, taking care to highlight potential difficulties (Section 3.1.3) and specific parameters (Section 3.2) arising from the construction of the system on the Fast Analog Computing with Emergent Transient States (FACETS) Stage 1 Hardware. Here, three tasks will also be defined (Section 3.3) to test the abilities of the classifier in three different ways.

3.1. Object Classification with a Liquid State Machine

This section introduces the realisation of a full system-on-a-chip linear classifier, consisting of a Liquid State Machine (as first described in Sussillo et al. (2007)) and conductance-based Tempotron (adapted from Gütig and Sompolinsky (2006)) that can learn to differentiate between input patterns based on their spatio-temporal structure.

3.1.1. Liquid Architecture

An issue with many neural network architectures is that they have to be optimized to function correctly when presented with a specific type of input. For example, the input could be too strong, causing a liquid like that proposed in Maass et al. (2002) to behave chaotically, or could be too weak to invoke any response at all.

Self-Stabilising Liquids Sussillo et al. (2007) proposes a solution to the task-specific tuning required to make a neural circuit function correctly. A liquid is built with two populations, one excitatory, one inhibitory. The populations are connected to one another through synapses which dynamically adjust their weights to stabilise the network according to the principles of short-term plasticity, therefore enabling it to compute a large range of inputs.

An instance of this network has been shown to function robustly on the chip-based hardware system in Bill (2008) and Bill et al. (2010). However, due to a design fault in the current version of the Spikey chip, one of the two chip's two cores has been rendered unaddressable. As the hardware was intended, it would have been possible to run self-stabilising liquid without adjustments. However with only one core in use, it is not possible to separate the inhibitory and excitatory populations in a way that would allow each to have unique Short-Term Plasticity (STP) properties. Depressing STP for the synapses from the excitatory populations is considered to be more important to maintain a stable network than the facilitating STP of the inhibitory synapses, therefore the inhibitory synapses applied have no STP dynamics, and the excitatory are given depressing STP characteristics. Figure 9 illustrates how this network is constructed on the chip-based hardware system.

3.1.2. Realising a Tempotron on Hardware

The readout lies at the heart of any LSM. In this case, a Tempotron (as described in Section 2.5) is adapted from the original current-based synapse models as described in Gütig and Sompolinsky (2006) to run in conjunction with a liquid on the hardware.

Synapses on the Spikey chip are conductance-based with 4-bit digital weights. This necessitates a reformulation of the original current-based synapse learning rule to suit the hardware as demonstrated by Jeltsch (2010). In the original current-based model, weights were initialised to be normally distributed around 0, with negative weights acting as inhibitory and positive as excitatory, which then evolve according to the Tempotron learning rule. Translating this

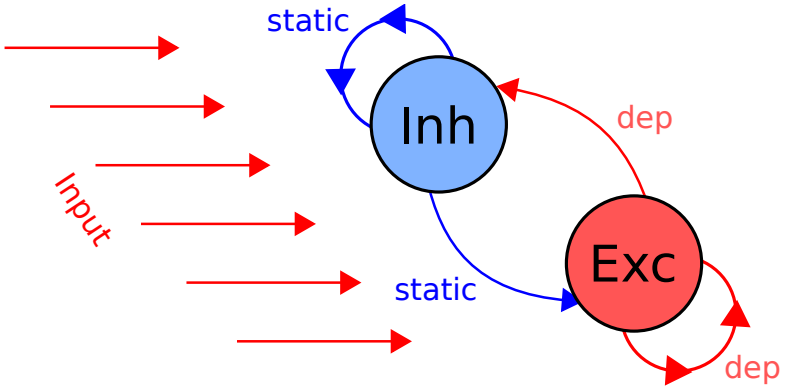


Abbildung 9: The self-stabilising liquid architecture implemented on the neuromorphic hardware by Jeltsch (2010), where each of the inhibitory and excitatory populations connect both to themselves and to the other, with the synapses of the excitatory neurons harnessing dampen strong inputs through depressing STP to ensure the overall fire rate of the liquid remains stable.

method to conductance-based synapses is not trivial, since the PSPs for excitatory and inhibitory conductance-based synapses are asymmetric.

The contribution of a single conductance-based synapse to the neuron membrane is the solution of the equation 1 with

$$I(t) = \omega_i g_0 \left(V_m(t) - E\right) \exp\left(-\frac{t - t_i}{\tau_s}\right) \qquad \text{for } t \geq t_i \qquad ,$$

which can be can be written as:

$$V_m(t) = E_l \exp\left(-\frac{t - t_i}{\tau_m}\right) + \frac{1}{C_m} \int_{t_i}^{t} \omega_i g_0 \left(V_m(t') - E\right) \exp\left(-\frac{t' - t_i}{\tau_s}\right) dt' \qquad .$$

Here ω_i the synaptic weight; g_0 the base synaptic conductance; t_i the time of the afferent spike which activates the synapse; E the reversal potential (either excitatory or inhibitory, depending on the synapse type); and τ_s the synaptic time constant. In the case of the current-based synapses, the voltage-dependent term in the integral disappears, making an analytic solution straightforward, as described in Section 2.5. Even assuming that the voltage term remains approximately constant during a single PSP (which may be plausible if there are many, relatively weak synapses, but is definitely not true for e.g. the triplet scenarios described in Gütig and Sompolinsky, 2006), synaptic contributions remain voltage-dependent:

$$V_m(t) \sim \omega_i \cdot \left(E - V_m(t_i)\right) \cdot K(t - t_i) \qquad ,$$

with the kernel K as described in Equation 7.

Out of the four possible outcomes of a trial on the Tempotron, it is only able to learn in two of them: when the Tempotron fires within the time window when it should not; and when it emits no spike in the window, but should have (see Table 1. In the latter case the Tempotron relies on being able to calculate the time of maximum voltage and raising the weights accordingly. This is the optimal rule for machine learning, but is thought to be biologically unrealistic since research has not shown the neuron to be able to accurately measure and store its membrane trace. It also

causes problems for the neuromorphic hardware too since a maximum of 4 membrane potentials (and hence 4 Tempotrons) can be simultaneously recorded through oscilloscopes, but in practice this is restricted by the number of scopes available and the volume of network traffic generated by recording voltage traces.

It is possible to train the Tempotron using a software simulator such as NEURON (Hines and Carnevale, 2006) or NEST (Gewaltig and Diesmann, 2007) and to translate the learnt weights to the hardware-realised Tempotron. Although this method can achieve high rates of classification, it is not able to learn from a real-time input, which is important for applications of neurally-inspired networks, such as handwriting and speech recognition, as well as being as fundamental property of a biological neural network. There is some inherent error in converting weights learnt on software to those learnt in hardware, since the hardware has not yet been calibrated to exhibit the same consistent behaviour as can be achieved in software simulators (See Figure 11).

The Modified Tempotron: a Quasi Spike Timing Based Linear Classifier By avoiding the measurement and storing of the membrane potential, the speed of classification by the Tempotron could be increased through the reduction of communication between the chip and the host computer, whilst incidentally making the learn rule more biologically feasible (though it is still by no means a biologically realistic). The learning rule for the outcome of a desired, but unachieved spike, X_0, was changed so that the time of peak membrane potential, t_{max}, was assigned with a random point in time inside the classification window. The complete learning rule is therefore given by Equation 11 and the modified t_{max} as follows:

$$t_{max}(O,T) = \Theta(T-O)\,\mathtt{rand}_p() + \Theta(O-T)\min\{t^i_{out}\} \qquad . \tag{12}$$

Where T, O are two binary variables which reflect the desired target response and actual response of the Tempotron respectively. Θ is the Heaviside step function, $\mathtt{rand}_p()$ is a sample drawn from a given random distribution p over $[0, t_{window}]$ and t^i_{out} are the output spikes of the tempotron recorded within its classification time window. Hence, if $O = T$ then t_{max} will be zero and no learning occurs. But if T and O differ, then either a random time within the window will be chosen as t_{max} (case X_0) or the time of the first output spike will be chosen respectively (case Y_1).

It was also considered to remove the learning rule for the case of X_0 altogether. If the weights were only be updated in the case of an unwanted spike and consequently inhibitory weights would only be strengthened, whilst excitatory weights would only be weakened. Considering a certain amount of noise, fluctuations and some chaotic behaviour in the liquid this would render all inhibitory inputs to reach their maximum efficacy and all excitatory inputs to die out. Such an unbalanced weight update rule can never reliably produce stable results and was therefore neglected.

To further improve the effectiveness of the Tempotron's learning, an exponentially decaying learn rate (a similar technique to the Thermal Perceptron Rule where the weights "cool off" with time Frean (1992)), shown to be effective in Jeltsch (2010) This "cooling" is applied to the Tempotron in order to improve classification.

3.1.3. Potential Difficulties of a System-on-a-Chip Linear Classifier

Synaptic Connections The topological constraints, caused by the chip design error has some further implications worthy of consideration. Since the synapse drivers can only be adjusted to a single set of parameters, any neuron in the excitatory population of the liquid can only be connected to other neurons through excitatory connections, and likewise for inhibitory neurons. This means that the type of synapse attached to the Tempotron must stay the same throughout

learning, contrasting to the original model, in which the synapses were freely able to switch between excitatory and inhibitory. It goes hand in hand that the ratio of excitatory to inhibitory synapses connected to the Tempotron must be the same as the ratio in the liquid.

Perhaps an even larger problem with the connectors comes from the STP setting. Since the liquid must be stabilised by connecting the excitatory population to the inhibitory using depressing synapses, the excitatory population must also connect to the Tempotron using the same dynamics. As can be seen in Figure 10, a persistent stimulus will cause the synaptic efficacy to be dynamically lowered. Since STP is implemented in the same synapse drivers from the excitatory population of the liquid, (where the effects of STP are essential to prevent it firing chaotically, as they are connected to the tempotron) it is likely that any fast-firing neuron in the liquid will have its synaptic weight to the tempotron dynamically depressed.

It must also be considered that the digital weights in hardware are defined by 4 bits, therefore providing a resolution of 16 possible weights.

Calibration of Hardware Biological neural networks are naturally inhomogeneous, noisy and unpredictable. The FACETS Stage 1 Hardware, Spikey, also has imperfections arising from the manufacturing process and the general nature of analogue circuitry. Although it can be argued that some of these imperfections make Spikey a more realistic biological emulator, they can also make running experiments on the chip difficult until they are properly calibrated.

The synaptic drivers and the threshold voltages are particularly capricious and in need of calibration if reliable experiments are to be run on Spikey. The hardware fluctuations are depicted in Figure 11.

3.2. Implementation of anLSM on Chip

Tempotron Parameters Earlier experiments of Jeltsch (2010) established optimal parameters for a hardware-realised Tempotron, the specific parameters of the LI&F neuron (see Table 2) are also applied to the modified Tempotron. There are many additional parameters with an influence on the learning of the modified Tempotron. The hardware weights in this study were initialised as a folded normal distribution with $\mu_\omega = 0\,\mathrm{nS}$ and σ_ω given as $\frac{\omega_{max}}{15}$, equal to the minimum non-zero weight available for a given chip (see Section 3.1.3). The learn rate of the modified Tempotron decayed exponentially over the learning period starting from an initial rate of 1 and a time constant $\tau = 700$.

Liquid Parameters Parameters for the liquid have already been determined (Bill, 2008) to produce a self-stabilising liquid on the hardware. Unless otherwise stated (such as in Section 4.2.5, these have also been implemented here. The liquid was composed of 191 neurons, divided into two populations of 143 excitatory and 48 inhibitory neurons. 64 external stimuli are applied, 32 connected through excitatory synapses and 32 through inhibitory to randomly selected neurons in the liquid, each given a weight of $\omega = 0.003\,\mathrm{nS}$.

3.3. Classification Tasks

The classifier was trained to classify three different tasks in order to explore its capabilities and limitations. Firstly, it was given the simple task of classifying between two trains of Poisson spikes, as used by Jeltsch (2010), followed by a task specially designed by Dimitri Probst to be unclassifiable by a lone Tempotron, and finally a simple digit recognition task to demonstrate a real-world application.

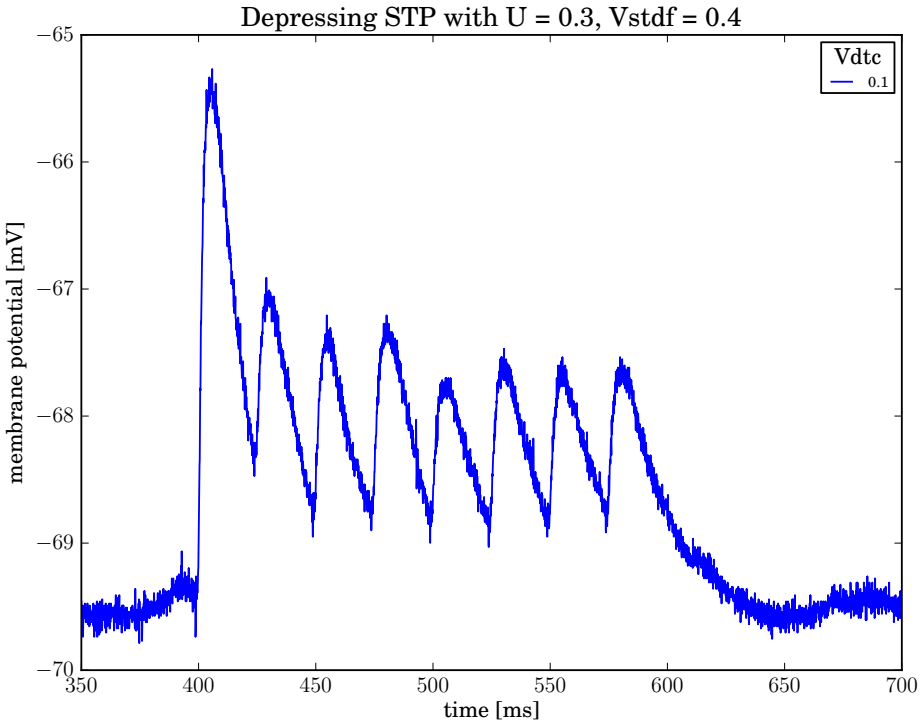


Abbildung 10: This plot shows the average of 10 membrane potential recordings of a hardware neuron under the influence of depressing STP. Presynaptic spikes of equal strength arrive at a frequency of 25 Hz. The impact of the PSP is diminished after each successive spike until a steady minimum is reached, thus reducing the probability of the neuron producing an action potential as would happen if the connections were through static synapses. Whilst the input frequency remains high, the limit of the membrane potential will remain at approximately the same value. After a period of no spikes, the amplitude of PSPs at an incoming synapse will recover towards its maximum at a rate defined by a time constant τ_{rec}. U,V_{stdf} and V_{dtc} correspond to three parameters of STP: Utilisation of synaptic efficacy, scale factor for STP strength and time constant for spike history respectively, applied according to the mechanism described in Tsodyks and Markram (1997)

For every task, one of two possible patterns was presented as stimuli to the liquid. One pattern, X or Y chosen at random, was presented per classification window. The response of the Tempotron was observed and the weights adjusted accordingly.

3.3.1. Simple Task

The first task involved the classification of two Poisson process spike sequences vectors (X and Y) Downarowicz (2008) (See Figure 12). These were generated for N spike sources with a mean frequency of 30 Hz and for a duration of 1250 ms. Each sequence was then cut into 50 ms slices

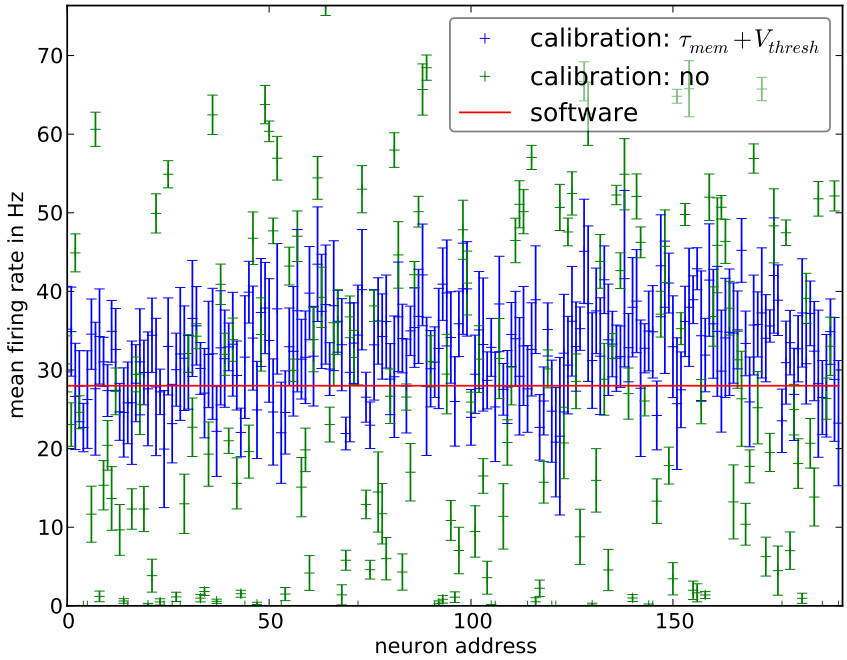


Abbildung 11: Illustration of the hardware mismatch. One can clearly see, how the calibration heavily reduces the deviation from the theoretical value given by the red line. Nevertheless, a certain amount of fluctuations remain. Besides, the error is given by the standard deviation. (Figure taken from Sebastian Jeltsch.)

where 50 ms corresponds to the time window of the Tempotron. A sequence of 1250 ms was built by appending 25 randomly chosen slices from the corresponding time slices of either pattern X or Y, one after another and labelling each slice with the pattern, from which it came. A Gaussian 'jitter' is applied to each spike time with a standard deviation of $\sigma = 2$ ms.

3.3.2. Difficult Task: Impossible for Tempotron Alone

The Tempotron is a very robust classifier, however, one specific task has been formulated which cannot be classified by a Tempotron alone. Pattern X is defined as all N spike sources firing at $10ms$ into the slice, and pattern Y firing 5 ms from the end. All spikes were again jittered with a standard deviation of $\sigma = 2ms$. The Tempotron is unable to recognise pattern Y since it falls too close to the end of the slice to induce a spike.

3.3.3. Real-Life Task: Character Recognition

Neural networks are commonly and successfully applied to problems which cannot easily be programmed with conventional algorithms. Face, speech and handwriting recognition, are among the tasks where neural networks achieve the best known results. A simple task was defined as the classification of two different handwritten characters generated as shown in Figure 13.

Tempotron		Liquid	
Parameter	Value	Parameter	Value
V_{reset}	$-63\,\text{mV}$	V_{reset}	$-80\,\text{mV}$
V_{thresh}	$-55\,\text{mV}$	V_{thresh}	$-55\,\text{mV}$
V_{rest}	$-58\,\text{mV}$	V_{rest}	$-58\,\text{mV}$
E^I_{rev}	$-80\,\text{mV}$	E^I_{rev}	$-80\,\text{mV}$
E_l	$-58\,\text{mV}$	E_l	$-58\,\text{mV}$
g_l	$20\,\text{nS}$	g_l	$40\,\text{nS}$
$\tau^E_{\text{syn}}/\tau^I_{\text{syn}}$	$2.5\,\text{mS}$	$\tau^E_{\text{syn}}/\tau^I_{\text{syn}}$	$2.5\,\text{mS}$

Tabelle 2: The default set of LI&F neuron parameters which are used for the following studies, unless otherwise stated.

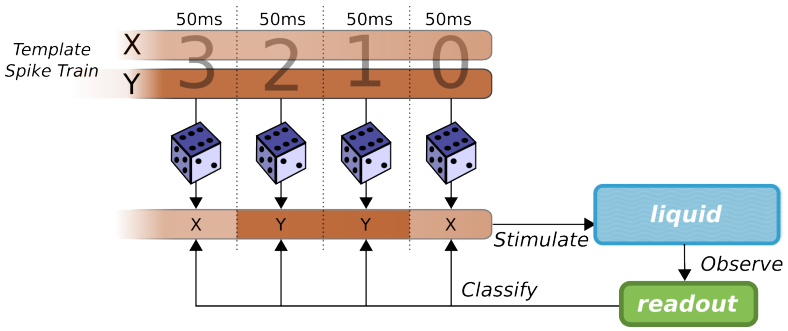


Abbildung 12: An illustration of the simple task. Two patterns (X and Y) are generated and sliced at 50 ms intervals. A spiketrain is then created by randomly choosing either pattern X or Y for any given interval and copying the slice from the same temporal position in the appropriate original pattern. This spiketrain is then presented to the liquid, and observed by the Tempotron. One Tempotron is required per readout frame from the recent past, making use of the fading memory of the liquid (Section 2.6).

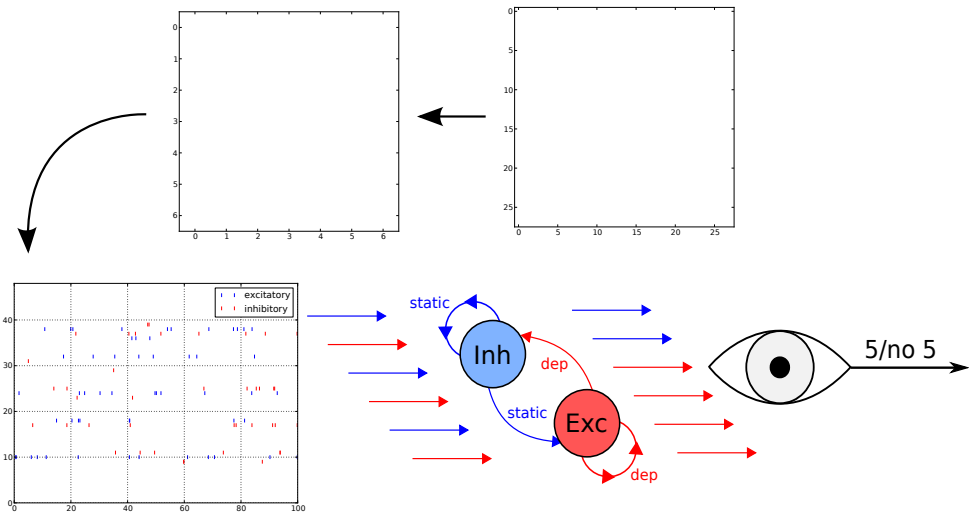


Abbildung 13: Conversion of handwritten characters into spike trains. The characters (here a '5' is shown) are recorded originally stored as 28x28 pixel images, which are shrunk to 7x7 pixels, then the tone of each pixel corresponds to exactly one input spiketrain with a Poisson generated frequency corresponding to the tone of the pixel. Figure used with friendly permission of Dimitri Probst.

4. Experimental Results and Discussion

The aim of these experiments was to construct, test and optimise an LSM, learning and classifying through spike time information, able to make use of the acceleration of the FACETS Stage 1 neuromorphic hardware chip. Results are shown demonstrating the classifier's abilities and limitations for performing the tasks described in Section 3.3. Firstly, the general behaviour of the LSM as a whole is analysed, then specific optimizations of the Tempotron are explored, and finally the power of the machine to classify between different types of patterns is presented. Where appropriate, notes will be made on the biological relevance of results.

A note on errors The most frequently applied measure to determine the capability of the classifier is correctness. All measurements of correctness are made from 40 classification trials in order to offer a low error without increasing the time of required for computation. The error is calculated as the standard error for N trials,

$$SD = \frac{\sigma}{\sqrt{N}} \tag{13}$$

where σ is the standard deviation of the population. In the case of 100% correctness, $\sigma = 0$ and therefore the standard error will be calculated to be 0. In this case it may be more appropriate to consider the maximum possible error as being the error on correctness that would be if the very next trial yielded a misclassification.

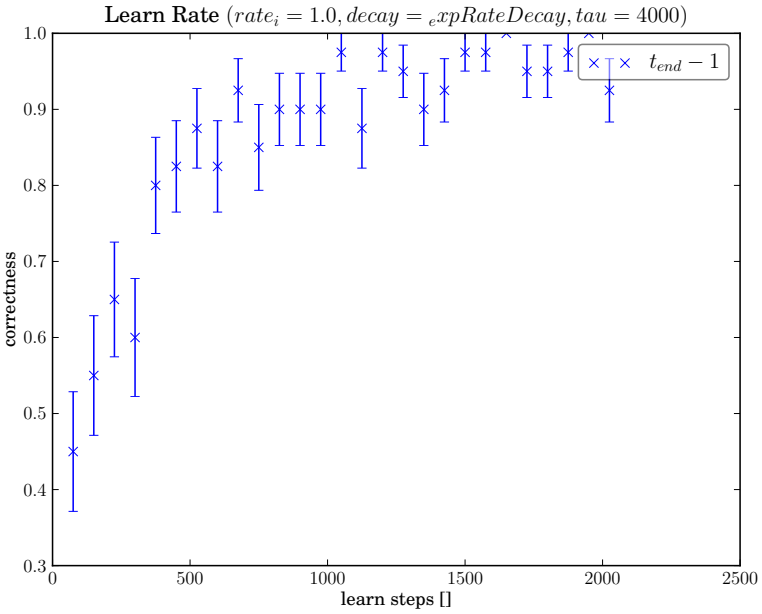
4.1. Preparatory Results: Towards the System-on-a-Chip

Building directly from the work of Jeltsch (2010), experiments were being carried out to optimise the speed of learning of the LSM. The machine was originally designed to train the Tempotron using a software simulator such as NEURON (Hines et al., 2008) from cached states of a hardware implemented liquid and translate the weights to a hardware-implemented Tempotron to be tested on a particular classification task. Learning was carried out in increments of 75 steps, after each increment the weights were translated to the hardware Tempotron and the correctness of classifying 40 presentations of a pattern was tested.

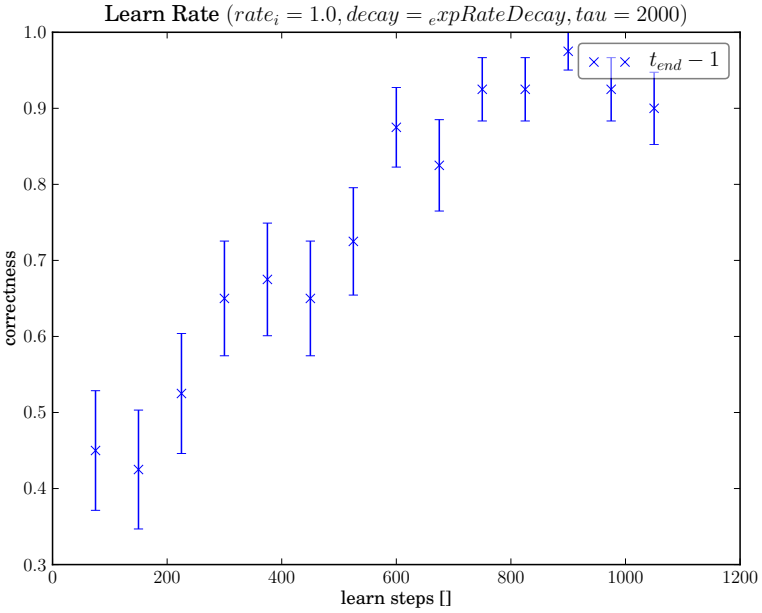
A self-stabilising liquid utilising all 192 neurons of the hardware, with an excitatory population of 144 and an inhibitory of 48, was run for 10,000 presentations, and the liquid state was recorded to a file each time, later to be presented to the Tempotron. This method of using cached liquid states were used in all preliminary experiments. These early studies were mostly made out based on a Tempotron observing *Frame-1* of the liquid, to test the ability of a classifier to utilise the fading memory of the liquid.

The very first trials of the Tempotron were carried out using NEURON simulated Tempotrons in order to explore the effect of different training parameters on its ability to classify simple patterns as defined in Section 3.3.1. This had already been demonstrated to achieve a high degree of correctness, however the trained weights of the Tempotron were mapped exclusively to excitatory synapses in hardware and loading cached liquid states led to long real time scales of training.

Due to the constraints of the hardware as described in Section 3.1.3, it was important for the training to be possible making use both excitatory and inhibitory synapses in order to utilise the full size of the liquid. This was first implemented using a model in NEURON and was shown to achieve a high rate of classification (see Figure 14a). The model was also realised for a hardware Tempotron using weights mapped from one trained in software without significant loss of classification efficacy (Figure 14b).



(a) Training of software simulated Tempotron to classify a pattern in *Frame-1* using states cached from a liquid run on hardware. It is clear that the correctness quickly rises from chance levels to above 90% between 500 and 1000 learn steps.



(b) Here a NEURON Tempotron was trained, then the weights were translated to hardware and its classification abilities were tested. It displays an learn rate of learning similar to the NEURON Tempotron shown in Figure 14a

Abbildung 14

Whilst performing experiments on this system, an error was made in the setup of an experiment which led to Tempotron being trained solely using the spike-time information generated by the hardware itself. According to the original Tempotron learning rule, the time of the maximum potential is required for learning in the X_0 case, in order update the weights as to increase the likelihood of the of a spike being emitted in subsequent trials. However, this was not possible in this experiment, since no oscilloscope was connected to the chip to record the membrane potential. Surprisingly the Tempotron was still able to learn to achieve a correctness of approximately 80% (Figure 15).

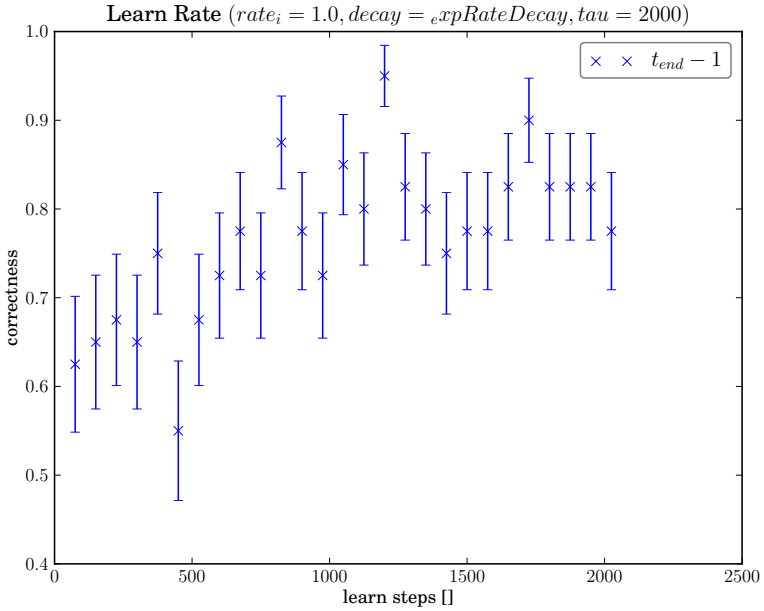
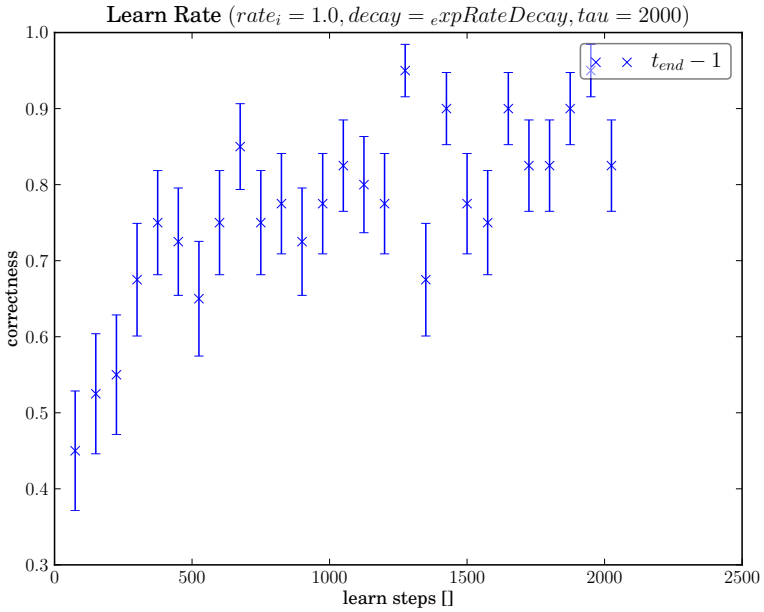


Abbildung 15: Due to an error in setup, an experiment was run whereby a Tempotron was learning in hardware purely in the case of outcome Y_1. This result was unexpected, since it had previously been assumed that the classification of a Tempotron would always remain around chance level unless the full Tempotron learning rule was applied.

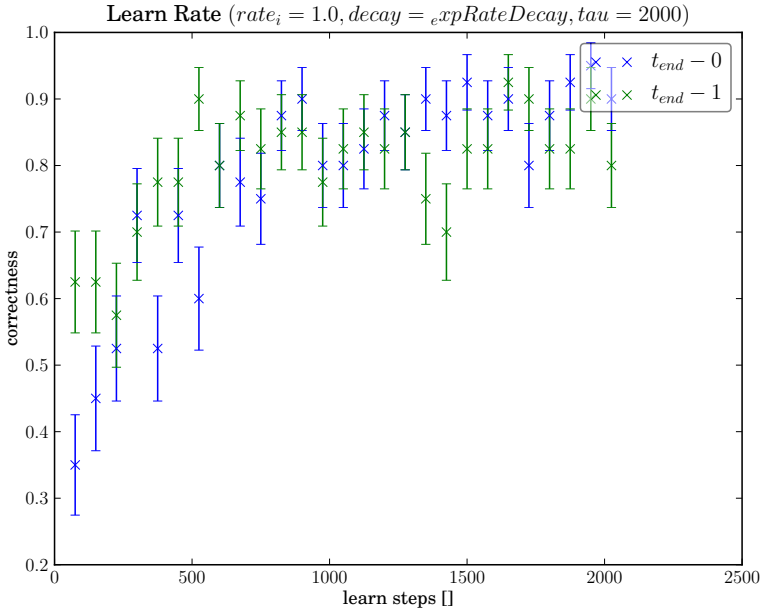
This prompted investigations into the efficacy of a Tempotron trained directly on hardware. A logical next step was to explore whether recording the membrane trace using an oscilloscope in order to determine the value of t_{max} could indeed improve the effectiveness of the classifier (Figure 16a). It was clear that Tempotron was able to learn classify patterns with a correctness of over 80%, although it was observed that the oscilloscope would often record several values for t_{max} in which case the t_{max} was redefined as the first instance of the maximum voltage.

As a reference, it was considered whether simply setting a t_{max} to be a random time within the window would produce similar results to recording the voltage trace using an oscilloscope. It was demonstrated (Figure 16b) that this led to results similar to training the Tempotron utilising an oscilloscope.

Given that implementation of an LSM on neuromorphic hardware, using purely hardware-generated spike data, had never previously been realised, and that it could enable fast learning and effective classification of patterns presented in real-time, it was decided to pursue this goal.



(a) Here the full Tempotron learning rule was applied on hardware by making use of an oscilloscope to record the membrane potential. Learning is clearly not as effective as that shown in Figure 14a, although the correctness of the Tempotron's classification does improve with time nonetheless. This could be arising from the imprecision of the instruments.



(b) Two Tempotrons are being trained with cached liquid states, one to classify patterns in *Frame-0* and the other in *Frame-1*. They are being trained purely on hardware by taking the time of maximum membrane potential in the case of X_0, t_{max} to be a random value within the time window. Learning to a high level of correctness takes longer than in the case of a NEURON-simulated Tempotron, however it is still able to reach a correctness of around 90%.

Abbildung 16

4.2. Tuning the Behaviour of the Liquid State Machine

4.2.1. Spiking Behaviour of the LSM

Aims In order to determine whether the LSM was functioning correctly, the spiking behaviour of all neurons in the population was observed. It was expected that the liquid would remain in a stable state throughout the duration of each trial.

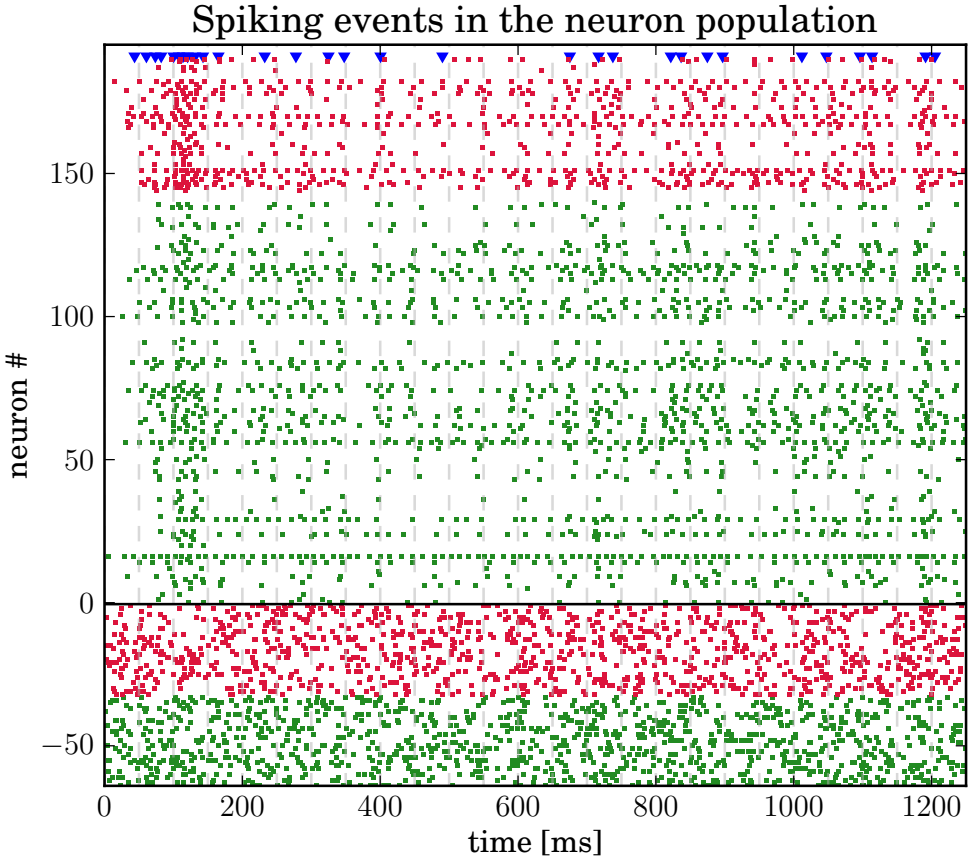


Abbildung 17: Spiking events in the population of neurons on the LSM receiving an input as described in Section 3.3.1. Excitatory and inhibitory populations are shown as green and red squares respectively. The blue triangles represent spikes in the Tempotron. Neuron numbers -64 to -1 emit spikes according to the pre-defined tasks, 0 till 190 are the neurons of the liquid and 191 is the Tempotron itself. The activity of the liquid remains more or less stable throughout the experiment, however note the brief swell of the liquid's activity which occurs at around 100 ms. Experiments carried out on the liquid, without STP enabled, showed that the liquid could quickly tend to permanent chaotic behaviour at this point

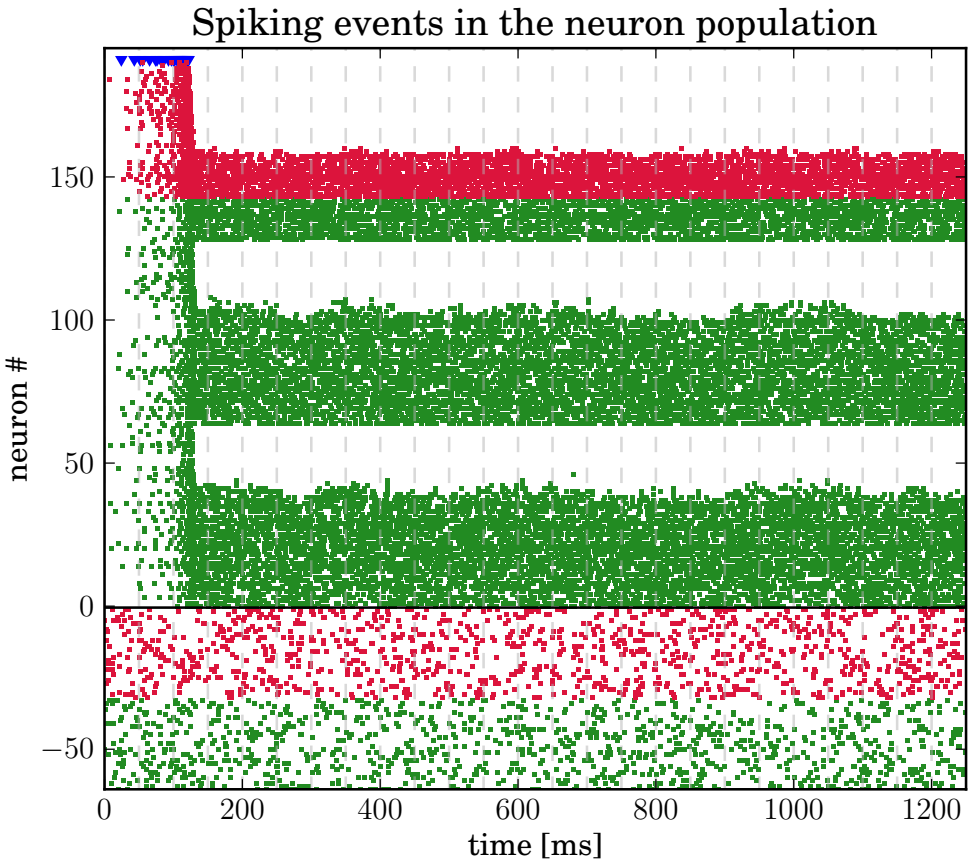


Abbildung 18: The Liquid State Machine with STP disabled. Here it can be seen that what was a brief swell in activity in Figure 17 quickly snowballs to chaotic firing without the depressing effects of STP applied. The plot is missing spikes which could not be recorded due to traffic constraints on the chip's buffers.

Results The expected behaviour of the LSM can be seen to be the case in Figure 17, where the liquid fires with a fairly stable frequency when presented with inputs and induces spikes in the Tempotron accordingly. It was also noticed that the liquid's stability did not hold through every run and would occasionally tend to chaotic behaviour as seen in Figure 18, which shows that the chip's buffers limit the number of spikes which can be recorded.

Discussion The reason for this sporadically chaotic behaviour is likely to arise from chip variations as seen in Figure 11. It could be that the liquid's parameters, taken as those used by Bill (2008), functioned well on a different chip, but are not appropriate for the chip I was using. However, the relative rarity of these events meant that slight adjustments to the carefully chosen parameters would not have been an efficient use of time in this study. Since the Tempotron is situated in a position on the chip where its spikes are lost, it will be taken as not to have fired in

this periodded, will be strongly adjusted accordingly in favour of inducing a spike for the next trial, according to Equation 11, since the firing rate of all neurons in the liquid is so high that $(t_{max} - t_i)$, the difference in time between a presynaptic and postsynaptic spike will be small no matter where t_{max} happens to fall. Therefore, these chaotic liquid events are likely to reduce the classification rate of the liquid should t_{max} coincide with an unwanted spike, or increase the classification rate if it happens to coincide with a correct spike.

Biological Relevance The qualitative spiking behaviour of the stable liquid is remarkably similar to those seen disassociated cortical cultures (Rolston et al., 2007), suggesting that biological results don't render the LSM biologically implausible.

4.2.2. Evolution of Weights over Time

Aims An important validation of the learning process of the Tempotron can be found through observing the course of the synaptic weights evolution during training. In Gütig and Sompolinsky (2006, Figure 7d), it can be seen that the originally proposed Tempotron that during a number of training runs the synaptic weights briefly remain at their initialised values before a handful are rapidly increased and the remainder are gradually suppressed towards 0, accompanied by a sharp decrease in false classifications. This evolution would be expected for any Tempotron.

Results The evolution of the weights in this system is qualitatively very similar, however not precisely as expected. The handful of weights rising to maximum are clear in Figure 19, however, as seen in Figure 20, distributions differ in that some weights rise only to remain at middle values rather than increasing to the maximum and that the low weights are not noticeably suppressed towards zero over the training period.

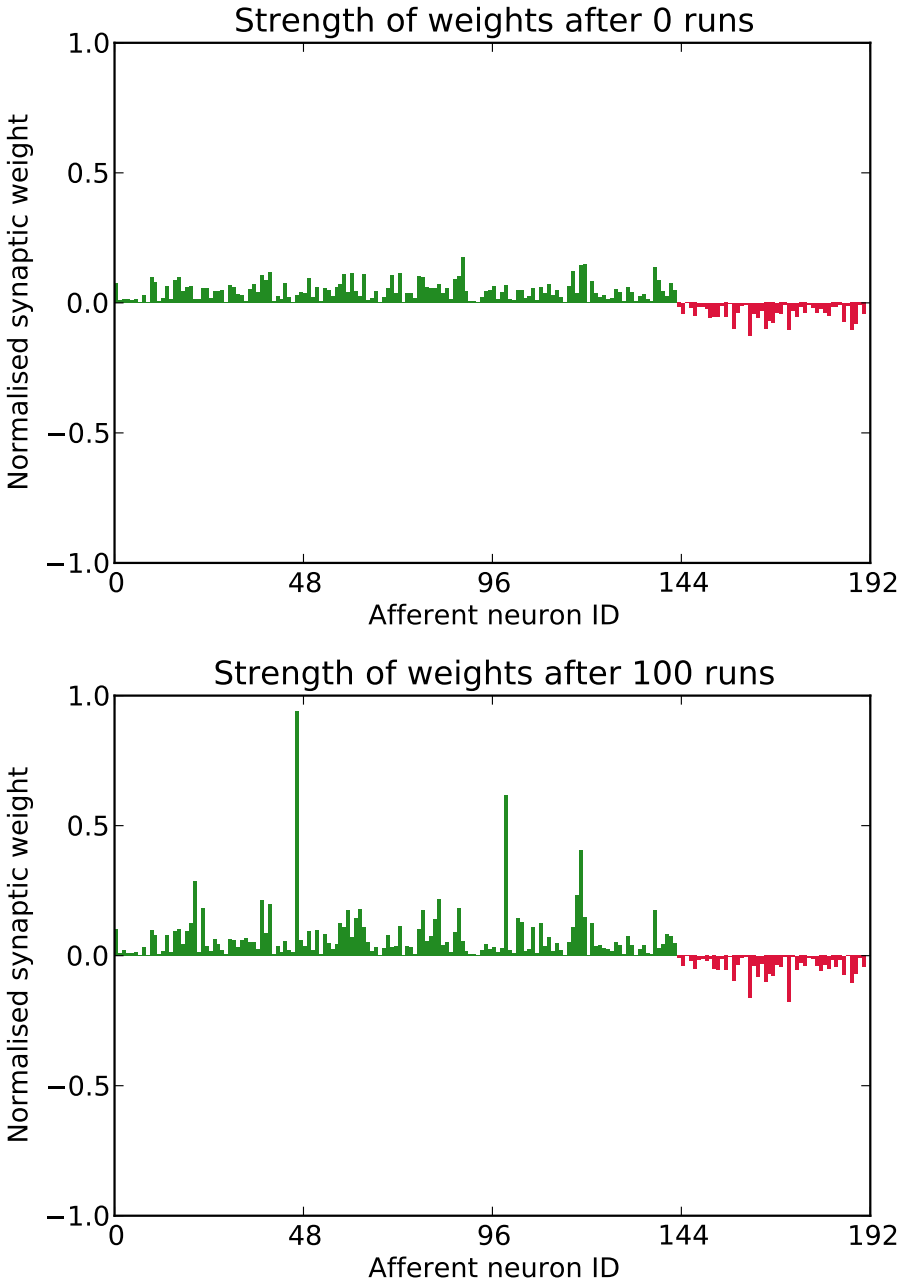
Discussion Although the evolution observed here is very similar to those known for the original Tempotron, it is useful to explore differences as potential causes of error learning and classification of the modified Tempotron. The differences could stem from the use of a liquid to process the signal before it is read out by the Tempotron, which can introduce additional noise, or the chaotic firing events as discussed in Section 4.2.1. Another possibility could be the discrete digital weights (Section 3.1.3), which means that some changes in weights in program do not lead to a change in the real synaptic weights in the hardware.

4.2.3. Investigating the Assignment of t_{max}

Aims The parameters for the choice of value for t_{max}, the time of maximum membrane potential in the case X_0, would seem to be the most important factor affecting the classification skills of this LSM. It was originally assumed a random assignment for this value would be appropriate since it would be likely to have a similar effect on learning as that of a low resolution membrane potential trace (see Figure 16a).

Effect of Distribution Width for the Assignment of a Random t_{max} on Learning Firstly an experiment was run to test the training efficacy for the random assignment of t_{max} according to different distributions. The values were randomly chosen according to a normal distribution centred around the centre of the time window at 25 ms and clipped between $t_{start} = 0$ ms and $t_{end} = 50$ ms for the values of σ_{tmax} shown in Figure 21.

Abbildung 19: The evolution of individual weights over the training of a Tempotron presented with patterns from the simple task. The weights have been normalised as such that 1 is the maximum possible weight. The excitatory synapses are shown in green and are positive, whilst the red bars represent the inhibitory weights and shown as negative for illustrative purposes, although in the conductance based synapse models as applied here all weights are positive. Note that it's only a few weights which increase, this is similar to the evolution to that seen in Gütig and Sompolinsky (2006, Figure 7d), where the Tempotron weights remain at their initial distribution before a select few of the weights rapidly rise, then plateau.



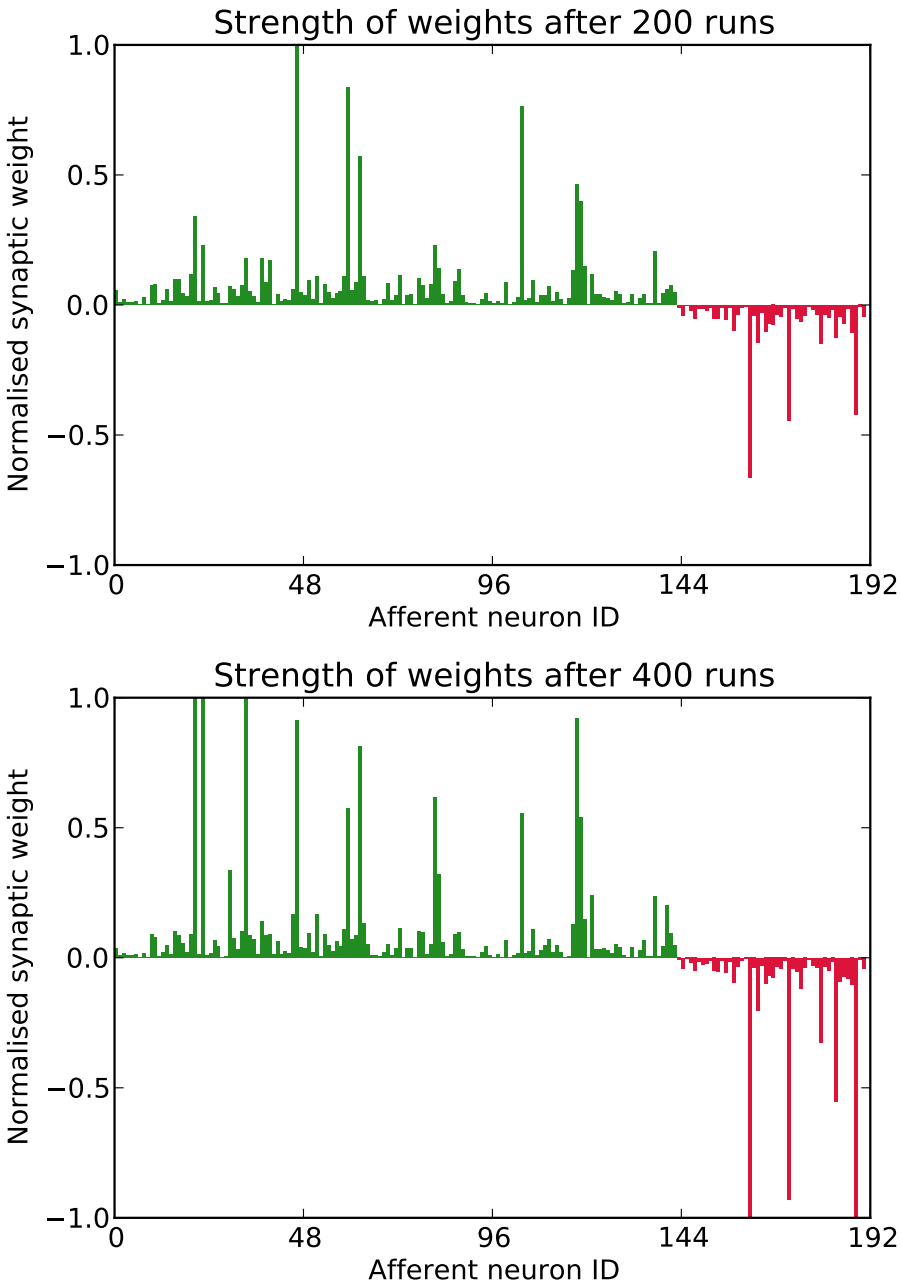


Abbildung 19: continued.

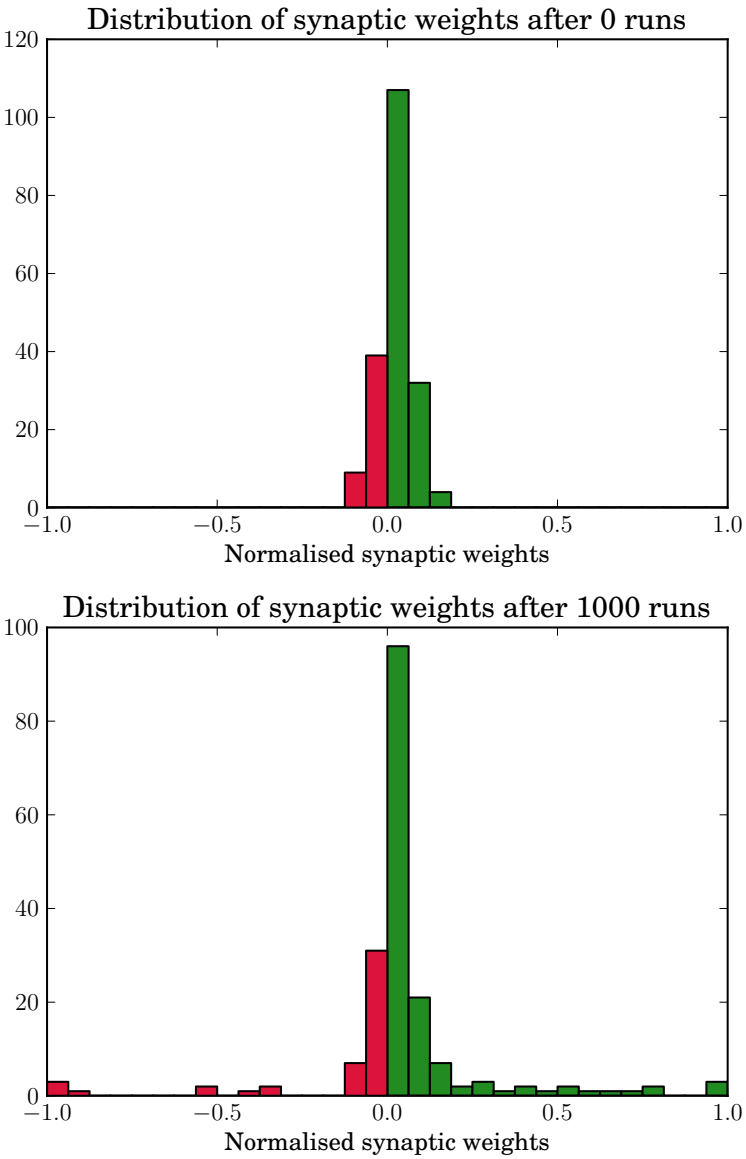


Abbildung 20: These histograms represent the initial and final distributions of afferent weights of the Tempotron. Most of the weights remain approximately the same, but a few evolve to the extremes. This plot shows a difference in the learning of weights when compared to that of the original Tempotron in that the unused weights do not fall to 0 and some weights of around half of the maximum can be found

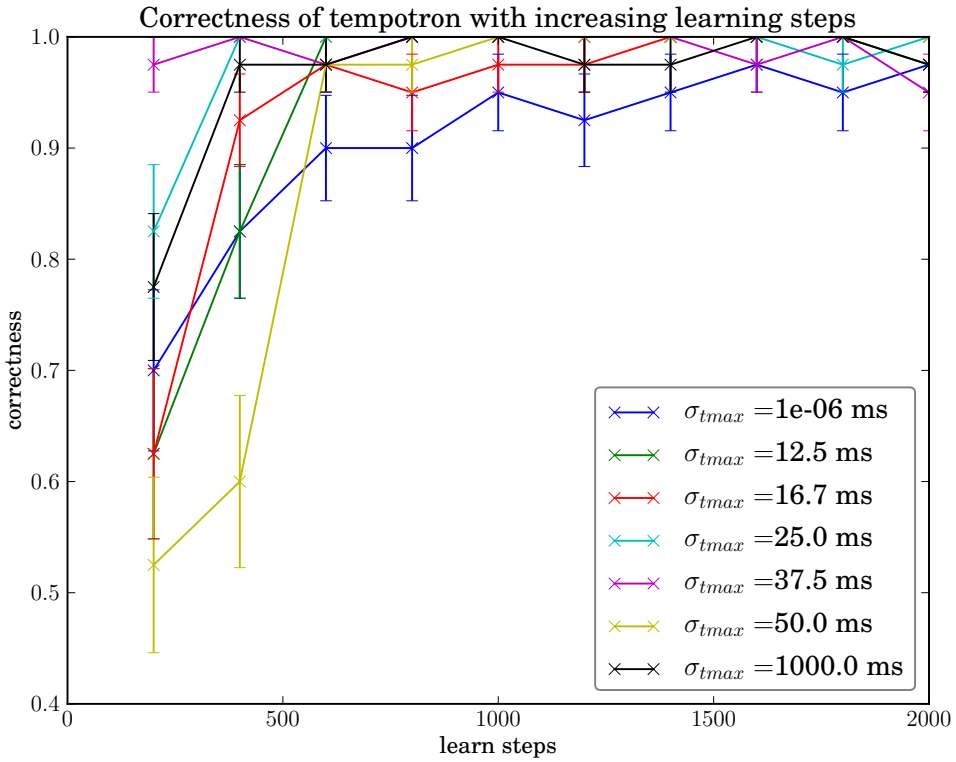


Abbildung 21: This figure shows the results of learning with a randomly assigned value for t_{max} in the case of outcome X_0. The random assignment is according to normally distributed values between 0 ms and 50 ms with $\mu_{tmax} = 25$ ms (halfway through the Tempotron's observational window) with increasing values for σ_{tmax} from what is effectively a delta function through to a uniform distribution. Although the normal distributions all behave differently, it is clear that from 600 training steps they reach a correctness of over 90% whilst the correctness of the delta function rises slower. *Note that lines are drawn to help distinguish between results, they do not represent experimental data.*

Results From the figure it appears that there is no clear correlation between breadth of distribution and speed of learning on the learning of the classifier except in the case of the delta function ($\sigma_{\mathrm{tmax}} = 1e-06$ ms) where its correctness was close to other distributions, however from 600-1600 steps the correctness was lower than every other sigma tried.

Discussion In this learning scheme, true Tempotron learning only happens in the case of outcome Y_1 which leads to the weakening of excitatory weights and the bolstering of inhibitory weights. If were not checked with a rule that caused the inverse to occur, assuming a non-decaying learn rate the excitatory weights would all tend to 0 and the inhibitory all to their maximum. A balancing term is required to prevent this. For lack of a membrane potential measurement which enables a

conventional tempotron to reach an effective weight balance, the modified Tempotron assigns a random value to $t_{\max}$ in the outcome X_0, thereby balancing through the adjusting a small subset of the weights corresponding to the inputs shortly before $t_{\max}$.

Effect of a Fixed t_{max} on Learning In order to determine a suitable centre for the random distribution an experiment was run whereby the value of t_{max} was always set to a certain time in the Tempotron's readout window (see Figure 22).

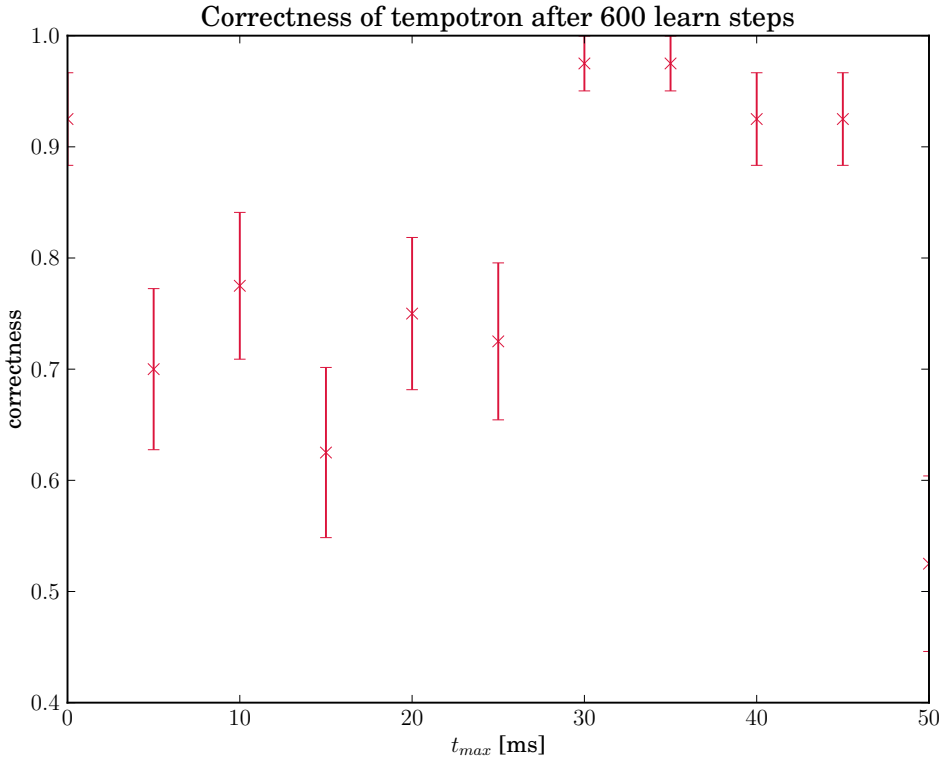


Abbildung 22: Here the value of t_{max} was set to a fixed value after an outcome of X_0. The learning was run over 600 steps, as this was seen to be time taken for the majority of cases to reach over 90% correctness in Figure 21. It appears that setting t_{max} to a time in the second half of the interval, the training was significantly more effective compared to setting it to to a time in the first (see text for explanation).

Results The experiment lasted 600 learn steps since it had been shown in the previous experiment that this classifier could achieve a 90% correctness on this timescale. The figure suggests a critical minimum value for the time of t_{max} of around 25 ms, after which the correctness appears to saturate .

Discussion An explanation for the improvement of classification increasing with the size of t_{max} could stem from the kernel function, which only adjusts the weights of the afferent synapses that spike in the period directly preceding the time of t_{max} (Equation 7). Should t_{max} be fixed to a time early in the classification window, then the weights of all synapses in before this time will be increased. The increased weights will lead to a higher membrane potential at the start of the time window, and since the *leak conductance* of the Tempotron is set to $g_l = 20\,\text{nS}$, corresponding to a time constant for the decay rate of the membrane potential, $\tau_{\text{mem}} = 10ms$, it is clear that on this time scale the membrane could remain excited above V_{rest}, and therefore more likely to spike for the whole period of the window as illustrated in Figure 23. This increased sensitivity could lead to a greater incidence of false classifications for a lower t_{max} due to the longer time of increased sensitivity. These results, however, further validation is required since these results were gathered only learning for the simple task and may vary for other patterns.

4.2.4. Classification of a Simple Pattern

Aims To initially test the tuned classifier, it was again presented with the same simple task (Section 3.3.1) as used in all preceding experiments until this point. This was tested for classifying patterns in *Frame-0* and *Frame-1* in order to determine whether the fading memory property of the liquid could be harnessed by the modified Tempotron.

The more complex task of training the classifier to identify patterns of *Frame-1*, lingering patterns from the previous time slice, proved to be harder for the LSM to achieve (Figure 25).

Results The classifier was trained identify patterns in the most recent *Frame-0* of the liquid over 1500 steps with the correctness being measured at 100 step intervals. It was observed that the classifier is capable of learning quickly and reaching 100% correctness after 300 steps, then classify consistently perfectly from 600 steps until the end of the experiment. This demonstrates the power of the LSM to learn to classify patterns to a high degree of accuracy based purely on spikes.

It appears the LSM was also able to utilise this fading memory of the liquid as explained in Section 2.6 to a degree, though the learn rate remained between 70% and 80% throughout the experiment. An LSM using cached liquid states, harnessing the modified Tempotron learning rule has been demonstrated to classify patterns from *Frame-1* with significantly better results.

Discussion The cause of this is likely to have arisen as a consequence from the move to an System-on-a-Chip (SoC), though it could be an effect of variations in the hardware (Figure 11) amplified through simultaneous use of every available neuron).

Biological Relevance The learn curve shown in Figure 24 has the characteristic sigmoid shape of which is expected from a capable classifier, which has also been observed in the learning of *ex vivo* cultures of cortical neurons (Shahaf and Marom (2001) Figure 6).

4.2.5. Difficult Task

Aims A task was specifically created to be unclassifiable by the tempotron alone, but straightforwardly when it acts in conjunction with a liquid. It has been previously described in Section 3.3.2.

Results Although the tempotron is already a very robust classifier, the results from this experiment demonstrate that it can be made more powerful as part of an LSM. Experiments by Dimitri Probst showed that a Tempotron alone was unable to learn to classify such a pattern after as

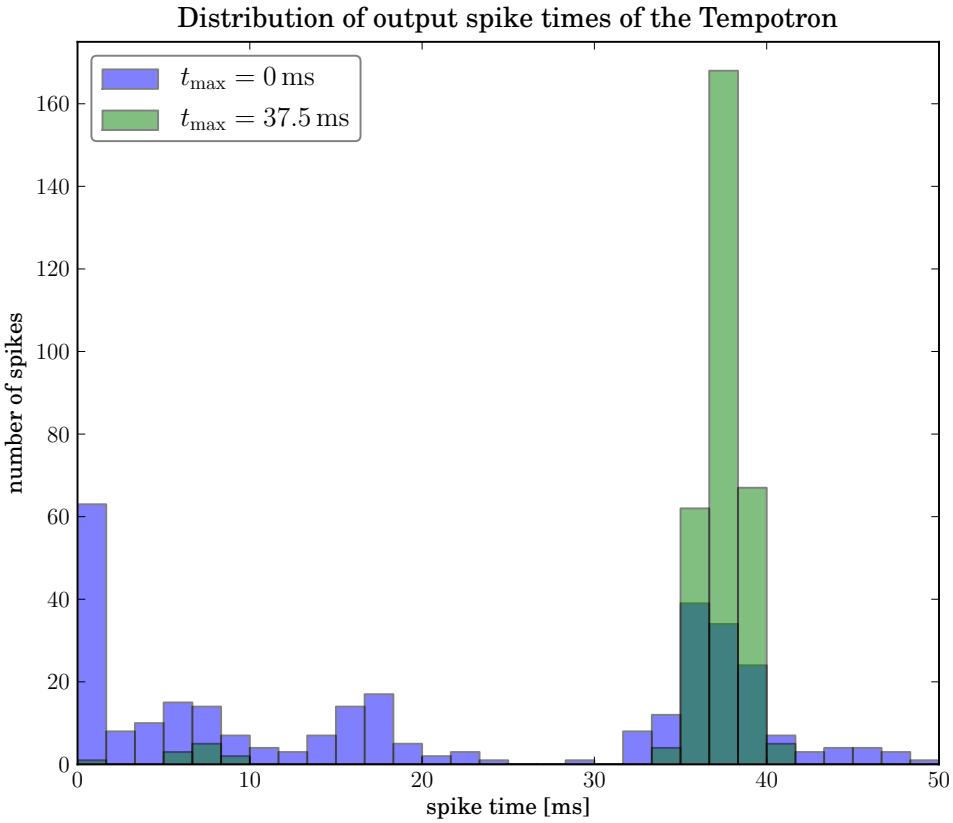


Abbildung 23: The difference in distribution of output spikes after 600 learn steps for $t_{max} = 0\,\text{msor}37.5\,\text{ms}$. This suggests that the if t_{max} is set to a time earlier in the window, the Tempotron is less likely to learn to respond to a single spike, whereas a later time leads to a higher likelihood of a single spike time being singled out. It is not completely clear however, whether this effect relies on one particular spike pattern (in this case, the simple task) to achieve this result and further experiments should be performed to corroborate or contradict this observation for different tasks.

many as 10,000 learn steps (unpublished results), whereas when learning and classifying as part of an LSM, a correctness of over 90% can be achieved (Figure 26).

Discussion In this experiment the frequency of the input spike train was low compared to that of the simple task of classifying Poisson spike trains, and the liquid was found to barely respond. In order to make the liquid and the tempotron fire, the simplest parameter adjustment was decided to be the lowering of V_{thresh} to -57.5 mV, just 0.5 mV above the resting potential and thus making all neurons in the system very sensitive to input stimuli.

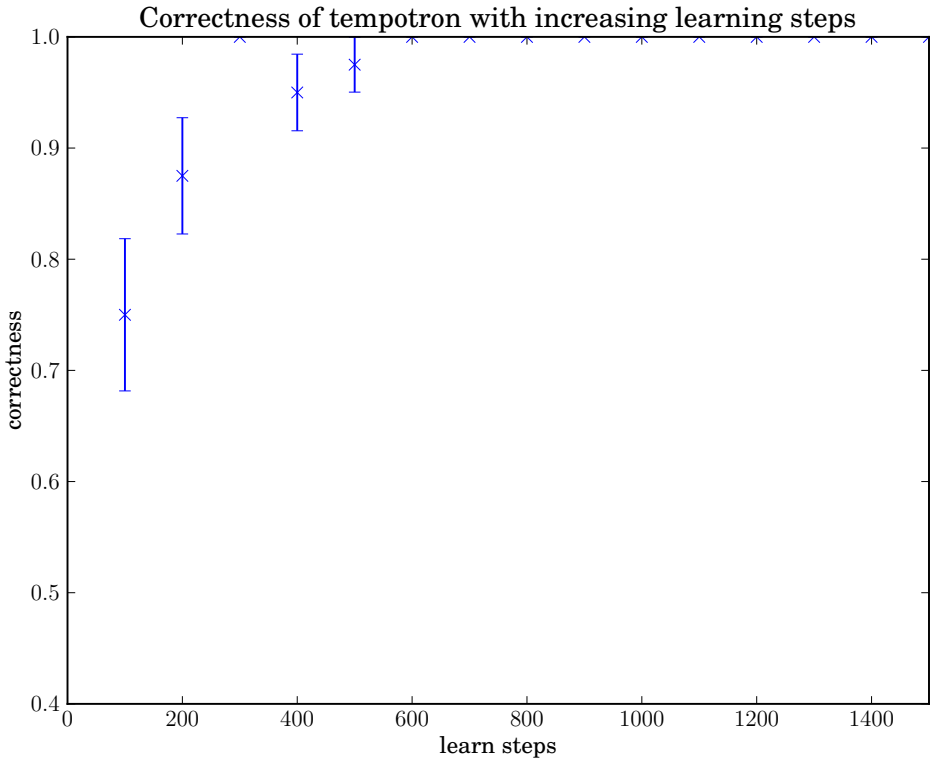


Abbildung 24: The optimised Tempotron was trained to classify two simple patterns (Section 3.3). The correctness quickly rises quickly to a steady 100% after 600 steps

Biological relevance Although this task has been intended as purely demonstrative, it is perhaps interesting to note that imposing a difference of only 0.5 mV between the *resting potential* and the *threshold potential* has not been observed in biology. Measurements *in vivo* of neocortical neurons have shown a difference closer to 5 mV for neurons in 'high-conductance states' (Destexhe et al., 2003).

4.2.6. Character recognition

Aims It was also useful to test the modified Tempotron has also been tested with the real-life character recognition task described in Section 3.3.3.

Results As can be seen in Figure 27a a good rate of classification could be quickly achieved by the LSM for the task of differentiating between 0's and 1's, however Figure 27b shows that differentiating between 0's and 2's was impossible for the classifier.

Discussion As with the task defined in Section 4.2.5 the signal on the liquid was too weak induce a useful number of spikes, and in this case was set to $V_{thresh} = -56\,\text{mV}$ for both the Tempotron and the liquid. The reason for the complete failure of the system to classify between 0 and 2 is

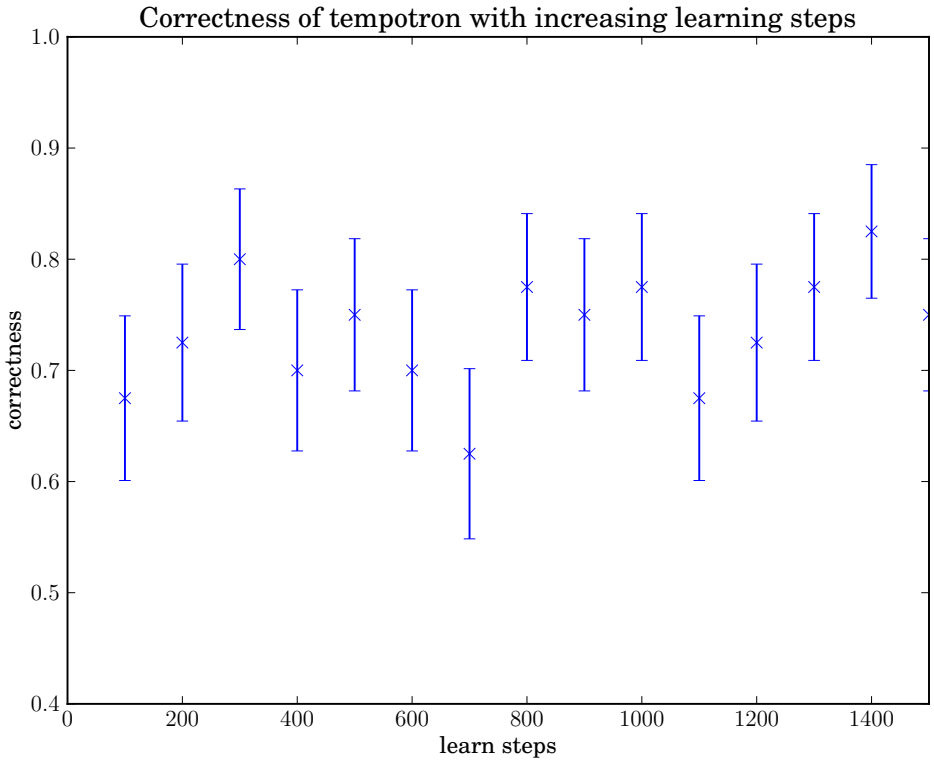


Abbildung 25: The classification efficacy of a Tempotron trained to classify a pattern from *Frame-1* of the liquid. The results suggest that the Tempotron is able to utilise the fading memory property of the liquid, however not as effectively as demonstrated in Figure 16b.

thought to be caused by spiketrains generated by the conversion of images of handwritten 0s and 2s as 7x7 pixels bitmaps into spike frequencies leading to very similar patterns. It is thought that the classification rate for the 0-2 trial could be improved by altering the method for the generation of spiketrains such that each digit leads to a more unique pattern for the Tempotron to classify.

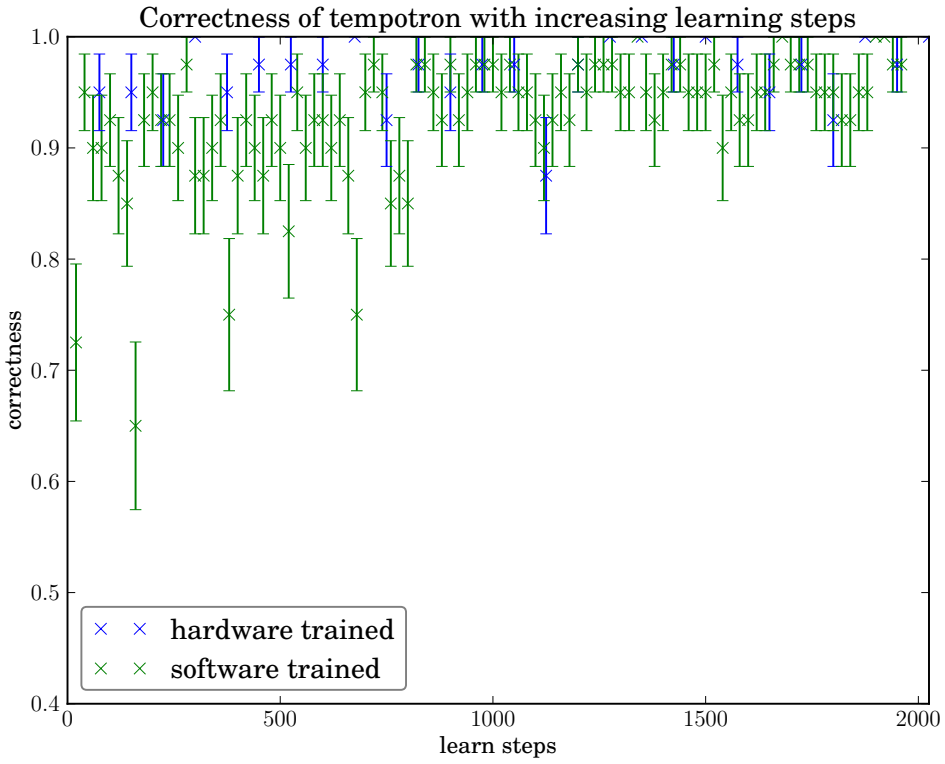
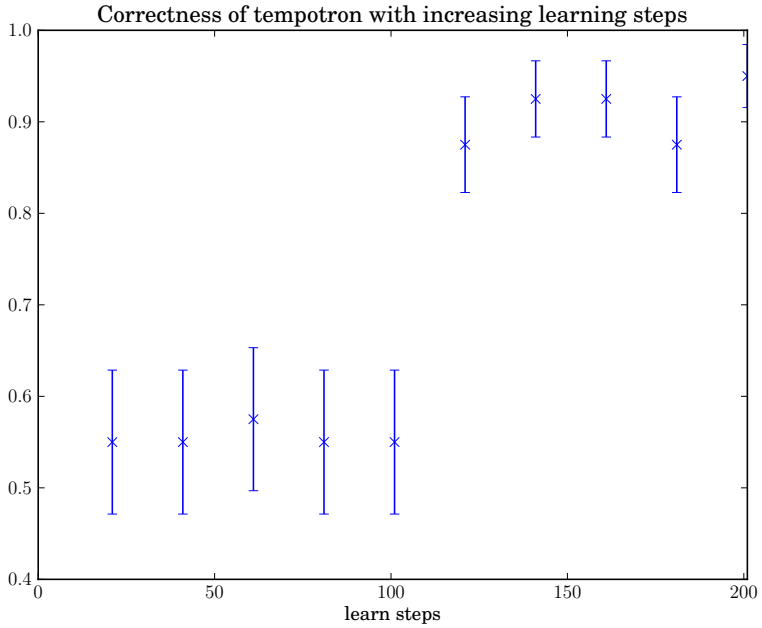
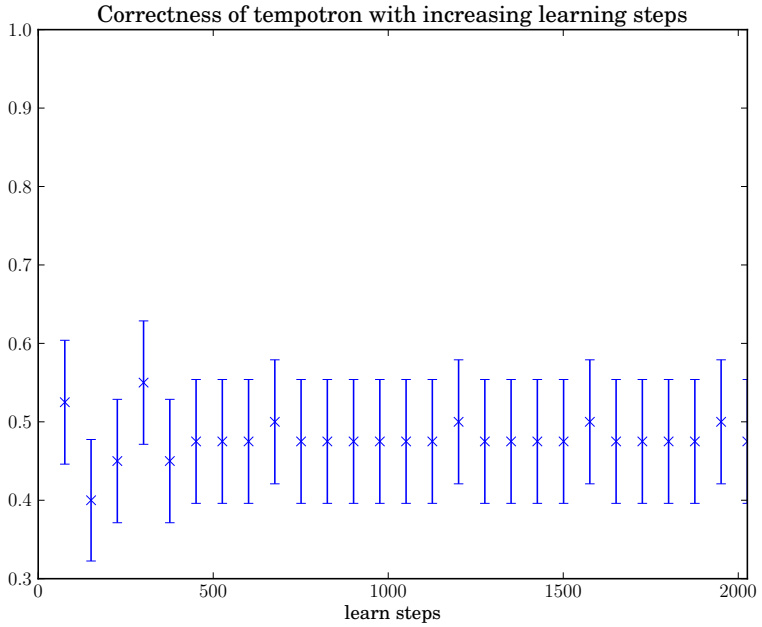


Abbildung 26: The task, which was impossible for a Tempotron alone is shown here to be well classifiable by an LSM results from a tempotron, which was trained offline on software are also shown here, with permission of Dimitri Probst. It appears that the SoC has similar classification capabilities to the offline system for this task.



(a) Classifying between handwritten 0's and 1's. The Tempotron's correctness jumps quickly from chance to near 90% between 100 and 120 learn steps and remained consistently 80–90% until the end of the experiment at 2000 steps (not shown).



(b) Classifying between handwritten 0's and 2's. It appears the SoC is unable to differentiate between these two handwritten digits.

Abbildung 27

4.3. Further Studies

After these initial investigations, there are many further studies which could be carried out to optimise the modified Tempotron and better understand its performance.

Change to weight update rule The tempotron rule applied in this paper, though effective, it is not as neat or powerful as the original tempotron rule. It is proposed that instead of the current learning scheme, it could be more effective to apply a uniform weight update to all afferent synapses within a window in the case of X_0.

Careful observation of learning process It would also be of great interest to monitor the results of every trial and record every change of weights after an unsuccessful classification to determine the discrete moments at which the weights evolve.

Initialisation of weights It has also been considered that the initial weights must play a large role in the speed and efficacy of learning. Since the learning rule receives a correct value of $t_{\max}$ only for the outcome of Y_1 and a random value otherwise, it will be likely to learn quicker if the outcome of the first trials is that of an undesired spike. This could easily be caused by initialising the excitatory weights with a higher mean value than the inhibitory, in which case learning would begin as accurate negative reinforcement learning and potentially quicker.

Purely excitatory weights Yet another option would be to ignore all inhibitory connections on to the tempotron and to connect purely through excitatory. If all synapses were initialised as maximum, then learning could be carried out which reduces the weights down to the correct distribution, rather than building them as was used in this paper.

Forced weight contrast It was noticed in Section 4.2.2 that the distribution of weights did not evolve to the same as was shown in Gütig and Sompolinsky (2006). Although it was suggested that differences could arise from the variations in the hardware, or the use of a liquid, it would be interesting to investigate whether artificially forcing each high synaptic weight to maximum, and every low to zero, would offer an improved classification rate of the LSM.

Implementation on a Multi-Spikey System A new development is on the horizon within the Electronic Vision(s) group called Multi-Spikey – a device built from a group of "Spikey" chips connected and acting together as one. When it is brought online, carrying out these studies with slackened dimensional constraints would allow for larger liquids, read by more Tempotrons, allowing for much more complex tasks to be achieved.

5. Conclusions

This paper aimed to implement a binary classifying LSM, fully on neuromorphic hardware and test its performance on a range of tasks. Through its development, it was discovered that the tempotron rule could be modified to dramatically reduce the volume of information communicated, demonstrating fast, successful learning and classification of 3 unique input patterns.

5.1. Achievements

The experiments shown above demonstrate that an LSM can be fully constructed on the hardware (Section 4.2.1) and learn binary classification tasks based purely on information recorded within the chip itself (Section 4.2.4). The originally proposed Tempotron (Gütig and Sompolinsky, 2006) was required to offer a precise time for the peak membrane potential, however the modified Tempotron, as proposed here, must only communicate the times of any spikes emitted during its observation window. The quantity of information is reduced by introducing a "rough guess" for the time of the maximum membrane potential instead of one that is known (Section 3.1.2), which, although crude, offers good results for a range of tasks (Section 4.2.5, 3.3.2. Through this reduction in the quantity of information required for the Tempotron to learn and classify, the potential acceleration of the hardware can be effectively utilised for learning and classification tasks, although measuring the exact increase in computational speed offered has yet to be done.

5.2. Outlook

Although the classifier could still not be considered a biologically accurate model, and is not as powerful as the original Tempotron (Section 4.2.4) it may interesting for future developments to note that it was possible to train a classifier without requiring any information about its membrane potential. The modified Tempotron as part of an LSM does offer a small step in the direction of a biologically plausible classification model, as well as a fast classifier for neurally-inspired computing tasks. Considering these models were built on hardware of 192 neurons, and future devices are planned to contain around 200,000 (Section 2.3), staggeringly large LSMs could be constructed, which would overcome issues of resolution and be able to learn to differentiate between many complex patterns at vastly accelerated speeds.

A. Acronyms

FACETS Fast Analog Computing with Emergent Transient States

FSM Finite-State Machine

LSM Liquid State Machine

PSP Postsynaptic Potential

I&F Integrate-and-Fire

LI&F Leaky Integrate-and-Fire

STDP Spike-Timing Dependent Plasticity

STP Short-Term Plasticity

SoC System-on-a-Chip

Literatur

J. Bill. Self-stabilizing network architectures on a neuromorphic hardware system. Diploma thesis (English), University of Heidelberg, HD-KIP-08-44, 2008.

J. Bill, K. Schuch, D. Brüderle, J. Schemmel, W. Maass, and K. Meier. Compensating inhomogeneities of neuromorphic VLSI devices via short-term synaptic plasticity. *Front. Comp. Neurosci.*, 4(129), 2010.

D. Brüderle, M. Petrovici, B. Vogginger, M. Ehrlich, T. Pfeil, S. Millner, A. Grübl, K. Wendt, E. Müller, M.-O. Schwartz, D. Husmann de Oliveira, S. Jeltsch, J. Fieres, M. Schilling, P. Müller, O. Breitwieser, V. Petkov, L. Muller, A. P. Davison, P. Krishnamurthy, J. Kremkow, M. Lundqvist, E. Muller, J. Partzsch, S. Scholze, L. Zühl, A. Destexhe, M. Diesmann, T. C. Potjans, A. Lansner, R. Schüffny, J. Schemmel, and K. Meier. A comprehensive workflow for general-purpose neural modeling with highly configurable neuromorphic hardware systems. 104:263–296, 2011. 10.1007/s00422-011-0435-9.

D. R. Chialvo and P. Bak. Learning from mistakes. *Neuroscience*, 90(4):1137–1148, 1999.

A. Destexhe, M. Rudolph, and D. Pare. The high-conductance state of neocortical neurons in vivo. *Nature Reviews Neuroscience*, 4:739–751, 2003.

T. Downarowicz. Law of series/poisson process. *Scholarpedia*, 3(11):3922, 2008.

D. A. Drachman. Do we have brain to spare? *Neurology*, 64(12):2004–2005, June 2005.

FACETS. Fast Analog Computing with Emergent Transient States – project website. `http://www.facets-project.org`, 2010.

J. A. Feldman and D. H. Ballard. Connectionist models and their properties. *Cognitive Science*, 6(3):205–254, 1982.

M. Frean. A "thermal" perceptron learning rule. *Neural Computation*, 4:946–957, Nov. 1992.

M.-O. Gewaltig and M. Diesmann. Nest (neural simulation tool). *Scholarpedia*, 2(4):1430, 2007.

A. Gierer. Spatial organization and genetic information in brain development. *Biological Cybernetics*, 59:13–21, 1988. doi: 10.1007/BF00336886.

F. Gomez-Rodriguez, L. Miro-Amarante, F. D. del Río, A. Linares-Barranco, and G. Jiménez. Real time multiple objects tracking based on a bio-inspired processing cascade architecture. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1399–1402, 2010.

R. Gütig and H. Sompolinsky. The tempotron: a neuron that learns spike timing-based decisions. *Nat Neurosci*, 9(3):420–428, Mar. 2006. ISSN 1097-6256. URL `http://dx.doi.org/10.1038/nn1643`.

P. Häfliger. Adaptive WTA with an analog VLSI neuromorphic learning chip. *IEEE Transactions on Neural Networks*, 18(2):551–72, 2007.

D. O. Hebb. *The Organization of Behaviour*. Wiley, New York, 1949.

M. Hines, J. W. Moore, and T. Carnevale. Neuron, 2008. URL `http://neuron.duke.edu`.

M. L. Hines and N. T. Carnevale. *The NEURON Book*. Cambridge University Press, Cambridge, UK, 2006. ISBN 978-0521843218.

IBM. System blue gene solution. `ibm.com/systems/deepcomputing/bluegene/`, 2010.

G. Indiveri, E. Chicca, and R. Douglas. Artificial cognitive systems: From VLSI networks of spiking neurons to neuromorphic cognition. *Cognitive Computation*, 1(2):119–127, Mar 2009.

S. Jeltsch. Computing with transient states on a neuromorphic multi-chip environment. Diploma thesis, Ruprecht-Karls-Universität Heidelberg, HD-KIP-10-54, `http://www.kip.uni-heidelberg.de/Veroeffentlichungen/details.php?id=2095`, 2010.

N. Kalisman, G. Silberberg, and H. Markram. The neocortical microcircuit as a tabula rasa. *Proceedings of the National Academy of Sciences of the United States of America*, 102(3): 880–885, Jan. 2005.

B. Katz and R. Miledi. The measurement of synaptic delay, and the time course of acetylcholine release at the neuromuscular junction. *Royal Society of London Proceedings Series B*, 161: 483–495, Feb. 1965.

M. A. Lewis, R. Etienne-Cummings, A. H. Cohen, and M. Hartmann. Toward biomorphic control using custom aVLSI chips. In *Proceedings of the International conference on robotics and automation*. IEEE Press, 2000.

W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11): 2531–2560, 2002.

C. A. Mead. *Analog VLSI and Neural Systems*. Addison Wesley, Reading, MA, 1989.

C. A. Mead. Neuromorphic electronic systems. *Proceedings of the IEEE*, 78:1629–1636, 1990.

C. A. Mead and M. A. Mahowald. A silicon model of early visual processing. *Neural Networks*, 1 (1):91–97, 1988.

A. B. Novikoff. On convergence proofs for perceptrons. In *Proceedings of the Symposium on the Mathematical Theory of Automata*, volume 12, pages 615–622, 1963. URL `http://citeseer.comp.nus.edu.sg/context/494822/0`.

S. Renaud, J. Tomas, Y. Bornat, A. Daouzli, and S. Saighi. Neuromimetic ICs with analog cores: an alternative for simulating spiking neural networks. In *Proceedings of the 2007 IEEE Symposium on Circuits and Systems (ISCAS2007)*, 2007.

J. D. Rolston, D. A. Wagenaar, and S. M. Potter. Precisely timed spatiotemporal patterns of neural activity in dissociated cortical cultures. *Neuroscience*, 148(1):294–303, Aug. 2007.

F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, 1958.

J. R. Sanes and J. W. Lichtman. Development of the vertebrate neuromuscular junction. *Annual Review of Neuroscience*, 22(1):389–442, 1999.

J. Schemmel, A. Grübl, K. Meier, and E. Muller. Implementing synaptic plasticity in a VLSI spiking neural network model. In *Proceedings of the 2006 International Joint Conference on Neural Networks (IJCNN)*. IEEE Press, 2006.

J. Schemmel, J. Fieres, and K. Meier. Wafer-scale integration of analog neural networks. In *Proceedings of the 2008 International Joint Conference on Neural Networks (IJCNN)*, 2008.

J. Schemmel, D. Brüderle, A. Grübl, M. Hock, K. Meier, and S. Millner. A wafer-scale neuromorphic hardware system for large-scale neural modeling. In *Proceedings of the 2010 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1947–1950, 2010.

G. Shahaf and S. Marom. Learning in networks of cortical neurons. *J Neurosci*, 21(22):8782–8788, Nov. 2001.

D. Sussillo, T. Toyoizumi, and W. Maass. Self-Tuning of Neural Circuits Through Short-Term Synaptic Plasticity. *J Neurophysiol*, 97(6):4079–4095, 2007. doi: 10.1152/jn.01357.2006.

M. Tsodyks and H. Markram. The neural code between neocortical pyramidal neurons depends on neurotransmitter release probability. *Proceedings of the national academy of science USA*, 94:719–723, Jan. 1997.

R. J. Vogelstein, U. Mallik, J. T. Vogelstein, and G. Cauwenberghs. Dynamically reconfigurable silicon array of spiking neuron with conductance-based synapses. *IEEE Transactions on Neural Networks*, 18:253–265, 2007.

R. W. Williams and K. Herrup. The Control of Neuron Number. *Annual Review of Neuroscience*, 11:423–53, 1988.

Acknowledgements

Daniel Brüderle for the friendly introduction to the group and to “Spikey”.

Sebastian Jeltsch for the guidance and support when needed most.

Thomas Pfeil for his keen eye and helpful words.

Erik Müller for the Linux wizardry.

Tobias and Janni, for the great chats in the Bachelorbüro.