Learning Hierarchically-Structured Concepts

Nancy Lynch[1] and Frederik Mallmann-Trenn[2]

[1]Massachusetts Institute of Technology, Cambridge, Massachusetts, USA
[2]King's College London, London, England

Abstract

We use a recently developed synchronous Spiking Neural Network (SNN) model to study the problem of learning hierarchically-structured concepts. We introduce an abstract data model that describes simple hierarchical concepts. We define a feed-forward layered SNN model, with learning modeled using Oja's local learning rule, a well known biologically-plausible rule for adjusting synapse weights. We define what it means for such a network to recognize hierarchical concepts; our notion of recognition is robust, in that it tolerates a bounded amount of noise.

Then, we present a learning algorithm by which a layered network may learn to recognize hierarchical concepts according to our robust definition. We analyze correctness and performance rigorously; the amount of time required to learn each concept, after learning all of the sub-concepts, is approximately $O\left(\frac{1}{\eta k}\left(\ell_{\max}\log(k) + \frac{1}{\varepsilon}\right) + b\log(k)\right)$, where k is the number of sub-concepts per concept, $\ell_{\max}$ is the maximum hierarchical depth, η is the learning rate, ε describes the amount of uncertainty allowed in robust recognition, and b describes the amount of weight decrease for "irrelevant" edges. An interesting feature of this algorithm is that it allows the network to learn sub-concepts in a highly interleaved manner. This algorithm assumes that the concepts are presented in a noise-free way; we also extend these results to accommodate noise in the learning process. Finally, we give a simple lower bound saying that, in order to recognize concepts with hierarchical depth two with noise-tolerance, a neural network should have at least two layers.

The results in this paper represent first steps in the theoretical study of hierarchical concepts using SNNs. The cases studied here are basic, but they suggest many directions for extensions to more elaborate and realistic cases.

Keywords: Hierarchical Concepts, Representing Hierarchical Concepts, Recognizing Hierarchical Concepts, Learning Hierarchical Concepts, Spiking Neural Networks, Brain-Inspired Algorithms

1 Introduction

We are interested in the general problem of *how concepts that have structure are represented in the brain*. What do these representations look like? How are they learned, and how do the concepts get recognized after they are learned? We draw inspiration from recent experimental research on computer vision in convolutional neural networks (CNNs) by Zeiler and Fergus [54] and Zhou, et al. [55]. This research shows that CNNs learn to represent structure in visual concepts: lower layers of the network represent basic concepts and higher layers represent successively higher-level concepts. This observation is consistent with neuroscience research, which indicates that visual processing in mammalian brains is performed in a hierarchical way, starting from primitive notions such as position, light level, etc., and building toward complex objects; see, e.g., [15, 14, 7]. More generally, we consider the thesis that *the structure that is naturally present in real-world concepts get mirrored in their brain representations, in some natural way that facilitates both learning and recognition*.

We approach this problem using ideas and techniques from theoretical computer science, distributed computing theory, and in particular, from recent work by Lynch, et al. on synchronous Spiking Neural Networks (SNNs) [28, 25, 27, 45, 13]. These papers began the development of an algorithmic theory of SNNs, developing formal foundations, and using them to study problems of attention and focus, neural representation, and short-term learning. Here we continue that general development, by initiating the study of long-term learning within the same framework.

We focus here on learning hierarchically-structured concepts. We capture these formally in terms of abstract *concept hierarchies*, in which concepts are built from lower-level concepts, which in turn are built from still-lower-level concepts, etc. Such structure is natural, e.g., for physical objects that are learned and recognized during human or computer visual processing. An example of such a hierarchy might be the following model of a *human*: A human consists of a *body*, a *head*, a *left leg*, a *right leg*, a *left arm*, and a *right arm*. Each of these concepts may consist of more concepts, allowing us to model a human to an arbitrary degree of granularity. Most concepts in the real world have additional structure, e.g., arms and legs are positioned symmetrically; however, we ignore such information for now and assume simply that each concept consists of sub-concepts. For this initial theoretical study, we make some additional simplifications: we fix a maximum level $\ell_{\max}$ for concept hierarchies, we assume that all non-primitive concepts have the same number k of "child concepts", and we assume that our concept hierarchies are trees, i.e., there is no overlap in the composition of different concepts at the same level of a hierarchy. We expect that these assumptions can be removed or weakened, but it seems useful to start with the simplest case.

This paper demonstrates theoretically, in terms of simple hierarchies, how hierarchically-structured data can be represented, learned, and recognized in feed-forward layered Spiking Neural Networks. Specifically, we provide formal definitions for *concept hierarchies* and *layered neural networks*. We define precisely what it means for a layered neural network to *recognize* a particular concept in a concept hierarchy. Our notion of recognition is *robust*: a concept is required to be recognized if the input is close to the ideal concept, and is required not to be recognized if the input is far from the ideal. We also define what it means for a layered neural network to *learn to recognize* a concept hierarchy, according to our robust definition of recognition.

Next, we present two simple, efficient algorithms (layered networks) that learn to recognize concept hierarchies; the first assumes reliability during the learning process, whereas the second tolerates a bounded amount of noise. An example of such learning is shown in Figure 1. We also provide a preliminary lower bound, saying that, in order to robustly recognize concepts with hierarchical depth 2, a neural network should have at least 2 layers. We discuss possible extensions of this bound to concepts with larger depth. We end with many directions for extending this work.

Note: We view this work as the first step in a general project to produce a theory for how logical concepts are represented, and learned, in the brain. Our general approach is to start with the simplest case, working out basic definitions, algorithms, and limitations for that case, and then to extend in many directions, step-by-step. We think such a stepwise approach will be effective in developing the theory. In addition, we hope that this first step, besides being of interest on its own, will provide a useful blueprint for later extensions.

In more detail: We describe our data model in Section 2. We assume a fixed maximum number $\ell_{\max}$ of levels in our concept hierarchies. Each concept hierarchy $\mathcal{C}$ has a fixed set C of concepts, organized into levels ℓ, $0 \leq \ell \leq \ell_{\max}$. These are chosen from some universal set D of *concepts*. Each concept at each level ℓ, $1 \leq \ell \leq \ell_{\max}$ has precisely k children, which are level $\ell - 1$ concepts. We assume here that each concept hierarchy is a tree, that is, there is no overlap among the sets of children of different concepts. Each individual concept hierarchy represents the concepts and child relationships that arise in a particular execution of the network (or lifetime of an organism). However, the chosen concepts and their relationships may be different in different concept hierarchies. Again we note that these assumptions are a considerable simplification of reality, but we regard them as a good starting point.

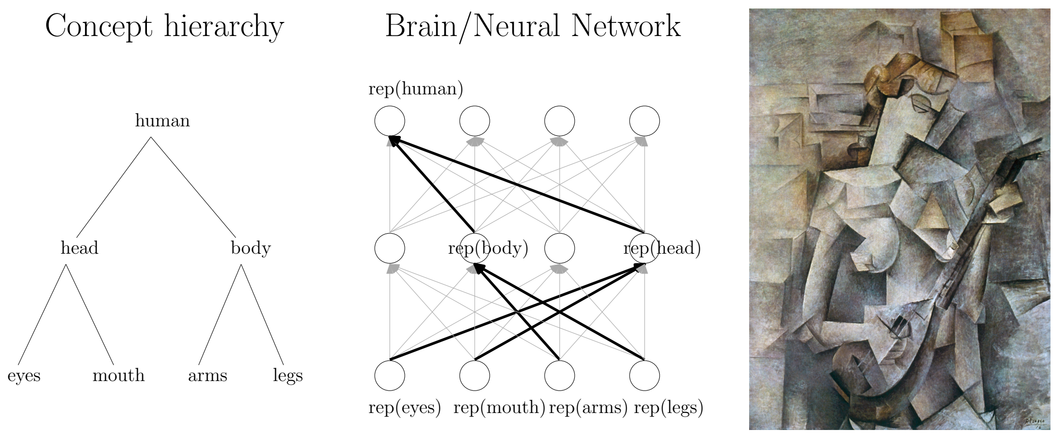

Figure 1: The leftmost figure shows the concept *human*, which consists of two sub-concepts, and so on. The second figure shows a network that has "learned" the concept "human" in the sense that, when the neurons representing the basic parts *eyes, mouth, arms, legs* are excited, then exactly one neuron u on the top layer will fire. Neuron u should also fire when "most" of the basic parts are excited, and u should not fire when few of the basic parts are excited. For example, the painting “Girl with a Mandolin” by Picasso[1] should cause u to fire despite the lack of a mouth and legs. The network accomplishes this by strengthening relevant synapses (bold edges) and weakening others (thin edges).

Next, in Section 3, we define a synchronous Spiking Neural Network model[2], derived from the one in [28, 27], but with additional structure to support learning. Namely, the new model incorporates edge weights (representing synapse strengths) into neuron states; this provides a convenient way to describe how those weights change during learning. We model learning using *Oja's rule*, a biologically-inspired rule that can be regarded as a mathematical formalization of Hebbian learning [18]. Oja's rule was first introduced in [35], and has since received considerable attention due its connections with dimensionality reduction; see, for example, [36, 8]. Although there is no direct experimental evidence yet that Oja's precise rule is used in the brain, its core characteristics such as long-term potentiation, long-term depression, and normalization are known to occur in brain networks, and have been studied thoroughly (e.g., [2, 1]). Interestingly, to the best of our knowledge, Oja's rule has so far been studied only in "flat" settings, where the network has only one layer. Moreover, previous work (e.g., [35]) has allowed the learning parameter η to be time-dependent, in order to achieve convergence. In this paper, we consider the multilayer setting, and we show convergence with a fixed learning rate.

In Section 4, we present our definitions for the robust recognition and noise-free learning problems. Thus, we define how an SNN represents a concept hierarchy; here we use the simplifying assumption that each concept is represented by just one neuron. We define what it means for an SNN to correctly recognize a concept hierarchy, including situations in which the network is required to recognize a concept c and situations where it is required not to do so. In particular, if a sufficiently large fraction r_2 of the children of concept c are recognized, then c should be recognized, whereas if fewer than a smaller fraction r_1 of the children of c are recognized, then c should not be recognized. We also define what it means for an SNN to learn to recognize a concept hierarchy, in the noise-free setting.

Then, in Section 5, we present algorithms that allow a network, starting from a default configuration, to recognize and to learn the concepts in a particular concept hierarchy. Our algorithms are efficient,

[2]A word about our use of the Spiking Neural Network terminology: Our model here is simpler than typical SNN models, in that neuron actions depend just on the previous state and not on a longer history. In some of our prior work, such as [45], we use a more elaborate version of the model in which neurons actions can depend on bounded history. This is useful for capturing aspects of neuron processing such as accumulating potential. In future extensions of the present work, we expect to use such elaborations. We use the SNN terminology here in an attempt to keep the terminology consistent across our papers.

in terms of network size and running time. In particular, a network with max layer $\ell_{\max}$ suffices to recognize a concept hierarchy with max level $\ell_{\max}$. Recognition happens within a very short time, proportional to the number of layers in the network. For learning, our algorithm converges reasonably quickly to a configuration that supports robust recognition. Our convergence time bound result for noise-free learning is Theorem 5.3. Our algorithms require the examples to be shown several times and in a constrained order: roughly speaking, we require the network to "learn" the children of a concept c first, before examples of c are shown. Thus, in our running example, we require enough examples of "head", "body", etc. to be able to learn those concepts before the network sees them all together as "human". Except for this constraint, concepts may be shown in an arbitrarily interleaved manner. In Section 6, we adapt our problem definitions and learning algorithm to a setting where the examples presented may be perturbed by noise. The modified algorithm still works, but now convergence requires the network to see more examples, compared to the noise-free case, as we show in Theorem 6.4. The detailed analysis needed to prove Theorems 5.3 and 6.4 appears in Sections A and B, respectively.

Once we see that a network with max layer $\ell_{\max}$ can easily learn and recognize any concept hierarchy with max level $\ell_{\max}$, it is natural to ask whether $\ell_{\max}$ layers are actually necessary. Certainly these networks yield natural and efficient representations, but it is still interesting to ask the theoretical question of whether shallower networks could accomplish the same thing. In Section 7, we give a preliminary lower bound result, showing that a two-layer concept hierarchy requires a two-layer network in order to solve the noisy recognition problem. We also discuss the possibility of extending this result to more levels and layers.

In summary, this paper is intended to show, using theoretical techniques, how structured concepts can be represented, recognized, and learned in biologically plausible neural networks. We give fundamental definitions and algorithms for particular types of concept hierarchies and networks. This represents a first step towards a theory of representation and learning for hierarchically-structured concepts in SNNs; it opens up many follow-on questions, which we discuss in Section 8.

Related work: Immediate inspiration from this work came from experimental computer vision research on "network dissection" by Zhou, et al. [55]. This work describes experiments that show that unsupervised learning of visual concepts in deep convolutional neural networks results in "disentangled" representations. These include neural representations, not just for the main concepts of interest, but also for their components and sub-components, etc., throughout a concept hierarchy. As in this paper, they consider individual neurons as representations for individual concepts. They find that the representations that arise are generally arranged in layers so that more primitive concepts (colors, textures,...) appear at lower layers whereas more complex concepts (parts, objects, scenes) appear at higher layers. Earlier work by Zeiler and Fergus [54] made similar observations. As we described earlier, this work is consistent with neuroscience research, which indicates that visual processing in mammalian brains is performed hierarchically [15, 14, 7]. Some of this work indicates that the network includes feedback edges in addition to forward edges; the function of the feedback edges seems to be to solidify representations of lower-level objects based on context [16, 33]. While we do not yet address feedback edges in this paper, that is one of our main intended future directions.

Brain-like hierarchical models have been studied before (e.g., [43] and [44]). The authors of [43] propose a model consisting of different kinds of cells to model image recognition in the brain. Another biologically-motivated line of research concerns synfire chains, which are essentially a feed-forward network of neurons. These networks are a predecessor of spiking neural networks (SNNs). An interesting work in this field is [44], which studies a hierarchical organization of synfire chains.

The SNN model [29, 30, 9, 17, 11], upon which all of our neural algorithms research is based, is a model for neural computation that balances biological plausibility with theoretical tractability. Our work is influenced by research of Maass et al. [30, 31, 32] on the computational power of SNNs, and by that of Valiant [47, 48, 49, 50] on learning in the *neuroidal model* of brain computation. Recent research by Papadimitriou, et al. [40, 42, 22, 41] on problems of learning and association of concepts is another source of inspiration.

Oja's learning rule [35, 36]. is a biologically plausible local rule for adjusting synapse weights during learning. As mentioned earlier, to the best of our knowledge, Oja's rule has so far been studied only in single-layered networks and with time-dependent learning rates ([35, 36, 8]. Other related learning rules include Hebbian variants [12, 23] or BCM learning [3].

The learning algorithms in this paper utilize a *Winner-Take-All* sub-network [21, 53, 46, 4, 32, 51, 37, 24], to help in selecting which neurons to engage in learning. Winner-Take-All is an important primitive in neural computation that is used to model visual attention and competitive learning. Maybe "Note that such engagement of a neuron to learn is also known in some of neuroscience literature eligibility traces (or synaptic flags); see [10] for experimental evidence of the existence of eligibility traces.

Work by Mhaskar et al. [34] is related to ours in that they also consider embedding a tree-structured concept hierarchy in a layered network. They also prove results saying that deep neural networks are better than shallow networks at representing a deep concept hierarchy, However, their concept hierarchies differ mathematically from ours, since they are formalized as compositional functions. Also, their notion of representation is different, corresponding to function approximation, and their proofs are based on approximation theory. Other related work appears in papers by Knoblauch and collaborators, e.g., [6, 19, 39]. These papers describe experimental work involving hierarchical concepts that are more general than ours (e.g., allowing overlap), networks that are more general (e.g., allowing feedback), and more robust types of representations (cell assemblies). They present this work in the context of an integrated robot system combining processing of visual and language input, decisions, and action). For us, this provides good inspiration for future theoretical work.

Acknowledgments: We thank Brabeeba Wang for helpful conversations and suggestions. we also thank an anonymous referee for much constructive feedback, and many suggestions for interesting extensions. The authors were supported in part by NSF Award Numbers CCF-1810758, CCF-0939370, CCF-1461559, and CCF-2003830.

2 Data Model

In this section, we define an abstract notion of a *concept hierarchy*, which represents all the concepts that arise in some particular "lifetime" of an organism, together with hierarchical relationships between them. As noted above, our definition is restricted to tree-structured hierarchical relationships; extensions are left for future work. We follow this with a definition for the notion of *support*, which indicates which lowest-level concepts are sufficient to trigger the recognition of higher-level concepts.

2.1 Preliminaries

We begin by defining some general notation. First, we fix four constants:

- $\ell_{\max}$, a positive integer, representing the maximum level number for the concepts we consider.
- n, a positive integer, representing the total number of lowest-level concepts.
- k, a positive integer, representing the number of top-level concepts in any concept hierarchy, and also the number of sub-concepts for each concept that is not at the lowest level.[3]
- r_1, r_2, reals in $[0, 1]$ with $r_1 \leq r_2$; these represent thresholds for noisy recognition.

We assume a predetermined universal set D of *concepts*, partitioned into disjoint sets $D_\ell, 0 \leq \ell \leq \ell_{\max}$. We refer to any particular concept $c \in D_\ell$ as a *level* ℓ *concept*, and write $level(c) = \ell$. Here, D_0 represents the most basic concepts and $D_{\ell_{\max}}$ the highest-level concepts. We assume that $|D_0| = n$.

2.2 Concept hierarchies

A *concept hierarchy* $\mathcal{C}$ consists of a subset C of D, together with a $children$ function. For each ℓ, $0 \leq \ell \leq \ell_{\max}$, we define C_ℓ to be $C \cap D_\ell$, that is, the set of level ℓ concepts in $\mathcal{C}$. For each concept $c \in C_\ell$, $1 \leq \ell \leq \ell_{max}$, we designate a nonempty set $children(c) \subseteq C_{\ell-1}$. We call each $c' \in children(c)$ a *child* of c. We require the following three properties.

1. $|C_{\ell_{\max}}| = k$.

[3] Assuming the same number k throughout is a simplification of what would be needed for applications; it should be easy to generalize this.

2. For any $c \in C_\ell$, where $1 \leq \ell \leq \ell_{\max}$, we have that $|children(c)| = k$; that is, the degree of any internal node in the concept hierarchy is exactly k.

3. For any two distinct concepts c and c' in C_ℓ, where $1 \leq \ell \leq \ell_{\max}$, we have that $children(c) \cap children(c') = \emptyset$; that is, the sets of children of different concepts at the same level are disjoint.

It follows that $\mathcal{C}$ is a forest with k roots and height $\ell_{\max}$. Also, for any $\ell, 0 \leq \ell \leq \ell_{\max}$, $|C_\ell| = k^{\ell_{\max} - \ell + 1}$. Note that our notion of concept hierarchies is quite restrictive, in that we allow no overlap between the sets of children of different concepts. Allowing overlap is an important next direction for future work.

We extend the $children$ notation recursively by defining a concept c' to be a $descendant$ of a concept c if either $c' = c$, or c' is a child of a descendant of c. We write $descendants(c)$ for the set of descendants of c. Let $leaves(c) = descendants(c) \cap C_0$, that is, all the level 0 descendants of c.

2.3 Support

Now we give a key definition that indicates which lowest-level concepts should be sufficient to trigger recognition of higher-level concepts.

We fix a particular concept hierarchy $\mathcal{C}$, with its concept set C partitioned into $C_0, \ldots, C_{\ell_{\max}}$. For any given subset B of the general set D_0 of level 0 concepts, and any real number $r \in [0, 1]$, we define a set $supported_r(B)$ of concepts in C. This represents the set of concepts $c \in C$, at all levels, that have enough of their leaves present in B to support recognition of c. The notion of "enough" here is defined recursively, based on having an r-fraction of children supported at every level.

Definition 2.1 (**Supported**). *Given $B \subseteq D_0$, define the following sets of concepts at all levels, recursively:*

1. *$B_0 = B \cap C_0$. That is, we restrict attention to just the level 0 concepts in C.*

2. *B_1 is the set of all concepts $c \in C_1$ such that $|children(c) \cap B_0| \geq rk$. That is, we consider the level 1 concepts in C for which at least an r-fraction of their children appear in B_0.*

3. *For $2 \leq \ell \leq \ell_{max}$, B_ℓ is the set of all concepts $c \in C_\ell$ such that $|children(c) \cap B_{\ell-1}| \geq rk$. That is, we consider the level ℓ concepts in C for which at least an r-fraction of their children appear in $B_{\ell-1}$.*

Define $supported_r(B)$ to be $\bigcup_{0 \leq \ell \leq \ell_{\max}} B_\ell$. We sometimes also write $supported_r(B, \ell)$ for B_ℓ.

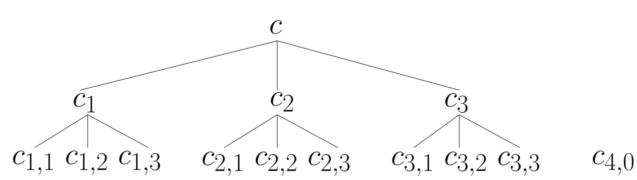

Figure 2: This example illustrates the $supported_r(B)$ definition, with $k = 3$ and $r = \frac{2}{3}$. We depict just a single level 2 concept c with children c_1, c_2, c_3 and grandchildren $c_{1,1}, c_{1,2}, c_{1,3}, c_{2,1}, c_{2,2}, c_{2,3}, c_{3,1}, c_{3,2}, c_{3,3}$. The set B consists of concepts $c_{1,1}$, $c_{1,2}$, $c_{3,1}, c_{3,3}$ plus an "extra" concept $c_{4,0}$ that is not a descendant of c. Then $B_0 = \{c_{1,1}, c_{1,2}, c_{3,1}, c_{3,3}\}$, $B_1 = \{c_1, c_3\}$, and $B_2 = \{c\}$.

The special case $r = 1$ is important as it corresponds to a "noise-free" notion of support, in which all the leaves of a concept must be present. That is:

Lemma 2.2. *For any $B \subseteq D_0$, $supported_1(B)$ is the set of all concepts $c \in C$ (at all levels) such that $leaves(c) \subseteq B$.*

3 Network Model

In this section, we define our network model. We first describe the network structure, then the individual neurons, and finally the operation of the overall network.

3.1 Preliminaries

We introduce four constants:

- ℓ'_{max}, a positive integer, representing the maximum number of a layer in the network.
- n, a positive integer, representing the number of distinct inputs the network can handle. This is the same n as in the data model, where it represents the total number of level 0 concepts in a concept hierarchy.
- τ, a real number, representing the firing threshold for neurons.
- η, a positive real, representing the learning rate for our learning rule.

3.2 Network structure

Our networks are directed graphs consisting of neurons arranged in layers, with edges directed from each layer to the next-higher layer; thus, they are feed-forward layered neural networks.

Specifically, a network $\mathcal{N}$ consists of a set N of neurons, partitioned into disjoint sets $N_\ell, 0 \leq \ell \leq \ell'_{max}$, which we call *layers*. We refer to any particular neuron $u \in N_\ell$ as a *layer ℓ neuron*, and write $layer(u) = \ell$. We assume (for simplicity) that each layer contains exactly n neurons, that is, $|N_\ell| = n$ for every ℓ. We refer to the n layer 0 neurons as *input neurons* and to all other neurons as *non-input neurons*. We assume total connectivity between successive layers, that is, each neuron in $N_\ell, 0 \leq \ell \leq \ell'_{max} - 1$ has an outgoing edge to each neuron in $N_{\ell+1}$, and these are the only edges.

We assume a one-to-one mapping $rep : D_0 \to N_0$, where $rep(c)$ is the neuron corresponding to concept c. That is, rep is a one-to-one mapping from the full set of level 0 concepts, D_0, to N_0, the set of layer 0 neurons, This will allow the network to receive an input corresponding to any level 0 concept. See Figure 3 for a depiction.

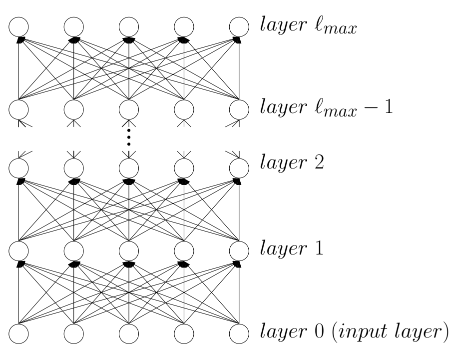

Figure 3: The figure depicts the general structure of a feed-forward network.

We "lift" the definition of rep to sets of level 0 concepts as follows: For any $B \subseteq D_0$, we define $rep(B) = \{rep(b) | b \in B\}$. That is, $rep(B)$ is the set of all $reps$ of concepts in B. (We will use analogous "lifting" definitions to extend other functions to sets.)

Since we know that $|C_0| = k^{\ell_{\max}+1}$, $C_0 \subseteq D_0$, and all elements of D_0 have $reps$ among the n neurons of N_0, it follows that $n \geq k^{\ell_{\max}+1}$. However, we imagine that n is much larger than this, because we imagine that the total number of possible level 0 concepts is much larger than the number that will arise in any particular execution of the network.

In Section 4, we will consider extensions of the $rep()$ function from level 0 concepts to higher-level concepts. Establishing such higher-level $reps$ will be the job of a learning algorithm.

3.3 Neuron states

We assume that the state of each neuron consists of several *state components*. Here we distinguish between input neurons and non-input neurons. Namely, each input neuron $u \in N_0$ has just one state component:

- *firing*, with values in $\{0, 1\}$; this indicates whether or not the input neuron is currently firing.

We denote the *firing* component of input neuron u at integer time t by $firing^u(t)$; we will sometimes abbreviate this in mathematical formulas as just $y^u(t)$.

Each non-input neuron $u \in N_\ell$, $1 \leq \ell \leq \ell'_{max}$, has three state components:

- *firing*, with values in $\{0, 1\}$, indicating whether the neuron is currently firing.
- *weight*, a real-valued column vector in $[0, 1]^n$ representing current weights on incoming edges.
- *engaged*, with values in $\{0, 1\}$; indicating whether the neuron is currently prepared to learn. As discussed in the intro, these model eligibility traces (see [10]).

We denote the three components of non-input neuron u at time t by $firing^u(t)$, $weight^u(t)$, and $engaged^u(t)$, respectively, and abbreviate these by $y^u(t)$, $w^u(t)$, and $e^u(t)$.

We also use the notation $x^u(t)$ to denote the column vector of *firing* flags of u's incoming neighbor neurons at time t. That is, $x^u(t) = [y^{v_1}(t)y^{v_2}(t) \dots y^{v_n}(t)]^T$, where $\{v_i\}_{i \leq n}$ are the incoming neighbors of u, which are exactly all the nodes in the layer below u.

3.4 Neuron transitions

Now we describe neuron behavior, specifically, we describe how to determine the values of the state components of each neuron u at time $t \geq 1$ based on values of state components at the previous time $t - 1$ and on external inputs. Again, we distinguish between input neurons and non-input neurons.

Input neurons: If u is an *input neuron*, then it has only one state component, the $firing$ flag. Since u is an input neuron, we assume that the value of the $firing$ flag is controlled by the network's environment and not by the network itself, that is, the value of $y^u(t)$ is set by some external input signal, which we do not model explicitly.

Non-input neurons: If u is a *non-input neuron*, then it has three state components, $firing$, $weight$, and $engaged$. Whether or not neuron u fires at time t, that is, the value of $y^u(t)$, is determined by its incoming *potential* and its *activation function*.

The potential at time t, which we denote by $pot^u(t)$ is given by the dot product of the weights and inputs at neuron u at time $t - 1$, that is,

$$pot^u(t) = w^u(t-1)^T \cdot x^u(t-1) = \sum_{j=1}^{n} w_j^u(t-1)x_j^u(t-1).$$

The activation function, which defines whether or not neuron u fires at time t, is then defined by:

$$y^u(t) = \begin{cases} 1 & \text{if } pot^u(t) \geq \tau, \\ 0 & \text{otherwise}, \end{cases}.$$

where τ is the assumed firing threshold.

We assume that the value of the $engaged$ flag of u is controlled by u's environment, that is, for every t, the value of $e^u(t)$ is set by some input signal, which may arise from outside the network or from another part of the network. For example, the $engaged$ flag could be used to ensure that, in any round, only one neuron is prepared to learn.[4] This neuron might be selected by a separate "Winner-Take-All" sub-network.

[4] We use the term "round" to represent the activity between two consecutive times. In particular, "round t" refers to the activity that takes the system from time $t - 1$ to time t. Thus, the potential in round t means the same thing as the potential at time t, captured by $pot^u(t)$.

Finally, for the weights, we assume that each neuron that is engaged at time t determines its weights at time t according to Oja's learning rule. That is, if $e^u(t) = 1$, then

$$\textit{Oja's rule}: w^u(t) = w^u(t-1) + \eta\, z(t-1) \cdot (x^u(t-1) - z(t-1) \cdot w^u(t-1)), \tag{1}$$

where η is the assumed learning rate and $z(t-1) = pot^u(t)$.[5] Thus, the weight vector is adjusted by an additive amount that is proportional to the learning rate and the potential, and depends on the input firing pattern, with a negative adjustment that depends on the potential and the prior weights.

3.5 Network operation

During execution, the network proceeds through a sequence of *configurations*, $Con(0), Con(1), Con(2), \ldots$, where $Con(t)$ describes the configuration at nonnegative integer time t. Each configuration specifies a state for every neuron in the network, that is, values for all the state components of every neuron.

As described above, the y values for the input neurons are specified by some external source. The y, w, and e values for the non-input neurons are defined by the network specification at time $t = 0$. For times $t > 0$, the y and w values are determined by the activation and learning functions described above. The e values (engagement flags) are determined by special inputs arriving from outside the network or from other sub-networks. In our algorithms in Sections 5.2 and 6.2, they will arrive from Winner-Take-All sub-networks.

4 Problem Statements

In this section we define our two main problems: *recognizing concept hierarchies*, and *learning to recognize concept hierarchies*. Our notion of recognition is robust to a bounded amount of noise. The notion of learning we define in this section corresponds to noise-free learning; we extend this to noisy learning in Section 6. In all cases, we assume that each item is represented by exactly one neuron; considering more elaborate representations is another direction for future work.

4.1 Preliminaries

Throughout this section, we fix constants $\ell_{\max}$, n, k, r_1, and r_2 according to the definitions for a concept hierarchy in Section 2. We consider a concept hierarchy $\mathcal{C}$, with concept set C and maximum level $\ell_{\max}$, partitioned as usual into $C_0, C_1, \ldots, C_{\ell_{\max}}$. We also fix constants ℓ'_{max}, n, τ, and η as in the definitions for a network in Section 3, and consider a network $\mathcal{N}$ as described earlier. Thus, we allow the maximum layer number ℓ'_{max} for $\mathcal{N}$ to be different from the maximum level number $\ell_{\max}$ for $\mathcal{C}$, but the number n of input neurons is the same as the number of level 0 items in $\mathcal{C}$.

The following definition will be useful in defining our recognition and learning problems. It expresses what it means for a particular subset B of the level 0 concepts to be "presented" as input to the network, at a certain time t.

Definition 4.1 (Presented). *If $B \subseteq D_0$ and t is a non-negative integer, then we say that B is* presented at time t *(in some particular execution) if, for every layer 0 neuron u, the following hold:*

1. *If $u \in rep(B)$ then $y^u(t) = 1$.*
2. *If $u \notin rep(B)$ then $y^u(t) = 0$.*

That is, all of the layer 0 neurons in $rep(B)$ fire at time t, and no other layer 0 neuron fires at time t.

4.2 Robust recognition

Here we define what it means for network $\mathcal{N}$ to recognize concept hierarchy $\mathcal{C}$. We assume that every concept $c \in C$, at every level, has a unique representing neuron, $rep(c)$; this extends the $rep()$ function from level 0 concepts to higher-level concepts. For this definition, we also assume that,

[5] The $z(t-1)$ notation is standard for Oja's rule, so we use that in the rest of this paper when we analyze network behavior based on this rule.

during the entire recognition process, the $engaged$ flags of all neurons are off, i.e., for every neuron u with $layer(u) > 0$, and every t, $e^u(t) = 0$.

The following definition uses the two assumed values $r_1, r_2 \in [0, 1]$, with $r_1 \leq r_2$. r_2 represents the fraction of children of a concept c at any level that should be sufficient to support firing of $rep(c)$. r_1 is a fraction below which $rep(c)$ should not fire.

Definition 4.2 (**Robust recognition problem**). *Network $\mathcal{N}$ (r_1, r_2)-*recognizes *a concept c in concept hierarchy $\mathcal{C}$ provided that $\mathcal{N}$ contains a unique neuron $rep(c)$ such that the following holds. Assume that $B \subseteq C_0$ is presented at time t.*

Then:

1. When $rep(c)$ must fire: *If $c \in supported_{r_2}(B)$, then $rep(c)$ fires at time $t + layer(rep(c))$.*
2. When $rep(c)$ must not fire: *If $c \notin supported_{r_1}(B)$, then $rep(c)$ does not fire at time $t + layer(rep(c))$.*

*We say that $\mathcal{N}$ (r_1, r_2)-*recognizes $\mathcal{C}$ *provided that it* (r_1, r_2)-recognizes *each concept c in $\mathcal{C}$.*

The special case of $(1, 1)$-recognition is interesting, since it is equivalent to the requirement that all level 0 descendants of a concept must be present for recognition:

Lemma 4.3. *Network $\mathcal{N}$ $(1, 1)$-recognizes a concept c in concept hierarchy $\mathcal{C}$ if and only if $\mathcal{N}$ contains a unique neuron $rep(c)$ such that the following holds. If $B \subseteq D_0$ is presented at time t, then $rep(c)$ fires at time $t + layer(rep(c))$ if and only if $leaves(c) \subseteq B$.*

Proof. By the definition of the robust recognition problem and Lemma 2.2. □

4.3 Noise-free learning

In the learning problem, the network does not know ahead of time which particular concept hierarchy might be presented in a particular execution. It must be capable of learning *any* concept hierarchy.

In our algorithm in Section 5.2, in order for the network to learn a concept hierarchy $\mathcal{C}$, it must receive inputs corresponding to all the concepts in C. Here we define how individual concepts are "shown" to the network, and then give constraints on the order in which the concepts are shown. Such constraints are captured by the notion of a *bottom-up training schedule*. Then we state our learning guarantees, assuming a bottom-up training schedule for $\mathcal{C}$.

We begin by describing how an individual concept c is "shown" to the network. Recall that $leaves(c)$ is defined to be $descendants(c) \cap C_0$.

Definition 4.4 (**Showing a concept**). *Concept c is* shown *at time t provided that the set $B = leaves(c)$ is presented at time t. That is, for every input neuron u, $y^u(t) = 1$ if and only if $u \in rep(leaves(c))$.*

Learning a concept hierarchy will involve showing all the concepts in the hierarchy. Informally speaking, we assume that the concepts are shown "bottom-up". For example, before the network is shown the concept of a head, it is shown the lower-level concepts of mouth, eye, etc. And before it is shown the concept of a human, it is shown the lower-level concepts of head, body, legs, etc. More precisely, to enable network $\mathcal{N}$ to learn the concept hierarchy $\mathcal{C}$, we assume that every concept in its concept set C is shown at least σ times, where σ is a parameter to be specified by a learning algorithm. Furthermore, we assume that any concept $c \in C$ is shown only after each child of c has been shown at least σ times. We allow the concepts to be shown in an arbitrary order and in an interleaved manner, provided that these constraints are observed.

Definition 4.5 (**σ-bottom-up training schedule**). *A* training schedule *for $\mathcal{C}$ is any finite list $c_0, c_1, \ldots, c_m$ of concepts in C, possibly with repeats. A training schedule is σ-*bottom-up*, where σ is a positive integer, provided that each concept in C appears in the list at least σ times, and no concept in C appears before each of its children has appeared at least σ times.*

Any training schedule $c_0, c_1, \ldots, c_m$ generates a corresponding sequence $B_0, B_1, \ldots, B_m$ of sets of level 0 concepts to be presented in a learning algorithm. Namely, B_i is defined to be $rep(leaves(c_i))$.

Definition 4.6 (**(r_1, r_2, σ)-learning**)**.** *Network $\mathcal{N}$ (r_1, r_2, σ)-learns concept hierarchy $\mathcal{C}$ provided that the following holds. At any time after a training phase in which all the concepts of $\mathcal{C}$ are shown according to a σ-bottom-up training schedule, network $\mathcal{N}$ (r_1, r_2)-recognizes $\mathcal{C}$.*

5 Algorithms for Recognition and Noise-Free Learning

We give algorithms for both of the problems described in Section 4.

5.1 Recognition

Fix a concept hierarchy $\mathcal{C}$ with concept set C, and $r_1, r_2 \in [0, 1]$, with $r_1 \leq r_2$. Recognition can be achieved by simply embedding the digraph induced by $\mathcal{C}$ in the network $\mathcal{N}$. See Figure 1 for an illustration. For every ℓ and for every level ℓ concept c of $\mathcal{C}$, we designate a unique representative $rep(c)$ in layer ℓ of the network. Let R be the set of all representatives, that is, $R = rep(C) = \{rep(c) \mid c \in C\}$. We use rep^{-1} with support R to denote the corresponding inverse function that gives, for every $u \in R$, the unique concept $c \in C$ with $rep(c) = u$.

If u is a layer ℓ neuron and v is a layer $\ell + 1$ neuron, then we define the edge weight $weight(u, v)$ by:

$$weight(u, v) = \begin{cases} 1 & \text{if } rep^{-1}(v) \in children(rep^{-1}(u)), \\ 0 & \text{otherwise.} \end{cases}.$$

That is, we define the weights of edges corresponding to child relationships in the concept hierarchy to be 1, and the weights of other edges to be 0.

Finally, we set the threshold τ for every non-input neuron to be $\frac{(r_1+r_2)k}{2}$. It should be clear that the resulting network $\mathcal{N}$ solves the (r_1, r_2)-recognition problem:

Theorem 5.1. *Network $\mathcal{N}$ (r_1, r_2)-recognizes $\mathcal{C}$.*

Recall that the definition of recognition, Definition 4.2 says that each individual concepts c in the hierarchy is recognized. For a level ℓ concept c, the definition includes a time bound of $layer(rep(c)) = level(c) = \ell$ for recognizing concept c.

We note that our choice of weights in $\{0, 1\}$ here is for simplicity. Other combinations are possible, and in fact, our learning algorithm below results in different weights, approximating $\frac{1}{\sqrt{k}}$ and 0.

5.2 Noise-free learning

Now we move from the simple recognition problem to the harder problem of learning. Now we must design a network $\mathcal{N}$ that can learn an arbitrary concept hierarchy $\mathcal{C}$ with parameters as listed in Section 2 and Section 3, and with $\ell_{\max} \leq \ell'_{max}$. Our algorithm utilizes *Winner-Take-All (WTA)* sub-networks [21, 53, 46, 4, 32, 51, 37, 24].

Winner-Take-All sub-networks: Our algorithm uses Winner-Take-All sub-networks to select which neurons are prepared to learn at different points during the learning process. In this paper, we abstract from these sub-networks by simply describing their effects on the $engaged$ flags in the non-input neurons. We give the precise requirements in Assumption 5.2.

While the network is being trained, example concepts are "shown" to the network, one example at each time t, according to a σ-bottom-up training schedule as defined in Section 4.3. We assume that, for every example concept c that is shown, exactly one neuron at the appropriate layer will be engaged; this layer is the one with the same number as the level of c in the concept hierarchy. Furthermore, the neuron on that layer that is engaged is the one that has the largest potential pot^u. More precisely, in terms of timing, we assume:

Assumption 5.2 (**Winner-Take-All assumption**)**.** *If a level ℓ concept c is "shown" at time t, then at time $t + \ell$, exactly one layer ℓ neuron u has its* engaged *state component equal to* 1*, that is, it has $e^u(t + \ell) = 1$. Moreover, u is chosen so that $pot^u(t + \ell)$ is the highest potential at time $t + \ell$ among all the layer ℓ neurons.*

Main algorithm: We assume that the network $\mathcal{N}$ starts in a clean state in which, for every neuron u in layer 1 or higher, $w^u(0) = \frac{1}{k^{\ell_{\max}+1}}\mathbf{1}$, where $\mathbf{1}$ is the n-dimensional all-one vector. We set the threshold τ for all neurons to be $\frac{(r_1+r_2)\sqrt{k}}{2}$, and the learning rate η to be $\frac{1}{4k}$. The initial condition, threshold, learning rate, Assumption 5.2, and the general model conventions for activation and learning suffice to determine how the network behaves, when shown a particular series of concepts. Our main result is:

Theorem 5.3 (**Noise-Free Learning Theorem**). *Let $\mathcal{N}$ be the network described above, with maximum layer ℓ'_{max}. Let b be an arbitrary positive real ≥ 2. Let r_1, r_2 be reals with $0 < r_1 < r_2 \leq 1$; assume that $r_1 k$ is not an integer, and $r_1 k - \lfloor r_1 k \rfloor \geq \frac{\sqrt{k}}{k^{b-1}}$. Also assume that r_2 and k satisfy the inequality $\frac{1}{\sqrt{k}} + \frac{1}{k} \leq \frac{r_2\sqrt{k}}{2}$.* [6] *Let $\varepsilon = \frac{r_2 - r_1}{r_1 + r_2}$.*

Let $\mathcal{C}$ be any concept hierarchy, with maximum level $\ell_{\max} \leq \ell'_{max}$. Let $\sigma = \frac{4}{3\eta k}((\ell_{\max}+1)\log(k)) + \frac{3}{\eta k \varepsilon} + \frac{b\log(k)}{\log(\frac{16}{15})}$. Thus, σ is $O\left(\frac{1}{\eta k}\left(\ell_{\max}\log(k) + \frac{1}{\varepsilon}\right) + b\log(k)\right)$.

Then $\mathcal{N}$ (r_1, r_2, σ)-learns concept hierarchy $\mathcal{C}$.

That is, unwinding the definition of (r_1, r_2, σ)-learning, at any time after a training phase in which all the concepts of $\mathcal{C}$ are shown according to a σ-bottom-up training schedule, network $\mathcal{N}$ (r_1, r_2)-recognizes $\mathcal{C}$.

A rigorous analysis can be found in Appendix A; the main idea of the analysis is as follows. We first prove some direct consequences of Oja's rule (Lemma A.1, Lemma A.2, and Lemma A.3). These quantify the weight changes for a single neuron involved in learning a single concept, assuming that all of its child concepts have already been learned. In particular, we show that the weights change quickly so that they approximate either $1/\sqrt{k}$ or 0, depending on whether or not the weights correspond to neurons that represent child concepts.

We next build on these lemmas to describe, in Lemma A.6, the learning (i.e., weight changes) that occur throughout the network in the course of the entire execution. What makes this challenging is that we allow "incomparable" concepts to be shown in an interleaved manner; the only constraint is that, for every concept c, child concepts of a concept c must be shown sufficiently many times before c is shown. In order to prove that all concepts are learned correctly despite these challenges, we use an involved yet elegant five-part induction. Finally, in Section A.3 we put everything together and show that the network successfully (r_1, r_2, σ)-learns the concept hierarchy.

6 Extension to Noisy Learning

We extend our model, algorithm, and analysis to noisy learning. The idea is that we should be able to learn a concept even if we do not see all the child concepts at every time. For example, we could expect to learn the concept of a "human" even if we sometimes see only the "legs" and "body", and other times see only the "head" and "legs" etc.

To model this, we assume that, in order to show a concept c, we show a random p-fraction of its sub-concepts. Formally, we use the following recursive marking procedure to determine which inputs should be presented to the network: We begin by marking c. Then, proceeding recursively, for any marked concept, we mark a random p-fraction of the sub-concepts. The recursion terminates when a subset of the leaves of c are marked. The inputs presented to the network are the representations of the marked leaves of c.

6.1 Modifications to the model

Formally, our model is as follows. Recall that in Definition 4.4, we assumed that when a concept c is shown, that *all reps* of the leaves of c fire. We now weaken this assumption, as follows.

Definition 6.1 (**p-noisy-showing a concept**). *Concept c is p-*noisy-shown *at time t, where $p \in (0,1]$, provided that a subset $B \subseteq leaves(c)$ produced by the random function $mark(c,p)$ is presented at*

[6] This last assumption can be satisfied by a variety of different combinations of assumptions on r_2 and k individually, such as $r_2 \geq \frac{1}{2}$ and $k \geq 6$, or $r_2 \geq \frac{1}{4}$ and $k \geq 11$.

time t.

Random function $mark(c,p)$ is defined recursively based on the level of c: If $level(c) = 0$, then $mark(c,p) = \{c\}$. If $level(c) \geq 1$, then choose a subset C' consisting of exactly $\lceil pk \rceil$ children of c, uniformly at random, and let $mark(c,p) = \bigcup_{c' \in C'} mark(c',p)$.

In the noisy case, we need an upper bound (σ_2 in the following definition) on the number of times a concept is noisy-shown. See the discussion in the footnote before Theorem 6.4 for more details.

Definition 6.2 (**(σ_1, σ_2)-bottom-up training schedule**). *A training schedule is* (σ_1, σ_2)-bottom-up*, where σ_1 and σ_2 are positive integers, $\sigma_1 \leq \sigma_2$, provided that each concept in C appears in the list at least σ_1 times and no more than σ_2 times, and no concept in C appears before each of its children has appeared at least σ_1 times.*

Definition 6.3 (**($r_1, r_2, \sigma_1, \sigma_2, p$)-noisy learning**). *Network $\mathcal{N}$* ($r_1, r_2, \sigma_1, \sigma_2, p$)-noisy-learns *concept hierarchy $\mathcal{C}$ provided that the following holds. At any time after a training phase in which all the concepts of $\mathcal{C}$ are p-noisy-shown according to a (σ_1, σ_2)-bottom-up training schedule, network $\mathcal{N}$ (r_1, r_2)-recognizes $\mathcal{C}$.*

6.2 Noisy Learning Algorithm

The algorithm is exactly the same as in Section 5.2, except that here we use p-noisy showing (Definition 6.1) instead of ordinary showing (Definition 4.4). We prove that our modified algorithm is robust in that it works even for our notions of noisy showing and noisy learning.

Our theorem for noisy learning, Theorem 6.4, differs from Theorem 5.3 in that we guarantee "correctness" only in cases where each concept is noisy-shown at most n^6 times, that is, in cases where the network $(r_1, r_2, \sigma, n^6, p)$-noisy learns the concept hierarchy. [7] Let $\bar{w} = 1/\sqrt{pk + 1 - p}$. Our algorithm uses the learning rate $\eta = \frac{(\frac{\delta p \bar{w}}{20})^3}{64 T k^2 p^3}$ and the firing threshold $\tau = r_2 k(\bar{w} - 2\delta)$, where $\delta = \bar{w}(r_2 - r_1)/50$.

We now state our main theorem in the noisy-learning setting.

Theorem 6.4 (**Noisy-Learning Theorem**). *Let $\mathcal{N}$ be the network described in Section 3, with maximum layer ℓ'_{max}. Let r_1, r_2 be reals with $0 < r_1 < r_2 \leq 1$; assume that $r_2 - r_1 \geq 1/k$ and $k \geq 2$. Let $\mathcal{C}$ be any concept hierarchy, with maximum level $\ell_{\max} \leq \ell'_{max}$ and a total of $|C|$ concepts. Let $\sigma = c' \frac{k^6}{p^6 \delta^3} \left(\ell_{\max} \log(k) + \log(|C| n/\delta)\right)$, for some large enough constant c'.*

Then, w.h.p., $\mathcal{N}$ $(r_1, r_2, \sigma, n^6, p)$-noisy-learns concept hierarchy $\mathcal{C}$.[8]

6.3 Proof idea

In the presence of noise, many of the properties of the noise-free case no longer hold, rendering the proof significantly more involved. Here we give a rough outline of our proof; details appear in Appendix B.

In the analysis we only consider the learning of one concept, as the interleaving of different concepts is no different than in the noise-free case and hence we do not repeat that analysis. Therefore, in the reminder we fix one concept.

First, we bound the worst-case change of potential during a period of T rounds (where the concept is shown), provided it is initially within certain bounds. We later show that it will stay throughout the first n^6 rounds where the concept is shown.

We aim to derive bounds on the change of the weight of a single edge during such a period. It turns out that the way the weights change depends highly on the other weights, which makes

[7] Note that we assume that every concept is shown at most n^6 times. This is natural since if we consider a number T of rounds that is of order exponential in n, then at some point $t \leq T$ it is very likely that the weights will be unfavorable for recognition. This can happen since in such a large time frame, it's very likely that there will be a long sequence of runs in which the same representatives are simply (due to bad luck) not shown. The network will forget about their importance. This is also partly the reason why the learning rate in the following theorem is smaller than the one of the noise-free counterpart: the smaller learning rate guarantees that during the first n^6 rounds no unlikely sequence occurs that is very 'bad'.

[8] We define w.h.p in this paper to be $1 - \frac{1}{n}$.

the analysis non-trivial. For this reason, we refrain from showing convergence of each weight separately. Instead we use the following potential function ψ. to show that the max and min weight convergence towards $\bar{w} = \frac{1}{\sqrt{pk+1-p}}$ and 0 respectively. Fix an arbitrary time t and let $w_{min}(t)$ and $w_{max}(t)$ be the minimum and maximum weights among $w_1(t), w_k(t), \ldots, w_k(t)$, respectively. Let $\psi(t) = \max\left\{\frac{w_{max}(t)}{\bar{w}}, \frac{\bar{w}}{w_{min}(t)}\right\}$.

Note that, in contrast to the noise-free case, weights belonging to representatives of sub-concepts converge to $\bar{w}$ instead to $1/\sqrt{k}$.

Our goal is to show that the above potential decreases quickly until it is very close to 1. Showing that the potential decreases is involved, since one cannot simply use a worst-case approach, due to the terms in Oja's rule being non-linear and potentially having a high variance, depending on the distribution of weights. Instead, the key to showing that ψ decreases is to carefully use the randomness over the input vector and to carefully bound the non-linear terms. Bounding these non-linear terms tightly presents a major challenge. To overcome it, we show that the changes of the weights form a Doob martingale allowing us to use Azuma-Hoeffding inequality to get asymptotically almost tight bounds on the change of the weights during the T rounds. The proof can be found in Appendix B.

7 A Lower Bound

Our results so far demonstrate how concept hierarchies with $\ell_{\max}$ levels can be represented robustly by networks with the same number of layers, and how such representations can be learned, even in the presence of noise. We would also like lower bound theorems saying that $\ell_{\max}$ layers are necessary for robust representation, under suitable restrictions.

In this section, we give a first step toward such a result, Theorem 7.1. It says that a network $\mathcal{N}$ with maximum layer 1 cannot recognize a concept hierarchy $\mathcal{C}$ with maximum level 2. This bound depends only on the requirement that $\mathcal{N}$ should recognize $\mathcal{C}$ according to our definition for noisy recognition in Definition 4.2. That definition says that the network must tolerate bounded noise, as expressed by the ratio parameters r_1 and r_2. Our result assumes reasonable constraints on the values of r_1 and r_2. Note that the bound does not involve learning, only recognition.

A preliminary generalization of this result to more levels and layers appears in [26]. However, in addition to the basic definition of noisy recognition, this generalization uses a strong technical assumption about disjointness of certain sets of triggered neurons. This assumption might be reasonable, in that it is guaranteed by our learning algorithms in Section 5.2; however, we think it is too strong and would prefer to weaken it to, say, a simple limitation on the number of neurons at each layer in the network. We leave this task for future work.

7.1 Assumptions for the lower bound

Here we list explicitly the assumptions that we use for our lower bound result, Theorem 7.1. We state these assumptions in a general way, in terms of a particular concept hierarchy $\mathcal{C}$ with concept set C and any number $\ell_{\max}$ of levels, and an arbitrary network $\mathcal{N}$ with any number ℓ'_{max} of layers. However, our lower bound result, Theorem 7.1, refers to just the special case of two levels and one layer. These assumptions capture the idea that concept hierarchy $\mathcal{C}$ is (r_1, r_2)-recognized by network $\mathcal{N}$.

1. Every concept $c \in C$ has a unique designated neuron $rep(c)$ in the network. (In general, it might be in any layer, regardless of the level of c.)
2. Let B be any subset of C_0. If $c \in supported_{r_2}(B)$, then presentation of B at time t results in firing of $rep(c)$ at time $t + layer(rep(c))$.
3. Let B be any subset of C_0. If $c \notin supported_{r_1}(B)$, then presentation of B at time t does not result in firing of $rep(c)$ at time $t + layer(rep(c))$.

Throughout this section, we assume the model presented in Section 2 and Section 3. Furthermore, since we are considering recognition only, and not learning, we assume that the *engaged* state components are always equal to 0. Also throughout this section, we assume that r_1 and r_2 satisfy the following constraints:

1. $0 \leq r_1 \leq r_2 \leq 1$.
2. $r_1 k$ is not an integer; define r'_1 so that $r'_1 k = \lfloor r_1 k \rfloor$.
3. Define r'_2 so that $r'_2 k = \lceil r_2 k \rceil$.
4. $(r'_2)^2 \leq 2r'_1 - (r'_1)^2$.

we think these constraints are reasonable. For example, for $k = 10$, $r_1 = .51$ and $r_2 = .8$ satisfy these conditions. Or $r_1 = \frac{1}{3}$ and $r_2 = \frac{2}{3}$.

7.2 Impossibility for recognition for two levels and one layer

We consider an arbitrary concept hierarchy $\mathcal{C}$ with maximum level 2 and concept set C. We assume a (static) network $\mathcal{N}$ with maximum layer 1, and total connectivity from layer 0 neurons to layer 1 neurons. For such a network and concept hierarchy, we get a contradiction to the noisy recognition problem in Section 4.2, for any values of r_1 and r_2 that satisfy the constraints given in Section 7.1. For the problem requirements, we use only Assumptions 1-3 from Section 7.1.

Theorem 7.1. *Assume that $\mathcal{C}$ has maximum level 2 and $\mathcal{N}$ has maximum layer 1. Assume that r_1, r_2, r'_1, r'_2 satisfy the constraints in Section 7.1. Then $\mathcal{N}$ does not recognize $\mathcal{C}$, according to Assumptions 1-3.*

Proof. Assume for contradiction that $\mathcal{N}$ recognizes $\mathcal{C}$. Let c denote any one of the concepts in C_2, i.e., a level 2 concept in C. Then c has k children, each of which has k children of its own, for a total of k^2 grandchildren.

Each of the k^2 grandchildren must have a rep in layer 0, but neither c nor any of its k children do, because layer 0 is reserved for level 0 concepts. So in particular, $rep(c)$ is a layer 1 neuron. By the structure of the network, this means that the only inputs to $rep(c)$ are from layer 0 neurons. Since we assume total connectivity, we have an edge from each layer 0 neuron to $rep(c)$. We define:

- $W(b)$, for each child b of c in the concept hierarchy: The total weight of all edges $(u, rep(c))$, where u is a layer 0 neuron that is the rep of a child of b.
- W: The total weight of all the edges $(u, rep(c))$, where u is a layer 0 neuron that is a rep of a grandchild of c. In other words, $W = \Sigma_{b \in children(c)} W(b)$.

We consider two scenarios. In Scenario A (the "must-fire scenario"), we choose input set B to consist of enough leaves of c to force $rep(c)$ to fire, that is, we ensure that $c \in supported_{r_2}(B)$, while trying to minimize the total weight incoming to $rep(c)$. Specifically, we choose the $r'_2 k \geq r_2 k$ children b of c with the smallest values of $W(b)$. And for each such b, we choose its $r'_2 k$ children with the smallest weights. Let B be the union of all of these $r'_2 k$ sets of $r'_2 k$ grandchildren of c. Since $r'_2 k \geq r_2 k$, it follows that $c \in supported_{r_2}(B)$.

Claim 1: In Scenario A, the total incoming potential to $rep(c)$ is at most $(r'_2)^2 W$.

In Scenario B (the "can't-fire scenario"), we choose input set B to consist of leaves of c that force $rep(c)$ not to fire, that is, we ensure that $c \notin supported_{r_1}(B)$, while trying to maximize the total weight incoming to $rep(c)$. Specifically, we choose the $r'_1 k < r_1 k$ children b of c with the largest values of $W(b)$, and we include all of their children in B. For each of the remaining $(1 - r'_1)k$ children of c, we choose its $r'_1 k < r_1 k$ children with the largest weights and include them all in B. Since $r'_1 k$ is strictly less than $r_1 k$, it follows that $c \notin supported_{r_1}(B)$.

Claim 2: In Scenario B, the total incoming potential to $rep(c)$ is at least $(r'_1)W + (1 - r'_1)r'_1 W = (2r'_1 - (r'_1)^2)W$.

Proof of Claim 2: We define:

- W_1: The total of the weights $W(b)$ for the $r'_1 k$ children b of c with the largest values of $W(b)$.
- $W_2 = W - W_1$: The total of the weights $W(b)$ for the remaining $(1 - r'_1)k$ children of c.

- W_3: We know that $W_1 \geq r_1' W$, since W_1 gives the total weight for the $r_1' k$ children of c with the largest weights, out of k children. Define $W_3 = W_1 - r_1' W$; then W_3 must be nonnegative.

Then the total incoming potential to $rep(c)$ is

$$\begin{aligned}
&\geq W_1 + r_1' W_2, \\
&= r_1' W + W_3 + r_1'(W - W_1), \\
&= r_1' W + W_3 + r_1'(W - W_3 - r_1' W), \\
&= 2r_1' W - (r_1')^2 W + (1 - r_1') W_3, \\
&\geq 2r_1' W - (r_1')^2 W, \\
&= (2r_1' - (r_1')^2) W,
\end{aligned}$$

as needed.

End of proof of Claim 2

Now, Claim 1 implies that the threshold τ of neuron $rep(c)$ must be at most $(r_2')^2 W$, since it must be small enough to permit the given B to trigger firing of $rep(c)$. On the other hand, Claim 2 implies that the threshold must be strictly greater than $(2r_1' - (r_1')^2)W$, since it must be large enough to prevent the given B from triggering firing of $rep(c)$. So we must have

$$(2r_1' - (r_1')^2)W < \tau \leq (r_2')^2 W,$$

which implies that

$$2r_1' - (r_1')^2 < (r_2')^2.$$

But this contradicts our assumption that $(r_2')^2 \leq 2r_1' - (r_1')^2$. □

8 Conclusions and Future Work

In this paper, we have proposed a theoretical model for recognizing and learning hierarchically-structured concepts in synchronous, feed-forward layered Spiking Neural Networks. Our networks use Oja's learning rule for adjusting synapse weights. Based on this model, we have presented two learning algorithms, one for noise-free learning and one that allows bounded noise. Both algorithms learn concepts in a bottom-up manner, but allow arbitrary interleaving in learning of incomparable concepts. We have analyzed both algorithms in detail.

The representations produced by these algorithms are certain types of embeddings of the hierarchical concept structure in the neural network. These representations support robust concept recognition, even when some of the inputs are missing. We have also provided a preliminary lower bound on the number of layers, saying that two-level concepts cannot be recognized robustly in one-level networks.

This paper represents a first step towards a theory of representation and learning for hierarchically-structured concepts in SNNs. In the longer term, we are interested in theoretical models that capture key features of real computer vision algorithms and brain networks. Our current model is highly abstract and makes many simplifying assumptions: for instance, we assume that concepts are strictly tree-structured, that every concept has the same number of children, that the number of network layers is at least as large as the number of concept levels, that the networks are feed-forward, and that the learning rule is applied without error. To make the results more realistic, one should loosen all of these these assumptions, systematically.

The results in this paper suggest numerous directions for future research:

Extensions to our results: One can consider more flexible orders in which concepts in a hierarchy can be learned, based on a larger class of training schedules. Is it possible to learn higher-level concepts before learning low-level concepts? How does the order of learning affect the time required to learn? Another interesting issue is robustness of the networks, for example, to presentation of a few "extraneous" inputs that are not part of the concept being shown, to noise in calculating potentials, or to failures of neurons or synapses.

Also, our algorithms use some auxiliary capabilities, such as Winner-Take-All, in order to select neurons for learning; it would be interesting to combine our algorithms with network implementations of these auxiliary capabilities in order to obtain complete, self-contained networks that solve the learning problem "from scratch". Finally, we would like to strengthen the lower bound results to apply to many levels and layers.

Variations in the network model: Our networks have a simple layered structure; it would be interesting to consider some natural variations. For example, instead of all-to-all connections between consecutive layers, what happens to the results if one assumes a smaller number of randomly-determined connections between layers? Also, in our networks, all edges go from one layer ℓ to the next higher layer $\ell + 1$. How do the results change if one allows edges to go from layer ℓ to any higher layer?

What would be the impact on the results of allowing feedback edges from each layer ℓ to the next-lower layer $\ell - 1$? How would the costs of recognizing and learning concepts change based on feedback from representations of higher-level concepts?

What would be the effect on the results of using other incremental learning rules besides Oja's rule? In an extreme case, what happens to the results if learning occurs all at once, rather than incrementally? In general, how can we compare the computational power of incremental learning models vs. one-shot learning models such as the one in [52, 20]?

Variations in the data model: Another interesting research direction is to consider variations on the structure of concept hierarchies. How do the results change if we allow different numbers of children for different concepts? It is not clear how one can set the firing thresholds in this case. Perhaps these thresholds could be 'learned'. Another interesting extension is to allow a level ℓ concept to have children at any level smaller than ℓ, rather than just level $\ell - 1$? What happens if a concept hierarchy need not be a tree, but may include a bounded amount of overlap between the sets of children of different concepts?

It would be interesting to understand more generally what kinds of logical structures can be learned by synchronous SNNs. In our concept hierarchies, each level $\ell + 1$ concept corresponds to the "and" of several level ℓ concepts. What if we allow concepts that correspond to "ors", or "nors", of other concepts? Similar questions were suggested by Valiant [47], in terms of a different model. Also, in addition to learning individual concepts, it would be interesting to consider learning relationships between concepts, such as association, causality, or sequential order.

Different forms of representation: In this paper, each concept c is represented by just one neuron $rep(c)$. An interesting extension, which may be more biologically plausible, would be to allow the representation of each concept c to be a more elaborate "code" consisting of a particular set of neurons that fire. Important examples here are representations based on "cell assemblies" [38, 39]. What are the theoretical advantages and costs of such codes, compared to simpler single-neuron representations? Another type of extension would be to "time-share" the network, allowing the same layer of the network to represent different levels of the concept hierarchy at different times. Ideas from [6, 19] on state machine simulations in neural networks may be useful here.

Experimental work: All the ideas we have presented in this paper are purely theoretical. It would be valuable to complement this work with experiments to evaluate the performance and robustness of the algorithms presented here, as well as future algorithms.

References

[1] Alain Artola, S Bröcher, and Wolf Singer. Different voltage-dependent thresholds for inducing long-term depression and long-term potentiation in slices of rat visual cortex. *Nature*, 347(6288):69, 1990.

[2] Alain Artola and Wolf Singer. Long-term depression of excitatory synaptic transmission and its relationship to long-term potentiation. *Trends in neurosciences*, 16(11):480–487, 1993.

[3] Elie L Bienenstock, Leon N Cooper, and Paul W Munro. Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex. *Journal of Neuroscience*, 2(1):32–48, 1982.

[4] Robert Coultrip, Richard Granger, and Gary Lynch. A cortical model of winner-take-all competition via lateral inhibition. *Neural Networks*, 5(1):47–54, 1992.

[5] Devdatt P Dubhashi and Alessandro Panconesi. *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press, 2009.

[6] Rebecca Fay, Ulrich Kaufmann, Andreas Knoblauch, Heiner Markert, and Günther Palm. Combining visual attention, object recognition and associative information processing in a neurobotic system. In *Biomimetic neural learning for intelligent robots*, pages 118–143. Springer, 2005.

[7] Daniel J Felleman and DC Essen Van. Distributed hierarchical processing in the primate cerebral cortex. *Cerebral cortex (New York, NY: 1991)*, 1(1):1–47, 1991.

[8] Peter Földiák and Peter Fdilr. Adaptive network for optimal linear feature extraction. 1989.

[9] Wulfram Gerstner and Werner M. Kistler. *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge University Press, 2002.

[10] Wulfram Gerstner, Marco Lehmann, Vasiliki Liakoni, Dane Corneil, and Johanni Brea. Eligibility traces and plasticity on behavioral time scales: experimental support of neohebbian three-factor learning rules. *Frontiers in neural circuits*, 12:53, 2018.

[11] Stefan Habenschuss, Zeno Jonke, and Wolfgang Maass. Stochastic computations in cortical microcircuit models. *PLoS Computational Biology*, 9(11):e1003311, 2013.

[12] D. O. Hebb. *The Organization of Behavior*. Wiley and Sons, New York, 1949.

[13] Yael Hitron, Nancy A. Lynch, Cameron Musco, and Merav Parter. Random sketching, clustering, and short-term memory in spiking neural networks. In *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*, pages 23:1–23:31, 2020. URL: https://doi.org/10.4230/LIPIcs.ITCS.2020.23, doi:10.4230/LIPIcs.ITCS.2020.23.

[14] D. Hubel and T. Wiesel. Receptive fields, binocular interaction, and functional architecture in the cat's visual cortex. *Journal of Physiology*, 160:106–154, 1962.

[15] D. H. Hubel and T. N. Wiesel. Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, 148(3):574–591, 1959. URL: https://physoc.onlinelibrary.wiley.com/doi/abs/10.1113/jphysiol.1959.sp006308, arXiv:https://physoc.onlinelibrary.wiley.com/doi/pdf/10.1113/jphysiol.1959.sp006308, doi:10.1113/jphysiol.1959.sp006308.

[16] JM Hupé, AC James, BR Payne, SG Lomber, P Girard, and J Bullier. Cortical feedback improves discrimination between figure and background by v1, v2 and v3 neurons. *Nature*, 394(6695):784, 1998.

[17] Eugene M. Izhikevich. Which model to use for cortical spiking neurons? *IEEE Transactions on Neural Networks*, 15(5):1063–1070, 2004.

[18] Richard Kempter, Wulfram Gerstner, and J Leo Van Hemmen. Hebbian learning and spiking neurons. *Physical Review E*, 59(4):4498, 1999.

[19] Andreas Knoblauch, Heiner Markert, and Günther Palm. An associative cortical model of language understanding and action planning. In *International work-conference on the interplay between natural and artificial computation*, pages 405–414. Springer, 2005.

[20] Andreas Knoblauch, Günther Palm, and Friedrich T Sommer. Memory capacities for synaptic and structural plasticity. *Neural Computation*, 22(2):289–341, 2010.

[21] John Lazzaro, Sylvie Ryckebusch, Misha Anne Mahowald, and Carver A. Mead. Winner-take-all networks of $o(n)$ complexity. Technical report, DTIC Document, 1988.

[22] Robert A. Legenstein, Wolfgang Maass, Christos H. Papadimitriou, and Santosh S. Vempala. Long term memory and the densest k-subgraph problem. In *9th Innovations in Theoretical Computer Science (ITCS 2018)*, pages 57:1–57:15, Cambridge, MA, January 2018.

[23] S. Lowel and W. Singer. Selection of intrinsic horizontal connections in the visual cortex by correlated neuronal activity. *Science Magazine*, 255(5041):209–212, January 1992.

[24] Nancy Lynch, Cameron Musco, and Merav Parter. Computational tradeoffs in biological neural networks: Self-stabilizing winner-take-all networks. In *ITCS 2017*, 2017. Full version available at https://arxiv.org/abs/1610.02084.

[25] Nancy Lynch, Cameron Musco, and Merav Parter. Winner-take-all computation in spiking neural networks, April 2019. arXiv:1904.12591.

[26] Nancy A. Lynch and Frederik Mallmann-Trenn. Learning hierarchically structured concepts. *CoRR*, abs/1909.04559v3, 2019. URL: http://arxiv.org/abs/1909.04559v3, arXiv:1909.04559v3.

[27] Nancy A. Lynch and Cameron Musco. A basic compositional model for spiking neural networks. *CoRR*, abs/1808.03884, 2018. URL: http://arxiv.org/abs/1808.03884, arXiv:1808.03884.

[28] Nancy A. Lynch, Cameron Musco, and Merav Parter. Computational tradeoffs in biological neural networks: Self-stabilizing winner-take-all networks. In *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, pages 15:1–15:44, 2017. URL: https://doi.org/10.4230/LIPIcs.ITCS.2017.15, doi:10.4230/LIPIcs.ITCS.2017.15.

[29] Wolfgang Maass. On the computational power of noisy spiking neurons. In *NIPS1996*, 1996.

[30] Wolfgang Maass. Networks of spiking neurons: the third generation of neural network models. *Neural Networks*, 10(9):1659–1671, 1997.

[31] Wolfgang Maass. Neural computation with winner-take-all as the only nonlinear operation. In *NIPS 1999*, pages 293–299, 1999.

[32] Wolfgang Maass. On the computational power of winner-take-all. *Neural Computation*, 2000.

[33] Nikola T Markov, Julien Vezoli, Pascal Chameau, Arnaud Falchier, René Quilodran, Cyril Huissoud, Camille Lamy, Pierre Misery, Pascale Giroud, Shimon Ullman, et al. Anatomy of hierarchy: feedforward and feedback pathways in macaque visual cortex. *Journal of Comparative Neurology*, 522(1):225–259, 2014.

[34] Hrushikesh Mhaskar, Qianli Liao, and Tomaso Poggio. Learning functions: when is deep better than shallow. *arXiv preprint arXiv:1603.00988*, 2016.

[35] Erkki Oja. Simplified neuron model as a principal component analyzer. *Journal of mathematical biology*, 15(3):267–273, 1982.

[36] Erkki Oja. Principal components, minor components, and linear neural networks. *Neural networks*, 5(6):927–935, 1992.

[37] Matthias Oster and Shih-Chii Liu. Spiking inputs to a winner-take-all network. In *NIPS 2006*, page 1051, 2006.

[38] Günther Palm. *Neural assemblies: An alternative approach to artificial intelligence*, volume 7. Springer Science & Business Media, 2012.

[39] Günther Palm, Andreas Knoblauch, Florian Hauser, and Almut Schüz. Cell assemblies in the cerebral cortex. *Biological cybernetics*, 108(5):559–572, 2014.

[40] Christos H. Papadimitriou and Santosh S. Vempala. Cortical learning via prediction. *Proceedings of Machine Learning Research (PMLR)*, 40:1402–1422, 2015.

[41] Christos H. Papadimitriou and Santosh S. Vempala. Random projection in the brain and computation with assemblies of neurons. In *10th Innovation in Theoretical Computer Science (ITCS 2019)*, pages 57:1–57:19, San Diego, CA, January 2019.

[42] Christos H. Papadimitriou, Santosh S. Vempala, Daniel Mitropolsky, Michael Collins, and Wolfgang Maass. Brain computation by assemblies of neurons, December 2019. bioRxiv:10.1101/869156v1.

[43] Maximilian Riesenhuber and Tomaso Poggio. Hierarchical models of object recognition in cortex. *Nature neuroscience*, 2(11):1019–1025, 1999.

[44] Sven Schrader, Markus Diesmann, and Abigail Morrison. A compositionality machine realized by a hierarchic architecture of synfire chains. *Frontiers in Computational Neuroscience*, 4:154, 2011. URL: https://www.frontiersin.org/article/10.3389/fncom.2010.00154, doi:10.3389/fncom.2010.00154.

[45] Lili Su and Chia-Jung Chang amd Nancy Lynch. Spike-based winner-take-all computation: Fundamental limits and order-optimal circuits. *Neural Computation*, 31(12), December 2019. Published online. Also, arXiv:1904.10399.

[46] Simon J. Thorpe. Spike arrival times: A highly efficient coding scheme for neural networks. *Parallel Processing in Neural Systems*, pages 91–94, 1990.

[47] Leslie G. Valiant. *Circuits of the Mind*. Oxford University Press on Demand, 2000.

[48] Leslie G. Valiant. A neuroidal architecture for cognitive computation. *Journal of the ACM (JACM)*, 47(5):854–882, 2000.

[49] Leslie G. Valiant. Memorization and association on a realistic neural model. *Neural Computation*, 17(3):527–555, 2005.

[50] Leslie G. Valiant. The hippocampus as a stable memory allocator for cortex. *Neural Computation*, 24(11):2873–2899, 2012.

[51] Wei Wang and Jean-Jacques E. Slotine. k-winners-take-all computation with neural oscillators. *arXiv preprint q-bio/0401001*, 2003.

[52] David J Willshaw, O Peter Buneman, and Hugh Christopher Longuet-Higgins. Non-holographic associative memory. *Nature*, 222(5197):960–962, 1969.

[53] Alan L. Yuille and Norberto M. Grzywacz. A winner-take-all mechanism based on presynaptic inhibition feedback. *Neural Computation*, 1(3):334–347, 1989.

[54] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 818–833, Cham, 2014. Springer International Publishing.

[55] Bolei Zhou, David Bau, Aude Oliva, and Antonio Torralba. Interpreting deep visual representations via network dissection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(9):2131–2145, September 2019.

A Analysis of Noise-free Learning

Here we present our analysis for the noise-free learning algorithm in Section 5. In Section A.1, we describe how incoming weights change for a particular neuron when it is presented with a consistent input vector. In Section A.2, we prove our main invariant, saying how neurons get bound to concepts, when neuron firing occurs, and how weights change, during the time when the network is learning. In Section A.3, we use that invariant to prove Theorem 5.3.

A.1 Weight Change for Individual Neurons

In this subsection we give a series of three lemmas that describe how incoming weights change for a particular neuron when it is presented with a consistent input vector during execution of our noise-free learning network. Throughout this subsection, we consider a single neuron u with $layer(u) \geq 1$.

We begin by considering how weights change in a single round. Lemma A.1 describes how the weights change for firing neighbors, and for non-firing neighbors. In this lemma, we consider a neuron u with weight vector $w(t-1)$ and input vector $x(t-1)$, both at time $t-1 \geq 0$. Write $z(t-1)$ for the dot product of $w(t-1)$ and $x(t-1)$, which represents the incoming potential in round t. We assume that the $engaged$ component, $e(t)$, is equal to 1. We give bounds on the new weights for u at time t, given by $w(t)$.

Lemma A.1. *Let $F \subseteq \{1, \ldots, n\}$, with $|F| = k$. Assume that:*

1. *$x_i(t-1) = 1$ for every $i \in F$ and $x_i(t-1) = 0$ for every $i \notin F$. That is, exactly the incoming neighbors in F fire at time $t-1$.*
2. *All weights $w_i(t-1), i \in F$ are equal, and all weights $w_i(t-1), i \notin F$ are equal.*
3. *For every $i \in F$, $0 < w_i(t-1) < \frac{1}{\sqrt{k}}$.*
4. *For every $i \notin F$, $w_i(t-1) > 0$.*
5. *$0 < \eta \leq \frac{1}{4k}$.*

Then:

1. *All weights* $w_i(t), i \in F$ *are equal, and all weights* $w_i(t), i \notin F$ *are equal.*
2. *For every* $i \in F$, $w_i(t) > w_i(t-1)$.
3. *For every* $i \in F$, $w_i(t) < \frac{1}{\sqrt{k}}$.
4. *For every* $i \notin F$, $w_i(t) < w_i(t-1)$.
5. *For every* $i \notin F$, $w_i(t) > 0$.

Proof. Note that $z(t-1) < k\frac{1}{\sqrt{k}} = \sqrt{k}$, because of the assumed upper bound for each $w_j(t-1)$ and the fact that $|F| = k$. Similarly, we have that $z(t-1) > 0$.

Part 1 is immediate by symmetry—all components for $i \in F$ are changed by the same rule, based on the same information.

For Part 2, consider any $i \in F$. Since $z(t-1) < \sqrt{k}$ and $w_i(t-1) < \frac{1}{\sqrt{k}}$, the product $z(t-1)\, w_i(t-1) < 1$. Then by Oja's rule:

$$w_i(t) = w_i(t-1) + \eta z(t-1)(1 - z(t-1)w_i(t-1)) > w_i(t-1) + \eta z(t-1) \cdot 0 = w_i(t-1),$$

as needed.

For Part 3, again consider any $i \in F$. Since $w_i(t-1) < \frac{1}{\sqrt{k}}$, we may write $w_i(t-1) = \frac{1}{\sqrt{k}} - \lambda$ for some $\lambda > 0$. Then by symmetry, for every $j \in F$, we have $w_i(t-1) = \frac{1}{\sqrt{k}} - \lambda$. We thus have that

$$\begin{aligned}
w_i(t) &= w_i(t-1) + \eta z(t-1)(1 - z(t-1)w_i(t-1)) \\
&= w_i(t-1) + \eta k \cdot \left(\frac{1}{\sqrt{k}} - \lambda\right)\left(1 - k\left(\frac{1}{\sqrt{k}} - \lambda\right)^2\right) \\
&= w_i(t-1) + \eta k \cdot \left(\frac{1}{\sqrt{k}} - \lambda\right)\left(1 - k\left(\frac{1}{k} - \frac{2\lambda}{\sqrt{k}} + \lambda^2\right)\right) \\
&< w_i(t-1) + \eta k \cdot \left(\frac{1}{\sqrt{k}}\right) 2\lambda\sqrt{k} \\
&\le w_i(t-1) + \frac{\lambda}{2} \\
&< 1/\sqrt{k},
\end{aligned}$$

as needed.

For Part 4, consider any $i \notin F$. We have

$$\begin{aligned}
w_i(t) &= w_i(t-1) + \eta z(t-1)(0 - z(t-1)w_i(t-1)) \\
&= w_i(t-1)(1 - \eta z(t-1)^2) \\
&< w_i(t-1),
\end{aligned}$$

as needed.

Finally, for Part 5, again consider any $i \notin F$. We then have:

$$\begin{aligned}
w_i(t) &= w_i(t-1) + \eta z(t-1)(0 - z(t-1)w_i(t-1)) \\
&= w_i(t-1)(1 - \eta z(t-1)^2) \\
&> w_i(t-1)(1 - \eta k), \text{ since } z(t-1) < \sqrt{k} \\
&\ge w_i(t-1)(1 - \frac{k}{4k}), \text{ since } \eta \le \frac{1}{4k} \\
&= \frac{3}{4} w_i(t-1) \\
&> 0,
\end{aligned}$$

as needed. ∎

Lemma A.2 extends Lemma A.1 to any number of steps. This lemma assumes that the same x inputs are given to the given neuron u at every time. When we apply this later, in the proof of Lemma A.6, it will be in a context where these inputs may occur at separated times, namely, the particular times at which u is actually engaged in learning. At the intervening times, u will not be engaged in learning and therefore will not change its weights.

Lemma A.2. *Let $F \subseteq \{1, \ldots, n\}$, with $|F| = k$. Assume that:*

1. *For every $t \geq 0$, $x_i(t) = 1$ for every $i \in F$ and $x_i(t) = 0$ for every $i \notin F$.*
2. *All weights $w_i(0)$ are equal.*
3. *$0 < w_i(0) < \frac{1}{\sqrt{k}}$ for every i.*
4. *$0 < \eta \leq \frac{1}{4k}$.*

Then for any $t \geq 1$:

1. *All weights $w_i(t), i \in F$ are equal, and all weights $w_i(t), i \notin F$ are equal.*
2. *$0 < w_i(t) < \frac{1}{\sqrt{k}}$ for every i.*
3. *For every $i \in F$, $w_i(t) > w_i(0)$.*
4. *For every $i \notin F$, $w_i(t) < w_i(0)$.*

Lemma A.3 gives quantitative bounds on the amount of weight increase and weight decrease over many rounds, again for a single neuron u involved in learning a single concept. We use notation $w(t), x(t), z(t)$ as before. We assume that $x(t)$ is the same at all times $t = 0, 1, \ldots$, and assume that the engaged component $e(t)$ is equal to 1 at all times t.

Lemma A.3 (Learning Properties)**.** *Let $F \subseteq \{1, \ldots, n\}$ with $|F| = k$. Let $\varepsilon \in (0, 1]$. Let b be a positive integer. Let $\sigma = \frac{4}{3\eta k}((\ell_{\max} + 1)\log(k)) + \frac{3}{\eta k \varepsilon} + \frac{b \log(k)}{\log(\frac{16}{15})}$. Thus, σ is $O\left(\frac{1}{\eta k}\left(\ell_{\max}\log(k) + \frac{1}{\varepsilon}\right) + b\log(k)\right)$. Assume that:*

1. *For every $t \geq 0$, $x_i(t) = 1$ for every $i \in F$, $x_i(t) = 0$ for every $i \notin F$, and $e(t) = 1$.*
2. *All weights $w_i(0)$ are equal to $\frac{1}{k^{\ell_{\max}}}$.*
3. *$\eta = \frac{1}{4k}$.*[9]

Then for every $t \geq \sigma$, the following hold:

1. *For any $i \in F$, we have $w_i(t) \in [\frac{1}{(1+\varepsilon)\sqrt{k}}, \frac{1}{\sqrt{k}}]$.*
2. *For any $i \notin F$, we have $w_i(t) \in [0, \frac{1}{k^{\ell_{\max}+b}}]$.*

Proof. We first show Part 1. Lemma A.2 implies the upper bound of $\frac{1}{\sqrt{k}}$, so it remains to show the lower bound. We do this is two steps, first increasing the weight to an intermediate target value $\frac{1}{2\sqrt{k}}$ and then to the real target value $\frac{1}{(1+\varepsilon)\sqrt{k}}$. These two steps use different arguments.

For the first step, we begin with Claim 1, which bounds the number of rounds required to double the weight w_i, for $i \in F$, when w_i is not "too close" to the target weight $\frac{1}{\sqrt{k}}$.

Claim 1: Assume that $i \in F$. For any positive integer j, the number of rounds needed to increase w_i from $\frac{1}{2^{j+1}\sqrt{k}}$ to $\frac{1}{2^j\sqrt{k}}$ is at most $\frac{4}{3\eta k}$.

[9] This is a very precise assumption but it could be weakened, at a corresponding cost in running time.

Proof of Claim 1: Since all the weights are the same and $\frac{1}{2^{j+1}\sqrt{k}} \le w_i(t-1) \le \frac{1}{2\sqrt{k}}$, we get:

$$\begin{aligned} w_i(t) &= w_i(t-1) + \eta z(t-1)\cdot(1 - z(t-1)\cdot w_i(t-1)) \\ &= w_i(t-1) + \eta k w_i(t-1)(1 - k w_i^2(t-1)) \\ &\ge w_i(t-1) + \frac{\eta k}{2^{j+1}\sqrt{k}}(1 - k\frac{1}{4k}) \\ &= w_i(t-1) + \frac{\eta k}{2^{j+1}\sqrt{k}}(3/4). \end{aligned}$$

Increasing w_i from $\frac{1}{2^{j+1}\sqrt{k}}$ to $\frac{1}{2^{j}\sqrt{k}}$ means we must increase it by an additive amount of $\frac{1}{2^{j+1}\sqrt{k}}$. We have just shown that each round increases w_i by at least $\eta k \frac{1}{2^{j+1}\sqrt{k}}(3/4)$. Thus, the number of rounds required to double w_i from $\frac{1}{2^{j+1}\sqrt{k}}$ to $\frac{1}{2^{j}\sqrt{k}}$ is at most $\frac{1}{2^{j+1}\sqrt{k}}$ divided by $\eta k \frac{1}{2^{j+1}\sqrt{k}}(3/4)$, which is $\frac{4}{3\eta k}$.

End of proof of Claim 1.

Now we can prove the first step, bounding the number of rounds required for the weight to reach at least $\frac{1}{2\sqrt{k}}$:

Claim 2: For $i \in F$, the number of rounds required to increase w_i from the starting value $\frac{1}{k^{\ell_{\max}}}$ to the intermediate target value $\frac{1}{2\sqrt{k}}$ is at most $\frac{4}{3\eta k}((\ell_{\max}+1)\log(k))$.

Proof of Claim 2: By applying Claim 1 $(\ell_{\max}+1)\log(k)$ times.
End of Proof of Claim 2.

Next, for the second step, we bound the number of rounds required to increase w_i, $i \in F$, from $\frac{1}{2\sqrt{k}}$ to $\frac{1}{(1+\varepsilon)\sqrt{k}}$. This time, of course, depends on ε.

Claim 3: For $i \in F$, the number of rounds required to increase w_i from the intermediate target value $\frac{1}{2\sqrt{k}}$ to the final target value $\frac{1}{(1+\varepsilon)\sqrt{k}}$ is at most $\frac{3}{\eta k\varepsilon}$.

Proof of Claim 3: The argument is generally similar to that for Claim 1, but now using the fact that $\frac{1}{2\sqrt{k}} \le w_i(t-1) \le \frac{1}{(1+\varepsilon)\sqrt{k}}$:

$$\begin{aligned} w_i(t) &= w_i(t-1) + \eta z(t-1)(1 - z(t-1)w_i(t-1)) \\ &= w_i(t-1) + \eta k w_i(t-1)(1 - k w_i^2(t-1)) \\ &\ge w_i(t-1) + \frac{\eta k}{2\sqrt{k}}\left(1 - \frac{1}{(1+\varepsilon)^2}\right) \\ &= w_i(t-1) + \frac{\eta\sqrt{k}}{2}\left(1 - \frac{1}{(1+\varepsilon)^2}\right) \\ &\ge w_i(t-1) + \frac{\eta\sqrt{k}}{2}\frac{\varepsilon}{3}, \\ &= w_i(t-1) + \frac{\eta\sqrt{k}\varepsilon}{6}, \end{aligned}$$

where we used the fact that $(1 - 1/(1+x)^2) \ge x/3$ for $0 \le x \le 1$. It follows that the total time to increase w_i from its initial value $\frac{1}{2\sqrt{k}}$ to the target value $\frac{1}{(1+\varepsilon)\sqrt{k}}$ is at most

$$\left(\frac{1}{(1+\varepsilon)\sqrt{k}} - \frac{1}{2\sqrt{k}}\right)\cdot\frac{6}{\eta\sqrt{k}\varepsilon} = \frac{1-\varepsilon}{2(1+\varepsilon)\sqrt{k}}\cdot\frac{6}{\eta\sqrt{k}\varepsilon} = \frac{6(1-\varepsilon)}{2(1+\varepsilon)\eta k\varepsilon} \le \frac{3}{\eta k\varepsilon}.$$

End of Proof of Claim 3.

It follows that the total number of rounds for Part 1 is at most the sum of the bounds from Claims 2 and 3, or

$$\frac{4}{3\eta k}\left((\ell_{\max}+1)\log(k)\right)+\frac{3}{\eta k\varepsilon},$$

which is $O\left(\frac{1}{\eta k}(\ell_{\max}\log(k)+\frac{1}{\varepsilon})\right)$.

Note that once the weights for indices in F reach their target values, they never decrease below those values. This follows from strict monotonicity shown in Lemma A.2.

We now turn to proving Part 2. Lemma A.2 implies the lower bound, so it remains to show the upper bound.

We consider what happens after the increasing weights (for indices in F) have already reached the level $\frac{1}{2\sqrt{k}}$, and then bound the number of rounds for the decreasing weights to decrease to the desired target $\frac{1}{k^{\ell_{\max}+b}}$. The reason we choose the level $\frac{1}{2\sqrt{k}}$ for the increasing weights is that this is enough to guarantee that z is "large enough" to produce a sufficient amount of decrease. For this part, we use our assumed lower bound on η.

Claim 4: For $i \notin F$, the number of rounds required to decrease w_i from the starting weight $\frac{1}{k^{\ell_{\max}+1}}$ to $\frac{1}{k^{\ell_{\max}+b}}$ is at most $\frac{b\log_2 k}{\log_2\frac{16}{15}}$, which is $O(b\log(k))$.

Proof of Claim 4: Considering a single round, we get:

$$\begin{aligned}
w_i(t) &= w_i(t-1)(1-\eta z(t-1)^2)\\
&\leq w_i(t-1)\left(1-\frac{1}{4k}\left(\frac{\sqrt{k}}{2}\right)^2\right)\\
&= w_i(t-1)\left(1-\frac{1}{16}\right) = w_i(t-1)\frac{15}{16}.
\end{aligned}$$

The inequality uses the facts that $\eta \geq \frac{1}{4k}$ and $z(t-1) \geq k(\frac{1}{2\sqrt{k}}) = \frac{\sqrt{k}}{2}$.

Thus, the weight decreases by a factor of $15/16$ at each round. Now consider the number of rounds needed to reduce from $\frac{1}{k^{\ell_{\max}+1}}$ to the target weight $\frac{1}{k^{\ell_{\max}+b}}$. This number is bounded by $\frac{b\log_2 k}{\log_2\frac{16}{15}}$, which is $O(b\log(k))$, as claimed.
End of Proof of Claim 4.

Summing the bounds for Part 1 (increasing) and Part 2 (decreasing), we see that the total number of rounds to complete all the needed increases and decreases is at most

$$\frac{4}{3\eta k}\left((\ell_{\max}+1)\log(k)\right)+\frac{3}{\eta k\varepsilon}+\frac{b\log_2 k}{\log_2\frac{16}{15}},$$

which is $O\left(\frac{1}{\eta k}(\ell_{\max}\log(k)+\frac{1}{\varepsilon})+b\log(k)\right)$, as needed.

□

A.2 Main Invariants

In this section, we give a key lemma, Lemma A.6, which describes key properties of the algorithm with respect to engagement, weight settings, and firing. This lemma deals with the network as a whole, and draws upon the lemmas in Section A.1 for properties involving learning by individual neurons. Lemma A.6 relies on assumptions about the input, captured by our σ-bottom-up training definition, and also about the settings of $engagement$ flags.

For the rest of Appendix A, we use the following assumptions about the various parameter settings:

1. The concept hierarchy consists of $\ell_{\max}$ levels.
2. The network consists of ℓ'_{max} levels, with $\ell_{\max} \leq \ell'_{max}$.

3. b is a positive real ≥ 2.
4. r_1, r_2 satisfy $0 < r_1 < r_2 \leq 1$, and $r_1 k$ is not an integer; more strongly, we assume the technical condition that $r_1 k - \lfloor r_1 k \rfloor \geq \frac{\sqrt{k}}{k^{b-1}}$. Furthermore, we assume that $\frac{1}{\sqrt{k}} + \frac{1}{k} \leq \frac{r_2\sqrt{k}}{2}$.
5. $\varepsilon = \frac{r_2 - r_1}{r_1 + r_2}$.
6. $\tau = \frac{(r_1+r_2)\sqrt{k}}{2}$.
7. $\eta = \frac{1}{4k}$.
8. σ, for the σ-bottom-up training schedule definition, is equal to $\frac{4}{3\eta k}((\ell_{\max}+1)\log(k)) + \frac{3}{\eta k \varepsilon} + \frac{b\log(k)}{\log(\frac{16}{15})}$. Thus, σ is $O\left(\frac{1}{\eta k}\left(\ell_{\max}\log(k) + \frac{1}{\varepsilon}\right) + b\log(k)\right)$.

We use the following assumption about the settings of the engagement flags.

Assumption A.4. *For every time t and layer ℓ, a neuron u on layer $\ell \geq 1$ is engaged (i.e., $u.engaged = 1$) at time t, if and only if both of the following hold:*

1. *A level ℓ concept was shown at time $t - \ell$.*
2. *Neuron u is selected by the WTA at time t.*

Recall that, by Assumption 5.2, the WTA selects exactly one layer ℓ neuron at time t. This, together with Assumption 5.2, implies that exactly one layer ℓ neuron will be engaged at time t.

We also define the point at which a particular layer ℓ neuron u gets "bound" to a particular level ℓ concept c. Namely, we say that a layer ℓ neuron u, $\ell \geq 1$, "binds" to a level ℓ concept c at time t if c is presented for the first time at time $t - \ell$, and u is the neuron that is engaged at time t. At that point, we define $rep(c) = u$.

Here is a simple auxiliary lemma, about unbound neurons.

Lemma A.5. *Let u be a neuron with $layer(u) \geq 1$. Then for every $t \geq 0$, the following hold:*

1. *If u is unbound at time t, then all of u's incoming weights at time t are the initial weight $\frac{1}{k^{\ell_{\max}+1}}$.*
2. *If u is unbound at time t, then u does not fire at time t.*

We are now ready to prove our main lemma. It has five parts, whose proofs are intertwined.

Lemma A.6. *Consider any particular execution of the network in which inputs follow a σ-bottom-up training schedule. For any $t \geq 0$, the following properties hold.*

1. *The $rep()$ mapping from the set C of concepts to the set N of neurons a is one-to-one mapping; that is, for any two distinct concepts c and c' for which $rep(c)$ and $rep(c')$ are both defined by time t, we have $rep(c) \neq rep(c')$.*
2. *For every concept c with $level(c) \geq 1$, every showing of c at a time $\leq t - level(c)$, leads to the same neuron $u = rep(c)$ becoming engaged at time t.*
3. *For every concept c with $level(c) \geq 1$, and any $t' \geq 1$, if c is shown at time $t - level(c)$ for the t'-th time, then the following are true at time t:*
   (a) *Neuron $u = rep(c)$ has weights in $\left(\frac{1}{k^{\ell_{\max}+1}}, \frac{1}{\sqrt{k}}\right)$ for all neurons in $rep(children(c))$, and weights in $\left(0, \frac{1}{k^{\ell_{\max}+1}}\right)$ for all other neurons.*
   (b) *If $t' \geq \sigma$, then u with $u = rep(c)$ has weights in $\left[\frac{1}{(1+\varepsilon)\sqrt{k}}, \frac{1}{\sqrt{k}}\right]$ for all neurons in $rep(children(c))$, and weights in $\left[0, \frac{1}{k^{\ell_{\max}+b}}\right]$ for all other neurons.*
4. *For every concept c, if a proper ancestor of c is shown at time $t - level(c)$, then $rep(c)$ is defined by time t, and fires at time t.*
5. *For any neuron u, the following holds. If u fires at time t, then there exists c such that $u = rep(c)$ at time t, and an ancestor of c is shown at time $t - layer(u)$. (This ancestor could be c or a proper ancestor of c.)*

Proof. First observe that, by Assumption A.4, every representative $rep(c)$ is on the layer equal to $level(c)$. We prove the five-part statement of the lemma by induction on t.

Base: $t = 0$.

For Part 1, the only concepts for which $reps$ are defined at time 0 are level 0 concepts, and these all have distinct $reps$ by assumption. For Parts 2 and 3, note that $level(c) \geq 1$ implies that the times in question are negative, which is impossible; so these are trivially true. For Part 4, it must be that $level(c) = 0$ (to avoid negative times), and a proper ancestor of c is shown at time 0. Then the layer 0 neuron $rep(c)$ fires at time 0, by the definition of "showing".

For Part 5, first note that at time 0 no neurons at layers ≥ 1 are bound, so by Lemma A.5, they cannot fire at time 0. Since we assume that u fires at time 0, it must be that $layer(u) = 0$, which implies that $u = rep(c)$ for some level 0 concept c. Then, since u fires at time 0, by definition of "showing", an ancestor of c must be shown at time 0.

Inductive step: Assume the five-part claim holds for time $t - 1$ and consider time t. We prove the five parts one by one.

For Part 1, let c and c' be any two distinct concepts for which $rep(c)$ and $rep(c')$ are both defined by time t. We must show that $rep(c) \neq rep(c')$. If both $rep(c)$ and $rep(c')$ are defined by time $t - 1$, then by the inductive hypothesis, Part 1, $rep(c) \neq rep(c')$ at time $t - 1$. Since the $reps$ do not change, this is still true at time t, as needed. So the only remaining possibility for conflict is that one of these two concepts, say c', already has its rep defined by time $t - 1$ and the other concept, c, does not, and $rep(c)$ becomes defined at time t, to be the same neuron as $rep(c')$. But we claim that, because of the weight settings, $rep(c)$ must be defined at time t to be a neuron that is unbound at time $t - 1$.

So suppose that u is the neuron that gets defined to be $rep(c)$ at time t; we argue that u must be unbound at time $t - 1$. Write $\ell = level(c)$; then also $layer(u) = \ell$. By Assumption A.4, the $engaged$ flag gets set at time t for u, and for no other layer ℓ neurons. Since c is shown at time $t - \ell$, by the σ-bottom-up assumption, each child of c must have been shown at least σ times prior to time $t - \ell$. Then by the inductive hypothesis, Parts 4 and 5, the layer $\ell - 1$ neurons "fire correctly" at time $t - 1$, that is, all neurons in the set $rep(children(c))$ fire and no other layer $\ell - 1$ neuron fires, at time $t - 1$. This firing pattern implies that every layer ℓ neuron that is already bound strictly prior to time t has incoming potential in round t that is strictly less than k times the initial weight, by the inductive hypothesis Part 3(a) and by the disjointness of the concepts. On the other hand, every layer ℓ neuron that is unbound at time $t - 1$ has incoming potential equal to k times the initial weight, by Lemma A.5. By assumption, there must be at least one unbound neuron available. It follows that the neuron u that is chosen by the WTA is unbound at time $t - 1$, and so cannot be the same as the already-bound neuron $rep(c')$.

For Part 2, let c be any concept with $level(c) \geq 1$, and write $\ell = level(c)$. We must prove that any showing of c at any time $\leq t - \ell$ leads to the same neuron $u = rep(c)$ becoming engaged. If c is not shown at time precisely $t - \ell$, then the claim follows directly from the inductive hypothesis, Part 2. So assume that c is shown at time $t - \ell$. If $t - \ell$ is the first time that c is shown, then $rep(c)$ first gets defined at time t, so the conclusion is trivially true (since there is only one showing to consider).

It remains to consider the case where $rep(c)$ is already defined by time $t - 1$. Then, by the inductive hypothesis, Part 2, we know that any showing of c at a time $\leq t - 1 - \ell$ leads to neuron $rep(c)$ becoming engaged. We now argue that the same $rep(c)$ is also selected at time t. As in the proof of Part 1, the $engaged$ flag is set at time t for exactly one layer ℓ neuron; we claim that this chosen neuron is in fact the previously-defined $rep(c)$. As in the proof for Part 1, we claim that all neurons in the set $rep(children(c))$ fire and no other layer $\ell - 1$ neuron fires at time $t - 1$. Then $rep(c)$ has incoming potential in round t that is strictly greater than k times the initial weight, by the inductive hypothesis, Part 3(a). On other hand, every other layer ℓ neuron has incoming potential that is at most k times the initial weight, again by the inductive hypothesis, Part 3(a). It follows that $rep(c)$ has a strictly higher incoming potential in round t than any other layer ℓ neuron, and so is the chosen neuron at time t.

For Part 3, let c be any concept with $level(c) \geq 1$, and write $\ell = level(c)$. Let $t' \geq 1$. Assume that c is shown at time $t - \ell$ for the t'-th time. We must show:

(a) Neuron $u = rep(c)$ has weights in $\left(\frac{1}{k^{\ell_{\max}+1}}, \frac{1}{\sqrt{k}}\right)$ for all neurons in $rep(children(c))$, and weights in $\left(0, \frac{1}{k^{\ell_{\max}+1}}\right)$ for all other neurons.

(b) If $t' \geq \sigma$, then u with $u = rep(c)$ has weights in $\left[\frac{1}{(1+\varepsilon)\sqrt{k}}, \frac{1}{\sqrt{k}}\right]$ for all neurons in $rep(children(c))$, and weights in $\left[0, \frac{1}{k^{\ell_{\max}+b}}\right]$ for all other neurons.

For both parts, we use Part 2 (for t, not $t-1$) to infer that every showing of c at a time $\leq t - level(c)$ leads to the same neuron $u = rep(c)$ being engaged. Thus, neuron u has been engaged t' times as a result of showing c, up to time t.

For Part (a), fix any $t' \geq 1$. Then we may apply Lemma A.2, with $F = rep(children(c))$, to conclude that the incoming weights for u are in the claimed intervals. Here we use the fact that the initial settings $w_i(0)$ are equal to $\frac{1}{k^{\ell_{\max}+1}}$ For Part (b), assume that $t' \geq \sigma$. Then we may apply Lemma A.3, with $F = rep(children(c))$, to conclude that the incoming weights for u are in the claimed intervals.

For Part 4, let c be any concept, and assume that c^*, a proper ancestor of c, is shown at time $t - level(c)$. We must show that $rep(c)$ is defined by time t, and that it fires at time t.

Since c^* is shown at time $t - level(c)$, by the definition of a σ-bottom-up schedule, that means c was shown at least σ times by time $t - level(c) - 1$. This implies that $rep(c)$ is defined by time $t-1$, and so, by time t. Moreover, since c was shown at least σ times by time $t - level(c) - 1$, by the inductive hypothesis, Part 3(b), at time $t-1$, $rep(c)$ has incoming weights at least $\frac{1}{(1+\varepsilon)\sqrt{k}}$ for all neurons in $rep(children(c))$. By the inductive hypothesis, Part 4, the neurons in $rep(children(c))$ fire at time $t-1$ since c^* is also a proper ancestor of all children of c. Therefore, in round t, the potential of $rep(c)$ is at least $k \cdot \frac{1}{(1+\varepsilon)\sqrt{k}}$, which by our assumptions on the values of the parameters means that the potential is at least τ, which implies that u fires at time t.

For Part 5, fix an arbitrary neuron u and suppose that u fires at time t. We must show that there is some concept c such that $u = rep(c)$ at time t, and a (not necessarily proper) ancestor of c is shown at time $t - layer(c)$. Since u fires at time t, by Lemma A.5, we know that u is bound at time t; let c be the (unique) concept such that $u = rep(c)$. The firing of u at time t is due to the showing of some concept, say c^*, at time $t - layer(u)$.

Let R be the subset of $rep(children(c))$ that fire at time $t-1$. We claim that $|R| \geq 2$; that is, at least two $reps$ of children of c must fire at time $t-1$. For, if at most one $rep(c')$ for a child of c fires at time $t-1$, then by the inductive hypothesis, Part 3(a), the total potential incoming to u in round t would be at most

$$\frac{1}{\sqrt{k}} + \frac{k^{\ell_{\max}}}{k^{\ell_{\max}+1}} = \frac{1}{\sqrt{k}} + \frac{1}{k} \leq \frac{r_2\sqrt{k}}{2} \leq \tau,$$

where τ is the threshold for firing.

Therefore, $|R| \geq 2$; let u' and u'' be any two distinct elements of R. Since u' and u'' fire at time $t-1$, by Lemma A.5, we know that both are bound at time $t-1$; let c' and c'' be the respective concepts such that $u' = rep(c')$ and $u'' = rep(c'')$. We know that $c' \neq c''$ because each concept gets only one rep neuron, by the way that rep is defined. Note that the firing of both u' and u'' must be due to the showing of the same concept c^* at time $(t-1) - (layer(u) - 1) = t - layer(u)$. Then by the inductive hypothesis, Part 5, applied to both u' and u'', we see that c^* must be an ancestor of both c' and c''. Therefore, c^* must be an ancestor of the common parent c of c' and c'', as needed.

This completes the overall proof of the lemma. □

A.3 Proof of Theorem 5.3

Now we use Lemma A.6 to prove our main theorem about noise-free learning, Theorem 5.3.

Proof. By assumption, all the concepts in the hierarchy are shown according to a σ-bottom-up training schedule. This implies, by Assumption A.4, that after the schedule, all the concepts in the hierarchy have $reps$ in the corresponding layers, that is, for each $c \in C$, $layer(rep(c)) = level(c)$. Also, by Lemma A.6, Part 3(b), the weights after the schedule are set as as follows: For every concept c with $level(c) \geq 1$, all incoming weights of $rep(c)$ from the $reps$ of its children, i.e., the neurons in $rep(children(c)$, are in the range $[\frac{1}{(1+\varepsilon)\sqrt{k}}, \frac{1}{\sqrt{k}}]$, and weights from all other neurons (on layer $level(c) - 1$) are in the range $[0, \frac{1}{k^{\ell_{\max}+b}}]$.

We must argue that the resulting network $\mathcal{N}$ (r_1, r_2)-recognizes the concept hierarchy $\mathcal{C}$, according to Definition 4.2. This has two directions, saying that certain neurons must fire and certain neurons must not fire, at certain times, when a particular subset $B \subseteq C_0$ is presented. So suppose that a particular subset $B \subseteq C_0$ is presented at time t.

Neurons that must fire: We must show that the rep of any concept c in $supported_{r_2}(B)$ fires at time $t + level(c)$ (see Definition 2.1 for the definition of $supported$). We prove this by induction on the level number ℓ, $1 \leq \ell \leq \ell_{\max}$, showing that the rep of each level ℓ concept in $supported_{r_2}(B)$ fires at time $t + level(c)$.

For the base case, consider a level 1 concept $c \in supported_{r_2}(B)$; then $rep(c)$ is in $layer$ 1. Since $c \in supported_{r_2}(B)$, it means that $|children(c) \cap B| \geq r_2 k$, that is, at least $r_2 k$ children of c are in B. As noted above, the rep of each of these children is connected to $rep(c)$ by an edge with weight at least $\frac{1}{(1+\varepsilon)\sqrt{k}}$, which yields a total incoming potential for $rep(c)$ in round 1 of at least

$$\frac{r_2 k}{(1+\varepsilon)\sqrt{k}} = \frac{r_2\sqrt{k}}{1+\varepsilon}.$$

To show that $rep(c)$ fires at time $t+1$, it suffices to show that the right-hand side is at least as large as the firing threshold $\tau = \frac{(r_1+r_2)\sqrt{k}}{2}$. That is, we must show that $\frac{r_2}{1+\varepsilon} \geq \frac{r_1+r_2}{2}$. Plugging in the expression for ε, we get that:

$$\frac{r_2}{1+\varepsilon} = \frac{r_2}{1+\frac{r_2-r_1}{r_1+r_2}} = \frac{r_1+r_2}{2},$$

as needed.

For the inductive step, consider $\ell \geq 2$ and assume by induction that the rep of any level $\ell - 1$ concept in $supported_{r_2}(B)$ fires at time $t + \ell - 1$. Consider a level ℓ concept $c \in supported_{r_2}(B)$. Since $c \in supported_{r_2}(B)$, it means that $|children(c) \cap B_{\ell-1}| \geq r_2 k$, using notation from Definition 2.1, that is, at least $r_2 k$ children of c are in $supported_{r_2}(B)$. By the inductive hypothesis, the $reps$ of all of these children of c fire at time $t + \ell - 1$. As noted above, the rep of each of these children is connected to $rep(c)$ by an edge with weight at least $\frac{1}{(1+\varepsilon)\sqrt{k}}$, which yields a total incoming potential for $rep(c)$ in round $t + \ell$ of at least

$$\frac{r_2 k}{(1+\varepsilon)\sqrt{k}} = \frac{r_2\sqrt{k}}{1+\varepsilon}.$$

Arguing as in the base case, this is at least as large as the firing threshold τ, as needed to guarantee that $rep(c)$ fires at time $t + \ell$.

Neurons that must not fire: We must show that the rep of any concept c that is not in $supported_{r_1}(B)$ does not fire at time $t + level(c)$. Again we prove this by induction on the level number ℓ, $1 \leq \ell \leq \ell_{\max}$, showing that the rep of each level ℓ concept that is not in $supported_{r_1}(B)$ does not fire at time $t + level(c)$.

For the base case, consider a level 1 concept $c \notin supported_{r_1}(B)$; then $rep(c)$ is in layer 1. Since $c \notin supported_{r_1}(B)$, it means that $|children(c) \cap B| < r_1 k$, which implies that $|children(c) \cap B| \leq \lfloor r_1 k \rfloor$. As noted above, the rep of each of these children is connected to $rep(c)$ by an edge with weight at most $\frac{1}{\sqrt{k}}$. Also, there are at most $k^{\ell_{\max}+1}$ other level 0 firing neurons, since $B \subseteq C_0$, and all the weights on edges connecting these to $rep(c)$ are at most $\frac{1}{k^{\ell_{\max}+b}}$. Therefore, the total incoming potential for $rep(c)$ in round $t+1$ is at most

$$\frac{\lfloor r_1 k \rfloor}{\sqrt{k}} + \frac{k^{\ell_{\max}+1}}{k^{\ell_{\max}+b}} = \frac{\lfloor r_1 k \rfloor}{\sqrt{k}} + \frac{1}{k^{b-1}}.$$

Now we use the technical assumption that $r_1 k - \lfloor r_1 k \rfloor \geq \frac{\sqrt{k}}{k^{b-1}}$. Then the right hand side of the last inequality is at most

$$\frac{r_1 k - \frac{\sqrt{k}}{k^{b-1}}}{\sqrt{k}} + \frac{1}{k^{b-1}} = r_1 \sqrt{k} < \frac{(r_1 + r_2)\sqrt{k}}{2} = \tau,$$

which implies that $rep(c)$ does not fire.

For the inductive step, consider $\ell \geq 2$ and assume by induction that the rep of any level $\ell - 1$ concept that is not in $supported_{r_1}(B)$ does not fire at time $t + \ell - 1$. Consider a level ℓ concept $c \notin supported_{r_1}(B)$. Since $c \notin supported_{r_1}(B)$, it means that $|children(c) \cap B_{\ell-1}| < r_1 k$, that is, the number of children of c that are in $supported_{r_1}(B)$ is less than $r_1 k$. As noted above, the rep of each of these children is connected to $rep(c)$ by an edge with weight at most $\frac{1}{\sqrt{k}}$.

Now consider the rest of the incoming edges to $rep(c)$. They may come from the $reps$ of children of c that are not in $supported_{r_1}(B)$, from layer $\ell - 1$ neurons that are bound to concepts that are not children of c, and from unbound layer $\ell - 1$ neurons. However, the $reps$ of children of c that are not in $supported_{r_1}(B)$ do not fire, by the inductive hypothesis, and the unbound neurons do not fire, by Lemma A.5. So that leaves us to consider the layer $\ell - 1$ neurons that are bound to concepts in C that are not children of c. There are at most $k^{\ell_{\max}} + 1$ such neurons. Since the weights of the edges connecting them to $rep(c)$ are at most $\frac{1}{k^{\ell_{\max}+b}}$, the total incoming potential for $rep(c)$ in round $t + \ell$ is at most

$$\frac{\lfloor r_1 k \rfloor}{\sqrt{k}} + \frac{k^{\ell_{\max}+1}}{k^{\ell_{\max}+b}} = \frac{\lfloor r_1 k \rfloor}{\sqrt{k}} + \frac{1}{k^{b-1}}.$$

As in the base case, this is strictly less than τ. Therefore, $rep(c)$ does not fire at time $t + level(c)$. ☐

B Analysis of Noisy Learning

Here we present our analysis for the noisy learning algorithm in Section 6. In Lemma B.1, we describe how incoming weights change for a particular neuron when it is noisy-shown. The proof can be found in Section B.4. Once we understand the weight changes of one neuron, we are able to use essentially the same invariants as in the noise-free case (Lemma A.6), describing how neurons get bound to concepts, when neuron firing occurs, and how weights change, during the time when the network is learning. In Section B.3, we put everything together to prove Theorem 6.4.

We start by giving a slightly more detailed proof overview than the one in Section 6.3.

B.1 Proof Overview

The overall proof of Theorem 6.4 is at its core similar to the proof of Theorem 5.3 presented in Appendix A. The main difference is that the weights of the neurons after learning are slightly different: following the notation of Lemma A.1, Lemma A.2 and Lemma A.3, we show that, for every $i \in F$, the weight will eventually approximate

$$\bar{w} = \frac{1}{\sqrt{pk + 1 - p}},$$

and for every $i \notin F$, the weight will eventually be in the interval $[0, 1/k^{2\,\ell_{\max}}]$. Note that, in this section, we set the parameter b, governing the desired decrease of unrelated weights, to be $b = \ell_{\max}$. Also note that we can recover the noise-free case by setting $p = 1$.[10]

The main difficulty in the noisy case is to establish a noisy version of Lemma A.3, which we do in Lemma B.1. Then, proving the main theorem is analogous to the noise-free case. This is because the behavior of this network is the same as that of the noise-free algorithm, except for how the weights of individual neurons are updated. Nonetheless, the same arguments as in the proof Lemma A.6 still hold. Therefore, the core of this section is to prove Lemma B.1. Due to the noise, main structural properties of the noise-free case, such as weights of neurons in F changing monotonically, do not hold anymore. To make matters worse, we cannot simply use Chernoff bounds and assume the

[10] In this case the probabilistic guarantees become deterministic guarantees.

worst-case distribution of the weight changes, since assuming worst-case in each round prevents the weights from converging. Instead, we use a fine-grained potential analysis.

We first bound the worst-case change of any weight w_i during a period of T rounds (Lemma B.2), assuming that the weight at the beginning of the period, $w_i(t)$, is in the interval $[\frac{\sqrt{p}}{4k}, \frac{4}{\sqrt{p}}]$. Namely, we show that for some small δ_1 (defined in Section B.2), we have $(1-\delta_1)w_i(t) \leq w_i(t+T) \leq (1+\delta_1)w_i(t)$. We later show that this assumption holds w.h.p. throughout the first n^6 rounds. It turns out that the way an individual weight changes depends strongly on the other weights in F and on the neurons of the previous layer that fire. More precisely, it depends on $z(t)$, which can change dramatically between rounds, rendering the analysis non-trivial. In order to show that the weights converge to $\bar{w}$, we use the potential function $\psi(\cdot)$. For any time t, let $w_{min}(t)$ and $w_{max}(t)$ be the minimum and maximum weight, respectively, among $\{w_i(t) \mid i \in F\}$. Let

$$\psi(t) = \max\left\{\frac{w_{max}(t)}{\bar{w}}, \frac{\bar{w}}{w_{min}(t)}\right\}.$$

Our goal is to show that this potential decreases quickly until it is very close to 1. Showing that the potential decreases is involved, since one cannot simply use a worst-case approach, due to the terms in Oja's rule being non-linear and potentially having a high variance, depending on the distribution of weights. Instead, we consider the terms $\bar{w}/w_{min}(t)$ and $w_{max}(t)/\bar{w}$ of the potential and consider four cases depending on whether these terms are small or large.

First, if the term $\bar{w}/w_{min}(t)$ is large and the term $w_{max}(t)/\bar{w}$ is small, then the minimum weight w_{min} increases and since the maximum weight w_{max} increases by at most a factor of $(1+\delta)$, the potential decreases. The second case, where the term $w_{max}(t)/\bar{w}$ is large and the term $\bar{w}/w_{min}(t)$ is small, can be bounded analogously. Finally, if $\bar{w}/w_{min}(t)$ and $w_{max}(t)/\bar{w}$ are both large and close to each other, then we show that both terms decrease. Note that if both terms are small, then the potential is small and we are done.

For example, to prove the first case, we first show that, for every $i \in F$ with $w_i(t) \geq (1+2\delta_1)w_{min}$, we have $w_i(t+T) \geq (1+\delta/2)w_{min}$, using the previously established bounds. As mentioned before, in order to prove that any such neuron i^* increases its weight, we cannot use worst-case bounds. Instead, we carefully use the randomness over the input vector x. To this end we define, for every $t' \geq 0$,

$$X(t') = z(t+t') \cdot (x_{i^*}(t+t') - z(t+t') \cdot w_{i^*}(t+t'))$$

and

$$S = \sum_{t'=1}^{T} X(t'). \tag{2}$$

Based on these terms we construct a Doob martingale (Lemma B.4), which allows us to get asymptotically almost tight bounds on S, To do this, we use the Azuma-Hoeffding inequality (Theorem C.1). Putting everything together, we see that $\psi(\cdot)$ decreases. This then allows us to prove Theorem 6.4.

B.2 Convergence of the Weights

We use the following assumptions about the various parameters:

1. $\delta = \bar{w}(r_2 - r_1)/50$,
2. $\delta_1 = \frac{\delta p \bar{w}}{20}$,
3. $T = \frac{7\log(|C|n)}{100p^3\delta_1^2}$,
4. The learning rate $\eta = \frac{\delta_1^3}{64Tk^2p}$.
5. The firing threshold $\tau = r_2 k(\bar{w} - 2\delta)$
6. $b = \ell_{\max}$.

The following lemma is the noisy counterpart to Lemma A.3.

Lemma B.1 (Learning Properties, Noisy Case)**.** *Let $F \subseteq \{1,\dots,n\}$ with $|F| = k$. Let $\varepsilon \in (0,1]$. Let $\sigma = c' \frac{k^6}{p^6 \delta^3} (\ell_{\max} \log(k) + \log(|C|n/\delta)$, for some large enough constant c'. Assume that:*

1. *For every $t \geq 0$, $x_i(t) = 0$ for every $i \notin F$, and $e(t) = 1$.*
2. *All weights $w_i(0)$ are equal to $\frac{1}{k}$.*
3. *η is defined above.*[11]

Then for every $t \in [\sigma, n^6]$, the following with high probability:

1. *For any $i \in F$, we have $w_i(t) \in [\bar{w} - 2\delta, \bar{w} + 2\delta]$.*
2. *For any $i \notin F$, we have $w_i(t) \leq \frac{1}{k^{2\,\ell_{\max}}}$.*

Proving Lemma B.1 is the main goal of the section and we need a series of properties to prove it. We give the proof in Section B.5. We now proceed by showing how Theorem 6.4 follows from this lemma.

B.3 Proof of Theorem 6.4, assuming Lemma B.1

As mentioned at the beginning of this section, it suffices to consider the learning of one concept. Generalizing to a concept hierarchy is analogous to the noise-free case (in particular the proof of Lemma A.6).

We now argue how the learning of one concept follows from Lemma B.1. By Lemma B.1, all weights in F are at least $\bar{w} - 2\delta$ and most $\bar{w} + 2\delta$. Hence, if $c \in \mathit{supported}_{r_2}(B)$, then we can show by a similar induction as in the proof of Theorem 5.3 that each rep fires since, the potential is at least $r_2 k(\bar{w} - 2\delta) = \tau$, which means that the corresponding rep fires. On other other hand, if $c \notin \mathit{supported}_{r_1}(B)$, then there will be a neuron that does not fire since all weights are, by Lemma B.1, at most $\bar{w} + 2\delta$.

Note that, by definition of δ,

$$\begin{aligned} r_1(\bar{w} + 2\delta) &= (r_2 - 50\delta/\bar{w})(\bar{w} + 2\delta) \\ &\leq r_2\bar{w} + 2\delta r_2 - 50\delta \\ &\leq r_2\bar{w} - 2\delta r_2 - 46\delta, \end{aligned}$$

since $r_2 \leq 1$. Therefore, the potential for $\mathit{rep}(c)$ will be at most

$$r_1 k(\bar{w} + 2\delta) + k^{\ell_{\max}} \frac{1}{k^{2\,\ell_{\max}}} < r_2 k \left(\bar{w} - 2\delta - \frac{46\delta}{r_2}\right) + \frac{1}{k} \leq r_2 k(\bar{w} - 2\delta) = \tau,$$

since $k 46\delta = k\frac{46}{50}\bar{w}(r_2 - r_1) \geq \frac{46}{50\sqrt{k}} \geq 1/k$, due to $\bar{w} \geq 1/\sqrt{k}$, $r_2 - r_1 \geq 1/k$ and $k \geq 2$. Thus, the neuron does not fire.

B.4 Towards Lemma B.1

In this subsection, we define a key property $\mathcal{E}_t$ that says that the weights remain within certain multiplicative bounds, for during the interval $[t, t+T]$ rounds. We show in Lemma B.2 that $\mathcal{E}_t$ holds with probability 1. Then we assume $\mathcal{E}$ and show Lemma B.3, which bounds the expected change of the terms in Oja's rule. To derive bounds on the actual change we first show how the changes form a Doob-martingale (Lemma B.4). Using this, we are finally able to show in in Lemma B.5 and Lemma B.6 that the potential decreases.

Let $\mathcal{E}_t$ be the event that for every $t' \in [t, t+T]$, we have

$$(1 - \delta_1)\, w_i(t) \leq w_i(t') \leq (1 + \delta_1)\, w_i(t).$$

[11] This is a very precise assumption but it could be weakened, at a corresponding cost in run time.

Lemma B.2. *Assume $w_i(t) \in [\frac{\sqrt{p}}{4k}, \frac{4}{\sqrt{p}}]$. Then, $\mathcal{E}_t$ holds.*

Proof. Let $w_{max}(t)$ denote the maximum weight at time t. We have $w_{max}(t+1) \leq w_{max}(t) + \eta z(t) \leq w_{max}(t) + \eta w_{max}(t)kp$. Thus, $w_{max}(t+t') \leq w_{max}(t)(1+\eta kp)^T = w_{max}(t)\left(1+\frac{\eta kpT}{T}\right)^T = w_{max}(t)e^x$ for $x = \eta kpT$. Since $p \geq 1/k$, we have $x < 1$, we have

$$w_{max}(t+t') \leq w_{max}(t)e^x \leq w_{max}(t)(1+x+x^2) \leq w_{max}(t)(1+2x).$$

this completes the upper bound of $\mathcal{E}_t$ since $2\eta kpT \leq \delta_1$.

We now consider the lower bound of $\mathcal{E}_t$. Similarly, if $w_{min}(t)$ denotes the minimum weight at time t, then

$w_{min}(t+1) \geq w_{min}(t) - \eta z^2(t) \geq w_{min}(t) - \eta w_{max}^2(t)k^2p^2 \geq w_{min}(t) - \eta 16k^2p$. Thus $w_{min}(t+1) \geq w_{min}(t) - T\eta 16k^2p \geq w_{min}(t) - T\eta 16k^2p\frac{1}{\sqrt{p}/(4k)}w_{min}(t) \geq w_{min}(t)\left(1-64\eta Tk^2\sqrt{p}\right) \geq w_{min}(t)(1-\delta_1)$, since $w_{min}(t) \geq \frac{\sqrt{p}}{4k}$. □

We define the following potential function

$$\phi(t) = \sum_{i\in F} w_i(t).$$

The following bounds the expected change of the weights.

Lemma B.3. *Suppose $\mathcal{E}_t$ holds. Then, we have*

1. $\mathbb{E}\left[\, z(t+t') \mid w(t+t'), \mathcal{F}_t \,\right] = p\phi(t+t')$
2. $\mathbb{E}\left[\, z(t+t')^2 w_{i^*}(t+t') \mid \mathcal{F}_t \,\right] \leq (1+\delta_1)^3 p\phi(t)\left((1-p)w_{max}(t)w_{i^*}(t) + pw_{i^*}(t)\phi(t)\right)$
3. $\mathbb{E}\left[\, z(t+t')^2 w_{i^*}(t+t') \mid \mathcal{F}_t \,\right] \geq (1-\delta_1)^3 p\phi(t)\left((1-p)w_{min}(t)w_{i^*}(t) + pw_{i^*}(t)\phi(t)\right).$

Proof. In the following, the randomness is over $x_i(t+t')$. We have,

$$\mathbb{E}\left[\, z(t+t') \mid w(t+t'), \mathcal{F}_t \,\right] = p\sum_{i\in F}\mathbb{E}\left[\, x_i(t+t') \,\right] w_i(t+t') = p\sum_{i\in F} w_i(t+t') = p\phi(t+t').$$

Moreover,

$$\begin{aligned}
\mathbb{E}\left[\, z(t+t')^2 \mid w(t+t'), \mathcal{F}_t \,\right] &= \sum_{i\in F}\left(pw_i(t+t')^2 + p^2 w_i(t+t')\sum_{j\in F, j\neq i} w_j(t+t')\right) \\
&= \sum_{i\in F}\left(pw_i(t+t')^2 - p^2w_i(t+t')^2 + p^2w_i(t+t')\phi(t+t')\right) \\
&= (p-p^2)\sum_{i\in F} w_i(t+t')^2 + p^2\phi(t+t')^2.
\end{aligned}$$

We suppose $\mathcal{E}_t$ holds, thus in every obtainable configuration it must hold that $(1-\delta_1)w_i(t) \leq w_i(t+t') \leq (1+\delta_1)w_i(t)$. Therefore, $(1-\delta_1)\phi(t) \leq \phi(t+t') \leq (1+\delta_1)\phi(t)$. Thus,

$$
\begin{aligned}
&\mathbb{E}\left[\, z(t+t')^2 w_{i^*}(t+t') \mid \mathcal{F}_t \,\right] = \\
&\quad = \sum_{w' \,\wedge\, w' \text{ obtainable}} \mathbb{E}\left[\, z(t+t')^2 w_{i^*}(t+t') \mid w(t+t') = w', \mathcal{F}_t \,\right] \mathbb{P}\left[\, w(t+t') = w' \,\right] \\
&\quad = \sum_{w' \,\wedge\, w' \text{ obtainable}} w'_{i^*}(t+t') \mathbb{E}\left[\, z(t+t')^2 \mid w(t+t') = w', \mathcal{F}_t \,\right] \mathbb{P}\left[\, w(t+t') = w' \,\right] \\
&\quad \le (1+\delta_1) w_{i^*}(t) \sum_{w' \,\wedge\, w' \text{ obtainable}} \left((p-p^2) \sum_{i \in F} w'_i(t+t')^2 + p^2 \phi(t+t')^2 \right) \mathbb{P}\left[\, w(t+t') = w' \,\right] \\
&\quad \le (1+\delta_1)^3 w_{i^*}(t) \left((p-p^2) \sum_{i \in F} w_i(t)^2 + p^2 \phi(t)^2 \right) \\
&\quad \le w_{i^*}(t)(1+\delta_1)^3 \left((p-p^2) w_{max}(t)\phi(t) + p^2 \phi(t)^2 \right) \\
&\quad \le (1+\delta_1)^3 p \phi(t) \left((1-p) w_{max}(t) w_{i^*}(t) + p w_{i^*}(t) \phi(t) \right).
\end{aligned}
$$

Similarly,

$$
\mathbb{E}\left[\, z(t+t')^2 w_{i^*}(t+t') \mid \mathcal{F}_t \,\right] \ge (1-\delta_1)^3 p \phi(t) \left((1-p) w_{min}(t) w_{i^*}(t) + p w_{i^*}(t) \phi(t) \right).
$$

□

In the following, we define a sequence of random variables $Y_1, Y_2, \ldots$ and show it forms a Doob martingale.

Lemma B.4. *Fix neuron i^*. Let X_i be the random choices of the pk children that fire in round i (in the definition of the noisy learning). Recall that $S = \sum_{t' \le T} z(t+t') \cdot (X_{i^*}(t+t') - z(t+t') \cdot w_{i^*}(t+t'))$. Let $Y_i = \mathbb{E}[\, S \mid X_i, \ldots, X_1 \,]$. Then the following holds*

1. *The sequence $Y_0, Y_1, \ldots, Y_T$ is a (Doob) martingale with respect to the sequence $X_0, X_1, \ldots X_T$.*
2. *For all i, $|Y_i - Y_{i+1}| \le 8k^2\sqrt{p}$.*
3. *$S = \mathbb{E}[\, S \mid X_T, \ldots, X_1 \,] = Y_T$.*

Proof. For the first part, we have, using the tower rule,

$$
\mathbb{E}[\, Y_i \mid X_{i-1}, \ldots, X_1 \,] = \mathbb{E}[\, \mathbb{E}[\, S \mid X_i, \ldots, X_1 \,] \mid X_{i-1}, \ldots, X_1 \,] = \mathbb{E}[\, S \mid X_{i-1}, \ldots, X_1 \,] = Y_{i-1}.
$$

For the second part, note that $w_i \le 2/\sqrt{p}$. Thus, $|Y_i - Y_{i+1}| \le z_{t+i}^2 w_{i^*} \le k^2 p^2 2^3/\sqrt{p}^3$.

The third part follows trivially.

□

Let

$$
\delta_2 = \left(k \frac{\sqrt{p}}{2k} \right) p^2 \left(\frac{20\delta_1}{p} \right) = 10 p^{3/2} \delta_1
$$

The following lemma shows that if the potential is large due to w_{min} being small, then the weight of the smallest neurons increases.

Lemma B.5. *Suppose $\mathcal{E}_t$ holds. Consider the neurons i^* with $w_{i^*}(t) \in [w_{min}, (1+2\delta_1)w_{min}]$ and $\bar{w} - w_{i^*}(t) \ge \delta$. Assume*

$$
\frac{\bar{w}}{w_{min}(t)} \ge (1-2\delta_1) \frac{w_{max}(t)}{\bar{w}}. \tag{3}
$$

Then, with probability at least $1 - 1/n^6$,

$$
w_{i^*}(t+T) \ge w_{i^*}(t) + T\eta\delta_2/2
$$

Proof. By the second part of Lemma B.3, for $t' \le T$

$$\mathbb{E}\left[\, z(t+t')^2 w_{i^*}(t+t') \,\right] \le (1+\delta_1)^3 p\phi(t) \left((1-p) w_{max}(t) w_{i^*}(t) + p w_{i^*}(t)\phi(t)\right).$$

We now bound the terms in the parentheses. First note that

$$w_{i^*}(t) w_{max}(t) \le (1+2\delta_1) w_{min}(t) w_{max}(t) \le \frac{1+2\delta_1}{1-2\delta_1}\bar{w}^2 \le (1+4.5\delta_1)\bar{w}^2,$$

since $\delta_1 \in [0, 1/18]$. Furthermore, for $\delta_1 \in [0, 1/9]$ we have $(1+4.5\delta_1)(1+\delta_1) \le (1+6\delta_1)$. Thus,

$$\begin{aligned} w_{i^*}(t)\phi(t) &\le (k-1)(1+\delta_1) w_{i^*}(t) w_{max} + (1+\delta_1) w_{i^*}(t) w_{i^*}(t) \\ &\le (k-1)(1+\delta_1)(1+4.5\delta_1)\bar{w}^2 + (1+\delta_1) w_{i^*}(t) w_{i^*}(t) \\ &\le (1+6\delta_1)\left((k-1)\bar{w}^2 + w_{i^*}(t)^2\right) \\ &= (1+6\delta_1)\left(k\bar{w}^2 + w_{i^*}(t)^2 - \bar{w}^2\right). \end{aligned}$$

Note that $(1-p)\bar{w}^2 + pk\bar{w}^2 = 1$. Thus,

$$\begin{aligned} (1-p) w_{max}(t) w_{i^*}(t) + p w_{i^*}(t)\phi(t) &\le (1+6\delta_1)\left((1-p)\bar{w}^2 + pk\bar{w}^2 + p(w_{i^*}(t)^2 - \bar{w}^2)\right) \\ &= (1+6\delta_1)\left(1 - p(\bar{w}^2 - w_{i^*}(t)^2)\right). \end{aligned}$$

Therefore,

$$\mathbb{E}\left[\, z(t+t')^2 w_{i^*}(t+t') \,\right] \le (1+10\delta_1) p\phi(t)\left(1 - p(\bar{w}^2 - w_{i^*}(t)^2)\right),$$

where we used that $(1+6x)(1+x)^3 \le (1+10x)$ for $x \le 0.045$.

Note that

$$\bar{w}^2 - w_{i^*}(t)^2 \ge \bar{w}^2 - w_{i^*}(t)\bar{w} = \bar{w}(\bar{w} - w_{i^*}(t)) \ge \bar{w}\delta = \bar{w}\frac{20}{\bar{w}p}\delta_1. \tag{4}$$

Finally, using the definition of S (Equation 2) and combining the above with the first part of Lemma B.3,

$$\begin{aligned} \mathbb{E}\left[\, S \,\right] &\ge T\left(\mathbb{E}\left[\, z(t+t') \,\right] - \mathbb{E}\left[\, z(t+t')^2 w_{i^*}(t+t') \,\right]\right) \\ &\ge T\phi(t)p\left(1 - (1+10\delta_1)\left(1 - p(\bar{w}^2 - w_{i^*}(t)^2)\right)\right) \\ &\ge T\phi(t)p^2\frac{\bar{w}^2 - w_{i^*}(t)^2}{2}, \end{aligned}$$

where we used that $1-(1+z)(1-x) = 1-(1-x+z-zx) = x-z+zx \ge x/2$ for $z \le x/2$. We define the sequence $Y_1, Y_2, \ldots$ of variabels as defined in Lemma B.4. By Lemma B.4, this sequence is a Doob martingale. Thus, we can apply Theorem C.1 to the Doob martingale $Y_T, Y_{T-1}, \ldots, Y_1$ with $|Y_i - Y_{i+1}| \le \delta_3$ for $\delta_3 = 8k^2\sqrt{p}$.

We derive using the lower bounds on the weights and Equation 4.

$$\begin{aligned} \mathbb{P}\left[\, |S - \mathbb{E}\left[\, S \,\right]| \ge \frac{\mathbb{E}\left[\, S \,\right]}{2} \,\right] &\le 2\exp\left(-\frac{2\left(\frac{\mathbb{E}[\, S\,]}{2}\right)^2}{T\delta_3^2}\right) \le 2\exp\left(-\frac{2\left(T\phi(t)p^2\frac{\bar{w}^2 - w_{i^*}(t)^2}{2}\right)^2}{T\delta_3^2}\right) \\ &\le 2\exp\left(-\frac{T\left(\phi(t)p^2\left(\bar{w}^2 - w_{i^*}(t)^2\right)\right)^2}{4\delta_3^2}\right) \le 2\exp\left(-7\,\ell_{\max}\log(|C|n)\right) \le \frac{1}{|C|n^6}, \end{aligned}$$

where the last inequality follows from

$$T\left(\phi(t)p^2\left(\bar{w}^2 - w_{i^*}(t)^2\right)\right)^2 \ge T\delta_2^2 = 100Tp^3\delta_1^2 = 7\log(|C|n).$$

Thus

$$w_{i^*}(t+T) \ge w_{i^*}(t) + \eta S \ge w_{i^*}(t) + \eta\mathbb{E}\left[\, S \,\right]/2 \ge w_{i^*}(t) + T\eta\delta_2/2$$

□

The following lemma is analogous to the previous one, with the difference that we analyse the case where ψ is dominated by large weights (rather than small) and show that these large weights decrease.

Lemma B.6. *Suppose $\mathcal{E}_t$ holds. Consider the neurons i^* with $w_{i^*}(t) \in [w_{max}(1-2\delta_1), w_{max}]$ and $w_{i^*}(t) - \bar{w} \geq \delta$. Assume*

$$\frac{w_{max}(t)}{\bar{w}} \geq (1-2\delta_1)\frac{\bar{w}}{w_{min}(t)} \tag{5}$$

Then, with probability at least $1 - 1/n^6$,

$$w_{i^*}(t+T) \leq w_{i^*}(t) - T\eta\delta_2/2$$

Proof. We have for all $i \in F$ with $w_i(t) \geq (1+2\delta_1)w_{min}$, we have $w_i(t+T) \geq (1+\delta_1/2)w_{min}$, since each weight can only decrease by a factor of $(1-\delta_1)$ and since $(1+2\delta_1)(1-\delta_1) = 1+\delta_1-2\delta_1 \geq (1+\delta/2)$. Thus, we only consider the neurons i^* with $w_{i^*}(t) \in [w_{min}, (1+2\delta_1)w_{min}]$. By the third part of Lemma B.3, for $t' \leq T$

$$\mathbb{E}\left[\, z(t+t')^2 w_{i^*}(t+t') \,\right] \geq (1-\delta_1)^3 p\phi(t)\left((1-p)w_{min}(t)w_{i^*}(t) + pw_{i^*}(t)\phi(t)\right).$$

We now bound the terms in the parentheses. First note that

$$w_{i^*}(t)w_{min}(t) \geq (1-2\delta_1)w_{min}(t)w_{max}(t) \geq (1-2\delta_1)^2\bar{w}^2 \geq (1-4\delta_1)\bar{w}^2,$$

since $\delta_1 \geq 0$.

Thus,

$$\begin{aligned}(1-p)w_{min}(t)w_{i^*}(t) + pw_{i^*}(t)\phi(t) &\geq (1-4\delta_1)\left((1-p)\bar{w}^2 + pk\bar{w}^2 + p(w_{i^*}(t)^2 - \bar{w}^2)\right) \\ &= (1-4\delta_1)\left(1 - p(\bar{w}^2 - w_{i^*}(t)^2)\right)\end{aligned}$$

Therefore,

$$\mathbb{E}\left[\, z(t+t')^2 w_{i^*}(t+t') \,\right] \geq (1-10\delta_1)p\phi(t)\left(1 - p(\bar{w}^2 - w_{i^*}(t)^2)\right),$$

where we used that $(1-4x)(1-x)^3 \geq (1-10x)$ for $x \geq 0$.

Note that

$$\bar{w}(w_{i^*}(t) - \bar{w}) \geq \bar{w}\delta = \bar{w}\frac{20}{\bar{w}p}\delta_1 \tag{6}$$

Finally, using the definition of S (Equation 2) and combining the above with the first part of Lemma B.3,

$$\begin{aligned}\mathbb{E}\left[\, S \,\right] &\leq T\left(\mathbb{E}\left[\, z(t+t') \,\right] - \mathbb{E}\left[\, z(t+t')^2 w_{i^*}(t+t') \,\right]\right) \\ &\leq T\phi(t)p\left(1 - (1-10\delta_1)\left(1 - p(\bar{w}^2 - w_{i^*}(t)^2)\right)\right) \\ &\leq 2T\phi(t)p^2\bar{w}^2 - w_{i^*}(t)^2 = -2T\phi(t)p^2(w_{i^*}(t)^2 - \bar{w}^2) \\ &\leq -2T\phi(t)p^2\bar{w}(w_{i^*}(t) - \bar{w}),\end{aligned}$$

where we used that $1-(1-z)(1-x) = 1-(1-x-z+zx) = x-z+zx \leq 2x$ for $z \leq 1$.

This allows us to apply Theorem C.1 and the rest is analogous.

Thus

$$w_{i^*}(t+T) \leq w_{i^*}(t) + \eta S \leq w_{i^*}(t) + \eta\mathbb{E}\left[\, S \,\right]/2 \leq w_{i^*}(t) - T\eta\delta_2/2$$

□

We have for all $i \in F$ with $w_i(t) \geq (1+2\delta_1)w_{min}$, we have $w_i(t+T) \geq (1+\delta_1/2)w_{min}$, since each weight can only decrease by a factor of $(1-\delta_1)$ and since $(1+2\delta_1)(1-\delta_1) = 1+\delta_1-2\delta_1 \geq (1+\delta/2)$.

Note that if neither Equation 3 nor Equation 5 applies, then both $w_{min}(t)$ and $w_{max}(t)$ must be close to $\bar{w}$ and the claim follows easily.

B.5 Proof of Lemma B.1

We argue by induction on j, that $\psi(j \cdot T) \leq \max(\psi(0) - jT\eta\delta_2/2, \bar{w} + 2\delta)$ with probability at least $1 - j/(|C|n^6)$. The base case is trivial. Assume the claim holds up to $j-1$. We have

$w_i((j-1)T) \in [\frac{\sqrt{p}}{4k}, \frac{4}{\sqrt{p}}]$. Therefore, by Lemma B.2 $\mathcal{E}_{(j-1)T,T}$ holds. This allows us to apply Lemma B.5 and Lemma B.6.

Consider the following equations

$$\frac{\bar{w}}{w_{min}(t)} \geq (1-2\delta_1)\frac{w_{max}(t)}{\bar{w}}. \tag{7}$$

$$\frac{w_{max}(t)}{\bar{w}} \geq (1-2\delta_1)\frac{\bar{w}}{w_{min}(t)} \tag{8}$$

We consider four cases based on whether or not the two equations Equation 7 and Equation 8 hold. In the first case Equation 7 holds and Equation 8 does not. In this case we can bound the drop of $\psi()$ by considering the the increase of $w_{min}()$ and we can disregard the increase of $w_{max}()$, since even if it increases by a factor of $(1+\delta_1)$, we have

$$\frac{w_{max}(jT)}{\bar{w}} \leq (1+\delta_1)\frac{w_{max}((j-1)T)}{\bar{w}} \leq (1+\delta_1)(1-2\delta_1)\frac{\bar{w}}{w_{min}((j-1)T)} \leq (1-\delta_1)\frac{\bar{w}}{w_{min}((j-1)T)}.$$

In the second case Equation 8 holds and Equation 7 does not. This case is analogous to the first case.

In the third case Equation 7 and Equation 8 hold. Here, one can show that both the minimum weight increases, and the maximum weight decreases.

In the fourth case, none of the equations hold. This yields a contradiction

$$\frac{\bar{w}}{w_{min}(t)} < (1-2\delta_1)\frac{w_{max}(t)}{\bar{w}} < (1-2\delta_1)^2\frac{\bar{w}}{w_{min}(t)}.$$

Thus we can disregard this case.

W.l.o.g. we assume the first case holds.

Consider the neurons i^* with $w_{i^*}(t) \in [w_{min}, (1+2\delta_1)w_{min}]$ and $\bar{w} - w_{i^*}(t) \geq \delta$. Then, by Lemma B.5, with probability at least $1 - 1/n^6$,

$$w_{i^*}(t+T) \geq w_{i^*}(t) + T\eta\delta_2/2 \geq w_{i^*}(t) + w_{i^*}(t)\frac{T\eta\delta_2}{2(4\sqrt{p})}.$$

Note that in the analogous cases two and three we have for any neurons i^* with $w_{i^*}(t) \in [w_{max}(1-2\delta_1), w_{max}]$ that

$$w_{i^*}(t+T) \leq w_{i^*}(t) - T\eta\delta_2/2 \leq w_{i^*}(t) - w_{i^*}(t)\frac{T\eta\delta_2}{2(4\sqrt{p})}.$$

Let $\delta_4 = T\eta\delta_2/(8\sqrt{p})$. Thus, either way

$$\psi(jT) \leq (1-\delta_4)\psi((j-1)T).$$

Using the fact that $\log(1+x) \geq 2x$ for $x \in (-1/2, 0)$, we get that after

$$j^* = \log_{1-\delta_4}(\delta/\psi(0)) = \frac{\log(\delta/\psi(0))}{\log(1-\delta_4)} \leq \frac{\log(\delta/\psi(0))}{-2\delta_4} = \frac{\log(\psi(0)/\delta)}{2\delta_4}$$

intervals of length T the $\psi()$ is within an error of at most 2δ and stays there by assumption for n^6 rounds. Thus the total number of rounds is Tj^*. The bound from the claim follows by observing that term $\eta T/\delta_4$ is a small polynomial in p and w and δ.

Finally, we consider the time required for weights $i \notin F$ to decreases below $k^{-2\,\ell_{\max}}$. After the weights in F are close to there target, we have that $z(t) \geq pk\bar{w}/2$. Thus at this point, the weights decrease changes as follows every round

$$w_i(t) = w_i(t-1)(1 - \eta z(t-1)^2) \geq w_i(t-1)(1 - \eta p^2 k^2 \bar{w}^2/4)).$$

Thus, the potential halves every $20/(\eta p^2 k^2 \bar{w}^2)$ rounds. Since the potential only needs to drop by a factor of $k^{2\,\ell_{\max}}$, the bound follows.

C Auxiliary Content

The following is a slightly modified version of Theorem 5.2 in [5], which we use in Lemma B.5 and Lemma B.6.

Theorem C.1 (Azuma-Hoeffding inequality - general version [5])**.** *Let $Y_0, Y_1, \ldots$ be a martingale with respect ot the sequence $X_0, X_1, \ldots$. Suppose also that Y_i satisfies $a_i \leq Y_i - Y_{i-1} \leq b_i$ for all i. As an example, the $engaged$ flag could be used to ensure that, in any round, only one neuron in the network is prepared to learn.*

$$\mathbb{P}\left[\,|Y_n - Y_0| \geq t\,\right] \leq 2\exp\left(-\frac{2t^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right).$$