What Do Hebbian Learners Learn? Reduction Axioms for Iterated Hebbian Learning

Caleb Schultz Kisby[1], **Saúl A. Blanco**[1], **Lawrence S. Moss**[2]
[1]Department of Computer Science, Indiana University
[2]Department of Mathematics, Indiana University
Bloomington, IN 47408, USA
{cckisby, sblancor, lmoss}@indiana.edu

Abstract

This paper is a contribution to neural network semantics, a foundational framework for neuro-symbolic AI. The key insight of this theory is that logical operators can be mapped to operators on neural network states. In this paper, we do this for a neural network *learning* operator. We map a dynamic operator $[\varphi]$ to *iterated Hebbian learning*, a simple learning policy that updates a neural network by repeatedly applying Hebb's learning rule until the net reaches a fixed-point. Our main result is that we can "translate away" $[\varphi]$-formulas via reduction axioms. This means that completeness for the logic of iterated Hebbian learning follows from completeness of the base logic. These reduction axioms also provide (1) a human-interpretable description of iterated Hebbian learning as a kind of plausibility upgrade, and (2) an approach to building neural networks with guarantees on what they can learn.

1 Introduction

The two dominant paradigms of AI, connectionist neural networks and symbolic systems, have long seemed irreconcilable. Symbolic systems are well-suited for giving explicit inferences in a human-interpretable language, but are brittle and fail to adapt to new situations. On the other hand, neural networks are flexible and excel at learning from unstructured data, but are considered black-boxes due to how difficult it is to interpret their reasoning. In response to this dichotomy, the field of *neuro-symbolic AI* has emerged — a community-wide effort to integrate neural and symbolic systems, while retaining the advantages of both. Despite the many different proposals for neuro-symbolic AI (too many to list! See (Bader and Hitzler 2005; Besold et al. 2017; Sarker et al. 2022)), there is little agreement on what the interface between the two ought to be. There is a clear need for a unifying theory that can explain the relationship between neural networks and symbolic systems (Harmelen 2022).

In fact, there *is* an up-and-coming foundational theory for neuro-symbolic systems, which we call *neural network semantics*. Its key insight is that neural networks can be taken as models for a formal logic. Moreover, logical operators can be mapped to operators on neural network states. Alternatively, we can *semantically encode* classical model operators into neural operators (and vice-versa).

The central questions this theory aims to answer are:

Soundness. What axioms are sound for neural network operators? Can neural operators be mapped to classical ones in a sound way? Note that checking soundness is equivalent to **formally verifying** properties of nets.

Completeness. What are the complete axioms for neural network operators? This is equivalent to **model building**: Can we build a neural network that obeys a set of logical constraints Γ? Can we build a neural network from a classical model?

We refer the reader to the landmark survey (Odense and d'Avila Garcez 2022), which shows that this framework encompasses a wide class of neuro-symbolic systems. We will discuss other work that we consider part of the core theory in the next section.

The standard example is the *forward propagation* operator Prop over a net $\mathcal{N}$. Active neurons in a state S successively activate new neurons until eventually the state of the net stabilizes — $\mathsf{Prop}(S)$ returns the state at the fixed point. A classic result from (Leitgeb 2001) is this: Say conditionals $\varphi \Rightarrow \psi$ are interpreted as

$$\mathcal{N} \models \varphi \Rightarrow \psi \text{ iff } \mathsf{Prop}(\llbracket\varphi\rrbracket) \supseteq \llbracket\psi\rrbracket$$

i.e., ψ is activated by input φ; or "the net classifies φ as ψ". Then, in a binary feed-forward net, Prop is completely axiomatized by the loop-cumulative conditional laws of (Kraus, Lehmann, and Magidor 1990). The result is robust, and can be extended to different choices of conditional axioms and neural network architectures (Leitgeb 2003). The general takeaway is that forward propagation corresponds to a non-monotonic conditional.

A central challenge for this theory is to do the same for neural network *learning* operators. Our previous work (Kisby, Blanco, and Moss 2022) considers a simple learning policy — naïve Hebbian update ("neurons that fire together wire together") — on a binary, feed-forward net. Although this work offers sound axioms for Hebbian learning, the question of completeness is left open.

Our Contribution. In this paper we tackle the completeness of Hebbian learning. We map a dynamic operator $[\varphi]$ instead to *iterated* Hebbian update Hebb^*, i.e., "update the net by repeatedly applying Hebb's learning rule until a fixed-point." Our main result is that we can "translate away" $[\varphi]$-formulas by reducing them to formulas that reason only about

forward propagation and graph reachability. It follows that iterated Hebbian learning is completely axiomatized by the reduction axioms we used in translation, plus whatever axioms the base logic needs. To our knowledge, this result is the first ever completeness theorem for any learning policy on neural networks.

There are two major upshots of this result. First, these reduction axioms give a complete and human-interpretable description of iterated Hebbian learning as a dynamic plausibility upgrade. Second, assuming we have model building for the base logic, we can use these reduction axioms to build neural networks with guarantees on what they can learn.

The proofs of our main theorem and its major supporting lemmas have been verified using the Lean 4 interactive theorem prover (Moura and Ullrich 2021). The code and installation instructions are available at

https://github.com/ais-climber/AAAI2024

2 Related Work

Neural Network Semantics. The idea that neural networks can be viewed as models for logic dates back to (McCulloch and Pitts 1943). But the neural network semantics we present here builds on a recent reimagining of this (Balkenius and Gärdenfors 1991; Leitgeb 2018), where logical formulas are mapped to *states* of the net rather than to individual neurons (thus avoiding the "grandmother cell" problem (Gross 2002)). Early work established the formal correspondence between forward propagation and conditional belief (Balkenius and Gärdenfors 1991; Leitgeb 2001, 2003; Blutner 2004). Note that all of this early work focuses on *binary* nets. More recently, (Giordano and Theseider Dupré 2021) and (Giordano, Gliozzi, and Theseider Dupré 2022) prove soundness for forward propagation over *fuzzy* neural networks. And as mentioned above, (Kisby, Blanco, and Moss 2022) shows soundness — but not completeness — for a simple Hebbian learning policy.

Dynamic Logics for Learning. Our approach to modeling iterated Hebbian learning takes inspiration from dynamic epistemic and doxastic logics (DELs) (Van Ditmarsch, van Der Hoek, and Kooi 2007; Van Benthem 2011). Two recent papers, (Baltag, Li, and Pedersen 2019; Baltag et al. 2019) present DELs that model an agent's learning. But it is unclear how these learning policies might relate to specific neural implementations of learning such as Hebbian update and backpropagation.

We also use the trick of completeness by translation, which has notably been used to prove completeness for public announcement logic without common knowledge (Baltag, Moss, and Solecki 1998; Plaza 2007), as well as plausibility upgrade operators such as lexicographic and elite upgrade (Van Benthem 2007). Perhaps the closest logics to ours are these logics for plausibility upgrade, especially *iterated* plausibility upgrade (Baltag and Smets 2009). But we leave open the precise relationship between these logics and Hebbian update.

3 Base Logic and Neural Network Semantics

Neural Network Preliminaries

For our base logic (without update), a model of our neural network semantics is just a special kind of artificial neural network (ANN), along with an interpretation function. First, we spell out precisely what class of neural networks Net we're talking about. In general,

Definition 1. An *ANN* is a pointed directed graph $\mathcal{N} = \langle N, E, W, A, \eta \rangle \in \mathsf{Net}$, where

- N is a finite nonempty set (the set of *neurons*)
- $E \subseteq N \times N$ (the set of *excitatory connections*)
- $W : E \to \mathbb{Q}$ (the *weight* of a given connection)
- $A : \mathbb{Q} \to \mathbb{Q}$ (the *activation function*)
- $\eta \in \mathbb{Q}, \eta \geq 0$ (the *learning rate*)

We write $m \in \mathrm{preds}(n)$, i.e., m is a *predecessor* of n, whenever $(m, n) \in E$. We also write $\deg(n)$ to indicate the degree (number of predecessors) of n.

We place the following restrictions on our nets $\mathcal{N} \in \mathsf{Net}$.

A **is binary.** $A : \mathbb{Q} \to \{0, 1\}$ is a binary activation function.

A **is nondecreasing.** $\forall x, y \in \mathbb{Q}$ if $x \leq y$ then $A(x) \leq A(y)$

A **has a threshold.** $\exists t \in \mathbb{Q}$ such that $A(t) = 1$.

$\mathcal{N}$ **is feed-forward.** The graph of $\mathcal{N}$ is acyclic.

$\mathcal{N}$ **is fully connected.** $\forall m, n \in N$, either $(m, n) \in E$, $(n, m) \in E$, or m and n have exactly the same predecessors and successors.

The first three conditions restrict A to binary step functions, which we need in order to match binary activations to binary truth values in the logic. But this assumption is clearly unrealistic in practice. Letting it go would instead require mapping fuzzy activations to fuzzy truth values (left to future work).

In machine learning practice, "fully connected" means that there is an edge from every node in layer l to every node in the *following* layer $l + 1$. But here we mean something much stronger: the graph is fully connected, including "highway edges" that cut between layers, as shown in Figure 1. (This intuition comes from work on highway networks (Srivastava, Greff, and Schmidhuber 2015).) This assumption is crucial for our results about iterated Hebbian learning, and we expect that letting it go will not be easy (see Section 7).

Since our nets are feed-forward, their nodes can be partitioned into layers. For every neuron $n \in N$, we define $\mathsf{layer}(n)$ to be the maximal length of a path from any node m to n, where m has no predecessors. Because our nets $\mathcal{N}$ are fully connected, for all $m, n \in N$ we have $m \in \mathrm{preds}(n)$ iff $\mathsf{layer}(m) < \mathsf{layer}(n)$.

Forward Propagation and Reachability

We now consider two fundamental neural network operators: Forward propagation Prop and graph reachability Reach. We formalize both of these as operators on the *state* of the neural network; a state is just a possible activation pattern of neurons in the net. Since our activation function A is binary, either a neuron is active (1) or it is not (0). So we can identify the states of $\mathcal{N}$ with sets of neurons.

$$\mathsf{State} = \{S \mid S \subseteq N\}$$

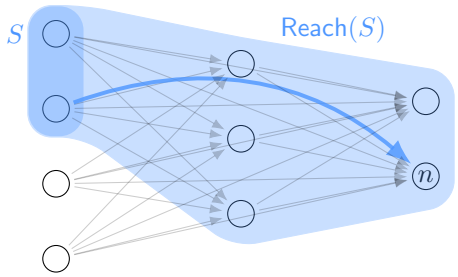

Figure 1: The graph reachability operator Reach.

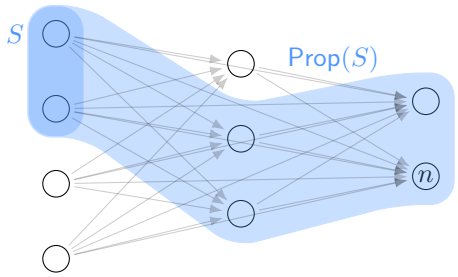

Figure 2: The forward propagation operator Prop.

If our activations were continuous $A \in [0,1]$, then we would identify states with *fuzzy* sets instead. We get the activation value of a particular neuron n in a state S as follows.

Definition 2. Let $S \in \mathsf{State}$. The characteristic function $\chi_S : N \to \{0,1\}$ is given by $\chi_S(n) = 1$ iff $n \in S$.

The Reach operator is just ordinary graph-reachability: $\mathsf{Reach}(S)$ returns the set of all neurons reachable from S (illustrated in Figure 1). Formally, $\mathsf{Reach}_{\mathcal{N}} : \mathsf{State} \to \mathsf{State}$ is given by $n \in \mathsf{Reach}(S)$ iff there exists $m \in S$ with an E-path from m to n.

Reach is not a very interesting operator in its own right, but we include it because graph reachability is necessary for reasoning about Hebbian learning. It's easy to check that Reach is an ordinary monotonic closure operator.

Proposition 1. For all $S, A, B \in \mathsf{State}$,

Inclusion. $S \subseteq \mathsf{Reach}(S)$
Idempotent. $\mathsf{Reach}(\mathsf{Reach}(S)) = \mathsf{Reach}(S)$
Monotonic. If $A \subseteq B$ then $\mathsf{Reach}(A) \subseteq \mathsf{Reach}(B)$.

The more important operator is Prop, which captures how activation patterns are "propagated forward" through the net. As we mentioned in the Introduction, the idea is that active neurons in a state S successively activate new neurons. The activation of each neuron n is a function of its predecessor's activations. $\mathsf{Prop}(S)$ returns the state at the fixed point of the process, i.e., the set of all neurons that are eventually activated on input S (illustrated in Figure 2).

We formalize forward propagation as follows, drawing heavily from (Leitgeb 2001).

Definition 3. Let $n \in N$, and let $\vec{m} = m_1, \ldots, m_{\deg(n)}$ list the predecessors of n. We define $\mathsf{Prop}_{\mathcal{N}} : \mathsf{State} \to \mathsf{State}$ recursively on $l = \mathsf{layer}(n)$ as follows.

Base ($l = 0$). $n \in \mathsf{Prop}_{\mathcal{N}}(S)$ iff $n \in S$
Constructor ($l \geq 0$). $n \in \mathsf{Prop}_{\mathcal{N}}(S)$ iff either $n \in S$, or n is activated by its predecessors $m_i \in \mathsf{Prop}_{\mathcal{N}}(S)$, i.e.,

$$A\Big(\sum_{i=1}^{\deg(n)} W(m_i, n) \cdot \chi_{\mathsf{Prop}_{\mathcal{N}}(S)}(m_i)\Big) = 1$$

If $\mathcal{N}$ is clear from context, we just write $\mathsf{Prop}(S)$.

Note that Prop is well-founded, since all predecessors $m_i \in \mathsf{preds}(n)$ have $\mathsf{layer}(m_i) < \mathsf{layer}(n)$. Also note that our definition differs somewhat from (Leitgeb 2001), which defines Prop over *inhibition nets* — weightless nets with both excitatory and inhibitory connections. But that paper also proves that inhibition nets and binary, feed-forward nets have the same Prop-structure. So we import the results here.

We can think of Prop as the nonmonotonic counterpart to Reach; it is not the case that for all $A, B \in \mathsf{State}$, if $A \subseteq B$ then $\mathsf{Prop}(A) \subseteq \mathsf{Prop}(B)$. This is because our net's weights can be negative, and so $\mathsf{Prop}(B)$ can inhibit the activation of new neurons that would otherwise be activated by $\mathsf{Prop}(A)$. Instead, we can characterize Prop as a loop-cumulative closure operator.

Proposition 2. (Leitgeb 2001) Let $S, S_1, \ldots, S_k \in \mathsf{State}$.

Inclusion. $S \subseteq \mathsf{Prop}(S)$
Idempotence. $\mathsf{Prop}(\mathsf{Prop}(S)) = \mathsf{Prop}(S)$
Cumulative. If $S_1 \subseteq S_2 \subseteq \mathsf{Prop}(S_1)$, then $\mathsf{Prop}(S_1) = \mathsf{Prop}(S_2)$
Loop. If $S_1 \subseteq \mathsf{Prop}(S_0), \ldots, S_k \subseteq \mathsf{Prop}(S_{k-1})$, and $S_0 \subseteq \mathsf{Prop}(S_k)$, then $\mathsf{Prop}(S_i) = \mathsf{Prop}(S_j)$ for all $i, j \in \{0, \ldots, k\}$

These Loop and Cumulative properties together are a well-known weakening of monotonicity (see (Kraus, Lehmann, and Magidor 1990)).

Syntax, Semantics, and Base Axioms

Now let us state formally the specific logic and neural network semantics we consider. Let $p, q, \ldots$ be finitely many atomic variables. These represent fixed states, corresponding to features in the external world that we know ahead of time. Usually these are input and output states, although they could be intermediate "hidden" states if we know these features ahead of time. For example, p might be the set of neurons that represent the color *pink*. For more complex formulas,

Definition 4. Formulas of our language $\mathcal{L}^*$ are given by

$$\varphi, \psi ::= p \mid \neg\varphi \mid \varphi \wedge \psi \mid \mathbf{K}\varphi \mid \mathbf{T}\varphi \mid [\varphi]\psi$$

We define $\top, \bot, \vee, \to, \leftrightarrow$, and the dual modal operators $\langle\mathbf{K}\rangle, \langle\mathbf{T}\rangle, \langle\varphi\rangle$ in the usual way. Additionally, let our base language $\mathcal{L}$ consist of all the $[\varphi]$-free formulas in $\mathcal{L}^*$.

First, let us give intuitive readings for these formulas; we will soon ground these readings with formal semantics. $\mathbf{K}\varphi$ reads "the agent *knows* φ," and $\mathbf{T}\varphi$ reads "the *typical* φ" or sententially as "*typically* φ." Note that we can express conditionals $\varphi \Rightarrow \psi$ in this language as $\mathbf{T}\varphi \to \psi$, i.e., "the typical φ is ψ." Finally, the dynamic operator $[\varphi]\psi$ reads "after the agent learns φ, ψ holds." In Section 6, we will discuss the sense in which this learning is a plausibility upgrade.

Formally, a model for this logic is just a net $\mathcal{N} \in \mathsf{Net}$ along with an interpretation function $[\![\cdot]\!]_{\mathcal{N}} : \mathcal{L} \to \mathsf{State}$ that maps each formula to the set of neurons it denotes. We drop the subscript when $\mathcal{N}$ is clear from context. For the base logic formulas (over $\mathcal{L}$), the idea is to map possible knowledge $\langle \mathbf{K} \rangle$ to Reach and possible typicality $\langle \mathbf{T} \rangle$ to Prop. In the next section, we will explain how we map $[\varphi]$ to Hebbian learning.

Definition 5. The semantics of our base logic is given recursively, using the dual forms $\langle \mathbf{K} \rangle$ and $\langle \mathbf{T} \rangle$, as follows.

$$\begin{array}{lcl} [\![p]\!] & \in & \mathsf{State} \text{ is fixed, nonempty} \\ [\![\neg\varphi]\!] & = & [\![\varphi]\!]^{\complement} \\ [\![\varphi \wedge \psi]\!] & = & [\![\varphi]\!] \cap [\![\psi]\!] \\ [\![\langle \mathbf{K} \rangle \varphi]\!] & = & \mathsf{Reach}([\![\varphi]\!]) \\ [\![\langle \mathbf{T} \rangle \varphi]\!] & = & \mathsf{Prop}([\![\varphi]\!]) \end{array}$$

To better understand this mapping, think of the net $\mathcal{N}$ loosely as a kind of possible worlds model. Its edges represent epistemic accessibility; the nodes reachable from $[\![\varphi]\!]$ are the nodes where it's *possible to know* φ. Similarly, the propagation of $[\![\varphi]\!]$, determined by the weights of the net, constrains which nodes it is *possible to find typical* of φ. In this sense, the propagation $\mathsf{Prop}([\![\varphi]\!])$ is dual to the notion of a *prototype* in psychology (Murphy 2004).

So far, we have defined formulas in terms of the features they denote, e.g., the color *pink*. But we can also read formulas sententially, e.g., "the current state is pink." We consider the sentence φ to be *true* in a net $\mathcal{N}$ as follows.

Definition 6. $\mathcal{N} \models \varphi$ iff $[\![\varphi]\!]_{\mathcal{N}} = N$

An important special case is the conditional $\varphi \Rightarrow \psi$. As before, we express this (over $\mathcal{L}$) as $\mathbf{T}\varphi \to \psi$. Observe that this is true whenever $\mathsf{Prop}([\![\varphi]\!]) \supseteq [\![\psi]\!]$, which tells us whether, given input $[\![\varphi]\!]$, the net's propagated state includes $[\![\psi]\!]$. So $\varphi \Rightarrow \psi$ conveniently expresses "the net *classifies* φ as ψ."

We define the logic's proof system in the usual way (the following applies to both $\mathcal{L}$ and $\mathcal{L}^*$). We have $\vdash \varphi$ iff either φ is an axiom, or φ follows from previously obtained formulas by one of the inference rules. For a set of formulas Γ, $\Gamma \vdash \varphi$ holds if there exist finitely many $\psi_1, \ldots, \psi_k \in \Gamma$ such that $\vdash \psi_1 \wedge \cdots \wedge \psi_k \to \varphi$. A set Γ is consistent if $\Gamma \nvdash \bot$.

We now list axioms for the base logic. We should emphasize that these axioms may not be complete, and we leave this question to future work. Rather, our focus in this paper is on the completeness of the dynamic operator $[\varphi]$, *given* the completeness of the base logic.

For $\mathbf{T}$ alone, (Leitgeb 2001) proves that the properties in Proposition 2 are complete for Prop over binary, feedforward nets. We transcribe these into our modal language.

Nec. From $\vdash \varphi$ we can infer $\vdash \mathbf{T}\varphi$
Dual. $\langle \mathbf{T} \rangle \varphi \leftrightarrow \neg \mathbf{T} \neg \varphi$
Refl. $\mathbf{T}\varphi \to \varphi$
Trans. $\mathbf{T}\varphi \to \mathbf{T}\mathbf{T}\varphi$
Cumulative. $(\varphi \to \psi) \wedge (\mathbf{T}\psi \to \varphi) \to (\mathbf{T}\varphi \to \psi)$
Loop. $(\mathbf{T}\varphi_0 \to \varphi_1) \wedge \cdots \wedge (\mathbf{T}\varphi_k \to \varphi_0) \to (\mathbf{T}\varphi_0 \to \varphi_k)$

But remember that our nets are also fully connected! So we need to modify the model construction from (Leitgeb 2001) by introducing a zero weight edge between every pair of previously unconnected nodes. Note that this change does not affect the Prop-structure of the net.

As for $\mathbf{K}$, we at least have the following sound axioms, transcribed from Proposition 1.

Nec. From $\vdash \varphi$ we can infer $\vdash \mathbf{K}\varphi$
Dual. $\langle \mathbf{K} \rangle \varphi \leftrightarrow \neg \mathbf{K} \neg \varphi$
Distr. $\mathbf{K}(\varphi \to \psi) \leftrightarrow (\mathbf{K}\varphi \to \mathbf{K}\psi)$
Refl. $\mathbf{K}\varphi \to \varphi$
Trans. $\mathbf{K}\varphi \to \mathbf{K}\mathbf{K}\varphi$

These axioms are the usual complete axioms for normal modal logic over reflexive and transitive frames. So far these axioms seem innocuous enough. But for completeness, the catch is that $\mathbf{K}$ and $\mathbf{T}$ may interact in ways that affect the model construction. For example, the following axiom is easy to check, but it is not clear whether it is sufficient.

Incl. $\mathbf{K}\varphi \to \mathbf{T}\varphi$

4 Dynamics of Iterated Hebbian Update

Single-Step Hebbian Update

The plan from here is to extend this base logic with a dynamic operator $[\varphi]$ for Hebbian update. Hebb's classic learning rule (Hebb 1949) states that when two adjacent neurons are simultaneously and persistently active, the connection between them strengthens. In contrast with, e.g., backpropagation, Hebbian learning is errorless and unsupervised. Another key difference is that Hebbian update is local — the change in a weight $\Delta W(m, n)$ depends only on the activation of the immediately adjacent neurons. For this reason, the Hebbian family of learning policies is often considered more biologically plausible than backpropagation. There are many variations of Hebbian learning, but we only consider the most basic form of Hebb's rule: $\Delta W(m, n) = \eta x_m x_n$, where η is the learning rate and x_m, x_n are the outputs of adjacent neurons m and n. Note that this is the *unstable* variation of Hebb's rule; repeatedly applying the rule will make the weights arbitrarily large. In this paper, we will not consider stabilizing variants such as Oja's rule (Oja 1982).

Before we formalize iterated Hebbian learning, let us consider a single step of Hebbian update Hebb. Given a net $\mathcal{N}$ and a state S, we first forward-propagate S through $\mathcal{N}$. Intuitively, $\mathsf{Hebb}(\mathcal{N}, S)$ returns the net that we obtain by applying Hebb's rule: Any edges that are involved in the propagated activation pattern simply have their weights strengthened. This is illustrated in Figure 3.

Definition 7. Let $\mathsf{Hebb} : \mathsf{Net} \times \mathsf{State} \to \mathsf{Net}$ be given by $\mathsf{Hebb}(\langle N, E, W, A, \eta \rangle, S) = \langle N, E, W', A, \eta \rangle$, where

$$W'(m, n) = W(m, n) + \eta \cdot \chi_{\mathsf{Prop}(S)}(m) \cdot \chi_{\mathsf{Prop}(S)}(n)$$

For propositions p we define $[\![p]\!]_{\mathsf{Hebb}(\mathcal{N},S)} = [\![p]\!]_{\mathcal{N}}$.

Note that Hebb does not affect the edges or activation function. This means the resulting net is still binary, feedforward, and fully connected, and so Hebb is well-defined. This also means Hebb does not affect the Reach operator.

Proposition 3. $\mathsf{Reach}_{\mathsf{Hebb}(\mathcal{N},A)}(B) = \mathsf{Reach}_{\mathcal{N}}(B)$

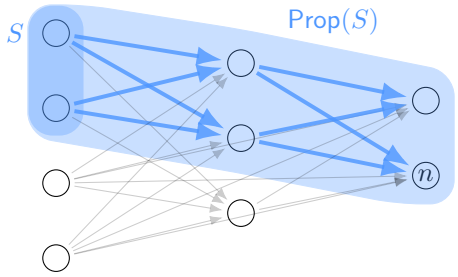

Figure 3: Hebb strengthens those edges whose neurons are active in $\mathsf{Prop}(S)$. The fixed-point operator Hebb^* repeats this update until the edges are "maximally" high.

The following is easy to see (since $\eta \geq 0$).

Proposition 4. Let $m, n \in N$. We have:

- $W_{\mathcal{N}}(m,n) \leq W_{\mathsf{Hebb}(\mathcal{N},S)}(m,n)$
- If either $m \notin \mathsf{Prop}(S)$ or $n \notin \mathsf{Prop}(S)$, then

$$W_{\mathsf{Hebb}(\mathcal{N},S)}(m,n) = W_{\mathcal{N}}(m,n)$$

Iterated Hebbian Update

We now turn to iterated Hebbian update Hebb^*. Recall that our single-step Hebbian update is unstable. So if we repeat Hebb on a single input state S, the net's weights within $\mathsf{Prop}(S)$ will be so high that *any* activation pattern that makes contact with $\mathsf{Prop}(S)$ will "rip through" it entirely. Repeating Hebb on S further will not change the Prop-structure, i.e., the update has reached a fixed point. Hebb^* returns the net at this fixed point. Intuitively: If $\varphi = \psi_1 \wedge \cdots \wedge \psi_k$ is a dataset of inputs over which we train our net $\mathcal{N}$, then $\mathsf{Hebb}^*(\mathcal{N}, [\![\varphi]\!])$ is the net that has "fully internalized" this training set φ.

Why do we study the fixed-point update, rather than single-step Hebb? Our main reason is that we have a plan of attack for completeness. As we will see, this fixed point is simple enough to completely describe using just Reach and Prop.

Constructing the Fixed Point. Rather than reason abstractly about the fixed point, we can explicitly define the number of iterations iter needed to reach it. The idea is to set iter to be high enough such that all updated weights $W(m,n)$ overpower any negative weights that would otherwise cancel their effect. The following definition captures the idea of the lowest possible value a weighted sum could have.

Definition 8. Let $n \in N$, and let $\vec{m} = m_1, \ldots, m_k$ list the predecessors of n. The *negative weight score* of n is the sum of all the negative weights of n's predecessors, i.e.,

$$\mathsf{nws}(n) = \sum_{i=1}^{\deg(n)} \begin{cases} W(m_i,n) & \text{if } W(m_i,n) < 0 \\ 0 & \text{otherwise} \end{cases}$$

The *minimum* negative weight score is simply

$$\mathsf{mnws} = \min_{n \in N} \mathsf{nws}(n)$$

Recall that an activation function A has a threshold, i.e., there is some $t \in \mathbb{Q}$ with $A(t) = 1$. We set the number of iterations iter to be exactly

$$\mathsf{iter} = \begin{cases} \lceil \frac{t - |N| \cdot \mathsf{mnws}}{\eta} \rceil & \text{if } \geq 1 \\ 1 & \text{otherwise} \end{cases}$$

Note that iter will always be a positive integer, and so iterating iter times is well-defined. This choice for iter may seem opaque, but we will see in Lemma 1 why it guarantees that the updated weights overpower competing edge weights.

Defining the Update. We are now ready to define Hebb^*. $\mathsf{Hebb}(\mathcal{N}, S)$ simply returns the net that we obtain by applying Hebb's rule iter times.

Definition 9. Let $\mathsf{Hebb}^* : \mathsf{Net} \times \mathsf{State} \to \mathsf{Net}$ be given by $\mathsf{Hebb}^*(\langle N, E, W, A, \eta \rangle, S) = \langle N, E, W', A, \eta \rangle$, where

$$W'(m,n) = W(m,n) + \mathsf{iter} \cdot \eta \cdot \chi_{\mathsf{Prop}(S)}(m) \cdot \chi_{\mathsf{Prop}(S)}(n)$$

Again, for propositions p we define $[\![p]\!]_{\mathsf{Hebb}^*(\mathcal{N},S)} = [\![p]\!]_{\mathcal{N}}$.

Notice that for each iteration, we're always updating by $\mathsf{Prop}(S)$ in the *original* net. We might worry that a single iteration of Hebb would affect $\mathsf{Prop}(S)$, and that we would have to track those changes every iteration. Fortunately, this is not the case. For all $S \in \mathsf{State}$,

Proposition 5. $\mathsf{Prop}_{\mathsf{Hebb}(\mathcal{N},S)}(S) = \mathsf{Prop}_{\mathcal{N}}(S)$

As with Hebb, for Hebb^* we have

Proposition 6. $\mathsf{Reach}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B) = \mathsf{Reach}_{\mathcal{N}}(B)$

Proposition 7. Let $m, n \in N$. We have:

- $W_{\mathcal{N}}(m,n) \leq W_{\mathsf{Hebb}^*(\mathcal{N},S)}(m,n)$
- If either $m \notin \mathsf{Prop}(S)$ or $n \notin \mathsf{Prop}(S)$, then

$$W_{\mathsf{Hebb}^*(\mathcal{N},S)}(m,n) = W_{\mathcal{N}}(m,n)$$

The following fact about Hebb^* is the most important. It is a formal expression of our statement before: Updated weights $W_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$ are so high that if m is active in $\mathsf{Hebb}^*(\mathcal{N}, A)$ then n must be as well.

Lemma 1. Let $A, B \in \mathsf{State}$, $m, n \in N$. If $m \in \mathsf{preds}(n)$, $m, n \in \mathsf{Prop}(A)$, and $m \in \mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$, then

$$A\Big(\sum_{i=1}^{\deg(n)} W_{\mathsf{Hebb}^*(\mathcal{N},A)}(m_i,n) \cdot \chi_{\mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)}(m_i)\Big) = 1$$

Proof Sketch. Since the activation function A has a threshold, $\exists t \in \mathbb{Q}$ with $A(t) = 1$. Since A is nondecreasing, it's enough to show that this weighted sum $\geq t$. From here we can pull the m-term out of the weighted sum, then apply the definition of Hebb^* and the fact that $m, n \in \mathsf{Prop}(S)$, $m \in \mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$ to eventually get

$$\sum_{i=1}^{\deg(n)} W_{\mathsf{Hebb}^*(\mathcal{N},A)}(m_i,n) \cdot \chi_{\mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)}(m_i) \geq |N| \cdot \mathsf{mnws} + \mathsf{iter} \cdot \eta$$

So we just need to show

$$t \leq |N| \cdot \mathsf{mnws} + \mathsf{iter} \cdot \eta$$

but we chose iter to satisfy precisely this inequality! □

Formal Semantics for the Update. We can now spell out the semantics of the dynamic operator $[\varphi]$. Whereas our base operators **K** and **T** are interpreted as *states* in the underlying net, $[\varphi]$ changes the net itself. We formalize this $[\varphi]$ using the standard dynamic epistemic logic trick (Van Ditmarsch, van Der Hoek, and Kooi 2007), i.e.,

$$[\![[\varphi]\psi]\!]_{\mathcal{N}} \quad = \quad [\![\psi]\!]_{\mathsf{Hebb}^*(\mathcal{N},[\![\varphi]\!])}$$

In other words, in order to evaluate $[\![[\varphi]\psi]\!]$, we simply evaluate $[\![[\varphi]\psi]\!]$ in the Hebb^*-updated net.

The Reduction for Hebb^*

Our main technical result is that we can "translate away" $[\varphi]$-formulas by reducing them to formulas in the base logic. To do this, we first need to show how Hebb^* reduces to Reach and Prop. We already have Proposition 6, which says that Hebb^* does not affect Reach. In this section, we prove the following reduction theorem for Hebb^* over Prop.

$$\mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B) = \mathsf{Prop}(B \cup (\mathsf{Prop}(A) \cap \mathsf{Reach}(\mathsf{Prop}(A) \cap \mathsf{Prop}(B)))) \quad (\dagger)$$

This theorem is at the heart of the reduction axioms that we will use to reduce $[\varphi]$ (see Section 4). To prove it, we will first need the following algebraic properties for Hebb^*.

Lemma 2. Let $A, B \in \mathsf{State}$. Hebb^* satisfies the following algebraic properties.

1. $\mathsf{Prop}(A) \cap \mathsf{Prop}(B) \subseteq \mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$
2. $\mathsf{Prop}(A) \cap \mathsf{Reach}(\mathsf{Prop}(A) \cap \mathsf{Prop}(B)) \subseteq \mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$
3. $\mathsf{Prop}(A) \cap \mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B) \subseteq \mathsf{Prop}(A) \cap \mathsf{Reach}(\mathsf{Prop}(A) \cap \mathsf{Prop}(B))$

Proof Sketch. First, let us give some intuition for these properties. Part (2) expresses a lower bound for $\mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$, whereas (3) gives an upper bound; (1) is just used to show (2). We sketch the proof of (1) here, since the other two are similar.

Let $n \in \mathsf{Prop}(A) \cap \mathsf{Prop}(B)$, and proceed by induction on $\mathsf{layer}(n)$. The base step is trivial. At $\mathsf{layer}(n) \geq 0$, we case on the definition of Prop. If $n \in B$, then we just apply Inclusion. Otherwise, n is activated by its predecessors $m_i \in \mathsf{Prop}(B)$ in $\mathcal{N}$. By well-ordering, there is some $m \in \mathsf{Prop}(A) \cap \mathsf{Prop}(B)$ with the smallest layer. Since n is also in this intersection, $\mathsf{layer}(m) \leq \mathsf{layer}(n)$.

Case 1. $\mathsf{layer}(m) < \mathsf{layer}(n)$. Since $\mathcal{N}$ is fully connected, we must have $m \in \mathsf{preds}(n)$. From here we have exactly the right conditions for Lemma 1, from which we have $n \in \mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$.

Case 2. $\mathsf{layer}(m) = \mathsf{layer}(n)$. In this case, we can inductively argue that the weights of n's predecessors in $\mathsf{Hebb}^*(\mathcal{N}, A)$ are the same as their weights in $\mathcal{N}$, which gives us $n \in \mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$. □

We now have everything we need to prove the reduction.

Theorem 3 (Reduction). For all $A, B \in \mathsf{State}$, (†) holds.

Proof Sketch. For all $n \in N$, we show that $n \in$ the left-hand side of (†) iff $n \in$ the right-hand side, by induction on $\mathsf{layer}(n)$. The base case is easy. At $\mathsf{layer}(n) \geq 0$, we show each direction.

($\rightarrow$) Let $n \in \mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$. If $n \in B$, then we just apply Inclusion. Otherwise, n is activated by its predecessors m_i. From here we split into two more cases: If $n \in \mathsf{Prop}(A)$ *and* there is some active predecessor m_i, we apply our inductive hypothesis and part (3) of Lemma 2. Otherwise, we can argue that the two nets have exactly the same predecessor weights.

($\leftarrow$) Let $n \in \mathsf{Prop}(B \cup (\mathsf{Prop}(A) \cap \mathsf{Reach}(\mathsf{Prop}(A) \cap \mathsf{Prop}(B))))$, and case on the definition of Prop. If n is in this inner union term, then we just apply Inclusion or part (2) of Lemma 2, depending on the case. Otherwise, n is activated by its predecessors m_i. From here we just apply our inductive hypothesis and Proposition 7. □

Note that if $\mathsf{Prop}(A) \cap \mathsf{Prop}(B) = \emptyset$, then we have $\mathsf{Reach}(\mathsf{Prop}(A) \cap \mathsf{Prop}(A)) = \emptyset$ as well. If we substitute this into the statement of Theorem 3, we see the following. In words, if $\mathsf{Prop}(A)$ and $\mathsf{Prop}(B)$ never meet, then updating A will have no effect on the propagation of B.

Corollary 1. If $\mathsf{Prop}(A) \cap \mathsf{Prop}(B) = \emptyset$ then

$$\mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B) = \mathsf{Prop}(B)$$

5 Reduction Axioms and Completeness

The payoff of our reduction theorem is that we can now see how to translate $[\varphi]$ sentences into $[\varphi]$-free sentences. It will then follow that iterated Hebbian learning is completely axiomatized by the reduction axioms used in translation, plus whatever axioms the base logic needs. This is known as *completeness by translation* in the dynamic epistemic and doxastic logic literature. See (Van Ditmarsch, van Der Hoek, and Kooi 2007) for an introduction and (Van Benthem 2007) for a discussion on this strategy in belief revision and plausibility upgrade (we draw heavily from both of these sources).

First, we establish reduction axioms for $[\varphi]$.

Theorem 4 (Reduction Axioms). The following are sound.

$$\begin{array}{lcll} [\varphi]p & \leftrightarrow & p & \text{for propositions } p \\ [\varphi]\neg\psi & \leftrightarrow & \neg[\varphi]\psi & \\ [\varphi](\psi \wedge \rho) & \leftrightarrow & [\varphi]\psi \wedge [\varphi]\rho & \\ [\varphi]\mathbf{K}\psi & \leftrightarrow & \mathbf{K}[\varphi]\psi & \\ [\varphi]\mathbf{T}\psi & \leftrightarrow & \mathbf{T}([\varphi]\psi \wedge (\mathbf{T}\varphi \vee \mathbf{K}(\mathbf{T}\varphi \vee \mathbf{T}[\varphi]\psi))) & \end{array}$$

Proof Sketch. We check that each left-hand-side φ and right-hand-side ψ have the same interpretation $[\![\varphi]\!]_{\mathcal{N}} = [\![\psi]\!]_{\mathcal{N}}$. The first three cases are routine. The $\langle\mathbf{K}\rangle$ case follows immediately from Proposition 6, and the $\langle\mathbf{T}\rangle$ case follows immediately from Theorem 3 (the reduction theorem). □

Notice that these axioms compositionally break down the postconditions after $[\varphi]$, and "push in" the $[\varphi]$ operator in each case. Given a set of formulas $\Gamma \subseteq \mathcal{L}^*$, we can use these axioms to "translate away" all instances of dynamic formulas $[\varphi]\psi$ in Γ, resulting in $\Gamma^{\mathrm{tr}} \subseteq \mathcal{L}$.

It's easy to see intuitively how this translation should go. For example, given the formula $[p]([p]\mathbf{T}q \wedge \mathbf{K}p) \in \Gamma$, we

would recursively apply our reduction axioms, pushing $[p]$ further into the expression until we can eliminate the propositional cases $[p]q$ and $[p]p$. It is also intuitively clear that this process terminates, and we skip it here. But beware — actually proving that this process terminates is not trivial at all; see (Baltag, Moss, and Solecki 2023).

Observe that the formulas $\varphi \in \Gamma$ are provably equivalent to their translations $\varphi' \in \Gamma^{\mathrm{tr}}$. So a net models $\Gamma \subseteq \mathcal{L}^*$ iff it models $\Gamma^{\mathrm{tr}} \subseteq \mathcal{L}$. This means that our nets already contain all information about what they learn after iterated Hebbian update. Moreover, model building over $\mathcal{L}^*$ follows from model building over our base language $\mathcal{L}$.

Theorem 5 (**Model Building**)**.** Suppose that we have model building for our base language $\mathcal{L}$, i.e., for all consistent $\Gamma \subseteq \mathcal{L}$ there is a net $\mathcal{N} \in \mathsf{Net}$ such that $\mathcal{N} \models \Gamma$. Then we have model building for our dynamic language as well: for all $\Gamma \subseteq \mathcal{L}^*$, there is $\mathcal{N}$ such that $\mathcal{N} \models \Gamma$.

Assuming we have completeness for the base logic, completeness for $\mathcal{L}^*$ then follows from model building.

Theorem 6 (**Completeness**)**.** Suppose we have a complete axiomatization for $\langle \mathbf{K} \rangle$ and $\langle \mathbf{T} \rangle$. Then the logic of iterated Hebbian learning $[\varphi]$ is completely axiomatized by these laws, plus the above reduction axioms: for all consistent $\Gamma \subseteq \mathcal{L}^*$, if $\Gamma \models \varphi$ then $\Gamma \vdash \varphi$.

Proof. Suppose contrapositively that $\Gamma \nvdash \varphi$. Then $\Gamma \cup \{\neg\varphi\}$ is consistent. We can apply the translation above to $\Gamma \cup \{\neg\varphi\}$ to obtain $\Gamma^{\mathrm{tr}} \subseteq \mathcal{L}$. Since we assumed the base logic is complete, we have a net $\mathcal{N} \models \Gamma^{\mathrm{tr}}$. By Theorem 5, $\mathcal{N} \models \Gamma \cup \{\neg\varphi\}$. But then $\mathcal{N} \models \Gamma$ yet $\mathcal{N} \not\models \varphi$, which is what we wanted to show. □

6 Discussion

Interpreting the Reduction Axioms

One major goal of neuro-symbolic AI is to make neural networks and their learning algorithms more interpretable. Neural network semantics provides a direct way to do this, by proving correspondences between neural network operators and more interpretable logical operators. We have shown in particular that iterated Hebbian update Hebb^* corresponds to a dynamic operator $[\varphi]$ that is characterized by our reduction axioms. What do these reduction axioms teach us about iterated Hebbian learning?

First, notice the form of these axioms. Each expresses what is true *after* the net learns φ in terms of what was true *before* learning φ. But the only operator that changes is typicality $\mathbf{T}$. So we can think of Hebb^* as a plausibility upgrade operator. The final line

$$[\varphi]\mathbf{T}\psi \leftrightarrow \mathbf{T}([\varphi]\psi \wedge (\mathbf{T}\varphi \vee \mathbf{K}(\mathbf{T}\varphi \vee \mathbf{T}[\varphi]\psi)))$$

reveals the plausibility upgrade policy that Hebb^* uses.

Let's unpack this. This axiom says that whether the agent found ψ plausible *before* learning φ, *after* learning φ she now also expects this $\mathbf{T}\varphi \vee \mathbf{K}(\mathbf{T}\varphi \vee \mathbf{T}[\varphi]\psi)$ term to be true. And the $\leftrightarrow$ indicates that the agent learns *only* this term. So what exactly has she learned to think is plausible? This complicated inner term states: Either (1) the agent found φ plausible in the first place (i.e., she learns nothing), or (2) she now *knows* about her prior expectations regarding φ and ψ.

So iterated Hebbian learning revises an agent's plausibility beliefs by expanding what she *thinks she can plausibly introspect on*. And although this is a mouthful, with some effort these reduction axioms give a human-interpretable description of what a Hebbian learner learns.

Why Bother with Completeness?

To our knowledge, Theorem 6 is the first ever completeness theorem for any learning policy on neural networks. Soundness alone for neural networks is interesting in its own right, since sound axioms give us formally verified guarantees about the neural network's behavior (Albarghouthi et al. 2021; d'Avila Garcez, Broda, and Gabbay 2001).

But for neural network semantics, completeness has an important practical consequence. Completeness is equivalent to neural network model building, i.e., building a neural network that obeys a set of constraints Γ. Using $[\varphi]$, our constraints Γ can express guarantees about the net at the fixed point of Hebbian learning. For example, we can build a net that models $(\mathbf{T}\psi \to \rho) \wedge [\varphi](\mathbf{T}\psi \to \rho)$, which says that the net classifies the input ψ as ρ, and iterated Hebbian learning preserves this fact.

The importance of this for learning policies used in practice (e.g., backpropagation) is hard to understate. The problem of AI alignment is largely a matter of building neural networks with these kinds of guarantees. But this idea is in its early stages, and many crucial details still need to be worked out. For example, we often want to build neural networks that obey constraints *at each update step*, rather than at some theoretical fixed point. In the next section, we mention future research directions that could make this approach more useful in practice.

7 Conclusions and Future Directions

In this paper, we mapped a dynamic logic operator $[\varphi]$ to a simple neural network learning policy, iterated Hebbian learning. We gave reduction axioms that "translate away" $[\varphi]$-formulas; consequently, completeness for iterated Hebbian learning follows from completeness for the base logic.

Our reduction axioms characterize iterated Hebbian learning as a type of plausibility upgrade. This raises the question of whether there a general correspondence between neural network learning and plausibility upgrade policies.

Our work also provides proof of concept that we can build neural networks that obey constraints on their learning. But more work needs to be done to make this useful in practice. As we mentioned before, we often want guarantees for what the neural network learns *at each step*. What we would need is a complete logic for single-step Hebb, but this is non-trivial.

It is natural to consider whether this kind of logical analysis could be applied to backpropagation. This is currently an open question, and it is a major long-term goal of neural network semantics. This will require new ideas, since the framework has yet to address supervised learning and convergence.

Acknowledgements

C. Schultz Kisby and S. Blanco were supported in part by the US Department of Defense [Contract No. W52P1J2093009]. We also thank the anonymous reviewers for their helpful suggestions.

References

Albarghouthi, A.; et al. 2021. Introduction to neural network verification. *Foundations and Trends® in Programming Languages*, 7(1–2): 1–157.

Bader, S.; and Hitzler, P. 2005. Dimensions of neural-symbolic integration-a structured survey. In *We Will Show Them! Essays in Honour of Dov Gabbay, Volume 1*, 167–194. College Publications.

Balkenius, C.; and Gärdenfors, P. 1991. Nonmonotonic inferences in neural networks. In *KR*, 32–39. Morgan Kaufmann.

Baltag, A.; Gierasimczuk, N.; Özgün, A.; Sandoval, A. L. V.; and Smets, S. 2019. A dynamic logic for learning theory. *Journal of Logical and Algebraic Methods in Programming*, 109: 100485.

Baltag, A.; Li, D.; and Pedersen, M. Y. 2019. On the right path: a modal logic for supervised learning. In *International Workshop on Logic, Rationality and Interaction*, 1–14. Springer.

Baltag, A.; Moss, L. S.; and Solecki, S. 1998. The logic of public announcements, common knowledge, and private suspicions. In *Proceedings of the 7th conference on Theoretical aspects of rationality and knowledge*, 43–56.

Baltag, A.; Moss, L. S.; and Solecki, S. 2023. Logics for epistemic actions: completeness, decidability, expressivity. *Logics*, 1(2): 97–147.

Baltag, A.; and Smets, S. 2009. Group belief dynamics under iterated revision: fixed points and cycles of joint upgrades. In *Proceedings of the 12th Conference on Theoretical Aspects of Rationality and Knowledge*, 41–50.

Besold, T.; d'Avila Garcez, A.; Bader, S.; et al. 2017. Neural-Symbolic Learning and Reasoning: A Survey and Interpretation. In *Neuro-Symbolic Artificial Intelligence*.

Blutner, R. 2004. Nonmonotonic inferences and neural networks. *Synthese*, 142: 143–174.

d'Avila Garcez, A.; Broda, K.; and Gabbay, D. M. 2001. Symbolic knowledge extraction from trained neural networks: A sound approach. *Artificial Intelligence*, 125(1-2): 155–207.

Giordano, L.; Gliozzi, V.; and Theseider Dupré, D. 2022. A conditional, a fuzzy and a probabilistic interpretation of self-organizing maps. *Journal of Logic and Computation*, 32(2): 178–205.

Giordano, L.; and Theseider Dupré, D. 2021. Weighted defeasible knowledge bases and a multipreference semantics for a deep neural network model. In *Logics in Artificial Intelligence: 17th European Conference, JELIA 2021, Virtual Event, May 17–20, 2021, Proceedings 17*, 225–242. Springer.

Gross, C. G. 2002. Genealogy of the "grandmother cell". *The Neuroscientist*, 8(5): 512–518.

Harmelen, F. 2022. Preface: The 3rd AI Wave Is Coming, and It Needs a Theory. In *Neuro-Symbolic Artificial Intelligence: The State of the Art*, V–VII. IOS Press BV.

Hebb, D. 1949. *The Organization of Behavior*. Psychology Press.

Kisby, C.; Blanco, S.; and Moss, L. 2022. The Logic of Hebbian Learning. In *The International FLAIRS Conference Proceedings*, volume 35.

Kraus, S.; Lehmann, D.; and Magidor, M. 1990. Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial intelligence*, 44(1-2): 167–207.

Leitgeb, H. 2001. Nonmonotonic reasoning by inhibition nets. *Artificial Intelligence*, 128(1-2): 161–201.

Leitgeb, H. 2003. Nonmonotonic reasoning by inhibition nets II. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 11(supp02): 105–135.

Leitgeb, H. 2018. Neural Network Models of Conditionals. In *Introduction to Formal Philosophy*, 147–176. Springer.

McCulloch, W. S.; and Pitts, W. 1943. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4): 115–133.

Moura, L. d.; and Ullrich, S. 2021. The Lean 4 theorem prover and programming language. In *Automated Deduction–CADE 28: 28th International Conference on Automated Deduction, Virtual Event, July 12–15, 2021, Proceedings 28*, 625–635. Springer.

Murphy, G. 2004. *The big book of concepts*. MIT press.

Odense, S.; and d'Avila Garcez, A. S. 2022. A Semantic Framework for Neural-Symbolic Computing. *ArXiv*, abs/2212.12050.

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

Plaza, J. A. 2007. Logics of public communications. *Synthese*, 158: 165–179.

Sarker, M. K.; Zhou, L.; Eberhart, A.; and Hitzler, P. 2022. Neuro-Symbolic Artificial Intelligence: Current Trends. *AI Communications*, 34.

Srivastava, R. K.; Greff, K.; and Schmidhuber, J. 2015. Training Very Deep Networks. In Cortes, C.; Lawrence, N.; Lee, D.; Sugiyama, M.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Van Benthem, J. 2007. Dynamic logic for belief revision. *Journal of applied non-classical logics*, 17(2): 129–155.

Van Benthem, J. 2011. *Logical dynamics of information and interaction*. Cambridge University Press.

Van Ditmarsch, H.; van Der Hoek, W.; and Kooi, B. 2007. *Dynamic epistemic logic*, volume 337. Springer.