Temporal stimulus segmentation and decision making via reinforcement learning in populations of spiking neurons

A Dissertation presented

by

Luisa Le Donne

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Doctor of Philosophy

in

Neuroscience

Stony Brook University

May 2017

Stony Brook University

The Graduate School

Luisa Le Donne

We, the dissertation committe for the above candidate for the

Doctor of Philosophy degree, hereby recommend

acceptance of this dissertation

This dissertation is accepted by the Graduate School

Charles Taber
Dean of the Graduate School

Abstract of the Dissertation

Temporal stimulus segmentation and decision making via reinforcement learning in populations of spiking neurons

by

Luisa Le Donne

Doctor of Philosophy

in

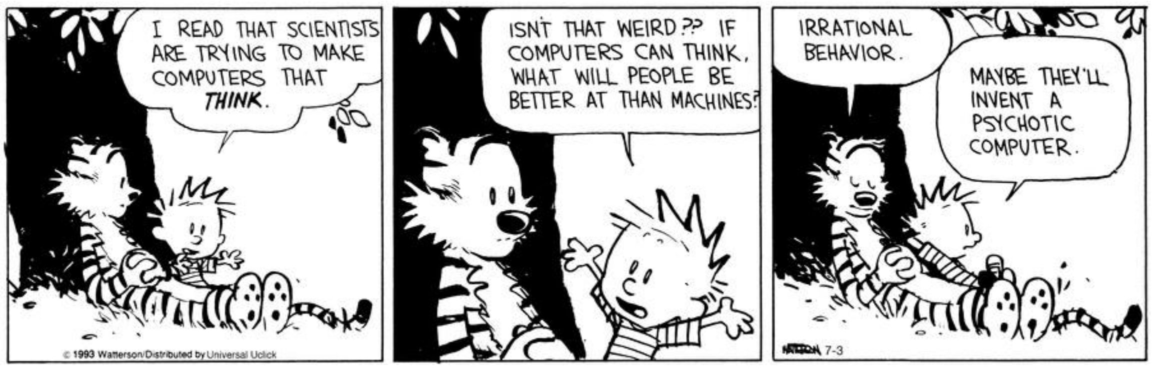
Neuroscience

Stony Brook University

2017

Stimulus identification is the process of picking out a particular stimulus among many other stimuli which may be present in the environment, for the purpose of performing a task. In the most interesting but also demanding scenario, the problem amounts to extracting action-relevant segments out of a noisy input stream, thus also involving the extrapolation of the onset and the end of the stimulus, not known a priori. Existing models in Computational Neuroscience and Artificial Intelligence have focused on the problem of discovering the correct decision in response to given stimuli – typically, for the purpose of getting reward. However, in these models the relevant stimuli (i.e., the stimuli that can trigger rewarded decisions) are known to the agent. For example, in many neural circuit models of decision-making, each stimulus is encoded by the activation of a predefined population of neurons representing that stimulus. Instead, an autonomous learning system should be able to identify any stimulus that is action-relevant without prior knowledge of it. Recently, a theory is emerging on how to address this problem using populations of spiking neurons. In this thesis, I study and extend a prototypical model based on populations of spiking neurons (the agent) able to identify the relevant stimuli and make the correct decisions. The agent is rewarded for making correct decisions at the right time; since the agent does not know, a priori, which stimuli are relevant or when they start or end, a decision is never enforced - contrary to most existing models. Instead, a decision is taken only when a readout of the decision neurons (a neural correlate of decision confidence) crosses a threshold. The learning rule implements a form of synaptic plasticity that tries to maximize the average reward obtained for correct decisions by following the gradient of the average reward. After showing the main features of the model, I characterize the dependence of its performance on crucial parameters including the number of stimuli, the type of stimuli (i.e., the way they are encoded), the number of decisions and the number of decision neurons. I then show that the model can handle natural stimuli recorded from the cortex of behaving rats. This model represents the first biologically plausible solution to the problem of stimulus segmentation and decision-making including multiple-choice decision-making.

Dedicated to Daniel Amit


I READ THAT SCIENTISTS ARE TRYING TO MAKE COMPUTERS THAT *THINK*.
ISN'T THAT WEIRD?? IF COMPUTERS CAN THINK, WHAT WILL PEOPLE BE BETTER AT THAN MACHINES?
IRRATIONAL BEHAVIOR.
MAYBE THEY'LL INVENT A PSYCHOTIC COMPUTER.
© 1993 Watterson/Distributed by Universal Uclick
WATTERSON 7-3

From Calvin and Hobbes by Bill Watterson, with permission.

Acknowledgements

Firstly, I would like to express my sincere gratitude to my advisor Prof. Giancarlo La Camera for his continuous support, for his motivation and ability to face challenges. His guidance helped me in developing critical thinking, motivation and love for this field. In addition, he helped me in growing not only in scientific aspects, but in life in general by teaching me how to be much more confident in myself, to work hard, to face stress, to grow as a person and to reach high maturation in facing hard moments: he helped me become a new person.

I would like to thank my thesis committee: Prof. Alfredo Fontanini, Prof. Mary Kritzer, and Prof. Christian Luhmann, for their insightful comments and encouragement, but also for the hard questions which motivated me to widen my research from various perspectives.

I thank my fellow classmates for the stimulating discussions, for the evenings we were working together before deadlines. A special thanks to Naz and Sneha. Also I thank my friends Tiffany, Upasana, Koushik, Pratik and Anusha for all the fun we have had.

I would like to thank my family: my parents, my brother and Sal for supporting me spiritually throughout writing this thesis and my life in general.

I am grateful to Prof. Daniel Amit for introducing me to scientific research. His enthusiasm for the field made a strong impression on me and I always carry positive memories of his teachings. This thesis is dedicated to his memory.

Contents

Chapter 1

Introduction

1.1 Modeling brain function

Exploring one of the most exciting and potentially rewarding areas of scientific research, neuroscientists focus on the study of the principles and mechanisms underlying brain function (Amit, 1992). Neuroscience has made enormous scientific contributions in the last 50 years, with many breakthrough findings about memory acquisition, learning, information processing, decision making, perception and integration of sensory information, object recognition, understanding language, emotions, cognitive disorders, and so on.

1.1.1 A brief history of neuroscience and the need for theoretical approaches

There has been an explosion of discoveries over the last several decades concerning the structure of the brain at the cellular and molecular levels. The discovery of voltage clamp by K. Cole and G. Marmont was the first technique allowing experimenters to measure current flowing through membrane channels (Cole and Marmont, 1942). They inserted an internal electrode into the giant axon of a squid and applied a current under conditions of controlled voltage. Using this tecnique, Hodgkin and Huxley conducted experiments aimed at determining the laws that govern the movement of ions in a nerve cell during an action potential. They developed their landmark model to explain the action potential generation in a squid giant axon in 1952. Their model, which was developed well before the advent of electron microscopes or large-scale computer simulations, was able to give scientists a basic understanding of how nerve cells work without having a detailed understanding of the cellular and molecular mechanisms at the ion channel level (Hodgkin and Huxley, 1952).

Hodgkin and Huxley were the first scientists to record an action potential (Fig. 1.1). This led to many studies on the neural activity of single cells, i.e. single unit recordings, that has become an important method for understanding mechanisms and functions of the nervous system. Hubel and Wiesel were

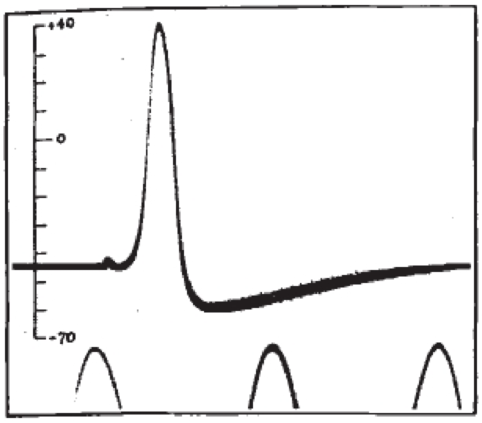


FIGURE 1.1: The first recording of an action potential using a microelectrode inserted into an axon was done by Alan Lloyd Hodgkin and Andrew Fielding Huxley in Plymouth in 1939 (Huxley, 2002; Hodgkin, 1976). The action potential was recorded from a squid giant axon. The vertical scale indicates the potential of the internal electrode (mV), the sea-water outside being taken to be at ground potential. Time marker at 500 Hz at bottom.

pioneers in recording neuronal activity. They inserted a microelectrode into the primary visual cortex of an anesthetized cat (Hubel and Wiesel, 1962) and projected patterns of light and dark on a screen in front of the cat. They found that some neurons respond primarily to bars of particular orientations ('simple cells'), while other cells, which they termed 'complex cells', detected edges regardless of where they were placed in the receptive field of the neuron and could detect motion in certain directions.

From the late 1970s, neuroscientists started to record neural activity during animal behavior. Goldberg and Wurtz performed single-neuron recordings, by inserting an electrode in the monkey visual cortex, while the animal was performing a saccade talk (Goldberg and Wurtz, 1972). Brinkman and Porter recorded neuronal activity in monkeys during a learned motor task (Brinkman and Porter, 1979). In 1978, Schmidt et al. implanted chronic recording micro-cortical electrodes into the cortex of monkeys (Schmidt et al., 1978) and showed that they could teach them to control neuronal firing rates, a key step to the possibility of recording neuronal signals and using them for brain-machine interfaces, or BMIs. The discovery and development, by Sakmann and Neher (Sakmann and Neher, 1984), of the "patch-clamp" method for recording from single-ion channels (the existence of which was hypothesized by Hodgkin and Huxley) provided decisive proof of the excitability of the neuronal membrane via voltage-sensitive ion-channels and allowed recording the neural activity of single cells and led to a long string of seminal studies on properties of excitable membranes, such as their resting potential and their ability to generate action potentials. In the 90s, the advent of functional MRI has allowed us to open a window into the activity of the whole human brain, a feat that continues today with calcium imaging techniques that allow visualizing an entire nervous system in simple organisms such as the zebrafish (e.g., (Ahrens et al., 2012)). In the mid 2000s, the new technique of optogenetics has combined genetic tools with electrophysiology to allow stimulating and tracking specific populations of neurons (Boyden et al., 2005; Boyle et al., 2013).

These technical innovations have provided a huge amount of data at all levels of description of neural activity, from molecular and cellular signaling inside neurons and synapses to the collective activity of populations of neurons. However, despite all this progress, we still have only a patchy knowledge of how the brain works. We do not yet understand how the brain enables us to see and hear, to learn skills and remember events, to plan actions and make choices (Sompolinsky, 2014; Churchland and Sejnowski, 2016).

In recent years, a number of researchers have emphasized the need for theoretical approaches in order to understand brain function ((Dayan and Abbott, 2001; William Bialek and de Ruyter van Steveninck, 1999). Theoretical analysis and computational modeling encourage understanding brain mechanisms by constructing compact representations of what has been learned, building bridges between different levels of description, and identifying unifying concepts and principles. Theoretical approaches come in different flavors, such as descriptive, explanatory, mechanistic and computational. Descriptive models summarize large amounts of experimental data compactly yet accurately, thereby characterizing what neurons and neural circuits do. For example, one could build a descriptive model of what the output of a neuron represents by presenting all possible classes of input to the animal and recording what sequences of spikes it produces. However, one can also build an explanatory model by working out what inputs the neuron has, what these inputs represent, and what transformations the neuron performs on these inputs. Mechanistic models address the question of how nervous systems operate on the basis of known anatomy, physiology, and circuitry. Such models often form a bridge between descriptive models couched at different levels (Dayan and Abbott, 2001). Experimental efforts are being more and more frequently complemented by modeling studies these days. Computational models, such as the one developed in this thesis usually combine the characteristic of several styles of modeling.

Computational models that describe the biophysical properties of the biological membrane channels were being developed since the 1950s. Different models were constructed at different levels of detail. One of the most famous of these is the model of the propagation of the nerve impulse (Hodgkin and Huxley, 1952). The advent of multi-electrode recordings and imaging techniques in the 1990s has enabled neuroscientists to obtain high spatial and temporal resolution of the electrical activity for populations of neurons in the brain. This has led to a resurgence of interest in large-scale models of neural activity (Bressler and Menon, 2010; Marrelec et al., 2006). This in turn has engaged the interest of physicists and mathematicians in the behavior of these systems. There are many levels in which one can model neural activity, from detailed models of single-ion channels and synapses (Destexhe et al., 1994) up to 'black box' models used to understand psychological phenomena (Rumelhart et al., 1988). An intermediate level of description makes use of 'neural networks', where each neuron is represented by a simple device with a restricted range of activations (e.g., binary values) or, more generally, by a low-dimensional dynamical system. In neural networks, neurons are connected by plastic synaptic weights which can be trained to solve a broad range of tasks. The setup of these models allows to unleash the power of the methods from statistical mechanics (Ermentrout, 1998; Amit, 1992), an approach that has led to landmark results including the famous Hopfield network (Hopfield, 1982), to this day a beautiful metaphor of how associative memory may be mediated by neural circuits (reviewed in Sec. 1.4.2). Neural networks have been a powerful metaphor of brain function and can be very useful devices for commercial applications ranging from speech and image recognition and robot behavior control to medical diagnosis and market predictions. The recent resurgence of methods relying on 'deep learning architectures' (networks with many intermediate layers of neurons) attests to the flexibility and usefulness of these methods. However, although inspired by the parallel organization of neural circuits, these methods are not representative of biological networks. For example, real synapses have only access to local activity from the pre- and post-synaptic neuron they connect, whereas in most neural networks synapses are changed based on the activity of other neurons, as e.g. in the famous back-propagation algorithm (Rumelhart et al., 1988a) (more will be said in Sec. 1.4.1). The 'neurons' themselves are simple devices that have nothing or little to do with real neurons – in particular, they do not generate spikes, but they communicate through analog or binary signals. The field of learning with populations of spiking neurons and local synapses is one of the least developed in theoretical neuroscience and, arguably, the closest to biological cortical circuits.

In this work, I present a biologically plausible model for learning a class of decision tasks that are important in everyday behavior. The model is based on populations of spiking neurons connected by plastic synapses that change their values during training. The plasticity rule on which learning is based is reminiscent of the empirical plasticity rules found in cortical-striatal synapses (Reynolds and Wickens, 2002). Neural decision-making has been studied extensively (Fellows, 2004; Shadlen and Kiani, 2013; Hilbert, 2012); for a number of reasons, ranging from experimental need to theoretical convenience, the overwhelming majority of these studies assume that the relevance and timing of the stimuli are known to the learning agent. Instead, in this work I focus on the problem of extracting action-relevant segments out of a noisy input stream, thus also requiring the extrapolation of the onset and the end of relevant stimulus features, not known a priori. I will introduce a biologically plausible autonomous learning system able to identify action-relevant stimuli without prior knowledge of them, while learning to ignore stimuli that are not behaviorally relevant.

1.2 Learning strategies

It is widely believed that learning is achieved by modifying connections between neuronal cells based on experience (known as 'synaptic plasticity'). Computational models and neural networks are outstanding tools to investigate how learning is achieved. The effect of different plasticity rules (or 'learning rules', i.e., the algorithms for changing the synaptic weights so as to accomplish the desired learning process) can be explored by using neural networks. Broadly speaking, learning strategies (and their corresponding 'learning rules') can be divided into 3 classes: *supervised learning*, *unsupervised learning*, and *reinforcement learning*.

In *supervised learning* (i.e. 'learning with a teacher') each stimulus (or 'pattern') to be learned comes labelled with the desired output (Fig. 1.2 left). In the supervised setting, the designer has to provide the correct label on a subset of situations. In an associative task, for example, we are requested to report the correct answer to each learned pattern. A supervised learning rule will compare the output to the presented input with the correct output (provided by the teacher). An example is the perceptron learning rule for binary classification (Rosenblatt, 1961).

In *reinforcement learning* (or reward-based learning) the teacher is only expected to provide a reward signal. For example in the case of chess, the reward really is trivial: +1 for winning the game, -1 for loosing the game. In the reinforcement learning setting, the teacher (modeled as the environment) tells us only whether the answer is right or wrong, by rewarding (or punishing) our decisions, but it doesn't tell us what the correct answer is. In other words, if the output is wrong, no additional information is given on what the correct answer is. Reinforcement learning proceeds by trial and error: every past action that led to a reward tends to be reinforced so as to increase the chance of obtaining reward in the future (Fig. 1.3), and similarly any past action that led to punishment tends to be discouraged, so as to be avoided in the future. In the animal kingdom, reinforcement learning underlies the learning of many important tasks that are crucial for survival, such as learning to orient to a relevant stimulus, forage for food, learn the meaning of cues that can predict the availability of rewards such as food or sex, and many other forms of everyday decisions. Despite some limitations of its applicability to real world problems (some reviewed in Sec. 1.6.1), reinforcement learning is a powerful framework and it will be the framework used in this work to attack the temporal stimulus segmentation problem.

Finally, in *unsupervised learning* the examples given to the learner are *unlabeled*, i.e., there is no error or reward signal to evaluate a potential solution (Fig. 1.2 right). Unsupervised learning algorithms tend to extract structure and regularities from the input patterns based on the statistics of the data. The learning system tends to self-organize and adapt its response to the stream of inputs to which it is exposed. An example of unsupervised learning rule is given by the 'Hebbian learning rule', in which a change in the strength w_{ij} of a connection from neuron j to a neuron i is a function of the pre- and post-synaptic neural activities:

$$\frac{dw_{ij}}{dt} = \eta A_{pre}(A_{post} - \theta), \tag{1.1}$$

where A_{pre} is a measure of the activity of the presynaptic neuron, A_{post} is a measure of the activity of the postsynaptic neuron, θ is an activity threshold for the postsynaptic neuron, and η is a parameter known as the *learning rate* because its value is related to the speed of learning. In this rule, the connections between neurons that are co-active ('fire together') are increased, whereas connections between uncorrelated neurons are weakened, obeying the original Hebb's proposal.[1]

[1]In this rule, the two neurons are co-active if the pre-synaptic neuron is active and the postsynaptic neuron's activity is above its threshold θ. Many different variations of this rule exist and some have some credible experimental support.

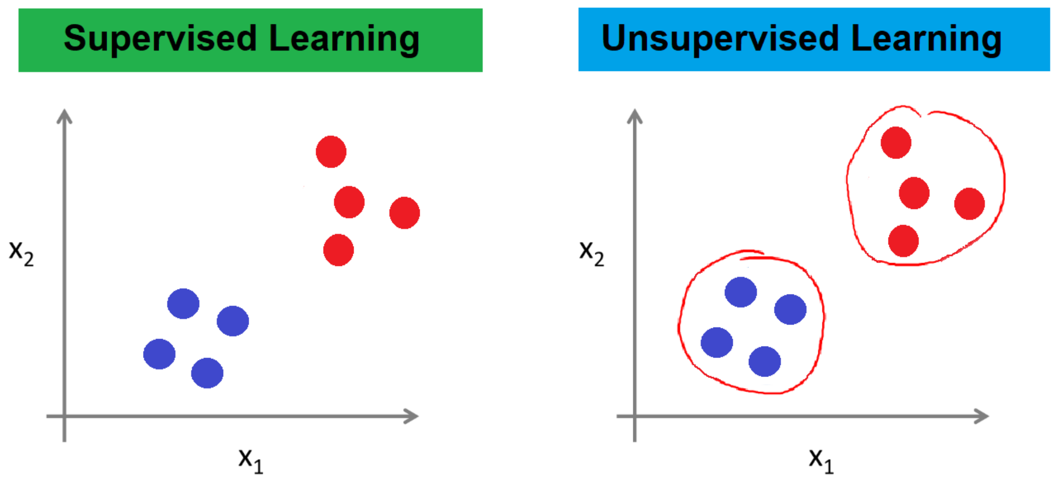


FIGURE 1.2: Symbols in the x_1-x_2-space represent training data. **Left:** In *supervised learning*, each training instance has a label: positive-labeled data (blue points) or negative-labeled data (red points). All data are labeled, i.e., the desired answer is provided, e.g. it may be $+1$ for blue points and -1 for red points, and the algorithm learns to predict the desired (or correct) output associated to each input datum. **Right:** In *unsupervised learning*, all data are unlabeled and the algorithm learns the inherent structure of the dataset (the regularities and correlations of the data) from the input data. (Adapted from Lecture 1 of Andrew Ng's Machine Learning Course on Coursera).

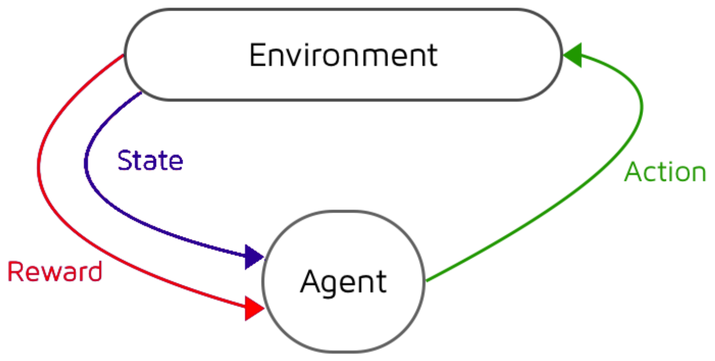


FIGURE 1.3: The reinforcement learning framework, in which an agent takes a series of actions, each of which takes the agent to a new state associated with a reward provided by the environment (Sutton and Barto, 1998).

Hebbian learning can for example learn to project a multidimensional input (such as a vector) onto a lower-dimensional subspace where most of the information about the input is retained. Hebbian learning can also provide a mechanism for a network of spiking neurons to learn to produce persistent activity in response to a stimulus after its removal – and thus provides a working memory for that stimulus (see e.g. Amit and Brunel (1997)). In these examples, the system extracts structure from the training data and there is no notion of right or wrong response to the inputs.

1.2.1 Reinforcement learning as reward maximization

Many models of reinforcement learning are built from the requirement of *maximizing the average reward collected from the environment*. A rigorous derivation of this procedure and its implementation in spiking neurons is presented in Appendix A and will be discussed more properly in Sec. 2.3. Here I show a heuristic way to try to accomplish this via the modification of 'pre-post' or 'covariance-based' rules such as the Hebbian rule Eq. 1.1 that include a modulatory term, the reward R. Most commonly, the reward is $+1$ if the output was correct, and -1 otherwise. For example, the generalized Hebb rule Eq. 1.1 would be modified as

$$\frac{dw}{dt} = \eta R A_{pre}(A_{post} - \theta), \tag{1.2}$$

where R is a *global* signal provided by the environment after an action or decision. Note that now the synaptic weights change only in the presence of a reward signal (either reward, $R = +1$, or punishment, $R = -1$). In the presence of reward (given when the output is correct), the connections between neurons that 'fire together' are increased, whereas connections between uncorrelated neurons are weakened, just as in the ordinary Hebbian rule. In the presence of a punishment ($R = -1$, given when the output is incorrect), the synaptic connections between neurons that are simultaneously active are now decreased rather than increased, contrary to the ordinary Hebbian prescription. In all cases, this learning rule will reinforce a post-synaptic activity that tends to produce a rewarding outcome (given some presynaptic activity $A^{pre} > 0$), and will counter post-synaptic activity that tends to produce punishment. This will occur after each stimulus presentation, until the synaptic weights converge to values that, in the best case scenario, will lead to correct decisions for all input stimuli. When this occur, all the reward obtainable from making correct decisions will be obtained, i.e., *the learning agent will maximize reward.*

Although Eq. 1.2 was derived heuristically and there is no guarantee that it will work as predicted, in a later section (Sec. 1.6) I will describe a similar learning rule that performs a form of reward optimization called *gradient ascent in the average reward.* With minor modifications, this is in fact the learning rule used in this work, and the comparison with Eq. 1.2 should serve to motivate it and provide some intuition as to why it works. Importantly, the same learning rule will be able to deal with a streaming input of spike trains, which is crucial for the problem of temporal stimulus segmentation which is the main object of this thesis.

1.3 Spiking neuron models

The aim of this thesis is to come up with a biologically plausible model of learning to solve the temporal stimulus segmentation problem. Biological plausibility demands that we deal with spiking neurons. The most convenient such model for the theory here developed is the so-called ‘spike response model’ (Gerstner and Kistler, 2002), which will be introduced shortly. This is a simplified neuron model that is very convenient when working with large networks of spiking neurons. Although it has some biophysical plausibility, it is not based on voltage-sensitive ion-channels and thus cannot, on its own, generate an action potential. To understand why this model is different from biophysically grounded model neurons, I will start from a brief review of the celebrated Hodgkin-Huxley model (Hodgkin and Huxley, 1952).

1.3.1 The Hodgkin and Huxley model

The first accurate model of membrane excitability was introduced by A.L. Hodgkin and A.F. Huxley in 1952. This model was based on biophysical mechanisms underlying Na^+ and K^+ conductances that generate action potentials in the giant axon of the squid. The postulate of the Hodgkin and Huxley (HH) model was that the membrane currents result from the assembly of gating particles freely moving in the membrane. The molecular components responsible for ionic permeabilities were later identified as being ion channels. The sensitivity of ion channels to voltage is a fundamental property that constitutes the core mechanism underlying the electrical excitability of biological membranes. The Hodgkin and Huxley model is based on a membrane equation describing three ionic currents in an isopotential compartment:

$$I = C_m \frac{\mathrm{d}V_m}{\mathrm{d}t} + g_K n^4 (V_m - V_K) + g_{Na} m^3 h (V_m - V_{Na}) + g_l (V_m - V_l), \tag{1.3}$$

where C_m is the membrane capacitance, V_m is the membrane potential, g_K, g_{Na} are the membrane conductances for potassium and sodium. V_K and V_{Na} are the potassium and sodium reversal potentials, respectively, and g_l and V_l are the leak conductance and leak reversal potential, respectively. I is an external current (if present) and the gating variables m, n and h obey the first-order ordinary differential equations

$$\begin{aligned}\frac{dm}{dt} &= \alpha_m(V)(1-m)-\beta_m(V)m, &(1.4)\\ \frac{dh}{dt} &= \alpha_h(V)(1-h)-\beta_h(V)h, &(1.5)\\ \frac{dn}{dt} &= \alpha_n(V)(1-n)-\beta_n(V)n, &(1.6)\end{aligned}$$

with transition rates $\alpha(V)$ and $\beta(V)$ given by

$$\begin{aligned}\alpha_m(V) &= \frac{0.1(V+40)}{1-e^{-0.1(V+40)}}\,; \quad \beta_m(V)=4e^{-0.0556(V+65)}, &(1.7)\\ \alpha_h(V) &= 0.07e^{-0.05(V+65)}\,; \quad \beta_h(V)=\frac{1}{1+e^{-0.1(V+35)}}, &(1.8)\\ \alpha_n(V) &= \frac{0.01(V+55)}{1-e^{-0.1(V+55)}}\,; \quad \beta_n(V)=0.125e^{-0.0125(V+65)}. &(1.9)\end{aligned}$$

The HH model was a breakthrough, a Nobel winning discovery, and one of the first triumphs of computational neuroscience. The action potential generation can be understood by using the HH model. This was a huge success and probably the main contribution of the HH model: to provide a clear understanding of how an excitable cell membrane can initiate and propagate (together with cable theory, Cole and Hodgkin (1939); Offner et al. (1940); Rushton (1951)) an action potential by means of a detailed mathematical model. Since then, the HH model has been extensively studied, modified, and simulated on a computer to explain a wide range of phenomena and predict new ones not yet observed empirically. To this day, the HH is still one of the best exemplifications of the contributions that computational neuroscience can make to the study of the brain. Hodgkin and Huxley predicted the existence of channels before they could have been observed with the patch clamp technique (E. Neher and B. Sakmann, developed it in the late 1970s and early 1980s and received the Nobel prize in 1991; when Hodgkin and Huxley proposed their model, channels were presumed to exist but the detailed description of single-channels kinetics wasn't known).

1.3.2 Simplified spiking neuron models

Despite its great success and its fundamental contribution to our understanding of excitable membranes, the HH model is a complex, multidimensional, non-linear model. It is a set of 4 nonlinear differential equations complemented by 3 constitutive equations for the transition rates. This makes this model neuron complex to analyze and computationally expensive to simulate on a computer. For this reason, several simplified neuronal models have been developed since the advent of HH, such as the FitzHugh-Nagumo model (FitzHugh, 1961; Nagumo et al., 1962) – for a review see Izhikevich and FitzHugh (2006) – or the Morris-Lecar model (Morris and Lecar, 1981), which produce action potentials by using only 2 differential equations. For the purpose of studying large networks of neurons, however, even those simplified models are too inconvenient. In this work, I will use an even simpler model that requires only one differential equation for the membrane potential, complemented by appropriate boundary conditions that formalize the process of action potential emission.

Such model is called the *spike response model* (SRM) (Gerstner and Kistler, 2002) and it is very convenient for the development of the learning rule used here to solve the temporal segmentation task (described in Ch. 2). The model can be understood as a generalization of the so-called *leaky integrate-and-fire* neuron (LIF) (Tuckwell, 2005), a model that dates back to Lapicque (1907) and that has enjoyed great popularity among theorists (examples from a virtually infinite literature are Usher et al. (1993); Abbott and van Vreeswijk (1993); Tsodyks et al. (1993); Hopfield and Herz (1995); Amit and Brunel (1997); for a comprehensive review see Burkitt (2006a,b)). Since the LIF neuron is more biologically grounded than the SRM, I will briefly first introduce the former.

1.3.3 The leaky integrate-and-fire model (LIF)

The idea behind the LIF model is to couple a description of the passive behavior of the neuronal membrane with some threshold device for the emission of the spike (through boundary conditions). The membrane potential evolves according to the original Lapique model

$$C\,\frac{dV}{dt} = -\,\frac{V - V_L}{R_m} + I, \tag{1.10}$$

where V is the membrane potential, C is the membrane capacitance, R_m is the membrane resistance, V_L is the resting potential, and I is the input current. Because of the linear term $(V - V_L)/R_m$ on the right hand side, this model can never produce an action potential (it is essentially the HH model without the voltage-sensitive sodium and potassium currents responsible for action potential generation). To endow the LIF model with the ability to produce action potentials (often called *spikes* in this context), one imposes the following *boundary conditions*:

$$if \quad V(t^*) = \theta \rightarrow \text{spike at time } t^* \quad ; \qquad V(t) = V_r \quad \text{for} \quad t^* < t < t^* + \tau_{arp}. \tag{1.11}$$

The first of these conditions expresses the fact that as V reaches a threshold value θ, usually situated around 20 mV above rest, *a spike is said to be emitted*, even though no action potential is actually being generated by Eq. 1.10. The second of these boundary conditions states that the membrane potential, after the emission of a spike at time t^*, is *clamped* to a *reset* value V_r for the duration of few milliseconds (τ_{arp}) to mimic the existence of an absolute refractory period.

The response of this model neuron to a regular train of action potentials is depicted in Fig. 1.4. This figure should also illustrate the origin of the term *integrate-and-fire* given to this model: presynaptic inputs are integrated (=accumulated) with a characteristic time of $\tau = R_m C$, the membrane time constant (see left panel); if enough inputs are received, they accumulate until the membrane voltage hits the threshold θ and a spike is emitted.

Despite its simplicity, the LIF model neuron can over-perform the HH model in some applications, for example in matching the distribution of interspike-intervals of real data (Ostojic, 2011).

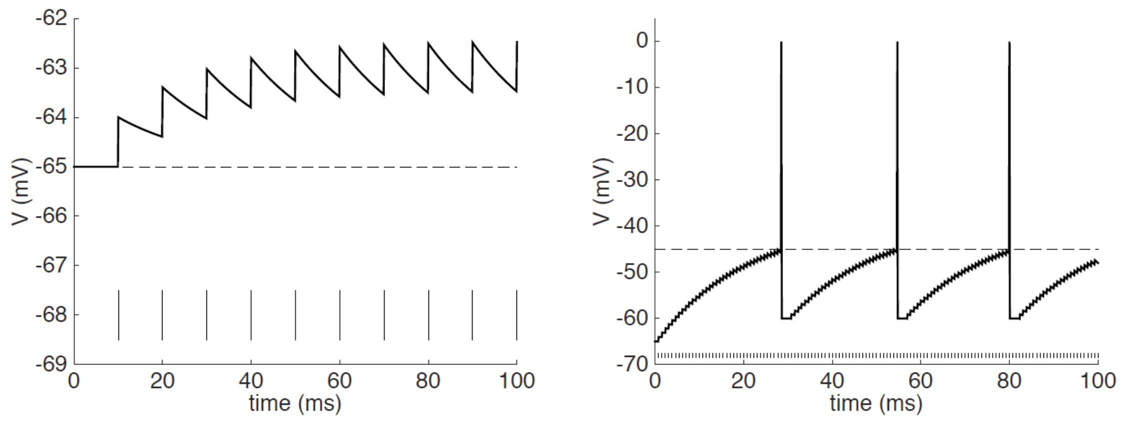


FIGURE 1.4: **Evolution of the LIF neuron Eq. 1.10 driven by a regular spike train.** The input spike trains are shown as a series of vertical 'ticks' below the membrane potential. They represent impulsive post-synaptic currents due to the arrival of presynaptic spikes. **Left:** Response to a spike train with firing rate $100Hz$ insufficient to drive the neuron above threshold for spike emission (here set at -45 mV). **Right:** Response to a spike train with firing rate $1300Hz$ leading to the appearance of spikes. After each spike the membrane potential is reset to $V_r = -60$ mV and 'clamped' there for a refractory period of $\tau_{arp} = 2$ ms. Adapted from La Camera (2017).

1.3.4 The Spike Response Model

The Spike Response Model (SRM) is a generalization of the leaky integrate-and-fire model. As in the LIF model, action potentials are generated when the voltage passes a threshold from below. Differently from the LIF model, the equation describing the sub-threshold behavior of the SRM is written directly for the membrane potential u, and not as a differential equation in u:

$$u(t) = u_{rest} + \sum_{i=1}^{M} w_i \sum_{s \in X_i} \frac{1}{\tau_m} e^{-\frac{t-s}{\tau_m}} \Theta(t-s) - \sum_{t^\nu \in Y} e^{-\frac{t-t^\nu}{\tau_m}} \Theta(t-t^\nu). \tag{1.12}$$

In this equation, u_{rest} is the resting potential, M is the number of presynaptic neurons, w_i is the synapse from the presynaptic neuron i, s is the spike time of a presynaptic neuron, X_i is the set of presynaptic spike times, τ_m the time constant of the membrane, the t^ν are the spike times of the neuron itself, Y is the set of all the neuron's spike times, $\Theta()$ is the Heaviside function ($\Theta(t) = 1$ for $t > 0$, and $\Theta(t) = 0$ otherwise). According to this equation, the neuron responds to an incoming spike with an instantaneous rise followed by an exponential increase (just like the LIF neuron, see Fig. 1.4), and is instantaneously reduced by 1 followed by an exponential relaxation to reset after each of its own spikes, akin to the post-spike reset of the LIF neuron. See Fig. 1.5 for an illustration.

A spike is emitted by this neuron when its membrane potential reaches a threshold, just as in the LIF neuron. However, in the variation used by many authors and in this thesis, a spike is emitted via an *escape noise* mechanism, rather than when a threshold is passed. According to the escape noise mechanism, a spike is emitted with a given probability $\phi(u)$ that depends on the membrane potential u. More precisely, the probability of emitting a spike in each tiny interval of time dt is given by $\phi(u)dt$. More details of the model will be given later.

The Spike Response Model allows very reliable prediction of many aspects of neuronal activity, such as timing of the spikes, membrane voltage, mean rate and CV of the inter-spike interval distribution

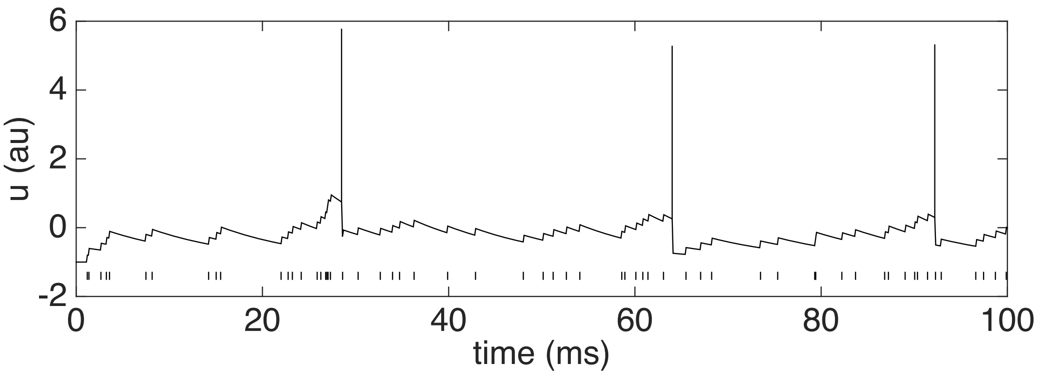


FIGURE 1.5: **Simulation of one SRM neuron driven by a spike train input.** The membrane potential of one spike response model neuron with exponential escape noise function obtained in response to the Poisson input spike train depicted at the bottom. Adapted from La Camera (2017).

(Jolivet et al., 2003). The SRM fits experimental data to a high degree of accuracy and predicts a large fraction of spikes with a precision of ±2 ms (Jolivet et al., 2006). The same procedure has also been used to approximate detailed neuron models of the Hodgkin-Huxley type by the Spike Response Model (Kistler et al., 1997; Jolivet et al., 2004), see e.g. Fig. 1.6. In the capability of predicting spike times under random conductance injection, the Spike Response Model has been a very successful model (Jolivet et al., 2008) whereas a standard leaky integrate-and-fire model is less adequate.

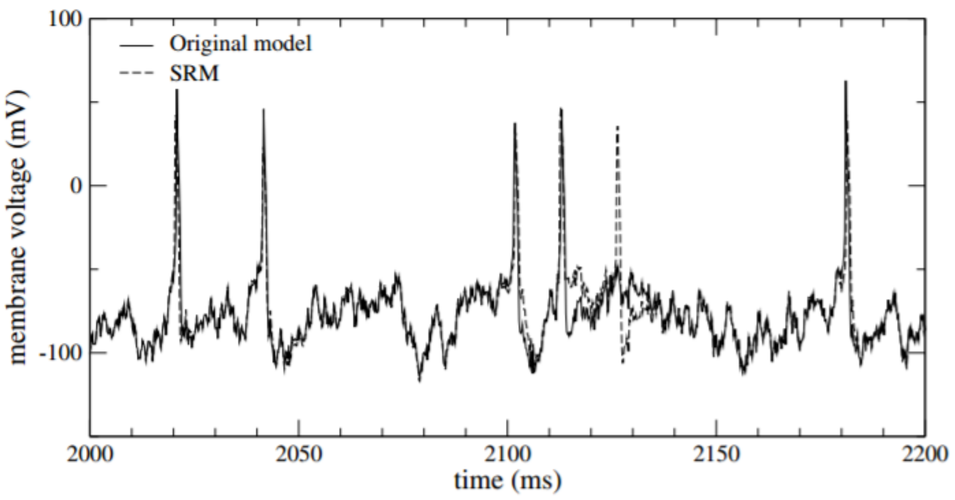


FIGURE 1.6: **Prediction of neuronal spike train data by the Spike Response Model is reliable.** The prediction of the SRM (dotted line) is compared to the target data (solid line). The model achieves very good prediction of the subthreshold behavior of the membrane voltage. The timing of all the spikes is predicted correctly, except that one extra spike is added at around 2125ms. Figure adapted from Jolivet et al. (2003).

1.4 Neural networks

For many applications, and in particular to solve the temporal stimulus segmentation problem of this thesis, what matters is the concerted activity of populations of spiking neurons, not just single neurons. Computational neuroscience has traditionally studied populations of neurons in terms of artificial neural networks. I give in the following sections a minimal background on some influential network models, from the *perceptron* of the 50s (Rosenblatt, 1961) to the more recent *tempotron* (Gütig and Sompolinsky, 2006), that for its biological plausibility is more closely related to the model developed here.

1.4.1 The perceptron: a model for classification of data

The perceptron was one of the first artificial neural networks to be produced. It is an information processing paradigm that can perform a classification task. The perceptron algorithm allows for online learning, in that it processes elements of a 'training set' of stimuli one at a time. The perceptron algorithm dates back to the late 1950s. It consists of N input neurons (x_i, $i = 1,..N$) connected to a 'McCulloch-Pitts neuron' y_j (the binary neuron of Eq. 1.13 that uses a step transfer function, (McCulloch and Pitts, 1943)) through synaptic weights w_{ij} (see Fig.1.7).

The perceptron is an algorithm for learning a binary classifier: a function that maps its input x (a real-valued vector) to an output value $y = f(x)$ (a single binary value):

$$f(x) = \begin{cases} 1 & \text{if } \sum_j w_j x_j + b > 0 \\ 0 & \text{otherwise} \end{cases} \tag{1.13}$$

where $\mathbf{w}$ is the vector of synaptic weights and b a parameter called 'bias'.

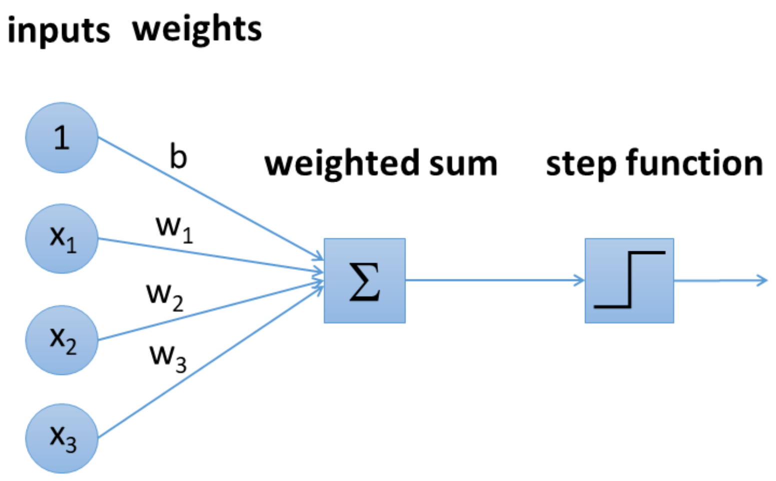


FIGURE 1.7: **Schematic of Rosenblatt's perceptron.** N neurons in the 'input layer' are connected to an output neuron where a weighted sum of the inputs is performed. The weights are called synaptic connections. Then a step function is performed, so the final value for the output neuron is a binary value.

The rule to update the synaptic connections $\{w_i\}$ is the following:

$$w_i(t+1) = w_i(t) + (d_j - y_j(t))x_i, \quad \textit{for all neuron inputs } 0 \le i \le n. \tag{1.14}$$

where d_j is the desired output and y_j is the actual output presented at time step t. The problem that the perceptron has to solve is illustrated in Fig. 1.8 (Wikipedia, 2017). This is the problem of separating a set of inputs in two classes. This is possible only if the stimuli are 'linearly separable', meaning (at least in 2 dimensions) that a straight line exists that separates them in the two desired classes, as shown in figure. When the stimuli are linearly separable, a beautiful theorem holds that says that the perceptron will find the solution in a finite number of steps (Amit, 1992)). During training, which involves repeated presentations on the input layer of all the stimuli to be learned, the weights and the 'bias' parameter b are modified according to the learning rule Eq. 1.14 (the learning rule for b differs only by the sign of the update). As the weights and b change, the position of the separating line changes until all stimuli are successfully separated (see Fig. 1.8). If they are not linearly separable, the algorithm will not converge to a complete solution, but it will still try to separate the stimuli as well as it can.

The problem of separating stimuli that are *not linearly* separable was solved in the 80s in multi-layer networks (where intermediate or 'hidden' layers of neurons are inserted between the input and output layer), thanks to a new learning rule called back-propagation (Rumelhart et al., 1988b). This method is still the basis of contemporary artificial neural networks known as *deep learning networks* due to presence of many intermediate layers (for a review on deep learning see (Schmidhuber, 2015)).

Note that a problem akin to the classification of stimuli in classes is considered in this thesis, but spiking neurons will be used instead of simple binary units, and streaming inputs of spike trains will be used instead of vectors of binary (or analog) values of artificial neural networks.

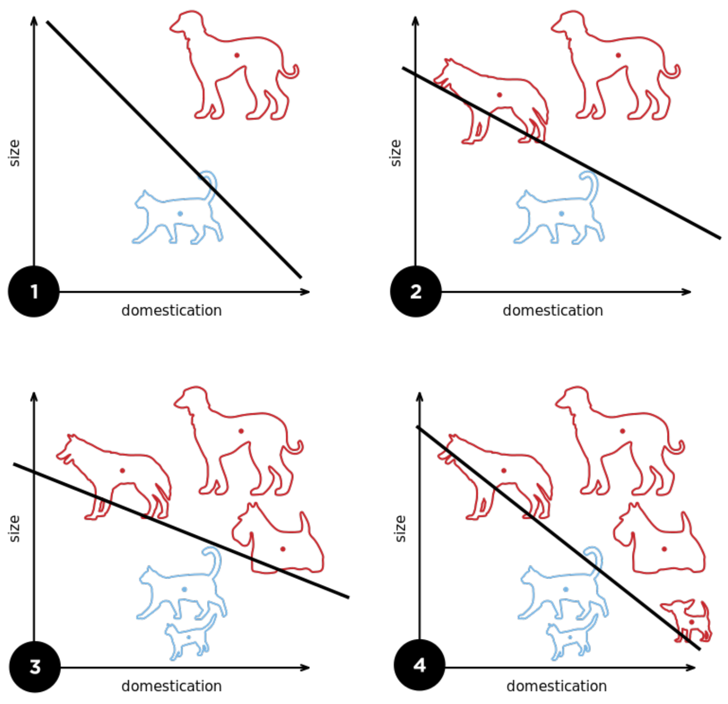


FIGURE 1.8: **Training a perceptron.** The classification problem is equivalent to the separation of two sets of data points in desired groups. A perceptron updates a linear separation boundary as more stimuli are presented during training. Adapted from Wikipedia (2017).

1.4.2 The Hopfield model: a model for associative memory

The Hopfield network is one of the most influential neural network models in the computational neuroscience community that has inspired many physicists to join and shape this field. Invented by John Hopfield in 1982, it is an recurrent neural network with symmetric connections that can serve as a model for associative memory.

Essentially, a Hopfield network contains N McCulloch-Pitts neurons all connected to one another with symmetric synaptic weights, i.e., $w_{ij} = w_{ji}$. At each time step t, the activity state of each neuron depends on the state of all other neurons at the previous time step according to:

$$s_i(t) \leftarrow \begin{cases} +1 & \text{if } \sum_j w_{ij} s_j(t-1) \geq \theta_i, \\ -1 & \text{otherwise.} \end{cases} , \tag{1.15}$$

where t is the current time step, w_{ij} is the strength of the connection from unit j to unit i, $s_j = \pm 1$ is the state of unit j, and θ_i is a 'threshold' for unit i analogous to the bias parameter b of the perceptron.

A 'memory' or 'pattern' of the network is a vector of neural activities $\xi_j = \pm 1$, $j = 1, \ldots, N$, and there can be up to 2^N such memories. The goal of the network is to be able to store as many memories as possible as attractors of the network's dynamics, Eq. 1.15. This means that, if starting from a network's activity close enough to one of the stored memories, the dynamics of the network Eq. 1.15 will converge to that memory. This is achieved by imposing synaptic values

$$w_{ij} = \frac{1}{M} \sum_{\mu=1}^{M} \xi_i^\mu \xi_j^\mu \tag{1.16}$$

where the sum over μ is over all patterns $\{\xi_i^\mu\}$ to be stored as attractors. If the number of memories, M, is smaller than $0.138N$ (Amit, 1992), starting from a pattern 'similar' to one of the patterns $\{\xi_i^\mu\}$, the network will converge to that memory pattern. This phenomenon can be interpreted as the 'recalling phase' of associative memory, wherein a memory is recalled after the network is presented with a stimulus that is similar to the memory itself.

The Hopfield network can function as an associative memory for two main reasons: 1) the symmetry of the weights, $w_{ij} = w_{ji}$, insures that the dynamics converge to a fixed point (Hopfield, 1982); 2) the Hebb-like prescription Eq. 1.16 guarantees, if M is below the critical capacity $\approx 0.138N$, that the desired memories $\{\xi_i^\mu\}$ are the attractors of the network.

The Hopfield model exploits a general feature of cortical circuits, i.e., recurrent connections among neurons in the same layer. The model developed in this thesis (and related models such as the tempotron introduced in the next section) does not use recurrent connections, in part because feedforward connections will prove sufficient to solve the problem, as shown in Chapter 3, in part because it is more difficult to develop the theory in the presence of recurrent connections (this is briefly discussed in Chapter 5). Moreover, note that *there is no learning* in the Hopfield network. The synaptic weights (see Eq. 1.16) are given once and for all and remain constant throughout, and therefore it is not a suitable model to solve the stimulus segmentation problem.

1.4.3 The tempotron: a model for classification of spatiotemporal spike patterns

The tempotron (Gütig and Sompolinsky, 2006) is a neural network that uses a supervised synaptic learning algorithm which is applied when the information is encoded in spatiotemporal spike patterns. It is one of the first neural networks able to deal with spike trains and where learning is performed by a spike timing-based learning rule. The neuron model consists of a model closely related to the SRM neuron (see Sec. 1.3.4) driven by N synaptic afferents. In the classification task considered, each input pattern belongs to one of two classes. A pattern consists of N input spike trains, one train for each afferent, arriving between 0 and $T = 500$ ms. The tempotron will integrate the input spikes and elicit a spike if its membrane potential hits a threshold. Contrary to the SRM and other standard models, however, after a spike emission the tempotron remains silent, i.e., it ceases to integrate its inputs and cannot emit another spike until the end of the stimulus.

This limitation was motivated by the specific learning rule put forward for the tempotron to learn the classification task: the learning rule changes the weights after each stimulus presentation according to

$$\Delta w_i = \eta^{\pm} \sum_{t_i < t_{max}} K(t_{max} - t_i), \tag{1.17}$$

where $K(t - t_i)$ is the normalized post-synaptic potential contributed by each incoming spike at time t_i, t_{max} denotes the time at which the postsynaptic potential reaches its maximal value, and $\eta^{\pm}$ is the learning rate. If a spike was required and was not emitted by the neuron, Eq. 1.17 is applied with $\eta = \eta^{+} > 0$, whereas if no spike was required and a spike was produced, Eq. 1.17 is applied with $\eta = \eta^{-} < 0$. Gütig and Sompolinsky (2006) have shown that this learning rule acts by minimizing the misclassification error. This was perhaps the first example of classification of spike patterns by a neuron-like device. Unlike the inputs to the perceptron, spike patterns involve temporally extended stimulus presentations, just like in the problem studied in this thesis. However, the tempotron learning rule is a supervised rule with offline updates (at the end of each stimulus presentation), which also requires ignoring all spikes but the first emitted by the neuron: this means that it is not suitable for solving the stimulus segmentation problem of this study.

Gütig and Sompolinsky (2006) also introduced a more biologically realistic implementation of their learning rule that does not require the knowledge of t_{max}. This new learning rule could be framed as a reinforcement learning rule and it could be shown that in that form it is more suitable to online learning and is more closely related to the learning rule used in this work. There are several differences, however, including the use of the learning rule in a population of neurons (rather than in a single neuron). This requires some crucial modifications to the learning rule as described in Ch. 2. Other differences with the tempotron will be further discussed in Ch. 5, after the model of this thesis has been thoroughly introduced and tested.

1.5 Homeostatic processes

Biological networks require homeostatic mechanisms to maintain a stable, relatively constant internal environment. Eve Marder, a pioneer of the study of homeostatic processes in neuroscience, defines the homeostatic regulation of neuronal excitability as “the collective phenomena by which neurons alter their intrinsic or synaptic properties to maintain a target level of electrical activity" (Williams et al., 2013). Many types of homeostatic processes have been observed in neurons in different experimental preparations and contexts. Synaptic and intrinsic homeostases cooperate to optimize single neuron response properties and tune integrator circuits, as shown e.g. by Cannon and Miller (2016). They also have shown that a set of homeostatic processes that appear to regulate mean firing rate may work together to control firing rate mean and variance.

Neurons are equipped with homeostatic mechanisms that counteract long-term perturbations of their average activity and thereby keep neurons in a healthy and information-rich operating regime (Harnack et al., 2015). The importance of such homeostatic mechanisms to the solution of the temporal segmentation problem will be clear later on (Sec. 3.4.2); here, I give a minimal account of some influential models of homeostatic plasticity.

In keeping with the experimental distinction, homeostatic models can be broadly separated into those that regulate synaptic properties and those that regulate intrinsic neuronal properties.

Homeostatic plasticity in neocortical circuits has been studied by several authors who observed compensatory changes in excitatory postsynaptic currents during protocols for synaptic potentiation (see e.g. Turrigiano and Nelson (2004)). This means that an increase or decrease in synaptic weight between two neurons could be compensated for by a regulatory mechanism on the postsynaptic side, acting to keep the post-synaptic neuron’s activity within a reasonable physiological range, see Fig. 1.9.

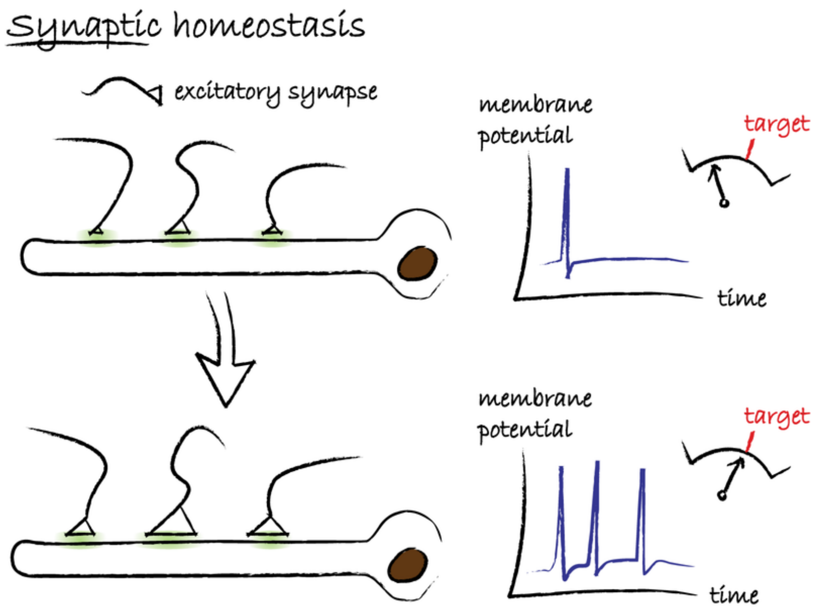


FIGURE 1.9: **Synaptic homeostasis.** A schematic illustration of a neuron with multiple excitatory synaptic inputs is shown on the left. If the average firing rate is too low (top), the excitatory inputs can be strengthened to increase the firing rate to an appropriate level (bottom). Figure adapted from Williams et al. (2013).

The importance of such mechanism had been highlighted for computational reasons much earlier on. Early computational models of neural networks that achieved learning by updating synaptic weights according to a learning rule (such as those reviewed in the previous sections) showed that these rules, left in isolation, are unstable. An increase in synaptic weight would produce a larger post-synaptic response, which would result in a larger synaptic weight via Hebbian learning, and so on, until the weight grows unbounded. Homeostatic synaptic plasticity rules were thus introduced to prevent this phenomenon. A famous example is Oja's rule, (Oja, 1982) which renormalizes the synaptic weights as they are updated. This rule lacks biological realism but has shown how Hebbian learning could be used to perform a dimensionality reduction on its inputs equivalent to principal component analysis. A more biologically plausible example of stable Hebbian learning is the so-called BCM rule, from the initials of the authors Bienenstock, Cooper and Munro (Bienenstock et al., 1982). This learning rule was conceived to explain the development of orientation specificity and binocular interaction in visual cortex. In the BCM rule the synaptic potentiation saturates as the post-synaptic activity increases. Both the Oja rule and the BCM rule also provide a mechanism for *competitive Hebbian learning*, wherein synaptic stability is further helped by the fact that synapses sharing the same postsynaptic neuron compete to drive its activity in different directions (some up, some down). These models also helped explain the experimentally-observed competitive processes that appeared to counter unstable growth of synaptic strength. In turn, this theoretical work was instrumental for our understanding that classical Hebbian learning required modification or an additional homeostatic process in a practical, biological implementation.

The learning rule used in this work is modulated by reward and decision activity. For this reason, no synaptic homeostatic process is required for its stability. However, the activity of some neurons could become extremely low during training, as shown in more detail in Ch. 3 (Sec. 3.4.2). What is needed, then, is a mechanism that keeps the activity of the neurons close to the threshold for making decisions. This is an example of *intrinsic homeostatic plasticity*.

The first model of intrinsic homeostatic plasticity was developed by LeMasson et al. (1993) and Abbott and LeMasson (1993). This model was inspired by the observation that in crustacean rhythm-generating circuits neurons appear to maintain characteristic electrical activity. It was proposed that neurons have built-in sensors that monitors electrical activity and adjust conductance densities to maintain a 'target' activity level (Fig. 1.10).

One such mechanism that is well known to neurobiologists is firing rate adaptation, the phenomenon through which the rate of action potential generation decreases over time during sustained stimulation. This phenomenon is mediated by a variety of ionic currents, notably sodium- and calcium-dependent potassium currents (Sah, 1996), and is well described by a minimal model of adapting cortical neurons (La Camera et al., 2004). However, firing rate adaptation only works in one direction, namely, to reduce the firing rate. If an increase in firing rate is required (for example, to compensate for a decrease in afferent synaptic weights), then the mechanism of firing rate adaptation will have no effect. Instead, what is needed is a bidirectional mechanism that increases the firing rate if the latter is below a target value, and decreases it if the firing rate is above the target value, just as pictured in Fig. 1.10.

Such a mechanism can be minimally implemented as described in Sec. 2.2.1. It is useful to anticipate, at this point, that modifying the synaptic weights for the purpose of regulating the activity of neurons interferes with learning. The latter requires its own synaptic changes aimed at learning the task at hand

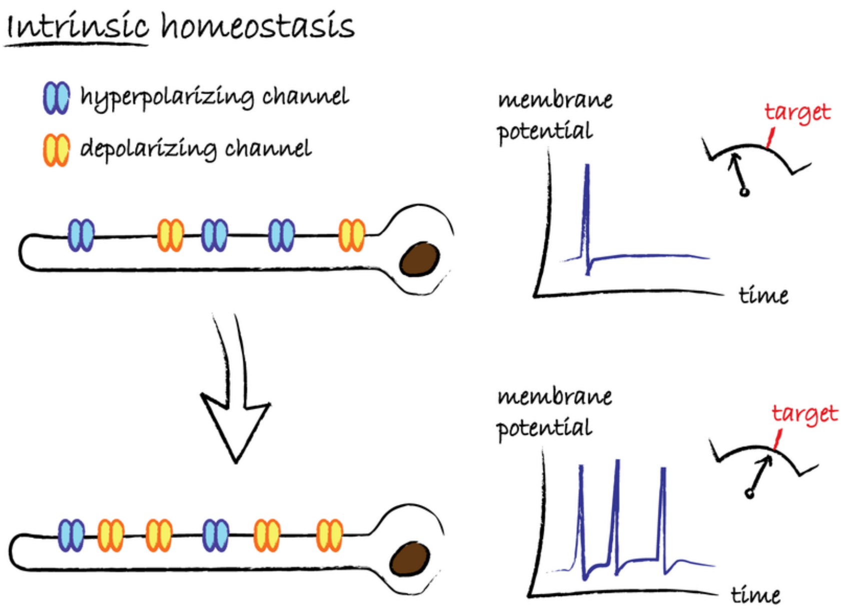


FIGURE 1.10: **Intrinsic homeostasis.** A schematic illustration of a neuron with a mix of inward and outward currents is shown on the left. If the average firing rate is too low (top), the inward currents can be up-regulated and the outward currents can be down-regulated to increase the firing rate to an appropriate level (bottom). Figure adapted from Williams et al. (2013).

(in our case, the segmentation of relevant segments in an input stream). Additional mechanisms of homeostatic plasticity will compete with the requirements of the learning rule and degrade learning. This was observed in a variety of homeostatic synaptic scenarios and led to the development of the intrinsic homeostatic mechanisms discussed in sections 3.4.2 and 3.4.3.

1.6 Reinforcement learning with spiking neurons

Reinforcement learning requires a source of stochasticity in the neuron's output, for example via stochastic synapses (Seung, 2003) or via the escape noise mechanism mentioned in Sec. 1.3.4. In this work I use the latter. As I will show in detail in Ch. 2, a reinforcement learning rule for a synapse w_{ij} connecting two spike response models with escape noise can be written as

$$\Delta w_{ij} = \eta \langle R(\delta_i - \phi_i(u)dt)PSP_j\rangle. \tag{1.18}$$

Here, Δw_{ij} is the variation in synaptic weight due to learning, R is the reward, $\delta_i = 1$ if the postsynaptic neuron i emits a spike in the current time bin (and zero otherwise), $\phi_i(u)dt$ is the probability that the postsynaptic neuron fires a spike in the same bin, dt is the (short) duration of the time bin, and PSP_j is the value of the post-synaptic potential in response to a spike from presynaptic neuron j. The brackets $\langle\cdot\rangle$ indicate an average across many trials, as is customary in *batch* updating. In this work, given the nature of the temporal segmentation problem, I will use instead an *online* version of the learning rule that updates the weights after each presentation, using the current value of $R(\delta_i - \phi_i(u)dt)PSP_j$ as a proxy of its average. This procedure, called *stochastic gradient ascent*, works rather well in practice.

Note that Eq. 1.18 is a biologically plausible implementation of the more heuristic Eq. 1.2. To see this, write Eq. 1.2 in the following form:

$$\Delta w_{ij} = \eta \langle R(A_{post} - \theta)A_{pre} \rangle. \tag{1.19}$$

Comparing this equation to Eq. 1.18,

$$\Delta w_{ij} = \eta \langle R(\delta_i - \phi_i(u)dt)PSP_j \rangle,$$

one notices that the postsynaptic activity A_{post} is represented there by the spike output δ_i of the neuron, the activity threshold θ is the probability of firing a spike, i.e. $\phi(u)dt$ (note how this can be approximated by the firing rate of the neuron itself), and the presynaptic activity A_{pre} is represented by the term PSP_j.

Although Eq. 1.2 was heuristically motivated, it will be shown in Appendix A that Eq. 1.18 performs gradient ascent in the average reward.

1.6.1 Limitations of reinforcement learning

Reinforcement learning is a powerful and general learning method, which however suffers from a number of issues that has often limited its applicability to real-life problems. Although these limitations will not affect my solution of the temporal segmentation problem, I list here for completeness the most important of them (see e.g. Alonso and Mondragón (2013)):

1. Exploration-exploitation balance: reinforcement learning assumes that, for optimal performance, agents explore (state-action pairs for which the outcome is unknown) and exploit (those state-action pairs for which rewards are known to be high). Finding the right balance between exploration and exploitation is not a straightforward exercise.

2. Temporal discounting: A discount factor is set for delayed rewards representing the fact that it is preferred to obtain the reward sooner rather than later. The problem is that small discounts can make the learner too greedy for present rewards and indifferent to the future, while large discounts slow learning down.

3. Generalization: the reinforcement learning approach does not allow for the ‘transfer’ of learning between different (yet still similar) situations. What is learned depends on the reward structure – if the rewards change, learning has to start over.

4. Large state spaces: for most practical tasks with large state spaces reinforcement learning fails to converge. Besides, it generates extreme computational costs when not dealing with small numbers of state-action pairs – which are very rare in any real learning scenario. For example, all state-action pairs must be repeatedly visited, which in practice means that many thousands of training iterations are required for convergence in even modest-sized problems.

1.7 The stimulus segmentation problem

The problem considered in this work is a combination of a decision task and a special type of stimulus identification task. Stimulus identification, illustrated in Fig. 1.11a), is the process of picking out a relevant stimulus among many other stimuli which may be present in the environment, for the purpose of performing a task. Here and in the following I will always assume that stimuli are relevant if they are *behaviorally relevant*, i.e., if a correct action in their presence leads to a rewarding outcome. In the figure this problem is illustrated as a fictitious laboratory task where a monkey is being presented with 4 different stimuli on a computer screen and must select which one would lead to reward.

In a more demanding scenario, this problem becomes a problem of stimulus *segmentation*, and amounts to extracting action-relevant segments out of a noisy input stream. This also involves the extrapolation of the onset and the end of the stimulus segments, not known a priori, and is akin to e.g. the problem of phoneme segmentation in speech recognition. In the stimulus segmentation problem the stimuli are not well defined, do not stand out from the environment as those in panel a) of Fig. 1.11 (which already hints at their possible relevance), and may be hidden in a spatio-temporal input. In particular, inside our brains everything is represented as a pattern of spike trains, which must be segmented to find those portions that represent the relevant stimuli, as shown in Fig. 1.11b). In this example, the segments shaded in pick represent relevant stimuli demanding one type of decision, while the segment in green demands a different decision. Thus, the problem is a combination of stimulus segmentation and decision making. A fictitious laboratory task corresponding to this problem is one in which a moving pattern moves very slightly and very slowly emerges on a computer screen from a noisy background, and the subject must learn to identify the occurrence of the pattern together with the action needed in response.

It is worth notice that no state-of-the-art artificial device (such as the celebrated ‘deep learning’) is able to deal with this problem. Most of the time the stimuli are known to the learning agent, and the task is to associate correct actions to given inputs. Here, timing and relevance of the stimuli are unknown, and learning amounts to a complete process of discovery. It is interesting that framing the problem in terms of reinforcement learning with spiking neurons not only represents a biologically plausible solution, but allows the online learning procedure that is necessary to solve this problem in real time. Machine learning algorithms commonly require off-line computations to solve segmentation problems. For example, this is the case of Hidden Markov Models widely used to solve speech recognition problems, see e.g. Rabiner et al. (1989) and Wilpon et al. (1990). Moreover, if the stimulus onset is not characterized by a changing in intensity (i.e., the firing rate of the spike trains or their cross-correlations), or it is not preceded by predictive cues, the problem of stimulus segmentation cannot be solved by unsupervised learning algorithms because those only look at deviations in the statistical structure of the stimuli. I will show in Ch. 3 that this is not a problem for the model introduced here.

As said in the previous sections, animals presumably learn to navigate their environments by exploring actions and repeating those actions that led to rewarding outcomes. For this reason, the problem of identifying relevant cues and stimuli can naturally be framed as a problem of reinforcement learning (see also Sutton and Barto (1998)). This work follows in the footsteps of a handful of previous works that have pioneered reinforcement learning with spiking neurons (Urbanczik and Senn, 2009; Friedrich et al., 2011; Pfister et al., 2006). Those works focused on learning classification tasks in the traditional

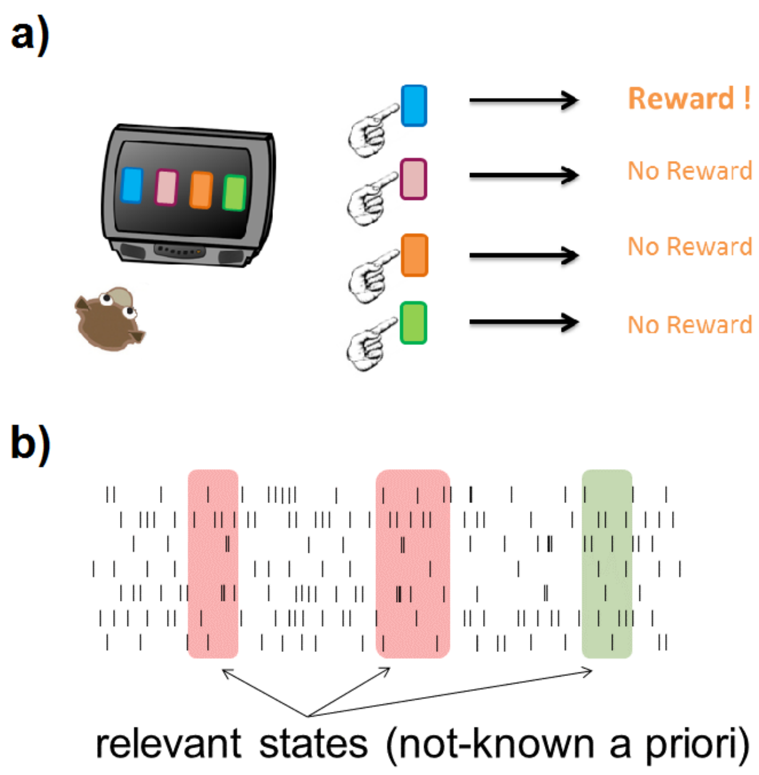


FIGURE 1.11: **Stimulus identification and stimulus segmentation problems.** **a)** Example of stimulus identification, which is the process of picking out a particular stimulus among many other stimuli which may be present in the environment, for the purpose of performing a task. Here this problem is illustrated as a fictitious laboratory task where a monkey is being presented with 4 different stimuli on a computer screen and must select which one would lead to reward. **b)** In the stimulus segmentation problem the stimuli are not well defined, do not stand out from the environment as those in panel a), and may be hidden in a spatio-temporal input pattern. See the text for more details.

setting in which stimuli's relevance and appearance are completely known to the learning agent. The novelty of those works, and especially of the Urbanczik and Senn (2009) paper, is the ability to deal with spatio-temporal spike patterns and was motivated by the need to build a biologically plausible theory of reinforcement learning in populations of spiking neurons. Following that study up, La Camera et al. (2013) noticed that the same framework *could be augmented to actually identify the relevant stimuli in the absence of any information on their identity and timing*. That work showed how to do so in a combined segmentation and binary decision making task, using a single population of decision neurons. Although promising, that model had unsatisfactory performance with non-relevant stimuli and could only deal with binary decisions (such as 'go left' vs. 'go right'). The objective of this project was to extend that model to deal with any number of different decisions while at the same time improving the performance with non-relevant stimuli. The key observation that made this possible is the discovery of the need of a slow homeostatic mechanism that allows the model to escape from local maxima of the average reward, and will be presented in detail in Ch. 3 (Sec. 3.4.2).

1.8 Outline of this work

In this chapter I have motivated the need for models in neuroscience by listing the contributions of some notable models to our current knowledge of neuroscience, and have discussed different types of learning strategies and neuronal models. I have also introduced the main topic of this thesis: the stimulus segmentation problem and the reinforcement learning method.

In Chapter 2 I will present the details of the network model, including its architecture, its model neurons, the learning rule and how the input stimuli are constructed.

In Chapter 3 I will show the ability of the model to learn to segment an input stream of spike trains, the main objective of this work. I will also show the robustness of the model to different encoding strategies of the input stimuli, l will analyze the role of some of the ingredients of the model, and I will introduce an alternative model.

In Chapter 4 I will demonstrate the ability of the model to identify cortical spike patterns recorded in behaving rats in response to taste stimuli, and I will show that the model is able to generalize past learning to novel stimuli.

Finally, in Chapter 5 I will summarize how this model is able to solve the stimulus segmentation and the multi-choice decision problems, and I will discuss its biological plausibility, its novelty, its weak and strong points, and where the research described in this work can be further taken from here.

Chapter 2

The model

In this chapter I introduce all the ingredients of the model before moving on to discuss its performance in the temporal segmentation task in Ch. 3.

As a reminder, the main objective is to develop a theory of stimulus segmentation by reinforcement learning in populations of spiking neurons. The aim of the decision maker is to learn to extract segments of a noisy input stream that are relevant for behavior. The relevant segments are not known a priori, and thus it is also assumed that the agent has no knowledge of the time at which those segments are presented. This is a major departure from existing models and requires the extrapolation of the onset and the end of each stimulus during learning. The network will be trained to identify the relevant stimuli (i.e., those that can trigger rewarded decisions) by reinforcement of correct decisions. The network architecture, the details of the neuron model, the learning rule and the construction of the stimuli will be all introduced in detail in this chapter.

2.1 Network architecture

The model network comprises N_{dec} populations of spiking neurons that receive input spike trains via plastic synapses (Fig. 2.1a). Each population μ codes for a specific decision and initiates the corresponding action whenever a measure of population activity P_s^μ crosses a decision threshold, i.e., a pre-defined level of activity θ (the same for all populations). As long as all population scores are below threshold, no decisions are taken and no feedback is given to the agent. The decision neurons producing scores P_s^μ were modeled as spike response models (Sec. 1.3.4). In this model, neuron ν is defined by membrane potential u^ν obeying the SRM equation 1.12 and depicted in Fig. 1.5:

$$u_i(t) \;=\; u_{rest} \;+\; \sum_{i=1}^{M} w_i \sum_{s \in X_i} \frac{1}{\tau_m} e^{-t/\tau_m} \;+\; \sum_{s \in Y} \frac{1}{\tau_m} e^{-t/\tau_m}.$$

Stimuli were continuously presented to the agent as streaming spatio-temporal patterns of spike trains (details in Sec. 2.4), and were divided into relevant and non-relevant stimuli. Each relevant stimulus was arbitrarily associated with a correct decision, one among the N_{dec} possible decisions. When a decision occurred, a rewarding feedback was given after a minor delay (30 *ms*), the stimulus was removed, and the population readout was reset to zero. Every decision (whether correct or incorrect) incurred a small cost (typically -0.1, to prevent the agent from continuously taking decisions); positive reward ($R = 1$) was given only for a correct decision in the presence of a relevant stimulus (netting a total reward of $R = 0.9$). Incorrect decisions were not punished (and thus only incurred the cost $R = -0.1$). The rationale for such choice is that an additional negative reward for an incorrect response to a relevant stimulus would signal the presence of a relevant stimulus at the time of the decision, aiding the solution of the segmentation task. In the (rare) case of multiple populations reaching threshold simultaneously, only one was selected pseudorandomly as the population responsible for the decision.

The network makes a decision when (and only when) the activity (to be defined shortly) of a population exceeds a threshold. For this reason, the population neurons are also called 'decision neurons' in the following. Note that the two decision populations do not influence each other and the two decision processes are independent.

2.1.1 Population scores and decision dynamics

The ongoing activity of population μ, defined as

$$A^{\mu}(t) = \int_0^t ds \sum_k \delta(s - t_k^{\mu}), \tag{2.1}$$

is a measure of the collective spike count of its neurons between times zero and t. The times t_k^{μ} are the times of the spikes emitted up to time t by the decision neurons of population μ, and $\delta(t)$ is Dirac's delta function. Since at the heart of the stimulus segmentation problem is the fact that the learning agent does not know when a stimulus starts or ends, $A^{\mu}(t)$ was approximated by a low-pass filter of its temporal derivative, $\dot{A}^{\mu}(t)$, to produce the online 'population score' P_s^{μ} responsible for initiating a decision:

$$\tau_P \frac{dP_s^{\mu}}{dt} = -P_s^{\mu} + \frac{dA^{\mu}}{dt}. \tag{2.2}$$

where τ_P was 500 *ms*, the same as the mean stimulus durations. A decision μ was said to occur at the first time t_d such that $P_s^{\mu}(t_d) > \Theta_D > P_s^{\neq\mu}(t_d)$ since the last decision, and at time $t_d^* \equiv t_d + 30$ *ms* all population scores P_s^{μ} were reset to zero. The 30 *ms* delay between the decision and the reset adds biological realism while allowing a confidence signal to build up, as follows. At time t_d^* following a decision by population μ, a feedback on the decision is given to all neurons in the population through the quantity $\tilde{P}_s^{\mu}$ obeying

$$\tilde{\tau} \frac{d\tilde{P}_s^{\mu}}{dt} = -\tilde{P}_s^{\mu} + \mathbf{1}_{50} \exp\left(-\left(\frac{P_s^{\mu}(t_d^*) - \Theta_D}{\Theta_D}\right)^2\right), \tag{2.3}$$

with $\tilde{\tau} = 50$ *ms* and $\mathbf{1}_{50} = 0$ except for 50 *ms* after t_d^*, when it is set to 1. $\tilde{P}_s^{\mu}$ has a double role: its magnitude can represent the population's own confidence in the decision at time t_d^* and will be used to attenuate learning (Eq. 2.19 and Eq. 2.20), whereas its sign will be used to build an individualized reward signal as proposed in Urbanczik and Senn (2009), Eq. 2.17.

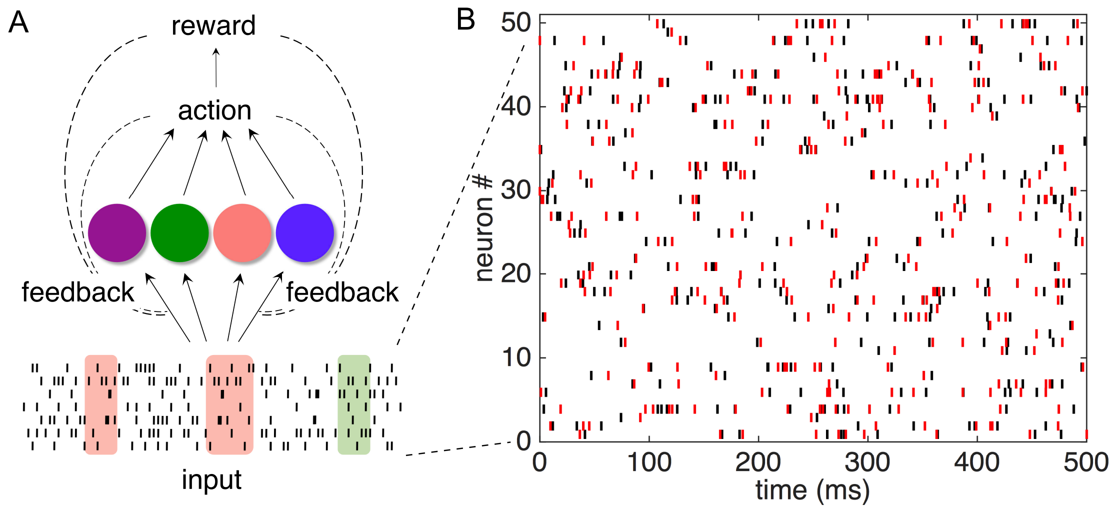


FIGURE 2.1: **A) Network architecture.** Spike trains ('input') from M input neurons project to N_{dec} populations of spiking neurons (colored circles) through plastic synapses. Each population is responsible for a different decision, which occurs when a measure of the spike count of each population first crosses a predefined threshold. After each decision (and only then), feedback from reward (right vs. wrong) and own population activity determines the change of synaptic weights between the input neurons and the population neurons responsible for the current decision, according to Eq. 2.19. **B)** One stimulus prototype (black) together with a jittered version of it (red; 10 ms random dispersion around the original spike times, see Sec. 2.4 for details).

2.2 The neuron model

The decision neurons producing scores P_s^μ were modeled as spike response models with fixed threshold (Pfister et al., 2006). In this model, neuron ν is defined by its membrane potential u^ν obeying Eq.1.12,

$$u^\nu(t|\mathbf{X}) = u_{rest} + \sum_i w_i^\nu PSP_i(t,\mathbf{X}) - \sum_{t^\nu \in Y} e^{-\frac{t-t^\nu}{\tau_m}} \Theta(t-t^\nu), \tag{2.4}$$

where: $u_{rest} = -1$ is the resting potential (in arbitrary units); w_i^ν is the synaptic weight connecting neuron ν to its presynaptic neuron i; $\mathbf{X} \equiv \{X_1, X_2, \ldots, X_M\}$ is the collection of the M presynaptic input spike trains, where $X_i = \{t_1^{(i)}, t_2^{(i)}, \ldots, t_n^{(i)}\}$ are the spike times of the i^{th} spike train; $PSP_i(t,\mathbf{X})$ is the postsynaptic potential due to the input spike train X_i,

$$PSP_i(t,\mathbf{X}) = \sum_{s \in X_i} \frac{1}{\tau_m} e^{-\frac{t-s}{\tau_m}} \Theta(t-s), \tag{2.5}$$

with $\tau_m = 10\ ms$ being the membrane time constant of the neuron; Y is the output spike train (comprising spike times $\{t^\nu\}$); and $\Theta()$ is the Heaviside function. For a discussion about the spike response model see Sec. 1.3.4. A neuron with membrane potential u emits a spike with probability

$$\phi(u)dt = ke^{\beta u}dt. \tag{2.6}$$

in a short time bin dt. Here, $\beta = 5$ controls the stochasticity of spike emission, $k = 0.01$ is a constant.

2.2.1 Homeostatic mechanism

Neurons' activities were kept close to the decision threshold via a homeostatic mechanism. This was implemented via the update of the parameter k in Eq. 2.6 controlling the neuron spike probabilities. Specifically, k^μ (the same for all neurons in population μ) was changed according to

$$\Delta k^\mu = \tilde{\eta}(\Theta_D - f_t^\mu), \tag{2.7}$$

where f_t^μ is a running average of the spike count similar to P_s^μ (Eq. 2.2) but without post-decision reset and a much longer time constant (4 seconds); k^μ is constrained to be positive. The update Eq. 2.7 occurs at random times sampled from a uniform distribution with average inter-update time of $4P$ stimulus presentations, where P is the total number of stimuli and $\tilde{\eta} = 0.01$. Thus, k was kept fixed over several stimulus presentations not to interfere with the learning of the weights. This homeostatic mechanism is beneficial because, as learning modifies the weights, it also induces changes in population activity. These may occasionally produce very low activity in some populations, resulting in lack of decisions for those populations. When this occurs, the agent is trapped in a local maximum of the average reward, and learning practically stops for that population. In such a case, the homeostatic mechanism slowly increases the firing probability in single neurons until the desired population activity level is restored and decisions are again possible. The parameter k is slowly modified during learning to help keep the decision populations' activities near the decision threshold.

2.3 The learning rule

In this section I introduce the learning rule used in this work. I shall present only the main facts and equations leading to it, deferring a detailed derivation of most results to Appendix A.

2.3.1 The basic rule

Given input spike pattern $\mathbf{X}$ lasting from 0 to t, the log-likelihood that a decision neuron produces the output spike train Y^ν, given the neuron model described in Sec. 2.2, is (Pfister et al., 2006):

$$\mathscr{L}_{\mathbf{w}^\nu}(Y^\nu|\mathbf{X}) \equiv \ln \mathscr{P}_{\mathbf{w}^\nu}(Y^\nu|\mathbf{X}) = \sum_{t^\nu \in Y^\nu} \ln \phi(u(t^\nu)) - \int_0^t dt' \phi(u(t')), \tag{2.8}$$

where $Y^\nu = \{t^\nu\}$ is the output spike train generate by neuron ν. The learning rule was designed to perform a variation of the method of gradient ascent in the average reward. Following Williams (1992), the gradient of the log-likelihood $\mathscr{L}_{\mathbf{w}^\nu}$ is required, which is in the case of the spike response model used here is

$$\frac{\partial}{\partial w_i^\nu}\mathscr{L}_{\mathbf{w}^\nu}(Y^\nu|\mathbf{X}) = \sum_{t^\nu \in Y^\nu} \beta PSP_i(t^\nu|\mathbf{X}) - \int_0^t \beta\phi(u(t'))PSP_i(t'|\mathbf{X})dt' = \tag{2.9}$$

$$= \int_0^t \beta \left(\sum_{t^\nu \in Y^\nu} \delta(t'-t^\nu) - \phi(u^\nu(t')) \right) PSP_i(t',\mathbf{X})dt'. \tag{2.10}$$

Note that I have introduced Dirac's delta function to make the term $\sum_{t^\nu \in Y^\nu} \beta PSP_i(t^\nu|\mathbf{X})$ appear under the integral.

As I show in Appendix A, an online learning rule performing gradient ascent in the average reward would read (Williams, 1992)

$$\frac{dw_i^\nu}{dt} = \eta R_t \frac{\partial}{\partial w_i^\nu}\mathscr{L}_{\mathbf{w}^\nu}(Y^\nu|\mathbf{X}), \tag{2.11}$$

where η is the learning rate and R_t is the reward at time t. The quantity $\frac{\partial}{\partial w_i^\nu}\mathscr{L}_{\mathbf{w}^\nu}(Y^\nu|\mathbf{X})$ is called the *eligibility trace* of the synapse because, according to this learning rule, the synapse is 'eligible' for change only when the eligibility trace is different than zero.

By using Eq. 2.10, this learning rule becomes

$$\frac{dw_i^\nu}{dt} = \eta R_t \int_0^t \beta \left(\sum_{t^\nu \in Y^\nu} \delta(t'-t^\nu) - \phi(u^\nu(t')) \right) PSP_i(t',\mathbf{X})dt'. \tag{2.12}$$

Note the integration times 0 (stimulus onset) and t (stimulus offset) in the right hand side: this learning rule would require the knowledge of when the stimulus is being presented. But the whole point of this work is to build a model capable of inferring the presence of relevant stimuli which are a priori unknown. *Telling the system when the stimuli are being presented gives it information that it is not supposed to have.* To avoid this problem I replace, in the learning rule above, the eligibility trace with a low-pass filter of its time-derivative, which I will indicate with the symbol E:

$$\tau_M \dot{E}_i^\nu = -E_i^\nu + \frac{d}{dt}\frac{\partial}{\partial w_i^\nu}\mathscr{L}_{\mathbf{w}}(Y^\nu|\mathbf{X}), \tag{2.13}$$

where $\frac{d}{dt}\frac{\partial}{\partial w_i^\nu}\mathscr{L}_{\mathbf{w}}(Y^\nu|\mathbf{X})$ is obtained immediately from Eq. 2.10:

$$\frac{d}{dt}\frac{\partial}{\partial w_i^\nu}\mathscr{L}_{\mathbf{w}}(Y^\nu|\mathbf{X}) = \beta \left(\sum_{t^\nu \in Y^\nu} \delta(t-t^\nu) - \phi(u^\nu(t)) \right) PSP_i(t,\mathbf{X}). \tag{2.14}$$

The modified learning rule is therefore

$$\frac{dw_i^\nu}{dt} = \eta R_t E_t, \tag{2.15}$$

with E_t given by Eq. 2.13.

Note that the right hand side of Eq. 2.14 contains the product of a postsynaptic term (the difference between the actual firing rate $\sum_{t^\nu} \delta(t-t^\nu)$ and the average firing rate ϕdt), and a presynaptic term (the postsynaptic potential PSP_i). Together with the reward-modulated term in Eq. 2.19, the latter is a three-factor synaptic plasticity rule as found e.g. in cortico-striatal synapses, a major locus for *reinforcement learning* (Reynolds and Wickens, 2002).

2.3.2 The learning rule with individualized reward

The learning rule of the previous section has been improved by Urbanczik and Senn (2009) to alleviate the spatial credit assignment problem. The problem is that, as the number of neurons in the network grows, the learning performance nevertheless deteriorates quickly with increasing N (see Fig. 2.2, panel a). The reason for this is rather simple: from the perspective of the single neuron, the global reward R is an unreliable performance measure, since the neuron may be punished for a correct response just because the majority of other neurons made a mistake. The odds of this happening increase with the size of population, the average single neuron response deteriorates and this is not compensated for by the boost provided via the population response (Urbanczik and Senn, 2009). For this reason I follow Urbanczik and Senn (2009) and train the neurons individually and only use the population read-out to improve the credit to each neuron. For this I assume an individual reinforcement signal $r^\nu = \pm 1$ that equals 1 if neuron ν took the right decision, and -1 otherwise (see section 'Individualized reward' below for its biological implementation). I then follow Urbanczik and Senn (2009) and replace the global reward with the individualized reward signal r^ν for neuron ν:

$$\frac{dw_i^\nu}{dt} = \eta |R_t| (r^\nu - 1) E_i^\nu, \tag{2.16}$$

where R_t is an estimate of reward at time t and is modeled as a low-pass filter of the actual reward delivered after each decision. Since $R_t \neq 0$ only around a decision time (when reward is given), the factor $|R_t|$ confines the synaptic update to a temporal window around reward delivery.

As shown by Urbanczik and Senn (2009), this modification speeds up learning when the population size increases, at least in a more traditional reinforcement learning task (see Fig. 2.2, compare panel a) with panel b)). Later, in Ch. 3 (Fig. 3.3), I will demonstrate that this is the case also for the temporal segmentation task which is the object of this thesis.

2.3.3 Biological implementation of individualized reward

The individualized reward signal could be made available locally at each synapse by broadcasting feedback from

- the population's own activity via the $\tilde{P}_s^\mu$ term defined in Eq. 2.3 (representing the deviation from equilibrium of the concentration of a neurotransmitter such as acetylcholine or serotonin);
- each neuron's own activity S_t^ν (e.g., through intracellular calcium transients), and
- the global reward feedback R_t (mediated e.g. by the concentration of dopamine).

Specifically, for the ν neuron in population μ (Urbanczik and Senn, 2009),

$$r^\nu = \operatorname{sign}(R_t \tilde{P}_s^\mu (S_t^\nu - \theta_S)), \tag{2.17}$$

where $\theta_S = e^{-1.1}$ is a threshold for S_t^ν. R_t is transiently driven by the global reward R (Sec. 2.1) for 50 *ms* after each decision time t_d^*, and then decays to zero exponentially with time constant of 10 *ms*. The neuron's activity S_t^ν was given by

$$\dot{S}_t^\nu = -\frac{S_t^\nu}{\tau_S} + (1 - S_t^\nu) \sum_{t^\nu} \delta(t - t^\nu), \tag{2.18}$$

where the $\{t^\nu\}$ are the neuron's output spike times, and $\tau_S = 500\ ms$. If, for example, the decision was correct ($R_t > 0$) and population μ was above threshold, its neurons which had been sufficiently active ($S^\nu > \theta_S$) had voted for the correct decision, hence for them $r^\nu = 1$. One can similarly work out all other possible scenarios and confirm that the Eq. 2.17 indeed provides the correct feedback in each case.

2.3.4 Learning attenuation

Although individual reward works far better than global reward, Urbanczik and Senn (2009) found that the performance nevertheless saturates rather quickly with increasing number of decision neurons N, and 100% performance is not reached (see Fig. 2.2, red curve in panel b). The problem is that the neurons are all trying to learn the same thing and this leads to correlations which are detrimental to the population performance.

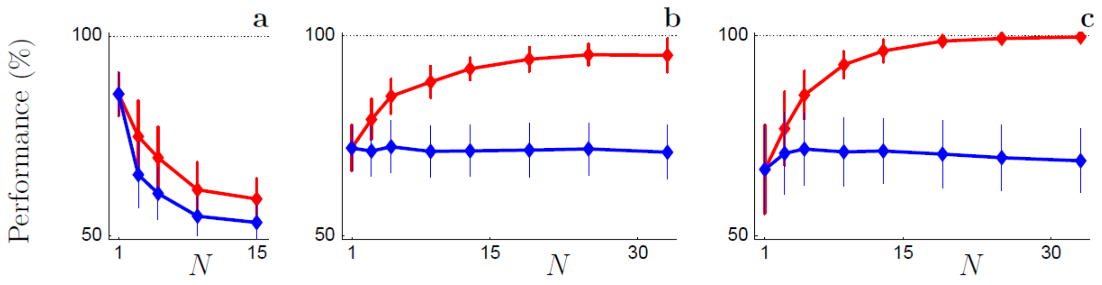


FIGURE 2.2: **The role of individualized reward and learning attenuation.** Learning performance in a 2-choice task with 30 patterns that had to be learned with target responses equally split between two output classes. The curves represent the performance (percentage of correct responses) after a fixed number of learning episodes as function of the population size N. The red curves show the performance of the population read-out, the blue ones show the average performance of the single neurons. **a)** Learning based just on global reward, Eq. 2.15. **b)** Learning with individual reward for each neuron, Eq. 2.16. **c)** Attenuated learning with individual reward, corresponding to the full learning rule Eq. 2.19 also used in this work. Figure adapted from Urbanczik and Senn (2009).

To address this problem, once again I follow Urbanczik and Senn and introduce in the learning rule an attenuation factor $a(\tilde{P}_s^\mu)$ based on the measure of population activity $\tilde{P}_s^\mu$ defined in Eq. 2.3, so that the learning rule Eq. 2.16 finally becomes

$$\boxed{\frac{dw_i^\nu}{dt} = \eta |R_t| a(\tilde{P}_s^\mu)(r^\nu - 1) E_i^\nu.} \tag{2.19}$$

The attenuation factor $a(\tilde{P}_s^\mu)$ is equal to 1 (no attenuation) for an incorrect decision ($R_t < 0$) and to $|\tilde{P}_s^\mu|$ for a correct decision ($R_t > 0$), with $\tilde{P}_s^\mu$ given by Eq. 2.3 (it is meant here that neuron ν belongs to population μ).

The attenuation factor $a(\tilde{P}_s^\mu)$ is responsible for reducing (attenuating) learning once the population response is reliable and correct. In the case of an *incorrect* decision, $a(\tilde{P}_s^\mu) = 1$ regardless of $\tilde{P}_s^\mu$, and a full synaptic weight update results from Eq. 2.19. In the case of a *correct* decision, the degree of synaptic change is modulated by (from Eq. 2.3)

$$a(\tilde{P}_s^\mu) = |\tilde{P}_s^\mu| \approx e^{-\left(\frac{P_s^\mu - \Theta_D}{\Theta_D}\right)^2}. \tag{2.20}$$

The rationale for this choice is the following: if P_s^μ is large or small compared to Θ_D soon after a *correct* decision, the population is highly confident of its vote and no further learning is required (and indeed $a(\tilde{P}_s^\mu) \approx 0$ in this case); otherwise, if $P_s^\mu(t_d^*) \approx \Theta_D$, the population is maximally undecided about the current decision, and full learning is required ($a(\tilde{P}_s^\mu) \approx 1$).

In the case of episodic learning, the learning rule Eq. 2.19 performs stochastic gradient ascent in a monotonic function of reward and population activity, as proved in Urbanczik and Senn (2009). This learning rule can be understood as an improvement over Williams's general gradient learning rule Eq. 2.15 and, with E_i^ν given by Eqs. 2.13 and 2.14, is the learning rule used in this work.

2.4 Input stimuli

The input stimuli were always collections of spike trains across N_i input neurons, where both N_i and the firing rates of the spike trains depended on the task. Stimuli were jittered versions of prototypical spatiotemporal patterns of 50 Poisson spike trains with constant firing rates sampled from a uniform distribution between 2 and 24 Hz (Fig. 2.3). Stimuli prototypes were initially generated and then, prior to each presentation, temporal jitter was added to the spike times to produce a noisy version of the prototypes. For each spike time t_{sp}, jitter was added by sampling from a Gaussian distribution centered on t_{sp} and a standard deviation of $\sigma = 10$ *ms* (Fig. 2.3a; for comparison, a prototype modified with a jitter having standard deviation of 100 ms is shown in panel b). The rationale behind using this kind of model stimuli is that *they closely relate to real patterns of spike trains recorded from sensory cortex*, as we will see in Ch. 4.

To check the *robustness* of the model to different ways of encoding stimuli, I also tested the model with 2 special types of stimuli. In the first case, all 50 spike trains in each prototype had the same firing rate of 12 Hz, and each new presentation added a random jitter of 10 ms, as described above (see Fig. 2.4a for an example). This is a more difficult problem to learn, since only the variations in the spike times allows to differentiate between any two stimuli. I refer to this case as **uniform firing rate stimuli**. This case is meant to emulate a *precise spike timing code*. In the second type of special stimuli, firing rates were distributed across the spike trains (uniformly between 2 and 24 Hz), but the spike times were generated anew at each presentation (see Fig. 2.4b). This is equivalent to having a uniformly distributed jitter on the spike times covering the whole duration of the stimulus. I refer to this type of stimuli as **firing rate patterns**, because the only quantity conserved across multiple presentations is the firing rate of each spike train. This coding strategy is the closest to the classical *rate coding* assumption in the theory of sensory coding. This case is also more difficult to handle than the standard encoding with a jitter of

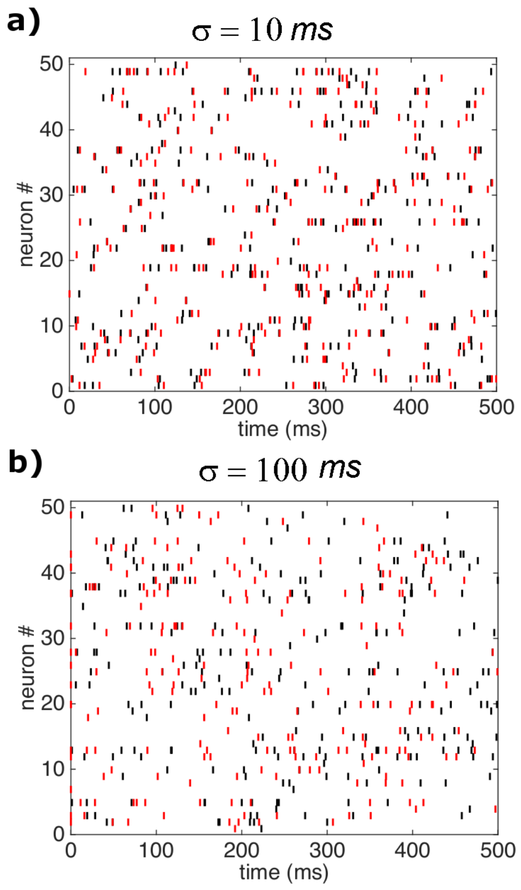


FIGURE 2.3: **Spike pattern stimuli used as inputs to the network.** Prototype stimuli (black) and their 'jittered' version (red) used in simulations of the temporal segmentation task. The stimuli are shown as 'raster plots': each row represents a neuron, whose firing times are marked by short vertical bars. A stimulus is defined by a collection of Poisson spike trains with constant firing rates ranging from 2 to 24 Hz. The spike times are jittered during each stimulus presentation, each time drawn from a Gaussian distribution with standard deviation σ centered around the original spike time in the prototype. For comparison, two prototypes jittered by 10 (a) and 100 ms (b) are shown here. See the text for more details.

10 ms. Comparison of the performance of the model with differently coded stimuli will be presented in the next chapter.

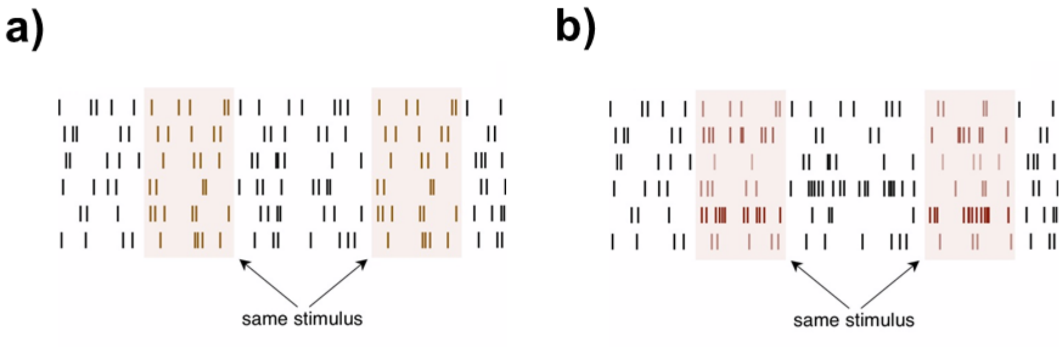


FIGURE 2.4: **Examples of stimuli used to test the robustness of the model. a)** *Uniform firing rate stimuli.* The shaded portions of the input stream represent the same stimulus. The firing rate is the same across all spike trains. For illustration purposes, no jitter has been added to the spike times, so that the two stimuli look identical, however in simulations each spike time was jittered by 10 ms with the same procedure described for the main stimuli. **b)**. Firing rate patterns. The shaded regions represent the same stimulus, even though the spike times were randomly drawn each time according to a Poisson process.

Chapter 3

Results

In this chapter I show that the model described in the previous chapter can successfully learn to solve the combined stimulus-segmentation and decision making problem by reinforcement learning. In later sections I will also consider the robustness of the model to different stimulus encoding strategies, and an analysis of the role of some key ingredients of the model, namely the role of the individualized reward and the role of the homeostatic mechanism that adapts the spike emission probability (described in Sec. 2.2.1).

All computer simulations described in the following were run in Matlab using custom computer code. A first-order Euler scheme was used for the numerical integration of differential equations with an elementary time step of 0.2 ms. The membrane potentials of the decision neurons were initially set to their resting values. Unless explicitly stated otherwise, initial synaptic values from pre-synaptic spike trains to post-synaptic decision neurons were sampled from a Gaussian distribution with zero mean and standard deviation of 2, and results are presented for simulations with 2000 stimulus presentations (either relevant or non-relevant).

The initial conditions were chosen so as to obtain similar initial performance across tasks (here, around 20%-correct with relevant stimuli). This is important to be able to compare the learning performance across tasks and simulations, because the initial performance of the network affects the learning time. For example, if the network makes a large number of decisions each second, it is not surprising that eventually it will learn to segment the temporal stream, and that it will do so fast. An initial performance of 20%-correct with relevant stimuli can insure that the initial rate of decisions is low, about one decision per second.

3.1 Illustrative examples

I first illustrate the problem in the simplest possible scenario of one relevant stimulus among $P = 4$ stimuli presented to the agent. The goal of the agent was to respond whenever the relevant stimulus was on. As shown in Fig. 3.1, the agent learned to successfully identify the relevant stimulus and respond

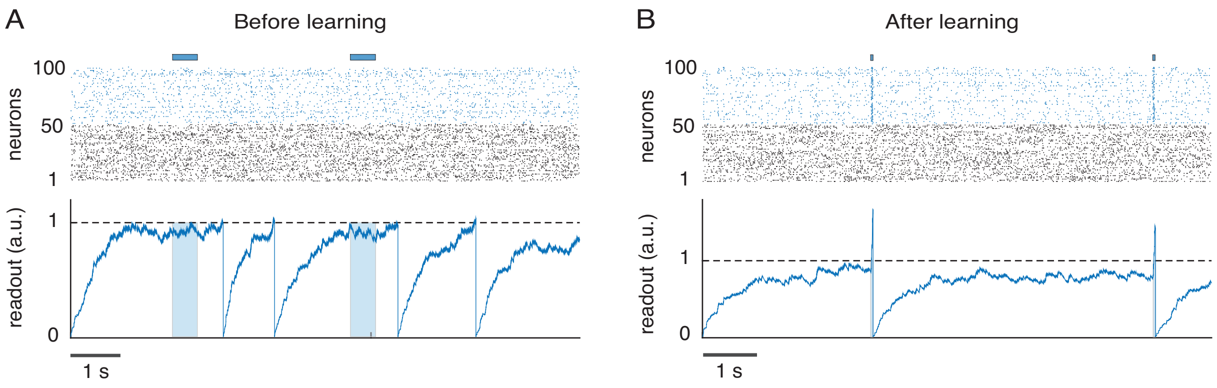


FIGURE 3.1: **Rasters and population activity vs. time in the basic segmentation task** with $N = 50$ decision neurons. **A.** Before learning. **B.** After learning. In each top panel: raster plot, with input spike train in black and decision neurons in blue. In each bottom panel: population score P_s^μ (blue; Eq. 2.2). A decision is taken when P_s^μ crosses a threshold $\theta_D = 1$ (dashed). Colored shading marks the presence of the relevant stimulus.

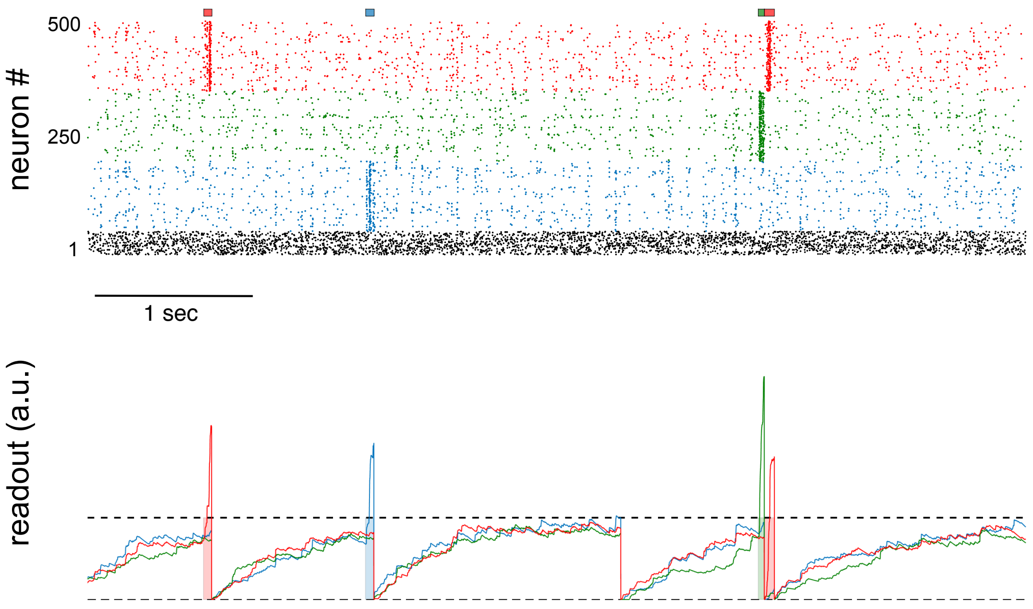


FIGURE 3.2: **Rasters and population activity in a three-way decision task** with $N = 150$ neurons per decision population. Same conventions as in Fig. 3.1, with the different stimuli demanding different decisions color-coded.

within 50 ms (or less) from its onset, while learning to ignore the other stimuli (Fig. 3.1, compare A with B).

Fig. 3.2 shows instead a simulation where *the temporal segmentation task occurs simultaneously with a three-way multiple-choice task* with 3 relevant stimuli and 9 non-relevant stimuli, respectively. The relevant stimuli required each a different decision. The behavior of the network is shown after sufficient training so that all relevant stimuli are correctly identified.

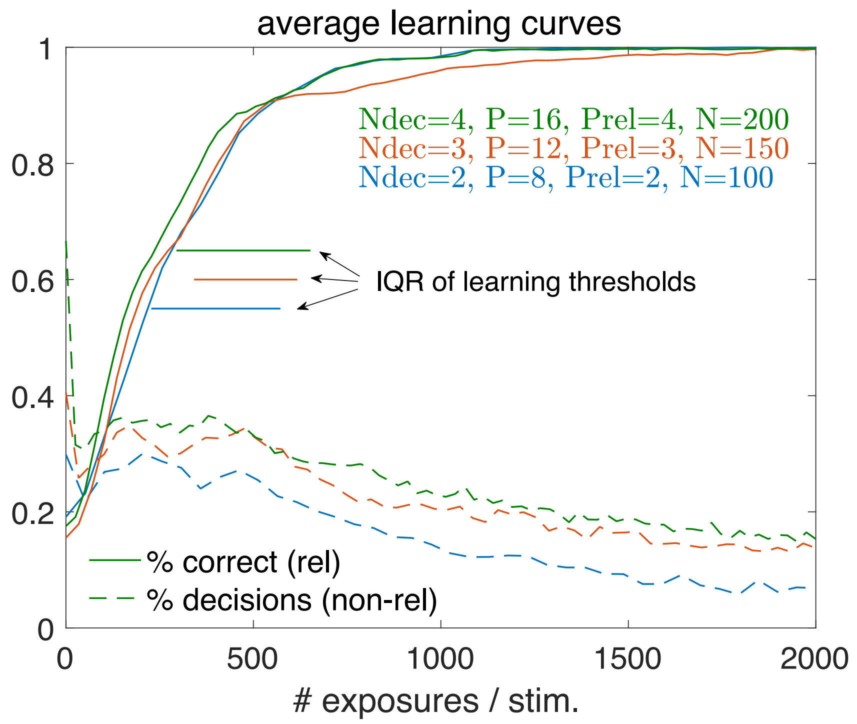


FIGURE 3.3: **Average learning curves** across 30 simulations for different tasks (color coded). *Solid curves:* average fraction of correct decisions with relevant stimuli; *dashed:* average fraction of (any) decisions in the presence of non-relevant stimuli (note that perfect performance with non-relevant stimuli is achieved when the dashed lines converge to zero). Horizontal segments denote the 25-75 percentiles (IQR = interquartile range) of the learning thresholds across simulations. Learning threshold is a measure of learning speed defined as the time (in number of relevant stimulus exposures) of the first of 100 consecutive correct decisions. *Key: Ndec*: number of different possible decisions; *P*: total number of stimuli; *Prel*: number of relevant stimuli (one for each decision); *N*: number of neurons in each decision population; *IQR*: interquartile range. Learning slows down with increasing task difficulty (i.e., in the presence of more decisions and/or stimuli). However, rescaling the number of decision neurons allows for similar learning times across tasks with different difficulty.

3.2 Average learning curves

The illustrative examples of the previous section do not prove that the model can reliably solve the segmentation problem, i.e., that training results in successful performance in a large fraction of tests. To prove this one must average the behavior of the model over a number of independent runs and independent problems. I show in this section that this is indeed the case.

Fig. 3.3 shows the *average learning curves* in various temporal segmentation tasks occurring simultaneously with a multiple-choice task with 2, 3 or 4 relevant stimuli (in the presence of 6, 9 and 12 non-relevant stimuli, respectively). The relevant stimuli required each a different decision.

The average learning curves shown in the figure report the average progress of the model performance during training. Specifically, learning performance with relevant stimuli was defined as the mean fraction f_r of correct decisions with relevant stimuli; learning performance with non-relevant stimuli was defined as $f_{nr} = 1 - f_{d,nr}$, where $f_{d,nr}$ was the mean fraction of decisions in the presence of non-relevant stimuli. f_r and $f_{d,nr}$ are shown in Fig. 3.3 as full and dashed curves, respectively; note that perfect performance is achieved when both $f_r = 1$ and $f_{d,nr} = 0$. Prior to plotting in Fig. 3.3, both measures of performance were averaged across multiple simulations, each with newly randomly generated prototype stimuli and synaptic weights, and then low-pass filtered according to $\bar{f}_n = (1-0.01)\bar{f}_{n-1} + 0.01 f_n$, where f_n is the performance during stimulus exposure # n and $\bar{f}_n$ its running average up to exposure # n.

The transition to good performance on relevant stimuli occurred at different times during training, depending on the initial weights, input prototypes and stochastic decisions. I call this transition the *learning threshold* and quantify it as the first of 100 consecutive correct responses to relevant stimuli. The average learning threshold was broadly distributed across independent simulations (Fig. 3.3, horizontal lines). This causes a lot of variability in learning performance across simulations, however the average learning curves appear to improve gradually over time (Fig. 3.3). Note that a variable learning threshold is an unavoidable consequence of the nature of the task: if the agent is randomly set to take very sparse decisions, feedback is rare and so are synaptic changes. However, once the agent reaches the learning threshold for relevant stimuli, it quickly reaches the maximal performance for those stimuli (Fig. 3.3).

3.3 Robustness of the model

In this section I consider the robustness of the model to two relevant factors, specifically different input encoding strategies and different ratios of relevant to non-relevant stimuli.

3.3.1 Different encoding strategies

The stimuli used in the main simulations of the previous section are only one way in which stimuli could be encoded as patterns of spike trains. Here I consider the performance of the model in two relevant cases, one resembling encoding by firing rates (see Fig. 3.4, *rate coding*) and one resembling encoding by precise spike timing (Fig. 3.4, *uniform rates*). Compared to the stimuli used so far (*non-uniform rates* in Fig. 3.4), these two new encoding methods can be seen as extreme cases. The details of how the new stimuli were produced were reported in Sec. 2.4. For concreteness, I used a 2-way decision task with 2 relevant stimuli and 6 non-relevant stimuli, with 100 decision neurons and 50 input spike trains per stimulus.

Fig. 3.4 shows that the model is successful also with the new encoding strategies and requires about the same learning time for both tasks (all simulations were obtained at parity of network parameters). The good learning performance with rate coding is rather impressive given that the learning rule was designed, in principle, for precisely timed spike patterns and not firing rate patterns.

It is instructive to compare the learning time with that of the main stimuli (*non-uniform rates* in Fig. 3.4). Recall that the main stimuli are comprised of 50 Poisson spike trains with different firing rates, with jitter in the spike times of the prototypes at each stimulus presentation. For this reason, different stimuli can be separated by two facts: 1) several of its spikes trains have different firing rates, and 2) the spike times are only jittered by ~ 10 ms. As a consequence, they represent an easier task to learn, and the learning time is faster.

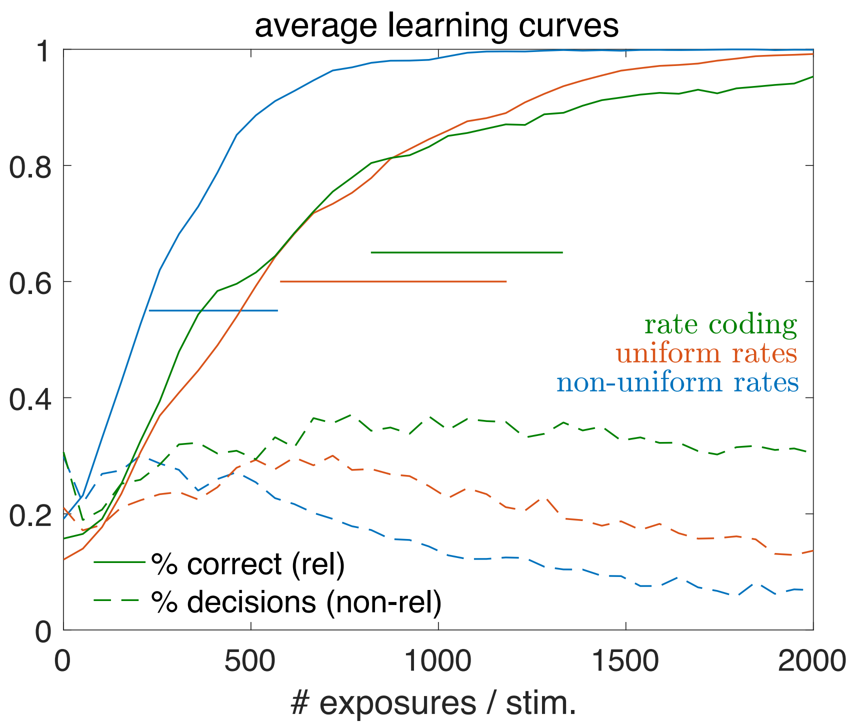


FIGURE 3.4: **Average learning curves for different input coding.** The network model had to solve a 2-choice task while learning to identify 2 relevant stimuli out of 8 stimuli, with 100 decision neurons and 50 input spike trains. The horizontal segments represents the interquartile range of the learning thresholds, see e.g. Fig. 3.3. All other conventions as in Fig. 3.3.

I also tested the model on real cortical data, i.e., spike trains simultaneously recorded from behaving rats presented with taste stimuli (Mazzucato et al., 2015; Samuelsen et al., 2012). The results for this case are deferred to the next chapter. I only anticipate here that segmentation of cortical data is faster and easier even for lower-dimensional input patterns. This is presumably due to the presence of larger structure in the data compared to the surrogate Poisson spike trains used here.

3.3.2 Different ratios of non-relevant to relevant stimuli

In the simulations shown so far, there was only one relevant stimulus per decision. Here I show that the model works also in the case of multiple stimuli demanding the same decisions, and/or for different ratios of relevant to non-relevant stimuli.

As shown in Fig. 3.5, the model can learn to identify multiple relevant stimuli per decision, and can do so among a larger pool of non-relevant stimuli. As expected, learning performance slows down as the number of non-relevant stimuli increases. However, for enough stimulus presentations (not shown here), nearly optimal performance can be achieved thanks to the homeostatic mechanism. The role for the latter will be clarified in the next section (subsec. 3.4.2), and a further demonstration of successful performance with an extensive number of stimuli (96 stimuli of which 48 relevant, thus a 1 : 1 ratio) is shown in Appendix B.

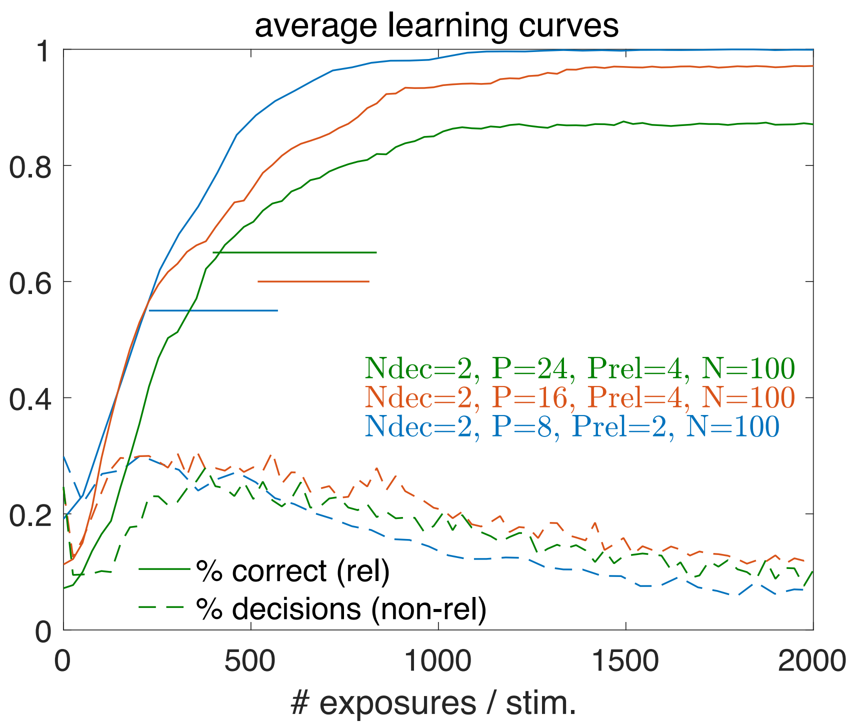


FIGURE 3.5: **Average learning curves for different ratios of relevant to non-relevant stimuli.** The network model had to solve a 2-choice task while learning to identify 2 or 4 relevant stimuli out of 8 stimuli, 16 or 24 stimuli, with 100 decision neurons and 50 input spike trains. The blue curve is the same as the blue curve in Fig. 3.5. Same key as in Fig. 3.3.

3.4 Analysis of the model

The previous sections have demonstrated that the model is successful at solving the stimulus-segmentation problem. One should always wonder though if all the ingredients of the model are strictly necessary to its success. Among those ingredients, the individualized reward signal and the homeostatic spike probability mechanism. The benefits of the individualized reward has been proved by Urbanczik and Senn (2009) in a simpler task (see rationale in Sec. 2.3.2). The homeostatic mechanism is instead a novel ingredient and its use should be fully justified.

In the next section I demonstrate the benefits of an individualized reward also for solving the temporal segmentation task, before moving on to the reasons for having a homeostatic mechanism.

3.4.1 The role of individualized reward: scaling up learning time with population size

Notice from Fig. 3.3 that the learning curves across tasks show a large overlap. This means that, despite some tasks are more difficult because they involve more decisions and/or more stimuli, the learning times are comparable. The reason is that the population size (the number of neurons in each decision population, N) has been each time increased to deal with more difficult tasks. The fact that, as N increases,

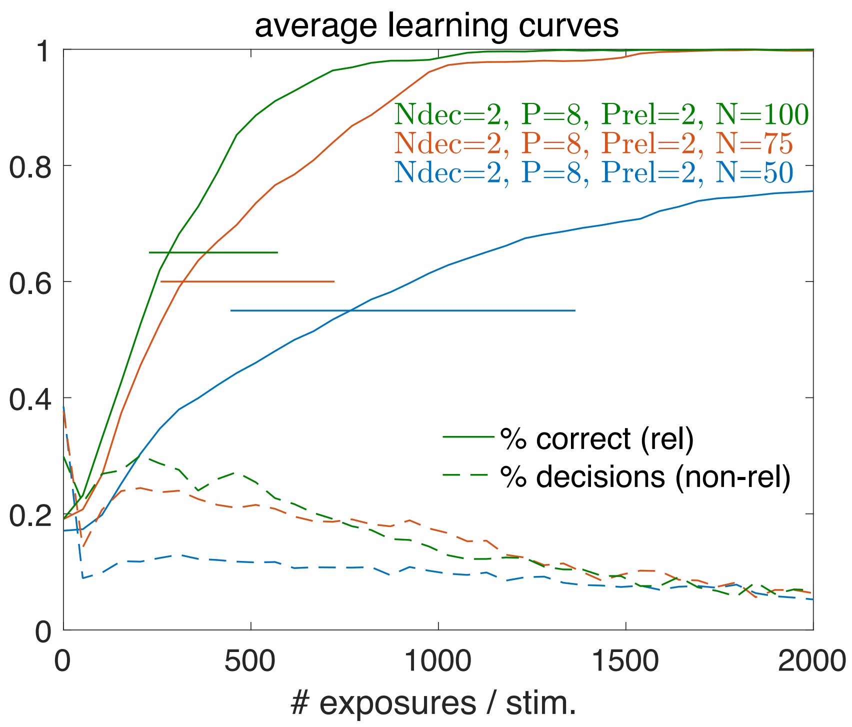


FIGURE 3.6: **Learning speeds up with population size.** Average learning curves across 30 simulations for a 2-choice task with 2 relevant and 6 non-relevant stimuli and variable decision population size *N* (reported in the legend). Same key as in Fig. 3.3.

learning speed is kept constant despite a more challenging task is the signature that a larger *N* on its own (i.e., at parity of task) will increase learning speed.

To prove this directly I compared simulations of the same task with different numbers of decision neurons. As shown in Fig. 3.6, at parity of task (a 2-choice task with 2 relevant and 6 non-relevant stimuli) learning speeds up with the number of decision neurons. This is a direct demonstration that learning speed increases with population size, and suggests that the model scales up well with the number of decisions and/or input stimuli as long as enough decision neurons are included. As proved by Urbanczik and Senn (2009), this is a consequence of having used individualized reward.

3.4.2 The need for a homeostatic mechanism

I found that, in the absence of a homeostatic mechanism, the model sometimes fails to learn the full task. The most pernicious situation is when one of the decision populations learns to quit its own activity, as shown in Fig. 3.7. The figure shows the behavior of a network with 2 decision populations after training for a segmentation task with 2 decisions, 2 relevant stimuli and 6 non-relevant ones. One can see that while one of the two populations (green) has successfully learned to respond to its relevant stimulus, the other (blue) has gone silent. This is probably the consequence of having taken a critical number of wrong decisions that has led to a decreased activity for that population. Eventually, things go so bad that the population activity is very far from having a chance to ever making a decision again.

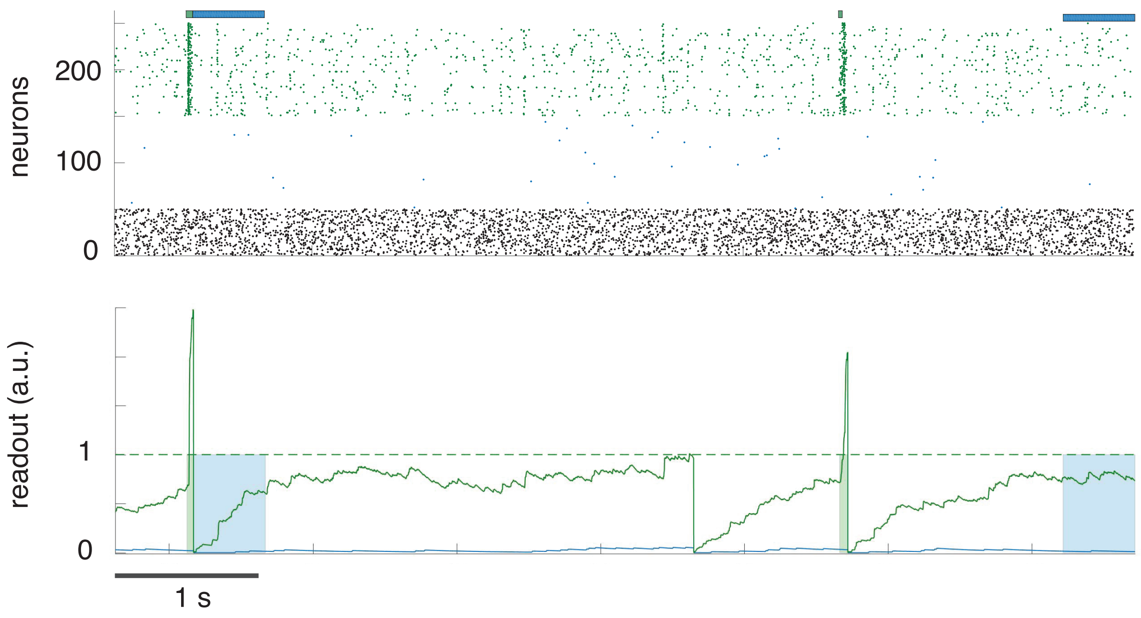


FIGURE 3.7: **Example of failure in the absence of a homeostatic mechanism**. The task is the same as in Fig. 3.6, i.e. a 2-choice task with 100 decision neurons. Same key as Fig. 3.1, except that now two decision populations are shown (in green and blue, respectively). Due to training, the green population has learned to respond to its corresponding relevant stimulus, while the blue population has gone silent. This is visible both from the raster (top panel) and from the population readout (bottom panel).

The presence of the homeostatic mechanism that changes the spike probability can prevent this phenomenon by increasing the probability of firing a spike when the population activity level is low. As explained in Sec. 2.2.1, the spike probability will adapt to the recent average activity level so as to keep the population firing rate close to the decision threshold. It may take many presentations before this occurs, but eventually this mechanism will prevent the situation depicted in Fig. 3.7.

The aggregate performance over many simulations reveals the difference between the models with and without the homeostatic spike probability. This is shown in Fig. 3.8 for the detection task of Fig. 3.1. In Fig. 3.8 the average learning curve of a non-homeostatic model (red and blue curves) is compared to the learning curves of the homeostatic model (green and purple). For each simulation I used two initial values for the parameter k appearing in the spike probability, Eq. 2.6,

$$\phi(u)dt = ke^{\beta u}dt. \tag{3.1}$$

The values are reported in parenthesis in the legend, and are 0.12 and 0.15. Although a larger initial value of k insures more frequent decisions and, thus, faster learning (compare purple and green curves), the non-homeostatic model (where k remains fixed to its initial value) converges in both cases to only about 48% of correct decisions with relevant stimuli, due to the phenomenon depicted in Fig. 3.7. The problem of decision populations going silent has been observed in all cases, regardless of the number of decisions or the number of stimuli.

The reason for the occasional failure of the non-homeostatic model is related to its inability to escape from local maxima of the average reward. When a population goes silent, it never incurs the cost of

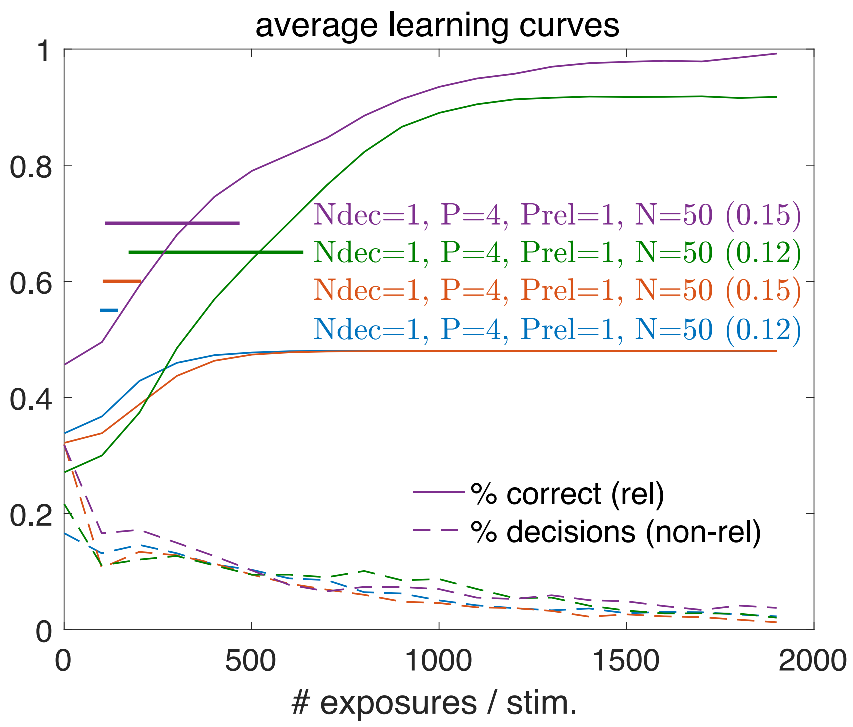
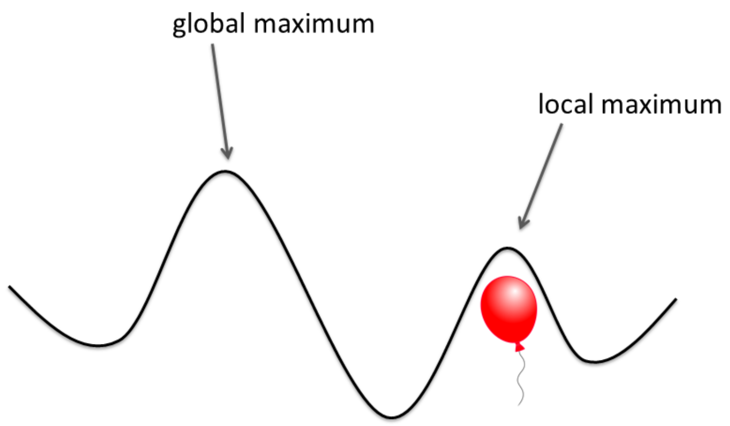


FIGURE 3.8: **Comparison of performance with and without a homeostatic mechanism.** Average learning curves across 30 simulations of the models with and without homeostatic mechanism, for two initial values of the parameter $k = 0.12, 0.15$ affecting the spike probability in Eq. 3.1. Same key as in Fig. 3.3 with, in addition, the initial value of k reported in parenthesis in the legend.

making a decision, at the price of not receiving reward for correct decisions. Especially in the presence of many incorrect decisions (the common situation during initial stages of learning), the overall reward rate when a population goes silent is not much smaller than the optimal reward rate obtained for perfect performance. In other words, the network has reached a *local maximum* of the average reward. This phenomenon can be intuitively understood with the illustration of Fig. 3.9.

In the figure an air ballon is flying upwards towards the sky, but gets stuck in a local maximum of a fictitious ceiling. The fictitious ceiling here represents the average reward function, which has a global maximum (achieved in the case of perfect performance) as well as a local maximum. Average reward functions for generic reinforcement learning problems have many such maxima. The exploration induced by the stochasticity of the model (in our case, by the probabilistic spiking mechanism) acts as a source of disturbance that will push the air ballon around. For strong enough pushes, the ballon can escape the local maximum and reach the global one. A homeostatic mechanism makes sure that this eventually will always happen: by increasing the spike probability when the population activity is too low (corresponding to the ballon being in a local maximum), the population eventually will start crossing the decision threshold again. This insures enough exploration to dislodge the ballon from being stuck in a local maximum. If necessary, this process will also occur for the other populations, and the global maximum of the average reward can eventually be reached.



$$\Delta w_{ij} = \eta \partial_w \langle R \rangle$$

FIGURE 3.9: **The problem of local maxima.** The same way a balloon in a room will reach the higher point in the ceiling, the network dynamics – directed by the learning rule Eq. 2.19 – push the system toward a maximum for the average reward, by following a gradient ascent method. The non-homeostatic network model is able to maximize the average reward reaching a local maximum (learning to respond correctly to some relevant stimuli indicates that the reward has been locally maximized). The network state is pushed toward a maximum, but it can get stuck in a local maximum as depicted here for the air balloon. The homeostatic mechanism insures enough perturbation of the ballon so that it can escape from the local maximum and eventually reach the global maximum of the average reward.

3.4.3 Alternative homeostatic mechanism: the variable decision threshold

A variable spike probability is not the only way to implement a homeostatic mechanism. Here I show an alternative model based on an adaptive decision threshold. The model is identical to the main model presented in Ch. 2 expect that the spike probability is fixed while the decision threshold Θ_D *of each decision population* adapts to the recent population activity. When the activity is low, the decision threshold decreases so that the population activity can still have a chance to make a decision. As with the spike probability, the crucial ingredient of this homeostatic mechanism is that it gets updated only once in a while, to allow for a decent estimate of the recent population activity. This model is perhaps less preferable to the variable spike probability model because the latter is somewhat more biologically plausible, being a mechanism occurring at the level of the single neurons that can read out their own activity level (e.g., through intracellular calcium levels), rather than at the level of a decision threshold for an entire population activity. Moreover, there is the problem of having to impose a minimal threshold to prevent Θ_D to decrease to zero values, which would make the model singular. However, as shown in Fig. 3.10, the variable-threshold model is equally successful at temporal segmentation, confirming our general understanding of the role of a slow homeostatic mechanism for escaping local maxima of the average reward. In Appendix B I further present examples of performance with an extensive number of stimuli, to demonstrate the ability of this homeostatic model to solve very difficult temporal segmentation tasks. In the next chapter, the same model will be tested on real cortical spike trains.

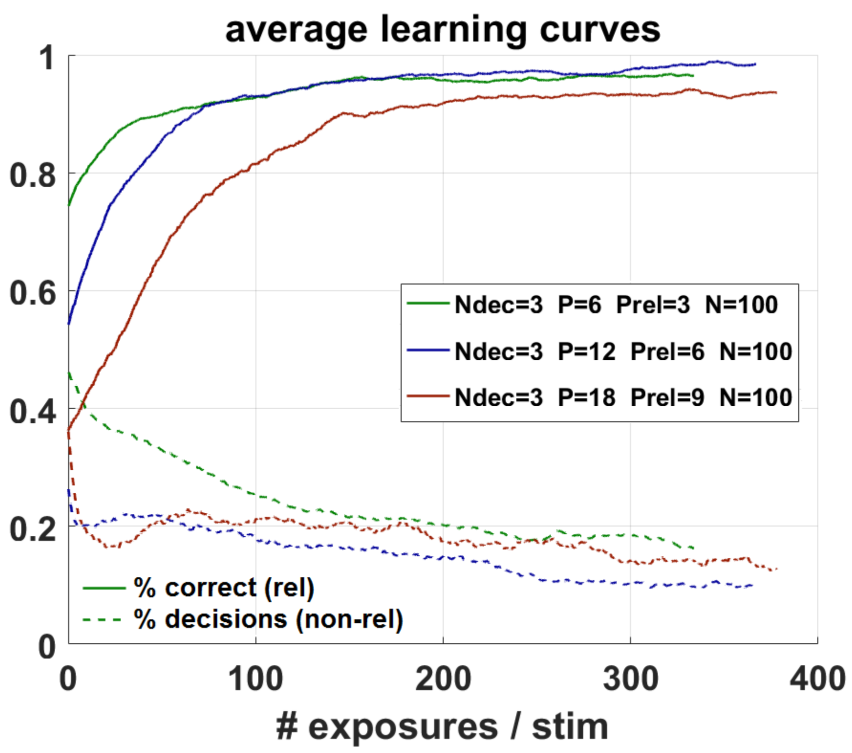


FIGURE 3.10: **Performance of the variable-threshold model.** Average learning curves across 10 simulations of a 3-choice task with 100 decision neurons for different number of input stimuli modeled as firing rate patterns (see Sec. 2.4). Because *N* is constant across tasks, learning slows down with increasing task difficulty (more decisions and/or stimuli). Same key as for Fig. 3.3.

Chapter 4

Application to real data

In this chapter I test the model on real cortical data to test the model's ability to detect and correctly decode natural sensory stimuli. The aim of this exercise is twofold: on the one hand, real data represent yet another coding scheme for the input stimuli, the most biologically plausible of all, so this will confirm the robustness of the model to different coding schemes addressed in Sec. 3.3.1. This will also prove the model's ability to detect and correctly respond to real data. On the other hand, proving that the model is able to identify cortical spike patterns observed in response to taste stimuli will also prove in a novel way that there is enough information about the stimuli in the observed patterns of neural activity. This can be (and has been) shown with statistical methods by several authors (e.g., Jones et al. (2007); Samuelsen et al. (2012)), but it would be worth to show that the information can be extracted by a neural system.

For concreteness, the model will be endowed with the homeostatic threshold discussed in Sec. 3.4.3, but this time probed with an ensemble of sensory neural activity. The dataset used here was recorded from the gustatory cortex of rats (Samuelsen et al., 2012) during a simple experiment wherein an alert rat receives one out of 4 tastes at random intervals with mean duration of about 40 seconds. In each trial, spike trains from 3 to 9 simultaneously recorded neurons were available depending on the recording session; an example is shown in Fig. 4.1.

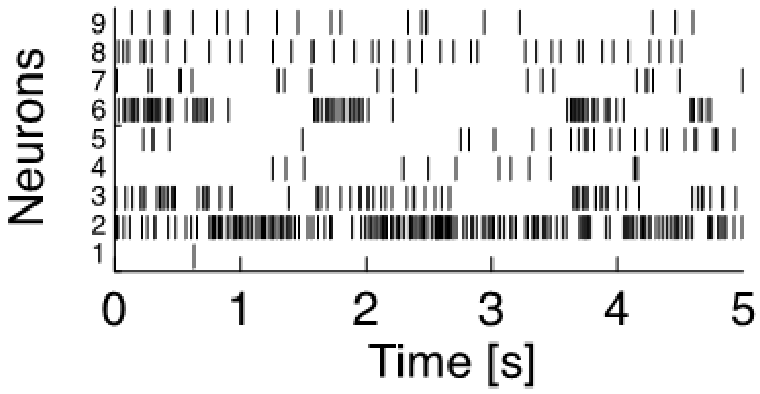


FIGURE 4.1: **A pattern of 9 simultaneously recorded spike trains** from the gustatory cortex of an alert rat. Data from Samuelsen et al. (2012).

Note the similarity between spike trains in Fig. 4.1 with the artificial spike patterns in Fig. 2.3. Since the network can learn to segment artificial Poisson-like stimuli, and since it is known that cortical neuronal spikes (particularly in gustatory cortex) are reasonably well approximated by a Poisson process (Shadlen and Newsome, 1998; Sakai et al., 2006; Stapleton et al., 2006), the network should be able to perform the segmentation and decision tasks using these natural inputs. I show first that this is the case, and then in Sec. 4.2 I will also investigate the ability of the network to generalize to novel stimuli after learning a set of stimuli, e.g. the network will be tested on a sucrose stimulus never seen before after training has been performed using sucrose stimuli from different trials. This procedure is called 'cross-validation' and is common in machine learning applications, where the main aim of a device is being able to generalize knowledge to similar but new stimuli and not just those used for training.

4.1 Stimuli and learning

The network will be trained as described in Ch. 2 to identify relevant stimuli by reinforcement of correct decisions. Spike patterns such as that shown in Fig. 4.1 will be used as relevant and non-relevant stimuli. Specifically, those patterns recorded in the presence of a taste stimulus (from 100 to 600 ms after taste delivery) will be considered relevant, and will be rewarded in case of a correct decision. Those patterns recorded during the 'ongoing' or 'spontaneous' activity (Mazzucato et al., 2015) in between taste deliveries will be used as non-relevant patterns (specifically, recording taken between -800 and -300 ms before stimulus delivery). Patterns recorded in response to the same stimulus (say, sucrose) will always demand the same response; patterns in response to a different stimulus (say, quinine) will demand a different response. This way, the natural variability of neural data in response to the same stimulus acts as the mechanism that adds jitter to the prototypical stimuli used for surrogate stimuli (see Sec. 2.4). Of course, the variability here is limited given that there were only 8 trials per stimulus: the same stimuli will have to be presented again and again to the network.

It turns out that with a limited number of simultaneously recorded neurons (a maximum of 9), the task is hard. The problem is the limited number of spike trains. Recall that with surrogate stimuli we used at least 50 input spike trains. Fig. 4.2 shows that the network can learn to segment three tastes correctly with only 9 input spike trains, however this was aided by using always the same stimulus for each taste (i.e., there was no variability across presentations of the same stimulus).

To handle the more general case I built higher-dimensional input patterns 'stacking' together simultaneously recorded spike trains (in response to the same taste) *across different sessions*. The procedure is illustrated in Fig. 4.3 and an example of 4 stimuli so built, together with the network's ability to learn them, is presented in Fig. 4.4. In this case, the same stimulus per taste was presented each time, i.e., no variability across presentations.

A case in which the same taste was represented by multiple presentations of 4 different patterns (all recorded in response to the same taste, but in 4 different trials across sessions), is shown in Fig. 4.5. The task was to learn to take one decision in response to every instance of sucrose ('S') and a different decision in response to every instance of citric acid ('C'). Similarly, 4 instances of spontaneous activity represented a non-relevant stimulus. Also in this case, the network could solve the task in reasonable

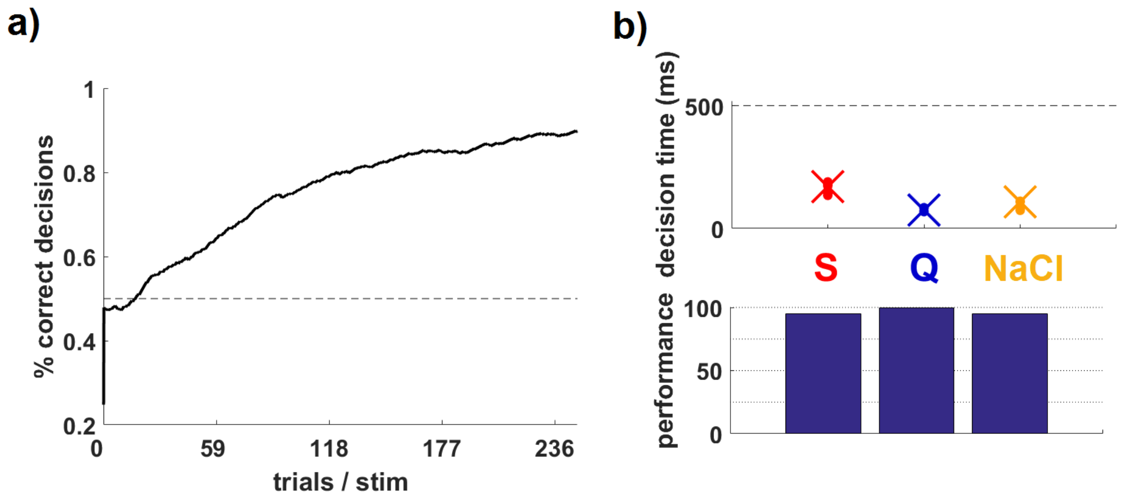


FIGURE 4.2: **Learning from recording data from 1 session using only the same data.** The network model learning to identify 3 tastant stimuli. Input data were neuronal recordings from gustatory cortex from a single trial from a single session (i.e., the same stimulus is presented repeatedly to the network). There are three possible decisions, 'S','Q', and 'NaCl' (which stand for sucrose, quinine and sodium chloride respectively). **a)** Learning curves, i.e. percent correct as a function of stimulus presentation number. **b)** Bottom: Network's performance for each stimulus (bars). The ability to correctly identify reward-predicting stimuli following training is 97%. Each color encodes for a different required decision. Top: sample (dots) and average (crosses) decision times per stimulus after learning. Though this simulation shows a successful learning, this is not generic due to the small number of input spike trains.

time ($\sim$ 500 trials). Aggregate learning curves across 10 simulations of a more difficult task with 6 instances per taste (see Fig. 4.6) demonstrate that this finding is generic.

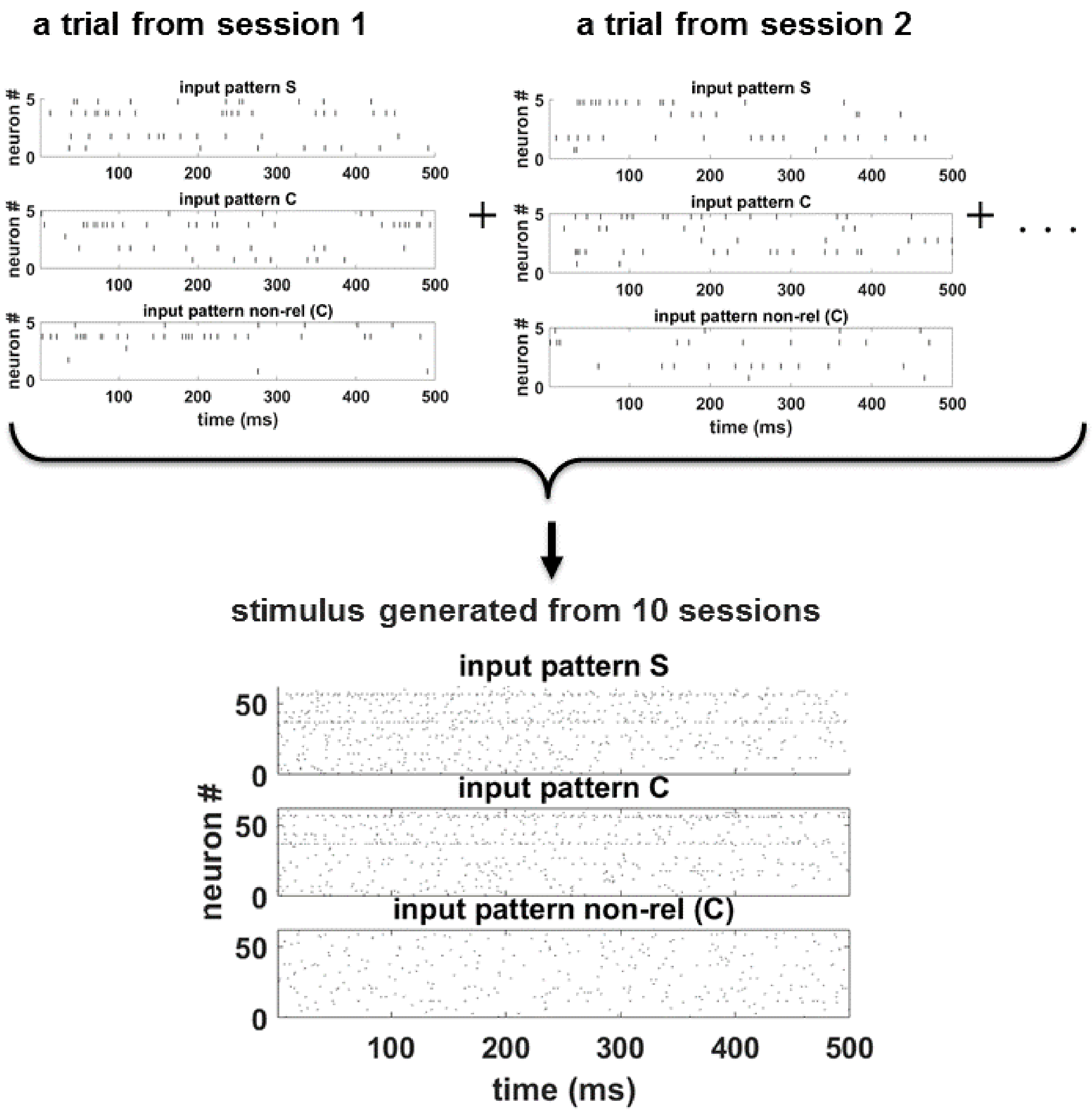


FIGURE 4.3: **Building stimuli across sessions.** An input stimulus is built from single trials in each of 10 sessions. Top: recordings from single sessions, one trial per session and stimulus. Neurons with a firing rate smaller than 2 Hz were removed. 'Input pattern S (C)' represents a spike train segment after the delivery of sucrose (citric acid). Bottom: recordings from single sessions are concatenated using as many sessions as needed to reach a chosen number of input spike trains (50 in this case).

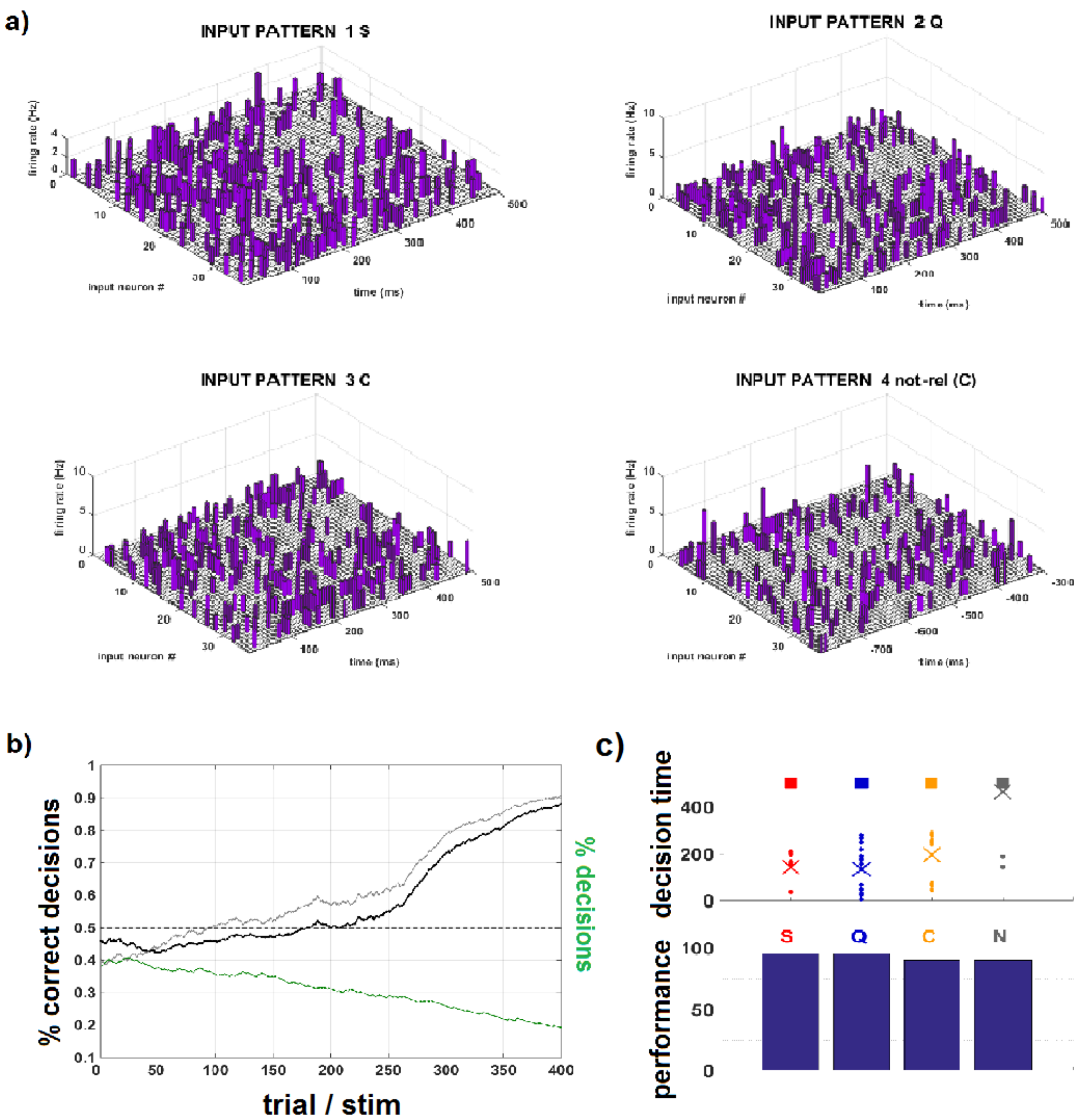


FIGURE 4.4: **Learning to identify stimuli built across sessions. a)** 4 input stimuli were built across 14 sessions as shown in Fig. 4.3, and are shown here as 2-dimensional histograms of firing rates across neurons and across time. Three of these stimuli were relevant (‘S’,‘Q’,‘C’) and one was not relevant (‘N’). There are three possible decisions, ‘S’,‘Q’, and ‘C’ (which stand for sucrose, quinine and citric acid respectively). **b)** Learning curves of the network during training, same key as in Fig. 3.3, with superimposed global performance across all stimuli in grey. **c)** Network’s performance for each stimulus (bars) and decision times after learning. The ability to correctly identify reward-predicting stimuli following training is 95%, the ability to ignore the non-relevant stimulus is 90%. Each color encodes for a different stimulus hence a different decision.

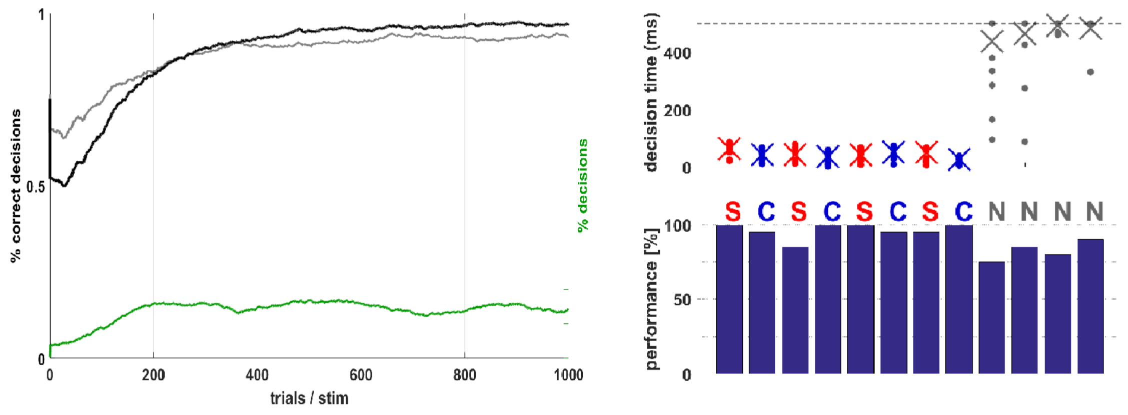


FIGURE 4.5: **Learning to identify cortical stimuli with trial-to-trial variability.** A network model learning to identify 2 tastant stimuli and ignore 1 non-relevant segment represented by repeated presentation of 4 instances obtained from 4 different trials across 15 sessions. **a)** Learning curves, same notation as in Fig. 4.4. **b)** Network's performance for each stimulus (bars) and decision times after learning. Each instance for each stimulus ('S', 'C', or 'N'=non-relevant) is shown separately.

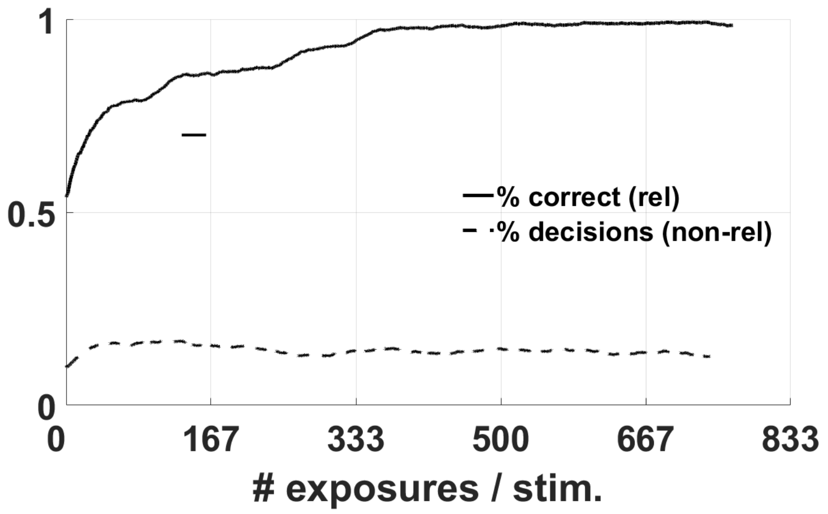


FIGURE 4.6: **Average learning curve with real data.** Learning curves averaged across 10 simulations. The task was a 2-choice decision between two tastes represented by 6 patterns from 6 different trials for each taste, and 1 non-relevant stimulus (also represented by multiple representations of 6 patterns). Each stimulus comprised 36 input spike trains build across sessions as shown in Fig. 4.3. The network had 50 decision neurons per population.

4.2 Generalization to novel stimuli

Is the network model able to generalize previous learning to inputs never seen before? To answer this question I trained the data using 4 trials out of the 7 available for each taste in each session. The stimuli were built collating spike trains across 19 recording sessions, as explained in Fig. 4.3. Training used only input patterns from those 4 trials and is shown in Fig. 4.7, top panels. After training, the network was tested on the patterns from one of the left-over trials not used for training. As shown in Fig. 4.7, bottom panels, the performance of the network was instantly high with the new stimuli, demonstrating a robust ability to generalize across stimuli obtained in response to the same taste. This is possible because the data contain enough information on the stimuli that had elicited them.

This result was confirmed across 7 independent problems where training was done on 6 trials and testing on a new trial not used for training, and the performance is summarized in Table 4.1.

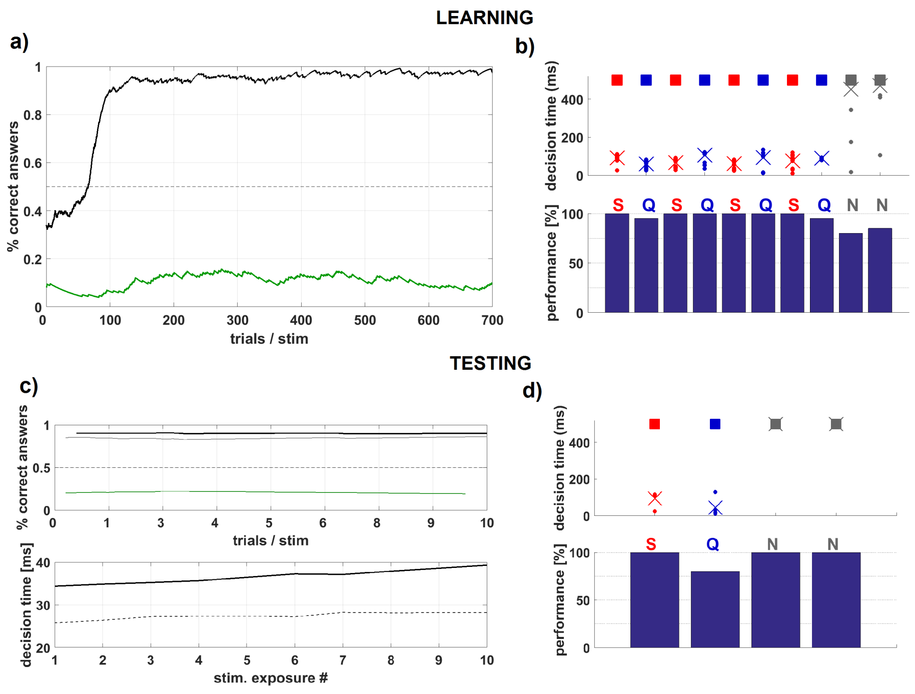


FIGURE 4.7: **Generalization.** Ability of the network to generalize past learning to novel stimuli. **Top,** training phase, using 4 instances (trials) for each stimulus. **Bottom,** testing performance on inputs never used for training. Average performance during testing was calculated as the number of correct decisions during 10 presentations per stimulus.

Tastant stimulus	Average performance on relevant novel stimuli	Average performance on non-relevant novel stimuli
Sucrose	100	95.0
Quinine	100	96.4
Citric Acid	100	95.5
Sodium Chloride	100	99.8

TABLE 4.1: **Mean generalization results.** Ability of the network to generalize past learning to novel stimuli. Testing performance on novel input x after learning has been performed using 6 inputs from 6 different trials (training the network for 100 presentations per stimulus). Average performance during testing was calculated as the number of correct decisions during 30 presentations per stimulus. The averaged performance was calculated as the average of the performances on all novel stimuli (average over 7 simulations).

Chapter 5

Discussion and conclusions

Learning to abstract relevant information from the environment is a crucial component of decision making; yet, current models typically assume that the relevant stimuli are known to the decision maker, which also knows when it is being presented with one. Here, I have put forward a spiking network model able to detect stimuli from the environment based on their behavioral relevance. Since the stimuli are presented in sequence in a continuous stream, with unknown starting and ending points, the task is akin to temporal stimulus segmentation, i.e., the task of discovering boundaries between successive stimuli.

In the model proposed here, the input segments are represented by spatio-temporal patterns of spike trains and are processed by a network of spiking neurons capable of online, spike-based learning. The network learns to segment the input stream by taking appropriate actions at the right time, using a spike-based synaptic plasticity rule that approximates gradient ascent on the average reward, i.e., by reinforcement learning, and makes use only of information locally available at each synapse. Importantly, the relevant segments are constructed so as to be behaviorally meaningful but not statistically different from irrelevant stimuli and noise, and therefore their boundaries are not detectable by standard pre-processing techniques. The network model developed here appears to be the first example of a biologically plausible, spiking-neuron based model of reinforcement learning capable of performing tasks as taxing and relevant as stimulus segmentation and multi-choice decisions.

5.1 Comparison with existing models

Traditional algorithms for stimulus segmentation that learn based just on the unsegmented input stream are unlikely to succeed in this task, especially if the relevant segments do not exhibit features that are detectable by some standard pre-processing strategy. As a consequence, existing models typically endow a learning agent with prior knowledge of what is relevant and focus on the problem of relating each segment with the outcome they predict (as e.g. in a categorization task; see, e.g., Sutton and Barto (1998) and Wang (2008) for reviews). Such models, of widespread use in computational neuroscience and machine learning, are unable to form or modify their own relevant segments, preventing the development of truly autonomous learning and decision-making devices.

In practical applications, segmentation tasks are typically solved by unsupervised methods such as Hidden Markov Models (HMMs) (Rabiner, 1989), which have recently been applied to the segmentation of spontaneously ongoing patterns of cortical data similar to the stimuli used here (Mazzucato et al., 2015). However, HMMs require a-priori knowledge of the structure of the relevant stimuli (e.g., Poisson-HMM rather than Gaussian-HMM); they also require knowledge of the total number of relevant stimuli, are not based on online algorithms, and lack biological plausibility (although see Kappel et al. (2014)). In contrast, the spiking network model studied here learns online, does not require a-priori knowledge of the relevant stimuli or their presentation times, it allows for direct comparison with neurobiological data and thus could help uncover potential correlates of decision confidence and other aspects of decision making.

Another hallmark of this study is the use of 'information-controlled' tasks, wherein subjects can respond whenever they feel confident (Zhang et al., 2009). The model introduced here differs from the class of neural-circuit models of decision making reviewed in Wang (2008), which require a neural population encoding each stimulus, and a priori knowledge of the relevant stimuli, and when they start and end. Following an approach more similar to ours, the 'tempotron' of Gütig and Sompolinsky (2006) (see Sec 1.4.3), can learn to separate spike patterns into two classes, which could be interpreted as 'relevant' vs. 'non-relevant'. However, the tempotron needs to know when stimuli start and end, and is given feedback for non-responses to relevant stimuli, which helps their identification. A modification of the tempotron able to perform 'aggregate-label' learning is closest to our problem and approach (Gütig, 2016). In aggregate-label learning, a neuron is trained to respond to different stimuli with a different number of spikes to each, with no knowledge a-priori of which stimuli are relevant and when they are being presented. However, there are important differences: the task is divided in trials and feedback is always given at the end of each trial. Moreover, this feedback (and the learning rule) is supervised in the total number of spikes required in the current trial. This number is externally calculated and fed back to the neuron for comparison with the actual number of emitted spikes during that trial. It seems implausible that somewhere in the brain knowledge of the exact number of spikes required to segment input stimuli during a specified period of time is held.

Instead, our model seems a good candidate for a biologically plausible theory of stimulus segmentation and decision making. Simulations show successful and robust results under different input coding scenarios and under modification of crucial parameters including the number of stimuli, the number of decisions, and the number of decision neurons. Based on these results, especially the ability of the network to handle stimuli coded in different ways (see Fig. 3.4 and Fig. 3.10), including real cortical patterns of spike trains as shown in Ch. 4, there are reasons to believe that this network model can provide a successful theory for stimulus segmentation and decision making.

5.2 Learning by reinforcement

My model uses a reinforcement learning algorithm. Reinforcement learning is one of the most active research areas in artificial intelligence. Reinforcement learning mimics how biological and artificial decision makers learn over time by trial and error so as to maximize their expected future reward (Sutton

& Barto, 1998). A large variety of reinforcement learning algorithms have been successfully applied to complex problems such as board games (Tesauro, 1995) and motor control tasks, either simulated (Sutton, 1996), or real (Morimoto and Doya, 1998). Reinforcement learning is different from supervised learning, the kind of learning studied in most current research in machine learning, statistical pattern recognition, and artificial neural networks. Supervised learning is learning from examples provided by an external supervisor. This is an important kind of learning, but alone it is not adequate for learning from interaction. In interactive problems it is often impractical to obtain examples of desired behavior that are representative of all the situations in which the agent has to act. In uncharted territory – where one would expect learning to be most beneficial – an agent must be able to learn from its own experience. In the biological and psychological sciences, a vast literature uses the framework of reinforcement learning to interpret behavioral and neural data (Montague et al., 1996; Daw and Doya, 2006; Johnson et al., 2007; O'doherty et al., 2007; Doya, 2008; Rushworth and Behrens, 2008). Accordingly, the way reinforcement learning may be implemented in the brain has been subject to ongoing research. In particular, dopamine has been found to play an important role as a potential reward signal presumably representing the current reward prediction error (Dayan and Niv, 2008; Dayan and Daw, 2008; Lee et al., 2012; O'doherty, 2012). To that effect, Hebbian learning rules have been extended to three-factor learning rules in which Hebbian synaptic plasticity is modulated by a reward signal (Xie and Seung, 2004; Pfister et al., 2006; Baras and Meir, 2007; Florian, 2007). A three-factor synaptic plasticity rule has been found in cortico-striatal synapses, a major locus for *reinforcement learning* (Reynolds and Wickens, 2002).

5.3 The importance of slow homeostatic mechanisms to learning

Adaptive mechanisms such as those used here have been used before to tackle similar challenges (Dean et al., 2005; Kobayashi et al., 2009; Brette and Gerstner, 2005; Lo and Wang, 2006; Simen et al., 2006; Fontaine et al., 2014) and are supported by experimental evidence. Most analysis techniques for recordings of extracellular neural activity begin with spike detection to identify the times at which action potentials occurred from one or more neurons. Typically spike detection is performed using simple threshold detectors, assuming the individual neuron threshold fixed, however more recent efforts are taking into account the possibility of threshold changes (Kim and McNames, 2007).

This work has demonstrated the need to endow a learning agent with a slow homeostatic mechanism that allows the population activities to stay close to the decision threshold. Without this mechanism, population activity could go silent, jeopardizing the normal functioning of the whole network (see Sec. 3.4.2). Many attempts to solve this problem via homeostatic synaptic plasticity failed for the reasons mentioned in Sec. 1.5, i.e., the interference of homoeostatic plasticity with learning. The latter requires its own synaptic changes and additional mechanisms of homeostatic plasticity will compete with the requirements of the learning rule and degrade learning. This was observed in a variety of homeostatic synaptic scenarios (not shown in this thesis), and led us to consider intrinsic (i.e., cellular) homeostatic mechanisms (also reviewed in Sec. 1.5). This eventually led to the two successful mechanisms described in Sec. 3.4.2 and 3.4.3, i.e., adjusting the neurons' probability of firing or the populations' decision thresholds. Both mechanisms are motivated by the necessity to keep the populations' activities close to the decision threshold, but whereas one acts at the cellular level (i.e., for each neuron), the adaptive threshold mechanism assumes the existence of a biological threshold that can be modified homeostatically.

In both cases, however, the crucial ingredient of the mechanism is that it needs to be slow compared to the dynamics of synaptic changes, not to interfere with the learning process itself. Understanding this issue was a turning point of this work and this result alone is an important finding of this thesis which generalizes widely to many learning problems.

5.4 Applications to real data

The network model was able to learn to identify the stimuli and correctly perform the decision task using the natural stimuli recorded from the gustatory cortex of rats. The segmentation of cortical data is faster and easier even for lower-dimensional input patterns. This is presumably due to the presence of a larger structure in the data compared to the surrogate spike trains. This result also emphasizes the mechanistic nature of the model: unlike descriptive models of neural activity, this model accounts for a possible mechanism the brain might use to solve problems involving temporal segmentation and decision making.

It is also worth mentioning again that the problem of relevant stimulus segmentation cannot be solved by supervised or unsupervised learning. A supervised method would make no sense as, by definition, it works by providing the correct answer after each decision, contrary to the assumption that the network does not know which stimuli are relevant and when they are being presented. Unsupervised methods such as Hidden Markov Models (HMMs) (Rabiner, 1989) have recently been applied to the segmentation of spontaneously ongoing patterns of cortical data similar to the stimuli used here (Mazzucato et al., 2015). However, HMMs require a-priori knowledge of the structure of the relevant stimuli (e.g., Poisson-HMM rather than Gaussian-HMM); they also require knowledge of the total number of relevant stimuli, are not based on online algorithms, and lack biological plausibility (although see Kappel et al. (2014)). In contrast, the spiking network model studied here learns online, does not require a-priori knowledge of the relevant stimuli or their presentation times, it allows for direct comparison with neurobiological data and thus could help uncover potential correlates of decision confidence and other aspects of decision making. In fact, the successful approach proposed here as a biologically plausible model of learning and classification of real data is also very suitable to be used as a novel tool for data analysis.

5.5 Issues

Although this model is successful and very promising, it has of course its shortcomings.

First, the number of possible decisions is known a priori by the agent, while a real decision maker should have the ability to discover how many decisions are possible in the environment and uncover new actions. The possible mechanism to achieve this goal is to give the network the ability to modify the partition of the decision neurons in different populations as new actions are discovered. The network might try to take a new action ('exploration' of the environment) every now and then in the presence of a non-relevant stimulus, and eventually a new subpopulation of neurons would emerge that will take decisions only in the presence of particular stimuli. This would represent the neural correlate of a new action that was not present in the original partition of the decision neurons in separate populations.

Secondly, although the information of the start and ending of the stimulus is never given to the network, in order for the model to succeed, all time parameters (e.g. τ_P in the low-pass filter of the decision population readouts, Eq. 2.2) has to be of the same order of the stimulus durations (here $\sim 500ms$). Without matching the duration of the stimuli it is difficult for the population signal to do its job correctly. This critique also applies to the estimation of the individualized reward signal (see Eq. 2.17 and Eq. 2.18), which requires a correct estimation of whether or not the neuron was active during the stimulus presentation. For large stimulus durations this would not be a problem and I anticipate that it becomes problematic for short stimuli prior to learning. After learning the decision times are very short (order of 50 ms, see e.g. Fig. 3.1B), and again those time constants are normally large enough for the decision process after training. However, this would be an issue during training.

A possible solution to this problem is to include the ability for different decision neurons to process different time scales. If all the time scales are represented in a large fraction of neurons, those with time constants matching the duration of a given stimulus should eventually learn to identify that stimulus, whereas the other neurons will only experience the decision process as noisy and contribute randomly to those stimuli. This solution, although possible in principle, requires a large number of population neurons so that all relevant time scales are represented, and would probably contribute a lot of noise during training, slowing down learning. Nevertheless, there is experimental evidence that multiple time scales exist and are represented in cortex (Bernacchia et al., 2011).

Finally, there is the issue of the network architecture. The model proposed here is a simple feedforward network with no feedback connections among its decision neurons. Although this was a convenient (and sufficient) starting point for solving the temporal segmentation problem, it is known that cortical networks contain large amount of recurrent connections in addition to feedforward inputs. Since the introduction of recurrent connections in the decision populations may interfere with the estimation of the individualized reward, introducing this element of realism in the model would probably require a novel way of estimating the feedback signals.

5.6 Possible extensions

This work can be extended in a number of directions. One could consider non-stationary stimuli; for example, a visual segmentation task wherein a sequence of images slowly appear and disappear on top of a noisy background, and the task of the agent is to identify the images that are action-relevant. Different subsets of sensory neurons would code for different stimuli, while the rest represent streaming noise.

Given its success in segmenting real data, this model could be extended into a device to decode sensory activity (in the context of no prior information about the relevance of the stimuli), and should be contrasted to machine learning devices that can learn to classify the stimuli by supervised learning (such as maximum likelihood or neural network decoding of the input spike patterns – see e.g. MacKay (2003)). In this application the hypothesis emerges that, after learning, the stimuli could be decoded from the patterns of activation of the decision neurons. That is, the decision neuron's activity could contain information not only on the correct decision in response to the stimulus, but also on the identity of the stimuli themselves. Extracting this information could be achieved by measuring the 'distance' between the spatio-temporal patterns of spike trains elicited by the stimuli in the decision neurons. Examples of distance range for a simple Euclidean distance (e.g., (Jezzini et al., 2013)) to various other forms of spike distances existing in the literature (Victor, 2005; Victor and Purpura, 1996, 1997).

Finally, as mentioned in Sec. 5.5, the model could be extended to discover its own actions and not just its own relevant stimuli; could include a range of time scales to deal with stimuli of different durations; and could include recurrent connections among its decision neurons to increase both the power and the biological plausibility of the network. So, despite being a successful first take at a very challenging problem, there are plenty of directions in which to develop and refine the model.

Appendix A

Reinforcement learning as gradient ascent in the average reward

Here I present a derivation of the reinforcement learning rule posited by Eqs. 2.13–2.15 for the spike response model introduced in Chap. 2. The learning rule is derived by using the method of gradient ascent in the average reward, which by definition is accomplished by having the change in synaptic weights follow the gradient of the average reward,

$$\frac{dw}{dt} = \eta \partial_w \langle R \rangle, \tag{A.1}$$

where η is the learning rate and I have introduced the notation $\partial_w f \doteq \frac{\partial f}{\partial w}$. I also drop pre- and post-synaptic indices in the following and simply use w rather than e.g. w_i^ν to simplify the notation.

I start from an observation due to Williams (1992) that allows writing the gradient of the average reward $\partial_w \langle R \rangle$ in a more convenient way.

A.1 Williams' formula

Consider the probability $\mathscr{P}_\mathbf{w}(y|x)$ that the learning agent performs action y conditioned on the input x, and for doing so it receives a reward $R(y,x)$. The average reward over all possible input-output pairs (x,y) is

$$\langle R \rangle = \sum_{x,y} R(y,x) \mathscr{P}_w(y|x) \mathscr{P}(x), \tag{A.2}$$

from which I can calculate $\partial_w \langle R \rangle$ as follows:

$$\partial_w \langle R \rangle = \sum_{x,y} R(y,x) \partial_w \mathscr{P}_w(y|x) \mathscr{P}(x) = \tag{A.3}$$

$$= \sum_{x,y} R(y,x) \frac{\partial_w \mathscr{P}_w(y|x)}{\mathscr{P}_w(y|x)} \mathscr{P}_w(y|x) \mathscr{P}(x) = \tag{A.4}$$

$$= \sum_{x,y} R(y,x) \partial_w \ln P_w(y|x) \mathscr{P}_w(y|x) \mathscr{P}(x) = \tag{A.5}$$

$$= \sum_{x,y} R(y,x) \partial_w \ln \mathscr{P}_w(y|x) \cdot \mathscr{P}_w(y,x) = \tag{A.6}$$

$$= \langle R \partial_w \ln \mathscr{P}_w(y|x) \rangle, \tag{A.7}$$

Note that in Eq. A.7 a new ingredient has appeared, the log-likelihood $\ln P_w(y|x)$ of having y in response to x. Thus, terms of $\partial_w \ln \mathscr{P}_w(y|x)$, the learning rule Eq. A.1 can be written as:

$$\frac{dw}{dt} = \eta \langle R \partial_w \ln \mathscr{P}_w(y|x) \rangle. \tag{A.8}$$

This is William's formula (Williams, 1992). In this work I use the *online version* of this learning rule, i.e., I sample the product $R \cdot \partial_w \ln \mathscr{P}_w(y|x)$ after each reward delivery, rather than performing the average across many such samples prior to updating the synapse. This way I end up with the learning rule

$$\frac{dw}{dt} = \eta R \partial_w \ln \mathscr{P}_{\mathbf{w}}(y|x). \tag{A.9}$$

Recall that the quantity $\partial_w \ln \mathscr{P}_{\mathbf{w}}(y|x)$ is called the *eligibility trace* because, according to this learning rule, the synapse is 'eligible' for change only when the eligibility trace is different than zero.

Because of the demands inherent to a temporal segmentation task, I now proceed as explained in Sec. 2.3.1 and replace the eligibility trace with a low-pass filter of its time-derivative, $\frac{d}{dt} \partial_w \ln \mathscr{P}_w(y|x)$, obtaining the learning rule

$$\frac{dw}{dt} = \eta R E, \tag{A.10}$$

where E obeys Eq. 2.13, which in our current notation reads

$$\tau_M \dot{E} = -E + \frac{d}{dt} \partial_w \ln \mathscr{P}_{\mathbf{w}}(y|x). \tag{A.11}$$

All is left to do is to derive the expression Eq. 2.14 for the spike response model.

A.2 Specialization to the spike response model

Here I specialize the learning rule of the previous section to the Spike Response Model (SRM) with escape noise (Pfister et al., 2006; Urbanczik and Senn, 2009), introduced in Sec. 1.3.4 of the main text. The main goal is to derive an expression for the time-derivative of the eligibility trace, $\frac{d}{dt} \partial_w \ln \mathscr{P}_{\mathbf{w}}(y|x)$.

I first derive an expression for the eligibility trace. To do so, I first discretize time in tiny, equally spaced intervals $(t, t+dt)$, inside which the neuron either fires a single spike or doesn't fire at all, and obtain the eligibility trace in each interval $(t, t+dt)$. [1]

First note that $\partial_w \ln \mathscr{P} = \frac{\partial_w \mathscr{P}}{\mathscr{P}}$. To separate the cases where there is a spike ($\mathscr{P}(y=1)$) from those in which there is no spike ($\mathscr{P}(y=0)$), let's define

$$P \equiv \mathscr{P}(y=1|x), \tag{A.12}$$

so that $1-P = \mathscr{P}(y=0|x)$. With this new notation, I can write

$$\partial_w \ln \mathscr{P} = \frac{\partial_w P}{P}\delta + \frac{\partial_w(1-P)}{(1-P)}(1-\delta), \tag{A.13}$$

where $\delta = 1$ if there is a spike in the current time bin, and $\delta = 0$ otherwise. Note that I have assumed that I am considering bins of 1ms or less, so that no more than 1 spike can occur in each bin.

Our neuron model emits a spike with probability given by Eq. 2.6,

$$P = \phi(u)dt = ke^{\beta u}dt, \tag{A.14}$$

where u is the membrane potential and depends on the synaptic weights only through the product $\sum_j w_j PSP_j$, see Eq. 2.4. Using the chain rule, I obtain

$$\frac{\partial_w P}{P} = \frac{1}{P}\frac{\partial P}{\partial u}\frac{\partial u}{\partial w} = \frac{1}{P}\cdot \beta P \cdot PSP(t) = \beta PSP(t) \tag{A.15}$$

where there is a spike, and

$$\begin{aligned} \frac{\partial_w(1-P)}{(1-P)} &= \frac{1}{(1-P)}\frac{\partial(1-P)}{\partial u}\frac{\partial u}{\partial w} = \\ &= -\frac{\beta\phi dt}{(1-\phi dt)}PSP(t) \approx -\beta\phi dt PSP(t), \end{aligned} \tag{A.16}$$

in the absence of spikes, where the last approximation is valid for small enough dt so that $1-\phi dt \approx 1$. Taking the sum of these two quantities I find, in bin $(t, t+dt)$,

$$\partial_w \ln \mathscr{P} = \beta\{\delta - \phi dt(1-\delta)\}PSP(t). \tag{A.17}$$

I next sum all the contributions over time bins covering the interval $(0,t)$. Summing up over all N bins (t_k, t_k+dt), with $t_0 = 0$ and $t_N = t$ and $\mathscr{P}_k$ the likelihood in bin k, I get

$$\sum_k \partial_w \ln \mathscr{P}_k = \sum_k \beta\{\delta_k - \phi dt(1-\delta_k)\}PSP(t_k) = \sum_k \beta PSP(t_k)\delta_k - \sum_k \beta\phi(1-\delta_k)PSP(t_k)dt, \tag{A.18}$$

where $\delta_k = 1$ if a spike has occurred in bin k, and $\delta_k = 0$ otherwise. The times t^ν are the output spike times observed in $(0,t)$. Performing now the limit $dt \to 0$, $N \to \infty$, for the first term on the right hand

[1] This route is a different way to obtain the eligibility trace compared to the one reported in Pfister et al. (2006). The advantage is that, along the way, it will provide directly the expression to be used in the numerical evaluation of the algorithm (Eq. A.17).

side I get

$$\sum_k \beta PSP(t_k)\delta_k \longrightarrow \sum_{t^\nu} \beta PSP(t^\nu), \tag{A.19}$$

where $\{t^\nu\}$ are the neuron's spike emission times. The second term turns into an integral (by definition of integral):

$$\sum_k \beta\phi(1-\delta_k)PSP(t_k)dt \longrightarrow \int_0^t dt'\beta\phi(u(t'))PSP(t'). \tag{A.20}$$

Here, the domain of integration should exclude those points $\{t^\nu\}$ in which there was an output spike (i.e., where $1-\delta_k = 0$). However, this is a finite set of points and, as I know from calculus, including those points will not affect the value of the integral.

Next, by introducing Dirac's delta functions I can also write the identity

$$\sum_{t^\nu} \beta PSP(t^\nu) = \int_0^t \beta PSP(t') \sum_{t^\nu} \delta(t'-t^\nu). \tag{A.21}$$

From the last two equations I obtain, in the continuum limit $dt \to 0$, $N \to \infty$, that $\sum_k \partial_w \ln \mathscr{P}_k$ in Eq. A.18 will converge to

$$\partial_w \ln \mathscr{P} = \int_0^t dt'\beta \left(\sum_{t^\nu} \delta(t'-t^\nu) - \phi(u(t')) \right) PSP(t'). \tag{A.22}$$

Finally, taking the derivative with respect to t I obtain Eq. 2.14:

$$\frac{d}{dt}\partial_w \ln \mathscr{P} = \beta \left(\sum_{t^\nu} \delta(t-t^\nu) - \phi(u(t)) \right) PSP(t). \tag{A.23}$$

A.3 Numerical implementation

The numerical implementation of the learning algorithm was done with a simple Euler algorithm with an elementary time bin of $dt = 0.2$ ms. Thus, the update rule for the eligibility trace, Eq. A.11, in each time bin was

$$E^{new} = E^{old} - \frac{E^{old}}{\tau_M}dt + \frac{1}{\tau_M}\frac{d}{dt}\partial_w \ln \mathscr{P}_{\mathbf{w}}(y|x)dt, \tag{A.24}$$

where

$$\frac{d}{dt}\partial_w \ln \mathscr{P}_{\mathbf{w}}(y|x)dt = \partial_w \ln \mathscr{P}_{\mathbf{w}}(y|x) = \beta\{\delta - \phi dt(1-\delta)\}PSP(t). \tag{A.25}$$

The last equality follows from Eq. A.17, that provides the value of the eligibility trace in the current time bin $(t, t+dt)$. Similarly, the synaptic weights (see Eq. 2.19) are updated in each time bin according to

$$w^{new} = w^{old} + \eta |R_t| a(\tilde{P}_s^\mu)(r^\nu - 1)Edt, \tag{A.26}$$

where E is given by Eq. A.24.

Appendix B

Learning many stimuli

In this appendix I show a few case studies demonstrating the network's ability to learn a 3-choice task with a large number of stimuli. Specifically, the network in Fig. B.1 learned 96 input stimuli divided into 48 relevant and 48 non-relevant stimuli, each comprising 300 spike trains. The network used 100 decision neurons per population and the homeostatic threshold mechanism introduced in Sec. 3.4.3. The second example (Fig. B.2) shows a network able to learn a 3-choice task with 48 input stimuli divided into 24 relevant and 24 non-relevant stimuli, each comprising 400 spike trains. The network used 400 decision neurons per population and the homeostatic threshold mechanism.

Both examples shown here used input patterns with equal firing rate of 12 Hz in all spike trains and jitter parameter $\sigma = 0$ (see Sec. 2.4), emulating a perfect *spike timing coding* of the stimuli. Because the firing rates were the same across all input spike trains, this problem is more difficult to learn than if using the main stimuli with non-uniform rates (see Fig. 3.4).

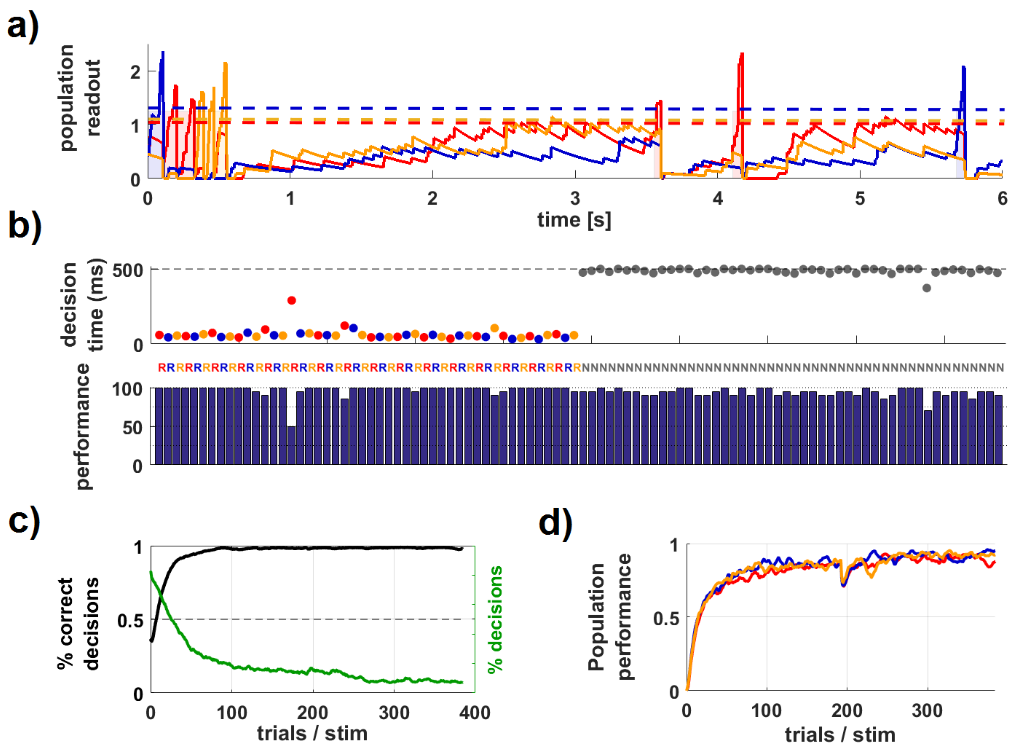


FIGURE B.1: **Example of learning an extensive number of stimuli.** A network learning 96 stimuli of which 48 relevant (see the text). **a)** Population readout (same key as in Fig. 3.2). **b)** Network' s performance for each stimulus (bars, bottom panel) and average decision times in response to each stimulus (top panel). The ability to respond correctly to relevant stimuli (in color) after training is nearly optimal (~98%), and the ability to ignore non-relevant stimuli is high (~90%). Each color encodes a different decision. **c)** Learning curves (same as in Fig. 3.3, with fraction of decisions with non-relevant stimuli in green). **d)** Learning curve plotted separately for each population. The performance across both relevant and non-relevant stimuli is shown for each curve.

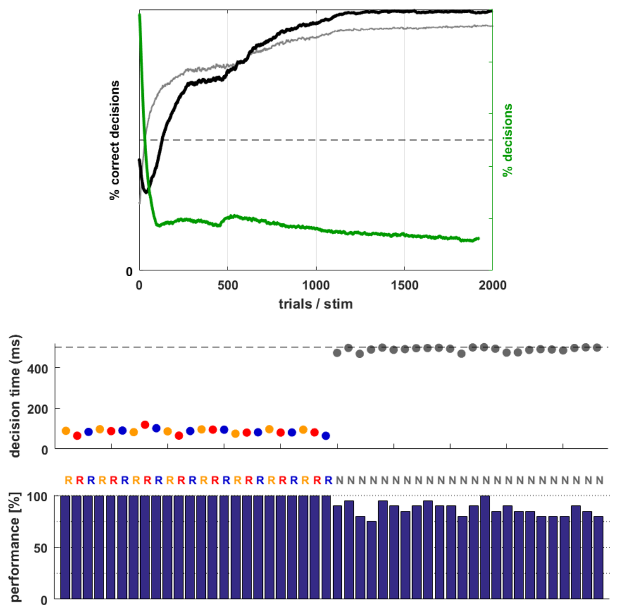


FIGURE B.2: **Example of learning an extensive number of stimuli.** A network learning 48 stimuli of which 24 relevant (see the text). **Top:** same as Fig. B.1, panel c), with superimposed the performance across all stimuli (grey). **Bottom:** same as Fig. B.1, panel b).

Bibliography

Abbott L, van Vreeswijk C (1993) Asynchronous states in networks of pulse-coupled oscillators. *Physical Review E* 48: 1483.

Ahrens MB, Li JM, Orger MB, Robson DN, Schier AF, Engert F, Portugues R (2012) Brain-wide neuronal dynamics during motor adaptation in zebrafish. *Nature* 485: 471–477.

Alonso E, Mondragón E (2013) Associative reinforcement learning a proposal to build truly adaptive agent and multi-agent system INTERNATIONAL CONFERENCE ON AGENTS AND ARTIFICIAL INTELLIGENCE.

Amit DJ (1992) *Modeling brain function: The world of attractor neural networks* Cambridge University Press.

Amit DJ, Brunel N (1997) Model of global spontaneous activity and local structured activity during delay periods in the cerebral cortex. *Cerebral cortex* 7: 237–252.

Baras D, Meir R (2007) Reinforcement learning, spike-time-dependent plasticity, and the bcm rule. *Neural Computation* 19: 2245–2279.

Bernacchia A, Seo H, Lee D, Wang XJ (2011) A reservoir of time constants for memory traces in cortical neurons. *Nature neuroscience* 14: 366–372.

Bienenstock E, Cooper L, Munro P (1982) Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex. *J Neurosci* 2: 32–48.

Boyden ES, Zhang F, Bamberg E, Nagel G, Deisseroth K (2005) Millisecond-timescale, genetically targeted optical control of neural activity. *Nature neuroscience* 8: 1263–1268.

Boyle PM, Williams JC, Ambrosi CM, Entcheva E, Trayanova NA (2013) A comprehensive multiscale framework for simulating optogenetics in the heart. *Nature communications* 4.

Bressler SL, Menon V (2010) Large-scale brain networks in cognition: emerging methods and principles. *Trends in cognitive sciences* 14: 277–290.

Brette R, Gerstner W (2005) Adaptive exponential integrate-and-fire model as an effective description of neuronal activity. *Journal of Neurophysiology* 94: 3637–3642.

Brinkman C, Porter R (1979) Supplementary motor area in the monkey: activity of neurons during performance of a learned motor task. *Journal of Neurophysiology* 42: 681–709.

Burkitt AN (2006a) A review of the integrate-and-fire neuron model: I. homogeneous synaptic input. *Biological cybernetics* 95: 1–19.

Burkitt AN (2006b) A review of the integrate-and-fire neuron model: Ii. inhomogeneous synaptic input and network properties. *Biological cybernetics* 95: 97–112.

Cannon J, Miller P (2016) Synaptic and intrinsic homeostasis cooperate to optimize single neuron response properties and tune integrator circuits. *Journal of Neurophysiology* 116: 2004–2022.

Churchland PS, Sejnowski TJ (2016) *The computational brain* MIT press.

Cole KS, Hodgkin AL (1939) Membrane and protoplasm resistance in the squid giant axon. *The Journal of general physiology* 22: 671–687.

Cole KS, Marmont G (1942) The effect of ionic environment upon the longitudinal impedance of the squid giant axon In *Fed. Proc*, Vol. 1, pp. 15–16.

Daw ND, Doya K (2006) The computational neurobiology of learning and reward. *Current opinion in neurobiology* 16: 199–204.

Dayan P, Abbott LF (2001) *Theoretical neuroscience*, Vol. 806 Cambridge, MA: MIT Press.

Dayan P, Daw ND (2008) Decision theory, reinforcement learning, and the brain. *Cognitive, Affective, & Behavioral Neuroscience* 8: 429–453.

Dayan P, Niv Y (2008) Reinforcement learning: the good, the bad and the ugly. *Current opinion in neurobiology* 18: 185–196.

Dean I, Harper NS, McAlpine D (2005) Neural population coding of sound level adapts to stimulus statistics. *Nature neuroscience* 8: 1684–1689.

Destexhe A, Mainen ZF, Sejnowski TJ (1994) Synthesis of models for excitable membranes, synaptic transmission and neuromodulation using a common kinetic formalism. *Journal of computational neuroscience* 1: 195–230.

Doya K (2008) Modulators of decision making. *Nature neuroscience* 11: 410–416.

Ermentrout B (1998) Neural networks as spatio-temporal pattern-forming systems. *Reports on progress in physics* 61: 353.

Fellows LK (2004) The cognitive neuroscience of human decision making: a review and conceptual framework. *Behavioral and cognitive neuroscience reviews* 3: 159–172.

FitzHugh R (1961) Impulses and physiological states in theoretical models of nerve membrane. *Biophysical journal* 1: 445–466.

Florian RV (2007) Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity. *Neural Computation* 19: 1468–1502.

Fontaine B, Peña JL, Brette R (2014) Spike-threshold adaptation predicted by membrane potential dynamics in vivo. *PLoS Comput Biol* 10: e1003560.

Friedrich J, Urbanczik R, Senn W (2011) Spatio-temporal credit assignment in neuronal population learning. *PLoS Comput Biol* 7: e1002092.

Gerstner W, Kistler WM (2002) *Spiking neuron models: Single neurons, populations, plasticity* Cambridge university press.

Goldberg ME, Wurtz RH (1972) Activity of superior colliculus in behaving monkey. i. visual receptive fields of single neurons. *J Neurophysiol* 35: 542–559.

Gütig R (2016) Spiking neurons can discover predictive features by aggregate-label learning. *Science* 351: aab4113.

Gütig R, Sompolinsky H (2006) The tempotron: a neuron that learns spike timing–based decisions. *Nature neuroscience* 9: 420–428.

Harnack D, Pelko M, Chaillet A, Chitour Y, van Rossum MC (2015) Stability of neuronal networks with homeostatic regulation. *PLoS Comput Biol* 11: e1004357.

Hilbert M (2012) Toward a synthesis of cognitive biases: how noisy information processing can bias human decision making. *Psychological bulletin* 138: 211.

Hodgkin AL, Huxley AF (1952) A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology* 117: 500–544.

Hodgkin AL (1976) Chance and design in electrophysiology: an informal account of certain experiments on nerve carried out between 1934 and 1952. *The Journal of physiology* 263: 1.

Hopfield JJ (1982) Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences* 79: 2554–2558.

Hopfield JJ, Herz AV (1995) Rapid local synchronization of action potentials: Toward computation with coupled integrate-and-fire neurons. *Proceedings of the National Academy of Sciences* 92: 6655–6662.

Hubel DH, Wiesel TN (1962) Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology* 160: 106–154.

Huxley A (2002) Classical perspectives. *Nature* 144: 710–711.

Izhikevich EM, FitzHugh R (2006) Fitzhugh-nagumo model. *Scholarpedia* 1: 1349.

Jezzini A, Mazzucato L, La Camera G, Fontanini A (2013) Processing of hedonic and chemosensory features of taste in medial prefrontal and insular networks. *Journal of Neuroscience* 33: 18966–18978.

Johnson A, van der Meer MA, Redish AD (2007) Integrating hippocampus and striatum in decision-making. *Current opinion in neurobiology* 17: 692–697.

Jolivet R, J. T, Gerstner W (2003) *The Spike Response Model: A Framework to Predict Neuronal Spike Trains*, pp. 846–853 Springer Berlin Heidelberg, Berlin, Heidelberg.

Jolivet R, Kobayashi R, Rauch A, Naud R, Shinomoto S, Gerstner W (2008) A benchmark test for a quantitative assessment of simple neuron models. *Journal of neuroscience methods* 169: 417–424.

Jolivet R, Lewis TJ, Gerstner W (2004) Generalized integrate-and-fire models of neuronal activity approximate spike trains of a detailed model to a high degree of accuracy. *Journal of Neurophysiology* 92: 959–976.

Jolivet R, Rauch A, Lüscher HR, Gerstner W (2006) Predicting spike timing of neocortical pyramidal neurons by simple threshold models. *Journal of Computational Neuroscience* 21: 35–49.

Jolivet R, Timothy J, Gerstner W (2003) The spike response model: a framework to predict neuronal spike trains In *Artificial Neural Networks and Neural Information ProcessingâĂŤICANN/ICONIP 2003*, pp. 846–853. Springer.

Jones LM, Fontanini A, Sadacca BF, Miller P, Katz DB (2007) Natural stimuli evoke dynamic sequences of states in sensory cortical ensembles. *Proceedings of the National Academy of Sciences* 104: 18772–18777.

Kappel D, Nessler B, Maass W (2014) Stdp installs in winner-take-all circuits an online approximation to hidden markov model learning. *PLoS Comput Biol* 10: e1003511.

Kim S, McNames J (2007) Automatic spike detection based on adaptive template matching for extracellular neural recordings. *Journal of neuroscience methods* 165: 165–174.

Kistler WM, Gerstner W, v. Hemmen JL (1997) Reduction of the hodgkin-huxley equations to a single-variable threshold model. *Neural Computation* 9: 1015–1045.

Kobayashi R, Tsubo Y, Shinomoto S (2009) Made-to-order spiking neuron model equipped with a multi-timescale adaptive threshold. *Frontiers in Computational Neuroscience* 3: 9.

La Camera G (2017) *Lecture notes for BIO 332: 'Computational modeling of physiological systems'*. Stony Brook University.

La Camera G, Rauch A, Lüscher HR, Senn W, Fusi S (2004) Minimal models of adapted neuronal response to in vivo–like input currents. *Neural computation* 16: 2101–2124.

La Camera G, Urbanczik R, Senn W (2013) Stimulus detection and decision making via spike-based reinforcement learning. *RLDM 2013* p. 183.

Lapicque L (1907) Recherches quantitatives sur lâĂŹexcitation électrique des nerfs traitée comme une polarisation. *J. Physiol. Pathol. Gen* 9: 620–635.

Lee D, Seo H, Jung MW (2012) Neural basis of reinforcement learning and decision making. *Annual review of neuroscience* 35: 287–308.

Lo CC, Wang XJ (2006) Cortico–basal ganglia circuit mechanism for a decision threshold in reaction time tasks. *Nature neuroscience* 9: 956–963.

MacKay DJ (2003) *Information theory, inference and learning algorithms* Cambridge university press.

Marrelec G, Bellec P, Benali H (2006) Exploring large-scale brain networks in functional mri. *Journal of Physiology-Paris* 100: 171–181.

Mazzucato L, Fontanini A, La Camera G (2015) Dynamics of multistable states during ongoing and evoked cortical activity. *Journal of Neuroscience* 35: 8214–8231.

McCulloch WS, Pitts W (1943) A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics* 5: 115–133.

Montague PR, Dayan P, Sejnowski TJ (1996) A framework for mesencephalic dopamine systems based on predictive hebbian learning. *Journal of neuroscience* 16: 1936–1947.

Morimoto J, Doya K (1998) Hierarchical reinforcement learning of low-dimensional subgoals and high-dimensional trajectories. In *ICONIP*, pp. 850–853. Citeseer.

Morris C, Lecar H (1981) Voltage oscillations in the barnacle giant muscle fiber. *Biophysical journal* 35: 193–213.

Nagumo J, Arimoto S, Yoshizawa S (1962) An active pulse transmission line simulating nerve axon. *Proceedings of the IRE* 50: 2061–2070.

O'doherty JP (2012) Beyond simple reinforcement learning: the computational neurobiology of reward-learning and valuation. *European Journal of Neuroscience* 35: 987–990.

O'doherty JP, Hampton A, Kim H (2007) Model-based fmri and its application to reward learning and decision making. *Annals of the New York Academy of sciences* 1104: 35–53.

Offner F, Weinberg A, Young G (1940) Nerve conduction theory: some mathematical consequences of bernstein's model. *The bulletin of mathematical biophysics* 2: 89–103.

Oja E (1982) Simplified neuron model as a principal component analyzer. *Journal of mathematical biology* 15: 267–273.

Ostojic S (2011) Interspike interval distributions of spiking neurons driven by fluctuating inputs. *Journal of Neurophysiology* 106: 361–373.

Pfister JP, Toyoizumi T, Barber D, Gerstner W (2006) Optimal spike-timing-dependent plasticity for precise action potential firing in supervised learning. *Neural computation* 18: 1318–1348.

Rabiner LR (1989) A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77: 257–286.

Rabiner LR, Lee C, Juang B, Wilpon J (1989) Hmm clustering for connected word recognition In *Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on*, pp. 405–408. IEEE.

Reynolds JN, Wickens JR (2002) Dopamine-dependent plasticity of corticostriatal synapses. *Neural Networks* 15: 507–521.

Rosenblatt F (1961) Principles of neurodynamics. perceptrons and the theory of brain mechanisms Technical report, DTIC Document.

Rumelhart DE, Hinton GE, Williams RJ (1988a) Learning representations by back-propagating errors. *Cognitive modeling* 5: 1.

Rumelhart DE, Hinton GE, Williams RJ (1988b) Learning representations by back-propagating errors. *Cognitive modeling* 5: 1.

Rumelhart DE, McClelland JL, Group PR et al. (1988) *Parallel distributed processing*, Vol. 1 IEEE.

Rushton W (1951) A theory of the effects of fibre size in medullated nerve. *The Journal of physiology* 115: 101.

Rushworth MF, Behrens TE (2008) Choice, uncertainty and value in prefrontal and cingulate cortex. *Nature neuroscience* 11: 389–397.

Sah P (1996) Ca 2+-activated k+ currents in neurones: types, physiological roles and modulation. *Trends in neurosciences* 19: 150–154.

Sakai Y, Okamoto H, Fukai T (2006) Computational algorithms and neuronal network models underlying decision processes. *Neural Networks* 19: 1091–1105.

Sakmann B, Neher E (1984) Patch clamp techniques for studying ionic channels in excitable membranes. *Annual review of physiology* 46: 455–472.

Samuelsen CL, Gardner MP, Fontanini A (2012) Effects of cue-triggered expectation on cortical processing of taste. *Neuron* 74: 410–422.

Schmidhuber J (2015) Deep learning in neural networks: An overview. *Neural networks* 61: 85–117.

Schmidt E, McIntosh J, Durelli L, Bak M (1978) Fine control of operantly conditioned firing patterns of cortical neurons. *Experimental neurology* 61: 349–369.

Seung HS (2003) Learning in spiking neural networks by reinforcement of stochastic synaptic transmission. *Neuron* 40: 1063–1073.

Shadlen MN, Kiani R (2013) Decision making as a window on cognition. *Neuron* 80: 791–806.

Shadlen MN, Newsome WT (1998) The variable discharge of cortical neurons: implications for connectivity, computation, and information coding. *Journal of neuroscience* 18: 3870–3896.

Simen P, Cohen JD, Holmes P (2006) Rapid decision threshold modulation by reward rate in a neural network. *Neural networks* 19: 1013–1026.

Sompolinsky H (2014) Computational neuroscience: beyond the local circuit. *Current opinion in neurobiology* 25: xiii–xviii.

Stapleton JR, Lavine ML, Wolpert RL, Nicolelis MA, Simon SA (2006) Rapid taste responses in the gustatory cortex during licking. *Journal of Neuroscience* 26: 4126–4138.

Sutton RS (1996) Generalization in reinforcement learning: Successful examples using sparse coarse coding. *Advances in neural information processing systems* pp. 1038–1044.

Sutton RS, Barto AG (1998) *Reinforcement learning: An introduction*, Vol. 1 MIT press Cambridge.

Tesauro G (1995) Temporal difference learning and td-gammon. *Communications of the ACM* 38: 58–68.

Tsodyks M, Mitkov I, Sompolinsky H (1993) Pattern of synchrony in inhomogeneous networks of oscillators with pulse interactions. *Physical review letters* 71: 1280.

Tuckwell HC (2005) *Introduction to theoretical neurobiology: volume 2, nonlinear and stochastic theories*, Vol. 8 Cambridge University Press.

Turrigiano GG, Nelson SB (2004) Homeostatic plasticity in the developing nervous system. *Nature Reviews Neuroscience* 5: 97–107.

Urbanczik R, Senn W (2009) Reinforcement learning in populations of spiking neurons. *Nat Neurosci* 12: 250–252.

Usher M, Schuster HG, Niebur E (1993) Dynamics of populations of integrate-and-fire neurons, partial synchronization and memory. *Neural Computation* 5: 570–586.

Victor JD (2005) Spike train metrics. *Current opinion in neurobiology* 15: 585–592.

Victor JD, Purpura KP (1996) Nature and precision of temporal coding in visual cortex: a metric-space analysis. *Journal of neurophysiology* 76: 1310–1326.

Victor JD, Purpura KP (1997) Metric-space analysis of spike trains: theory, algorithms and application. *Network: computation in neural systems* 8: 127–164.

Wang XJ (2008) Decision making in recurrent neuronal circuits. *Neuron* 60: 215–234.

Wikipedia (2017) Perceptron — wikipedia, the free encyclopedia [Online; accessed 15-May-2017].

William Bialek DW, de Ruyter van Steveninck R (1999) *Spikes: exploring the neural code* MIT press.

Williams AH, O'Leary T, Marder E (2013) Homeostatic Regulation of Neuronal Excitability. *Scholarpedia* 8: 1656 revision #140978.

Williams RJ (1992) Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8: 229–256.

Wilpon JG, Rabiner LR, Lee CH, Goldman E (1990) Automatic recognition of keywords in unconstrained speech using hidden markov models. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 38: 1870–1878.

Xie X, Seung HS (2004) Learning in neural networks by reinforcement of irregular spiking. *Physical Review E* 69: 041909.

Zhang J, Bogacz R, Holmes P (2009) A comparison of bounded diffusion models for choice in time controlled tasks. *Journal of mathematical psychology* 53: 231–241.