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

Anonymous submission

Abstract

This paper is a contribution to neural network semantics, a theoretical framework for neuro-symbolic AI. The key insight of this theory is that operators on neural network states can be formalized as logical operators via a straightforward semantic interpretation. Here, we do the same for a neural network *learning* operator. We consider a simple learning policy on neural networks: unstable Hebbian learning. We interpret the dynamic operator $\langle\varphi\rangle$ as iterated Hebbian update, i.e., "update the net by repeatedly applying Hebb's learning rule until a fixed-point." Our main result[1] is that we can "translate away" $\langle\varphi\rangle$-formulas via reduction axioms. This means that completeness for the logic of unstable Hebbian learning follows from completeness of the base logic. To our knowledge, this result is the first ever completeness theorem for any learning policy on neural networks. Finally, we discuss the interpretability of the reduction axioms, as well as the importance of completeness for logics of this kind.

1 Introduction

The two dominant paradigms of AI, connectionist neural networks and symbolic systems, have long been 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).

Enter neural network semantics, an up-and-coming semantic theory for neuro-symbolic AI. The key insight of this framework is that we can take the states of a neural network as the semantics for a formal logic. We then interpret logical operators straightforwardly as operators on neural network states. The central question is: Which neural operators correspond to which logical operators?

Consider for example the *forward propagation* operator $\mathsf{Prop} : \mathsf{State} \to \mathsf{State}$ 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 major result from (Leitgeb 2001) is this: Say we interpret conditional belief $[\mathbf{B}](\varphi, \psi)$ as

$$\mathcal{N} \models [\mathbf{B}](\varphi, \psi) \text{ iff } \mathsf{Prop}([\![\varphi]\!]) \supseteq \psi$$

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). This means that forward propagation corresponds to a kind of conditional belief.

In this paper, we do the same for a simple neural network *learning* operator. In particular, we consider unstable Hebbian learning ("neurons that fire together wire together") on a binary, feed-forward net. Analogous to the fixed-point operator $\mathsf{Prop} : \mathsf{State} \to \mathsf{State}$ for forward propagation, we model iterated Hebbian update as a fixed-point operator $\mathsf{Hebb}^* : \mathsf{Net} \times \mathsf{State} \to \mathsf{Net}$.

Our main result is that iterated Hebbian update Hebb^* corresponds precisely to a certain dynamic operator $\langle\varphi\rangle$. We show that we can "translate away" $\langle\varphi\rangle$-formulas by reducing them to formulas that reason only about forward propagation and graph reachability. It follows that unstable Hebbian learning is completely axiomatized by the reduction axioms we used in translation, plus whatever axioms the base logic needs. In other words, the reduction axioms give a complete and human-interpretable description of what an (unstable) Hebbian learner learns.

The rest of this paper is organized as follows. In Section 2 we introduce the base logic we use along with its neural network semantics. In Section 3 we define iterated Hebbian update Hebb^* and prove our main reduction theorem. In Section 4, we give the reduction axioms and show how completeness for $\langle\varphi\rangle$ follows from completeness for the base logic. In Section 5, we give a human-interpretable reading of these axioms and discuss the importance of completeness. In Section 6 we compare our work to related research. We summarize the paper and point to future research in Section 7.

[1] 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 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 indegree (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 require an analogous coupling of fuzzy activations with fuzzy-valued logic (left to future work).

Often "fully connected" means that there is an edge from every node m to every node in the following layer n. But here we mean something much stronger: the graph is fully connected, including "highway edges" that cut between layers (inspired by highway networks (Srivastava, Greff, and Schmidhuber 2015)). This assumption is crucial for our results about 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.

Definition 2. For every neuron $n \in N$, we define its *layer* in the net, $\mathsf{layer} : N \to \mathbb{N}$ as follows.

$$\mathsf{layer}(n) = \begin{cases} 0 & \text{if } n \text{ has no predecessors} \\ k & \text{otherwise, where } k \text{ is the maximal} \\ & \text{length of a path from any node } m \\ & \text{to } n, \text{with } \mathsf{layer}(m) = 0 \end{cases}$$

We have the following fact about layer. The $(\to)$ direction holds by definition, whereas the $(\leftarrow)$ direction follows from the fact that $\mathcal{N}$ is fully connected.

Proposition 1. For all $m, n \in N$,

$$m \in \mathrm{preds}(n) \text{ iff } \mathsf{layer}(m) < \mathsf{layer}(n)$$

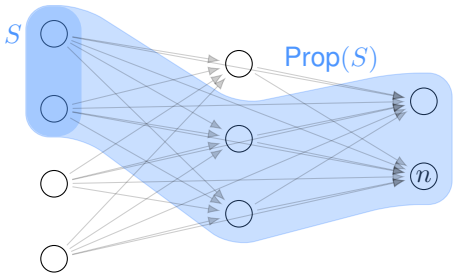

Figure 1: The forward propagation operator Prop.

Forward Propagation and Reachability

For our base logic (without update), we will need two operators on neural network states: Forward propagation Prop and (conditional) graph reachability Reach. But first, we need to specify what we mean by the *state* of a net. In general, 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.

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

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

Definition 3. Let $S \in \mathsf{State}$. The characteristic function $\chi_S : N \to \{0, 1\}$ is given by $\chi_S(n) = 1$ iff $n \in S$.

We are now ready to define Prop. As we mentioned before, the idea is that active neurons in a state S successively activate new neurons until eventually the state of the net stabilizes. $\mathsf{Prop}(S)$ is the fixed point of this process, the result of "propagating S forward" (illustrated in Figure 1).

Definition 4. 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 $\mathsf{Prop}(S)$.

Notice that the recursive call is somewhat obscured; it happens within $\chi_{\mathsf{Prop}_{\mathcal{N}}(S)}$. Prop is well-founded, since all predecessors $m_i \in \mathrm{preds}(n)$ have $\mathsf{layer}(m_i) < \mathsf{layer}(n)$.

We have the following properties for Prop. These come from (Leitgeb 2001), which actually proved that these properties hold for a closure operator Cl over inhibition nets, i.e., weightless nets with both excitatory and inhibitory connections. But this paper also proves that inhibition nets and binary, feed-forward nets are equivalent. Additionally, our Prop is exactly Cl. And so we import the result here.

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

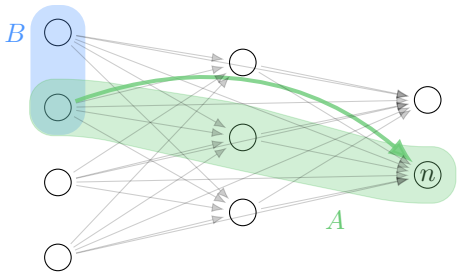

Figure 2: The conditional reachability operator Reach.

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 $i, j \in \{0, \ldots, k\}$

Prop is not monotonic; 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 $\mathsf{Prop}(A)$. But the Loop and Cumulative properties serve as a weakened form of monotonicity (see (Kraus, Lehmann, and Magidor 1990)).

The Reach operator is conditional graph-reachability: $\mathsf{Reach}(A, B)$ returns the set of all neurons reachable from B via paths running entirely through A (illustrated in Figure 2).

Definition 5. Let $\mathsf{Reach} : \mathsf{State} \times \mathsf{State} \to \mathsf{State}$ be defined as follows. $n \in \mathsf{Reach}(A, B)$ iff $\exists m \in B$ with an E-path from m to n, where all nodes on this path are in A.

The following properties for Reach are easy to check.

Proposition 3. For all $A, B, C \in \mathsf{State}$,

Layer 0. If $\mathsf{layer}(n) = 0$, $n \in \mathsf{Reach}(A, B)$ implies $n \in B$
Empty. If $A \cap B = \emptyset$ then $\mathsf{Reach}(A, B) = \emptyset$
Subsumption. $\mathsf{Reach}(A, B) \subseteq A$
Inclusion. $A \cap B \subseteq \mathsf{Reach}(A, B)$
Idempotent. $\mathsf{Reach}(A, \mathsf{Reach}(A, B)) = \mathsf{Reach}(A, B)$
Monotonic. If $B \subseteq C$ then $\mathsf{Reach}(A, B) \subseteq \mathsf{Reach}(A, C)$
Closed under union.
$\mathsf{Reach}(A, B \cup C) = \mathsf{Reach}(A, B) \cup \mathsf{Reach}(A, C)$

Unlike Prop, Reach is not a very interesting operator in its own right. It is just a conditionalized closure operator. But we include it because our reduction depends on it. As we will see in Section 3, conditional graph reachability is necessary for reasoning about Hebbian learning!

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 propositional variables, representing fixed states that we know ahead of time. These may be sets of neurons in the input or output layers, but could also include neurons in the hidden layer if we know what features these states represent. For more complex formulas,

Definition 6. Formulas of our language $\mathcal{L}^*$ are given by

$$\varphi, \psi ::= p \mid \neg\varphi \mid \varphi \wedge \psi \mid \langle \mathbf{K} \rangle(\varphi, \psi) \mid \langle \mathbf{B} \rangle \varphi \mid \langle \varphi \rangle \psi$$

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

For the rest of this section, we focus on the base logic (over $\mathcal{L}$). A model for this logic is just a net $\mathcal{N} \in \mathsf{Net}$ along with an interpretation function $[\![\cdot]\!]_{\mathcal{N}} : \mathcal{L} \to \mathsf{State}$. (We drop the subscript when $\mathcal{N}$ is clear from context.) The idea for $[\![\cdot]\!]_{\mathcal{N}}$ is simply to map $\langle \mathbf{K} \rangle$ to Reach and $\langle \mathbf{B} \rangle$ to Prop.

Definition 7. The semantics of our base logic is given recursively as follows.

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

Since Reach is a well-behaved closure operator, we can give $\langle \mathbf{K} \rangle(\varphi, \psi)$ an epistemic reading: "given φ, the agent possibly knows ψ." Since Prop is non-monotonic (it is revised when presented with new information), we give $\langle \mathbf{B} \rangle \varphi$ a doxastic reading: "the agent possibly believes φ." The dual $[\mathbf{B}]\varphi \equiv \neg\langle \mathbf{B} \rangle \neg\varphi$ expresses that the agent *must* believe φ. (The dual of $\langle \mathbf{K} \rangle(\varphi, \psi)$ is not such an easy story, but we leave this for future work.)

We consider φ to be true in a net $\mathcal{N}$ as follows.

Definition 8. $\mathcal{N} \models \varphi$ iff $[\![\varphi]\!]_{\mathcal{N}} = N$

For example, material implication $\mathcal{N} \models \varphi \to \psi$ holds iff $[\![\varphi \to \psi]\!] = [\![\varphi]\!]^{\mathsf{C}} \cup [\![\psi]\!] = N$, which holds iff $[\![\varphi]\!] \subseteq [\![\psi]\!]$. Using this trick, we can express conditional belief $[\mathbf{B}](\varphi, \psi)$ from (Leitgeb 2001) as $[\mathbf{B}]\varphi \to \psi$. In Leitgeb's semantics, $\mathcal{N} \models [\mathbf{B}](\varphi, \psi)$ iff $\mathsf{Prop}([\![\varphi]\!]) \supseteq [\![\psi]\!]$. Now observe that

$$\begin{array}{lll} \mathcal{N} \models [\mathbf{B}](\varphi, \psi) & \text{iff} & \mathsf{Prop}([\![\varphi]\!]) \supseteq [\![\psi]\!] \\ & \text{iff} & \mathcal{N} \models \psi \to \langle \mathbf{B} \rangle \varphi \\ & \text{iff} & \mathcal{N} \models [\mathbf{B}]\varphi \to \psi \end{array}$$

We define the 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 $\Gamma \nvdash \bot$.

Since our main focus in this paper is $\langle \varphi \rangle$, we won't worry about giving the complete axioms for $\langle \mathbf{K} \rangle$ and $\langle \mathbf{B} \rangle$. We take completeness of the base system for granted, and leave it to future work. But we should emphasize that completeness for the base system may be tricky. The story for $\langle \mathbf{B} \rangle$ alone is easy. (Leitgeb 2001) proves that the properties in Proposition 2 are complete for Prop, and we can just transcribe these into the modal language. Here we give the axioms for the dual $[\mathbf{B}]$.

Nec. From $\vdash \varphi$ we can infer $\vdash [\mathbf{B}]\varphi$
Dual. $\langle \mathbf{B} \rangle \varphi \leftrightarrow \neg[\mathbf{B}]\neg\varphi$
Refl. $[\mathbf{B}]\varphi \to \varphi$
Trans. $[\mathbf{B}]\varphi \to [\mathbf{B}][\mathbf{B}]\varphi$

Cumulative. $(\varphi \to \psi) \wedge ([\mathbf{B}]\psi \to \varphi) \to ([\mathbf{B}]\varphi \to \psi)$
Loop. $([\mathbf{B}]\varphi_0 \to \varphi_1) \wedge \cdots \wedge ([\mathbf{B}]\varphi_k \to \varphi_0) \to ([\mathbf{B}]\varphi_0 \to \varphi_k)$

On the other hand, for $\langle\mathbf{K}\rangle$ we at least have the following sound axioms (transcribed from Proposition 3).

Empty. $(\varphi \wedge \psi \leftrightarrow \bot) \to (\langle\mathbf{K}\rangle(\varphi,\psi) \leftrightarrow \bot)$
Subsump. $\langle\mathbf{K}\rangle(\varphi,\psi) \to \varphi$
Refl. $\varphi \wedge \psi \to \langle\mathbf{K}\rangle(\varphi,\psi)$
Trans. $\langle\mathbf{K}\rangle(\varphi,\langle\mathbf{K}\rangle(\varphi,\psi)) \to \langle\mathbf{K}\rangle(\varphi,\psi)$
Distr. $\langle\mathbf{K}\rangle(\varphi,\psi \vee \rho) \leftrightarrow (\langle\mathbf{K}\rangle(\varphi,\psi) \vee \langle\mathbf{K}\rangle(\varphi,\rho))$

This is likely not a complete list. For instance, we don't know what the appropriate necessitation rule and dual axioms should be for $\langle\mathbf{K}\rangle$. But because Reach is so well-behaved, we conjecture that there *is* a complete set of axioms for $\langle\mathbf{K}\rangle$. Finally, the trickiest step is combining $\langle\mathbf{B}\rangle$ and $\langle\mathbf{K}\rangle$, since the two may interact in ways not discussed here.

3 Dynamics of (Unstable) Hebbian Update

Single-Step Hebbian Update

The plan from here is to extend this base logic with a dynamic operator $\langle\varphi\rangle$ 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, i.e., 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).

Our goal is to model iterated Hebbian update Hebb^* as the fixed-point of repeatedly applying Hebb's rule. But first, we define a function Hebb for *single-step update*. $\mathsf{Hebb}(\mathcal{N}, S)$ strengthens those edges in a net $\mathcal{N} \in \mathsf{Net}$ whose neurons are active when we feed $\mathcal{N}$ a signal $S \in \mathsf{State}$. (Both Hebb and Hebb^* are illustrated in Figure 3.)

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

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

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 $\mathsf{Hebb}(\mathcal{N}, S)$ is well-defined; the new net is still binary, feed-forward, and fully connected. This also means Hebb does not affect the Reach operator.

Proposition 4. $\mathsf{Reach}_{\mathsf{Hebb}(\mathcal{N},S)}(A,B) = \mathsf{Reach}_{\mathcal{N}}(A,B)$

The following is easy to see from the definition and the fact that $\eta \geq 0$.

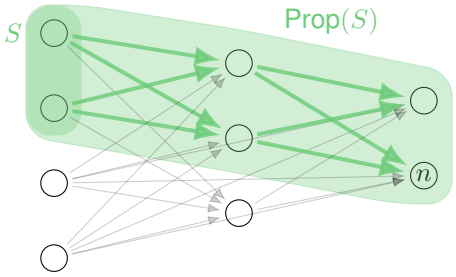

Figure 3: Hebb strengthens those edges whose neurons are active in $\mathsf{Prop}(S)$. The fixed-point operator Hebb^* repeats this update until the edges are "maximally" high.

Proposition 5. 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

With single-step update Hebb in hand, we now turn to iterated (fixed-point) update Hebb^*. Our plan again is to map $\langle\varphi\rangle$ directly to Hebb^*, i.e.,

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

Whereas our base operators $\langle\mathbf{K}\rangle$ and $\langle\mathbf{B}\rangle$ are interpreted as *states* in the underlying neural network, $\langle\varphi\rangle$ changes the net itself. We borrow this move from dynamic epistemic logic (Van Ditmarsch, van Der Hoek, and Kooi 2007).

Why should we study the fixed-point, rather than Hebb itself? Our main reason is that we have a plan of attack for completeness (since the fixed-point is well-behaved). But we also have a motivating intuition about Hebb^*. Imagine $\varphi = \psi_1 \wedge \cdots \wedge \psi_k$ is a dataset of inputs over which we would like to train our net $\mathcal{N}$. Then $\mathsf{Hebb}^*(\mathcal{N}, [\![\varphi]\!])$ is the net at the end of this training process, that has "fully internalized" its training set φ. In other words, $\langle\varphi\rangle\psi$ says what $\mathcal{N}$ has learned at the point where it has fully increased its belief in φ.

For *stable* learning policies (e.g., backpropagation), this fixed-point is the point of convergence. But for unstable Hebbian learning the weights diverge instead of converging. For our purposes, at this point of divergence all updated weights $W(m,n), m, n \in \mathsf{Prop}(S)$ are so high that if m is active in $\mathsf{Hebb}^*(\mathcal{N}, S)$, then n must be as well.

In fact, we can achieve this in a *finite* number of iterations! We define this number iter explicitly. Let $m, n \in \mathsf{Prop}(S)$ be nodes whose weight $W(m,n)$ was updated. In the worst case, the other active predecessor weights $W(m_i, n)$ sum to a very large (high magnitude) negative value. We need to set iter to be high enough that $W(m,n)$ overpowers this sum.

The following definition captures this idea of the lowest possible value any such weighted sum could have.

Definition 10. 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

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

We see that $\mathsf{mnws} \leq$ any particular weighted sum term.

Proposition 6. For all $S \in \mathsf{State}, m, n \in N$, we have

$$\mathsf{mnws} \leq W(m,n) \cdot \chi_S(m)$$

Recall that 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

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

This choice for iter may seem arbitrary, but we will see in a moment why it guarantees that the net diverges. For now, note that iter will always be a positive integer (we can't iterate a negative or continuous number of times).

At last, we are ready to define Hebb^*.

Definition 11. 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 Hebb would affect $\mathsf{Prop}(S)$. We would then need to define Hebb^* to track those changes. Fortunately, we don't have to worry about that. For a single iteration of Hebb we have, for all $S \in \mathsf{State}$,

Proposition 7. $\mathsf{Prop}_{\mathsf{Hebb}(\mathcal{N},S)}(S) = \mathsf{Prop}_{\mathcal{N}}(S)$

As with Hebb, for Hebb^* we have

Proposition 8. $\mathsf{Reach}_{\mathsf{Hebb}^*(\mathcal{N},S)}(A,B) = \mathsf{Reach}_{\mathcal{N}}(A,B)$

Proposition 9. 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 $m \in \mathrm{preds}(n)$, $m, n \in \mathsf{Prop}(A)$, 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 A has a threshold, there is $t \in \mathbb{Q}$ 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 Proposition 6, 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! □

The Reduction for Hebb^*

Our main technical result is that we can "translate away" $\langle\varphi\rangle$-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 8, which says that Hebb^* does not affect Reach. In this section, we prove the following reduction theorem for Hebb^* over Prop.

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

This theorem is at the heart of the reduction axioms that we will use to reduce $\langle\varphi\rangle$ (see Section 4). To prove it, we will first need the following algebraic properties for Hebb^*. Parts (1) and (2) express an upper bound for $\mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$, whereas (3) gives a lower bound. We plan to use (2) and (3) for the reduction; (1) is just used to prove (2).

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

Proof Sketch. We prove each in turn.

1. 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 $n \in \mathsf{Prop}(A) \cap \mathsf{Prop}(B)$, $\mathsf{layer}(m) \leq \mathsf{layer}(n)$.
   - **Case 1.** $\mathsf{layer}(m) < \mathsf{layer}(n)$. Since $\mathcal{N}$ is fully connected, we must have $m \in \mathrm{preds}(n)$. 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 $n \in \mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$.
2. Let $n \in \mathsf{Reach}(\mathsf{Prop}(A), \mathsf{Prop}(B))$. Then there is a path from some $x \in \mathsf{Prop}(B)$ to n running entirely through $\mathsf{Prop}(A)$. The proof goes by induction on the length of this path. In the base step, the path is from n to itself, i.e., $n \in \mathsf{Prop}(A) \cap \mathsf{Prop}(B)$. By (1) we have our goal. In the inductive step, the path is from x to m, with $m \in \mathrm{preds}(n)$. Applying our inductive hypothesis to m gives us $m \in \mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$, which sets up the conditions for Lemma 1. And so $n \in \mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$.
3. Let $n \in \mathsf{Prop}(A) \cap \mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$, and proceed by induction on $\mathsf{layer}(n)$. The base step follows from Inclusion for Prop and Reach. At $\mathsf{layer}(n) \geq 0$, we case on the definition of Prop. If $n \in B$, we just apply Reach Inclusion (since $n \in \mathsf{Prop}(A)$. Otherwise, n is activated by its predecessors $m_i \in \mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$ in $\mathsf{Hebb}^*(\mathcal{N},A)$. As with part (1), we have some $m \in$

$\mathsf{Prop}(A) \cap \mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$ with minimal layer, and we case on $\mathsf{layer}(m) \leq \mathsf{layer}(n)$. The only difference in this case is that if $\mathsf{layer}(m) < \mathsf{layer}(n)$, we only need to apply the definition of Reach. □

We now have everything we need to prove the reduction.

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

Proof. 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 $n \in \mathsf{Prop}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$, and case on the definition of Prop. 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:

- **Case 1.** We have $n \in \mathsf{Prop}(A)$ *and* there is some $m \in \mathrm{preds}(n)$ such that $m \in \mathsf{Prop}(A) \cap \mathsf{Prop}(B \cup \mathsf{Reach}(\mathsf{Prop}(A), \mathsf{Prop}(B)))$. We then apply our inductive hypothesis and part (3) of Lemma 2.
- **Case 2.** Either $n \notin \mathsf{Prop}(A)$ or $\forall m \in \mathrm{preds}(n)$, either $m \notin \mathsf{Prop}(A)$ or m is not active ($m \notin \mathsf{Prop}(A) \cap \mathsf{Prop}(B \cup \mathsf{Reach}(\mathsf{Prop}(A), \mathsf{Prop}(B)))$). In either case, the weights of the two nets are the same and we have our goal.

($\leftarrow$) Let $n \in \mathsf{Prop}(B \cup \mathsf{Reach}(\mathsf{Prop}(A), \mathsf{Prop}(B)))$, and case on the definition of Prop. If $n \in B \cup \mathsf{Reach}(\mathsf{Prop}(A), \mathsf{Prop}(B))$, then we just apply Inclusion or part (2) of Lemma 2, depending on the case. Otherwise, n is activated by its predecessors $m_i \in \mathsf{Prop}(B \cup \mathsf{Reach}(\mathsf{Prop}(A), \mathsf{Prop}(B)))$. In this case, we just apply our inductive hypothesis and Proposition 9. □

Note that if $\mathsf{Prop}(A) \cap \mathsf{Prop}(B) = \emptyset$, then by the Empty property of Reach, we have $\mathsf{Reach}(\mathsf{Prop}(A), \mathsf{Prop}(A)) = \emptyset$ as well. We can substitute this into the statement of Theorem 3 to get the following.

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

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

4 Reduction Axioms and Completeness

The upshot of our reduction theorem is that we can now see how to translate $\langle\varphi\rangle$ sentences into $\langle\varphi\rangle$-free sentences. It will then follow that unstable 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/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 (we draw heavily from both of these sources).

First, we establish reduction axioms for $\langle\varphi\rangle$.

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

$$\begin{array}{lcll}
\langle\varphi\rangle p & \leftrightarrow & p & \text{for propositions } p \\
\langle\varphi\rangle\neg\psi & \leftrightarrow & \neg\langle\varphi\rangle\psi & \\
\langle\varphi\rangle(\psi \wedge \rho) & \leftrightarrow & \langle\varphi\rangle\psi \wedge \langle\varphi\rangle\rho & \\
\langle\varphi\rangle\langle\mathbf{K}\rangle(\psi, \rho) & \leftrightarrow & \langle\mathbf{K}\rangle(\langle\varphi\rangle\psi, \langle\varphi\rangle\rho) & \\
\langle\varphi\rangle\langle\mathbf{B}\rangle\psi & \leftrightarrow & \langle\mathbf{B}\rangle(\langle\varphi\rangle\psi \vee \langle\mathbf{K}\rangle(\langle\mathbf{B}\rangle\varphi, \langle\mathbf{B}\rangle\langle\varphi\rangle\psi)) &
\end{array}$$

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 8, and the $\langle\mathbf{B}\rangle$ case follows immediately from Theorem 3 (the reduction theorem). □

This is an especially nice situation to have. Notice that these axioms compositionally break down the postconditions after $\langle\varphi\rangle$, and "push in" the $\langle\varphi\rangle$ operator in each case. If we read the axioms as rewrite rules from left-to-right, we can see that they each take a $\langle\varphi\rangle$ formula eventually (recursively) to a provably equivalent $\langle\varphi\rangle$-free formula. These rewrite rules are precisely our translation.

Observe that the translation φ^{tr} is provably equivalent to the original formula φ. So a net models φ^{tr} if and only if it models φ in the base language. This means that our nets $\mathcal{N} \in \mathsf{Net}$ already contain all information about what they learn after iterated Hebbian update. Moreover, model building for $\mathcal{L}^*$ follows from model building for 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 $\mathcal{N} \models \Gamma^*$.

Proof Sketch. Let $\Gamma^* \subseteq \mathcal{L}^*$. First, use the rewrite rules obtained from our reduction axioms to translate all the dynamic formulas $\langle\varphi\rangle\psi$, resulting in a $\langle\varphi\rangle$-free set Γ^{tr}. By our assumption, we have a net $\mathcal{N} \models \Gamma^{\mathrm{tr}}$. Since the formulas in Γ^* and Γ^{tr} are provably equivalent, this very same net $\mathcal{N} \models \Gamma^*$. □

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 $\langle\mathbf{B}\rangle$. Then the logic of iterated Hebbian learning $\langle\varphi\rangle$ 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. We give the standard proof. Suppose contrapositively that $\Gamma^* \nvdash \varphi$. Then $\Gamma^* \vdash \neg\varphi$, and so $\Gamma^* \cup \{\neg\varphi\}$ is consistent. By Theorem 5 we have a net $\mathcal{N} \models \Gamma^* \cup \{\neg\varphi\}$. But then $\mathcal{N} \models \Gamma^*$ yet $\mathcal{N} \not\models \varphi$, which is what we wanted to show. □

5 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 $\langle\varphi\rangle$ 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 (possible)

belief $\langle \mathbf{B} \rangle$. So we can think of Hebb^* as a belief revision operator. The final line

$$\langle \varphi \rangle \langle \mathbf{B} \rangle \psi \leftrightarrow \langle \mathbf{B} \rangle (\langle \varphi \rangle \psi \vee \langle \mathbf{K} \rangle (\langle \mathbf{B} \rangle \varphi, \langle \mathbf{B} \rangle \langle \varphi \rangle \psi))$$

reveals the belief revision policy that Hebb^* uses.

Let's unpack this. This axiom says that whatever the agent possibly believed *before* learning φ, *after* learning φ she now also considers this $\langle \mathbf{K} \rangle (\langle \mathbf{B} \rangle \varphi, \langle \mathbf{B} \rangle \langle \varphi \rangle \psi))$ term to be possible. And the $\leftrightarrow$ indicates that the agent learns *only* this term. So what exactly has she learned to believe is possible? This complicated inner term states: If it is possible for the neural agent to believe φ, then it is possible for her to *know* about her (possible) prior belief in ψ.

So iterated Hebbian learning revises an agent's (possible) belief by expanding what she *can possibly believe she can possibly 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 $\langle \varphi \rangle$, our constraints Γ can express guarantees about the net at the tail-end of iterated Hebbian learning. For example, we can build a net that models $([\mathbf{B}]\psi \to \rho) \wedge \langle \varphi \rangle ([\mathbf{B}]\psi \to \rho)$, which says that the net classifies the input ψ as ρ, and fixed-point 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 our work is an early proof of concept, 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 Section 7, we mention future research directions that could make this approach more useful in practice.

6 Related Work

The idea that we can view neural networks as semantics for symbolic reasoning 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 propositions 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). Similarly, (Blutner 2004) proved that activation patterns in Hopfield networks correspond to the logic of "weight-annotated Poole systems." 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) demonstrate how to extend this account with sound fuzzy and probabilistic semantics.

A recent survey (Odense and d'Avila Garcez 2022) shows that this framework encompasses a much wider class of neuro-symbolic methods than we had previously thought. [2] Despite this, our paper appears to be the first to give a complete correspondence for a neural network learning operator.

Our dynamic logic of $\langle \varphi \rangle$ takes inspiration from dynamic epistemic/doxastic logics (DELs) (Van Ditmarsch, van Der Hoek, and Kooi 2007). 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 learning policies expressed in these logics might relate to specific neural implementations of learning such as Hebbian update and backpropagation. The trick of completeness by translation has notably been used to prove completeness for public announcement logic without group knowledge (Baltag, Moss, and Solecki 1998; Plaza 2007), as well as belief revision operators such as lexicographic and elite upgrade (Van Benthem 2007). DELs for belief revision are perhaps the closest logics to ours; we leave open the question of how our logic with $\langle \varphi \rangle$ relates to these logics.

7 Conclusions and Future Directions

In this paper, we showed how a simple neural network learning policy, unstable Hebbian learning, could be modeled as a dynamic logical operator $\langle \varphi \rangle$. We gave reduction axioms that "translate away" $\langle \varphi \rangle$-formulas, providing a complete and human-interpretable description of what a Hebbian learner learns in terms of belief revision. Additionally, given model building for the static language, these reduction axioms in turn allow us to build neural networks with guarantees on what they can learn.

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 need for this is a complete logic for the *transitive reduction* of $\langle \varphi \rangle$. There has been work on completeness for the transitive *closure* $\Box^*$ of an operator $\Box$ (Kozen and Parikh 1981; Moss 2007), but to our knowledge the reverse direction has not been explored. Other exciting future directions include:

- Generalizing to fuzzy and probabilistic interpretations
- Exploring the relationship between Hebbian learning and other belief revision operators
- Generalizing to a broader class of neural networks
- Completeness for *stable* Hebbian learning
- Completeness for supervised learning policies, e.g., gradient descent

[2] Actually, the survey looks at *semantic encodings* that map neural network states to interpretations in a classical model. In contrast, neural network semantics uses $[\![\cdot]\!]$ to map sentences of the logic directly to neural network states. But any semantic encoding determines a neural network interpretation $[\![\cdot]\!]$, and vice-versa.

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.

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.

Kozen, D.; and Parikh, R. 1981. An elementary proof of the completeness of PDL. *Theoretical Computer Science*, 14(1): 113–118.

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.

Moss, L. S. 2007. Finite models constructed from canonical formulas. *Journal of Philosophical Logic*, 36: 605–640.

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.

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