Neural Network Semantics

Caleb Schultz Kisby

Submitted to the faculty of the University Graduate School

in partial fulfillment of the requirements

for the degree

Doctor of Philosophy

in the School of Informatics, Computing, and Engineering,

Indiana University

May 23, 2025

Accepted by the Graduate Faculty, Indiana University, in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Doctoral Committee

Lawrence S. Moss, PhD (Co-advisor)

Saúl A. Blanco, PhD (Co-advisor)

David B. Leake, PhD

Chung-chieh Shan, PhD

May 23, 2025

[Dedication page]

Acknowledgments

[Write the acknowledgments page]

Preface

[Write the preface, which includes the different conferences + workshops this work was presented at.]

Caleb Schultz Kisby

Neural Network Semantics

sdfafsfl;asjdfa;lkjf l;sdaj fl;asjf kl;asjfd ;lasjf ;lkasjf ;klsajf l;ksaj f;lksajf;klasj f;klsaj f;klasj flk;asj fkl;saj fl;ksaj f;lkjs a;flkj sa;lkfj s;klafj ;slakjf ;lasfj

References List:

History of neural network semantics. [44] [7] [38] [39] [17] [40] [34] [35] [28] [?] [29]

Social network semantics. [8]

Dynamic logics for learning. [11] [9] [10]

Dynamic epistemic logic; Belief revision. [69] [66] [71] [12] [13] [56] [36] [70] [68] [14]

General neuro-symbolic AI. [5] [60] [15] [31] [20] [26] [6] [27] [43] [19]

Neural network verification. [3]

AI/neural network Alignment.

Neural networks as automata. [46] [64] [48] [47]

Neural network descriptive complexity. [33] [42] [21]

Neural networks & Category theory.

General conditional logic. [37]

General modal logic. [49] [55]

Classic papers in AI. [32] [63] [30] [54] [61] [45] [59] [72] [65] [62]

General TSC/mathematics. [2] [57]

General cognitive science. [51]

Systems and frameworks. [50] [23] [1]

[Note to self so I don't forget—use new-dpage and new-dpage* commands! (follow them with no-indent!)]

Lawrence S. Moss, PhD (Co-advisor)

Saúl A. Blanco, PhD (Co-advisor)

David B. Leake, PhD

Chung-chieh Shan, PhD

May 23, 2025

Contents

Chapter 1

Introduction

In the last 15 years, modern machine learning systems have achieved unprecedented success in learning from data with little human guidance. Consider, for example, large language models such as Llama and GPT [1; 23; 72], which have taken the world by storm with their ability to learn to converse in English solely from unstructured text data scraped from the Internet. Or, consider AlphaGo [62], which learned to play Go at a human expert level by repeatedly playing against itself. These breakthroughs in machine learning are in large part thanks to the widespread use of neural networks—brain-inspired computational models that are flexible and excel at learning from unstructured data.

But the danger of neural networks is that they come with no guarantees of safety, reliability, or correctness. Neural systems can carry all sorts of misconceptions, make silly logical mistakes, and are quite happy to spread disinformation [25; 41; 52; 65]. Diagnosing, correcting, and understanding the source of these errors is not feasible, so in practice neural networks are treated as 'black-boxes' [cite]. In this sense, neural networks lack transparency and interpretability.

How can we "open the black box" to better understand, reason about, and guide the behavior of neural networks? The field of *neuro-symbolic AI* has emerged to answer this question [5; 15; 60]. Consider symbolic systems—logics—which have been used in theoretical computer science to reason about and guide the behavior of computational systems. Historically, logics were used to model intelligent behavior prior to the rapid growth of neural network systems [cite]. Whereas neural networks are not easily interpreted, logics provide transparent access to their reasoning via a human-interpretable language and compositional semantics. But also unlike neural networks, traditional logics fail to model flexible learning or update. (This is known as the *frame problem* in AI [22; 45; 61]).

The primary goal of neuro-symbolic AI is to combine the complementary strengths of neural networks and logic, retaining the advantages of both. There are now many distinct proposals for

neuro-symbolic systems—to many to count! To give a short but representative list, these systems include Logic Tensor Networks [6], Distributed Alignment Search [27], DeepProbLog [43], Logic Explained Networks [19], neural networks as automata [46; 48; 73], and neural network fibring [26]. These systems and algorithms are disparate, and there is little agreement among them on how to interface neural and symbolic systems. Together these proposals form a scattered picture, and some unifying perspective or theory is needed. In the preface to a recent neuro-symbolic survey book [15], Frank van Harmelen writes:

> What are the possible interactions between knowledge and learning? Can reasoning be used as a symbolic prior for learning … Can symbolic constraints be enforced on data-driven systems to make them safer? Or less biased? Or can, vice versa, learning be used to yield symbolic knowledge? … And how to avoid the inevitable bias seeping from the data into the resulting knowledge base? … **neuro-symbolic systems currently lack a theory that even begins to ask these questions, let alone answer them.**

In this dissertation, I will explore a class of neuro-symbolic systems that tightly couple logic with neural network dynamics. This may seem like adding yet another neuro-symbolic proposal to the pile. But in fact, the point of this work is to develop and promote a broader perspective that deepens our understanding of well-known neuro-symbolic systems and informs the development of neuro-symbolic interfaces in the future.

Here is the perspective I'm promoting: **Neural networks can be thought of as models for formal logic, i.e., logics can be interpreted directly on neural networks.** Speaking more technically, logical operators such as conditionals $\Rightarrow$, modalities $\Box$, dynamic updates $[P]$, and quantifiers $\forall x$ can be interpreted by semantics that refer to, e.g., the input-output behavior, signal propagation, or update of weights in a neural network. I will soon show that many neuro-symbolic systems can be considered from this point of view; I will call any specification of formal semantics that falls under this umbrella "*neural network semantics*."

In previous work, the input-output behavior of (binary) neural networks was shown to be an

appropriate semantics for defeasible conditionals $\varphi \Rightarrow \psi$ (read "typically, if φ then ψ") [7; 17; 20; 38; 39]. In this dissertation, I develop this same idea in more expressive logical contexts, including modal logic [16; 67], Dynamic Epistemic Logic (DEL) [68; 70; 71], and first-order logic (FOL) [24]. I will focus on three, highly nonstandard, logics that are interpreted on binary neural networks:

1. A defeasible modal logic whose central modality $\mathbf{C}\varphi$ is interpreted via the closure (or forward propagation) of the signal φ in a neural network. **(Chapters 3 and 4)**
2. A DEL whose update operator $[P]$ is interpreted as Hebbian update (on input P) [32], a simple neural network learning policy that changes weights according to the rule "neurons that fire together wire together." I will consider a logic of single-step Hebbian update, as well as a logic of iterated Hebbian update. **(Chapter 5)**
3. A defeasible FOL whose central quantifier $\forall x\varphi$ is interpreted via the closure (or forward propagation) of the signal φ in a specially-designed neural network that uses variable assignments in place of neurons. **(Chapter 6)**

As a logician, I will take each of these semantics seriously. I will develop them by mathematically proving formal logical results, including soundness, completeness, expressivity, and frame correspondence theorems. These may seem like results that merely satisfy a logician's curiosity. But my point is that, in the context of neural network semantics, these theorems enable us to directly answer fundamental questions about neural network inference, learning, and neuro-symbolic interfaces. Here is an explicit list of questions this perspective helps us answer, along with the chapter in which I answer them:

- What algebraic properties hold for neural network inference? Can we design an expressive logic that can be used to formally verify these properties? **(Chapter 3)**
- Can we build a neural network that obeys a given set of constraints on its inference (stated as formulas in an expressive logic)? **(Chapter 4)**
- How powerful are neural network models, when compared to other classes of models? What kinds of models can neural nets simulate? And vice-versa, what kind of models can

be simulated by neural nets? **(Chapter 4)**

- What algebraic properties hold for neural network learning policies? Can we design a *dynamic* logic that can be used to formally verify properties of neural network learning? **(Chapter 5)**
- In general, can we build a neural network that obeys a set of constraints on its inference *before and after learning takes place*? **Note:** This is one of the keys to the AI Alignment problem. **(Chapter 5)**
- Can we design a neuro-symbolic system that supports reasoning in full first-order logic? What kinds of neural network properties guarantee the soundness of FOL axioms? **(Chapter 6)**

In summary, I will defend the following thesis statement:

Thesis Statement. Neural networks can be treated as a class of models in formal logic, where logics are interpreted directly on the neural net ("*neural network semantics*"). By developing this idea in richer logical contexts, we are able to answer fundamental questions about neural network inference, learning, and neuro-symbolic interfaces in general.

Chapter 2

Background: Defeasible Reasoning in Artificial Intelligence

The connection between neural networks and formal logic begins with defeasible reasoning (aka nonmonotonic reasoning, or reasoning by default). In standard treatments of logic, the facts you infer are non-revocable, i.e., they cannot be withdrawn in light of new information. But we live in a world of change, partial information, and exceptions—in order to effectively reason, an agent must jump to conclusions about what is "normally" or "plausibly" the case, and be ready to withdraw these inferences. For these reasons, defeasible reasoning is a central to a theoretical understanding of artificially intelligent agents.

Here's a classic example: If you know Tweety is a bird, you should conclude (assuming we're in a "normal" situation) that Tweety flies. But if you then discover that Tweety is a penguin, you must retract that conclusion. The standard material implication fails to model this: If Tweety→penguin, penguin→bird, and bird→flies we must conclude that Tweety flies.

In this chapter, I will give a tour of many different ways to model defeasible reasoning in formal logic. I will focus on the "preferential" or "plausibility" approach to defeasible reasoning, which branches from the classic papers [37] and [cite Shoham 1988]. First, I will present the standard plausibility semantics for conditional logics (where $\varphi \Rightarrow \psi$ expresses "typically, φ are ψ"). Then I will discuss many different ways to transfer these semantics to more expressive modal logics. I will present the logic of $\mathbf{C}\varphi$ ("the current state is the best one where φ holds"), which forms the backbone of my work connecting neural networks and logic. Finally, I will introduce neural networks, and discuss how they may be seen as models of defeasible reasoning as well. This will set us up for the central plot of my thesis: Developing a neural network semantics for the logic of $\mathbf{C}\varphi$.

1 Defeasible Reasoning in Conditional Logic

I will now present the standard way to model nonmonotonic inference in conditional logic, in the KLM tradition [37]. The language is stratified—sentences are conditionals $\varphi \Rightarrow \psi$, where $\varphi, \psi \in \mathcal{L}_{\text{prop}}$ are propositional formulas connected by $\neg, \wedge, \rightarrow$ in the usual way. Sentences $\varphi \Rightarrow \psi$ cannot be nested within each other, nor within propositional formulas. This odd feature is due to the original conception in [37] that $\varphi \Rightarrow \psi$ specify inference rules, but are not themselves propositions. The intended meaning of $\varphi \Rightarrow \psi$ is "typically (normally), φ are ψ", e.g., bird⇒flies reads "typically, birds fly."

Kraus, Lehmann, and Magidor use the following models to interpret these conditional sentences. I will be moving on pretty quickly to modal logic syntax and semantics, so I won't dwell on these models too long. Let W be an underlying set of worlds (propositional valuations) for $\mathcal{L}_{\text{prop}}$ (not necessarily the set of all worlds for $\mathcal{L}_{\text{prop}}$).

Definition 1.1. A cumulative-ordered model is $\mathcal{M} = \langle \mathcal{S}, l, \prec \rangle$, where

- $\mathcal{S}$ is a nonempty set of states
- $l : \mathcal{S} \rightarrow \mathcal{P}(W) - \{\emptyset\}$ (a *labelling* of states)
- $\prec : \mathcal{S} \times \mathcal{S}$ (the *plausibility order*, or *preference relation*)

The plausibility order $\prec$ is required to be a strict order relation (irreflexive and transitive) satisfying the Smoothness Condition (which I will state shortly). $S_1 \prec S_2$ intuitively means that the agent considers the state $S_1 \in \mathcal{S}$ to be more plausible, or more normal, than $S_2 \in \mathcal{S}$. In order to reason about the most plausible (normal) states, we can look at the $\prec$-minimal states. Formally, each cumulative-ordered model determines a function $\mathsf{best}_{\prec} : \mathcal{S} \rightarrow \mathcal{S}$

$$\mathsf{best}_{\prec}(S) = \{w \in l(S) \mid \text{For all } u \in l(S), \neg u \prec w\}$$

For propositional formulas $\varphi \in \mathcal{L}_{\text{prop}}$, $[\![\varphi]\!] = \{S \in \mathcal{S} \mid w \models \varphi \text{ for all } w \in l(S)\}$, i.e., the set of states where φ is true everywhere. I said before that $\prec$ must satisfy the Smoothness Condition [37]—this condition says that for any propositional formula $\varphi \in \mathcal{L}_{\text{prop}}$, $[\![\varphi]\!]$ has no infinitely descending $\prec$-chains, i.e., every non-empty $[\![\varphi]\!]$ has at least one minimal element.

Postulate 1.2. For all cumulative-ordered models $\mathcal{M}$, states $S \in \mathcal{S}$, and all $\varphi \in \mathcal{L}_{\text{prop}}$, if $S \in [\![\varphi]\!]$ then either $S \in \mathsf{best}_{\prec}([\![\varphi]\!])$, or there is some $S' \prec S$ better than S that *is* the best, i.e. $S' \in \mathsf{best}_{\prec}([\![\varphi]\!])$.

I can now give the KLM intepretation of conditional sentences:

$$\mathcal{M} \models \varphi \Rightarrow \psi \text{ iff } \mathsf{best}_{\prec}(\llbracket \varphi \rrbracket) \subseteq \llbracket \psi \rrbracket$$

That is, in the most plausible (normal) states where φ holds, ψ holds, which was our intended reading. There is a lot more to say about conditional logics like these (expressivity, proof systems, soundness and completeness, their rich history), but I must move on. I will conclude with an example demonstrating that these semantics do in fact resolve our earlier issue with Tweety the penguin.

Example 1.3. [Give an example of these semantics successfully modeling defeasible reasoning. Maybe the Tweety example?]

2 Defeasible Reasoning in Modal Logic

The inability to nest conditionals $\varphi \Rightarrow \psi$ makes conditional logics somewhat flat and inexpressive. Additionally, $\varphi \Rightarrow \psi$ only allows us to refer to the plausibility of the premise φ, and not the antecedent ψ. For example, the following sentences are not expressible in the conditional language above:

- If birds typically fly, then Tweety does.
- The car normally drives, but the check engine light is always on.
- This was not your typical criminal.
- If this isn't normal, I don't know what is.

We can overcome this by transferring the main ideas of the semantics to a more expressive language—in particular, to modal logics. [is there anything more I need to say here to motivate the reader?]

2.1 A Brief Crash Course in Modal Logic

Let's briefly introduce the basics of modal logic. [cite a standard modal logic text or two!] A modal logic extends propositional logic with "modal formulas" $\Box\varphi$ and $\Diamond\varphi$ ($\Box\varphi$ is read "it is

necessary that φ"; $\diamond\varphi$ is read "it is possible that φ." Standard (normal) modal logics are interpreted using a relational (Kripke) model, which is just an ordinary graph equipped with a valuation of propositions.

Definition 2.1. A relational model is $\mathcal{M} = \langle W, R, V \rangle$, where W is a set of nodes (*worlds*, aka *states*), $R: W \times W$ an edge relation (the *accessibility relation*), and $V:$ propositions$\to\mathcal{P}(W)$ (the *valuation function*).

Definition 2.2. Let **Rel** be the class of all relational models, and let $\mathbf{Rel}_{S4}$ be the class of all whose accessibility relation R is reflexive and transitive.

Unlike conditional logic, in modal logics we evaluate a formula *locally*. That is, instead of φ being true or false, we consider the set of worlds where φ is true. We write $\mathcal{M}, w \Vdash \varphi$ to indicate that φ holds at world w. The semantics of propositions and boolean connectives $\neg, \wedge$ are what you might expect. $\diamond\varphi$ is defined as $\neg\Box\neg\varphi$. The key case is for $\Box\varphi$:

$$\mathcal{M}, w \Vdash \Box\varphi \text{ iff for all } u \text{ such that } wRu, \text{ we have } \mathcal{M}, u \Vdash \varphi$$

That is, $\Box\varphi$ holds if φ holds everywhere accessible from the current state (φ is *necessarily* true).

The accessibility relation can have many different interpretations depending on what phenomenon we are trying to model. For example, if R indicates which states are *possibly known* (i.e., *epistemically accessible*), then $\Box\varphi$ takes on the reading "φ is known (by some agent)," written boxφ. Similarly, $\Box$ can be cast as belief **B**, obligation **O**, provability **P**, etc. There may be one, or many, modal operators in a modal logic. We may also index modalities $\Box_i$, indicating a modal attitude for each agent i (in a multi-agent setting), or for different relations R_i within the same agent.

2.2 Defeasible Modal Logics

Unfortunately, this usual treatment of modal logics cannot model defeasible reasoning. [cite Chellas or something] This is because all normal modal logics satisfy the axiom

$$\Box(\varphi \to \psi) \to \Box\varphi \to \Box\psi$$

For concreteness, let's read $\Box$ as belief, and suppose $\Box$(bird$\rightarrow$flies), i.e., for all things we could possibly believe we see from the current state, if we see a bird then it flies. Now say we believe we see a bird ($\Box$bird). Then the axiom says we *necessarily* believe it flies ($\Box$flies), leaving no room for revoking our initial conclusion.

There is a wide variety of different ways to resolve this, to rework the idea of defeasibility into modal logic. This is not the right time or place for a thorough literature review, but I will tour a representative sample to give you a sense of what can be done. Certain approaches use the cumulative-ordered models defined for conditional logic above; others use relational models, but interpret the relation R to be a plausibility ordering; others still use both. For my purposes, I will define plausibility models as Krikpe models, but with an irreflexive plausibility relation $\prec$ (there is no distinction between epistemic accessibility and plausibility). [cite plausibility models, the word is used by Baltag & Smets]

Definition 2.3. A plausibility model is $\mathcal{M} = \langle W, \prec, V \rangle$, where

- W is a set of *worlds* or *states*
- $R: W \times W$ (the *epistemic accessibility relation*)
- $\prec: W \times W$ (the *plausibility order*)
- V: [todo] (the *propositional valuation*)

[(Basically a cumulative-ordered model along with an epistemic frame)] I require R to be reflexive and transitive [Connected? Symmetric?]. As with cumulative-ordered models, I require $\prec$ to be irreflexive, transitive, antisymmetric. In cases where we want to refer to the reflexive extension of $\prec$, I write $u \preceq v$ to mean $u \prec v$ or $u = v$. As before, each plausibility model determines a $\mathsf{best}_{\prec}$ function, whose definition now simplifies to

$$\mathsf{best}_{\prec}(S) = \{w \in S \mid \text{For all } u \in S, \neg u \prec w\}$$

Similarly, we require the $\mathsf{best}_{\prec}$ operator to satisfy a similar Smoothness condition. For *any* formula φ (not just propositional formulas), say we have defined some semantics $\Vdash$, and let $[\![\varphi]\!] = \{w \in W \mid \mathcal{M}, w \Vdash \varphi\}$. Smoothness says:

Postulate 2.4. For all plausibility models $\mathcal{M}$, $w \in W$, and formulas φ, if $w \in [\![\varphi]\!]$ then either $w \in \mathsf{best}_{\preceq}([\![\varphi]\!])$, or there is some $v \prec w$ better than w that *is* the best, i.e. $v \in \mathsf{best}_{\preceq}([\![\varphi]\!])$.

[Should I move this postulate down past the semantics? It would be nice if I defined the language first, so I could say $\varphi \in \mathcal{L}_{\mathrm{C}}$.]

Definition 2.5. Let **Plaus** be the class of all such plausibility models.

Here are some of the ways we can transfer defeasibility into a modal logic setting:

Boutilier's Modal Treatment.

Baltag & Smets' Safe Belief.

- There are many ways to rework the idea of typicality in modal logic: conditional belief $B^{\varphi}\psi$, regular belief $B\varphi$, typicality $\cdot, \boldsymbol{T}$, "defeasible modalities"

3 Dynamic Epistemic Logic and Belief Revision

[This is really one of the best upshots of modeling something using modal logic! If formulas are nestable, then we can nest things inside and within update operators.] [Introduce Dynamic Epistemic Logic, and various operators $\mathsf{Cond}, \mathsf{Lex}, \mathsf{Consr}$ for belief revision.]

[Include some parts of this in this section]

Other Dynamic Logics for Learning. My approach to modeling learning in neural networks takes inspiration from Dynamic Epistemic Logic (DEL) [68; 71]. Perhaps the closest logics to mine are the logics for plausibility upgrade, in particular conditionalization (Cond), lexicographic (Lex), and conservative (Consr) upgrade [66; 70]. In the Expressivity portion of my dissertation, I will explore whether Hebbian update $\mathsf{Hebb}^{\star}$ can be simulated by plausibility upgrade, and vice-versa whether Cond, Lex, Consr, and vice-versa whether these plausibility upgrades can be simula.

In my thesis, I mainly focus on the effects of single-step updates. But recent literature on learning in DEL goes beyond this by considering iterated update and convergence to the truth ("learning in the limit") [9; 10; 11; 14]. The key questions here are: How can we compare the

learning power of different iterated update policies? How can we axiomatize important properties of learning? These questions are answered in terms of updates on more classical graphs and plausibility structures. Although in this thesis I don't consider iterated update, I do lay down the groundwork to import neural network learning into this setting.

4 Defeasible Reasoning in Neural Networks

Introduce neural networks, for a broad audience (including logicians that know nothing about them). Explain how inference works in a neural network, and how neural networks can be thought of as performing defeasible reasoning. (Hannes explains it pretty well, maybe I should borrow his example.)

Explain learning in a neural network at a high level, both unsupervised (Hebbian learning is representative of unsupervised learning algorithms, mention the relationship with Principal Components Analysis) and supervised (backpropagation is representative here, and the efficient computation of backpropagation is what has made neural networks so successful).

Inference in a neural network is "like" a conditional inference—but this analogy goes further. Many authors have already studied a formal correspondence between the input-output behavior of a neural network and defeasible conditionals. [cite d'Avila Garces, Hannes, Giordano, really all the people here who have made the observation before me. This is a good time to break down the history of how it happened.] [Talk about both "soundness" and "completeness" here]

In the rest of this chapter, I will extend this analogy by giving a neural network interpretation for the more general logic of **C**. My main point in considering a modal language is the same as before: It buys us expressive power over conditionals, and in particular sets us up to express *neural network update* using Dynamic Epistemic Logic. Towards the end of this work, I will extend these neural network semantics for **C** with a dynamic operator for a simple Hebbian update policy over neural networks.

Chapter 3

Neural Network Semantics

1 Introduction

[I wrote the following short paragraph for my thesis proposal. But now I have more space to say more, slowly, about where neural network semantics comes from, what the underlying idea is. It would be nice to introduce it using an example of a neural network in practice—justify why binary, interpreted neural networks are the "right" thing to look at!]

I will now give an overview of the particular neural network semantics [rephrase]I've developed during my PhD. First, I will discuss the simplifying assumptions that make it possible to use neural networks as models, and introduce the *closure* (or forward propagation) of a signal in the net. This closure operator allows us to express the inference, or input-output behavior, of the net. I will give a modal logic whose key operator is given by this closure operator. I will then turn to dynamic update in neural networks and introduce iterated Hebbian learning, one of the simplest learning policies over nets. Finally, I will give a dynamic logic whose formulas express the behavior of a neural network before and after Hebbian update.

2 Neural Network Models

A model of neural network semantics is an artificial neural network (ANN), along with a valuation function which interprets propositions as sets of neurons. I will make a few more simplifying assumptions soon, but this is the basic idea.

Definition 2.1. A neural network model is $\mathcal{N} = \langle N, \mathsf{bias}, E, W, A, \eta, V \rangle$, where

- N is a finite nonempty set (the set of *neurons*)
- $\mathsf{bias} \in N$ is a fixed node (the *bias* node)
- Each $E \subseteq N \times N$ (the *edge relation*)
- $W : E \to \mathbb{Q}$ (the *edge weights*)
- $A : \mathbb{Q} \to \mathbb{Q}$ (the *activation function*)
- $\eta \in \mathbb{Q}$, $\eta \geq 0$ (the *learning rate*)
- $V : \mathcal{L}_{\text{prop}} \to \mathcal{P}(N)$ (the *valuation function*)

In general, a *state* is just a possible activation pattern or configuration of the net. In practice, a neural network's nodes take on fuzzy activation values in $[0, 1]$. But we would like to associate each state with a binary set of neurons—either a neuron is active (1) or it is not (0). To do this, I assume that the activation function A is a (nonzero) binary step function ($A : \mathbb{Q} \to \{0, 1\}$). [Definition: $\mathcal{N}$ has a threshold, $\exists t \in \mathbb{Q}$ with $A(t) = 1$; $\mathcal{N}$ is nondecreasing. These things amount to A being a binary step function.] It turns out this binary constraint is also a common theoretical assumption in work that analyzes neural networks as automata [46; 48; 73]. In their terminology, we say our nets are *saturated*.

- [Todo, put somewhere in this section, optional property]
- **$\mathcal{N}$ 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.
- 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 [DIAGRAM]. (This intuition comes from work on highway networks [63].)

This assumption is crucial for our results about iterated Hebbian learning, and we expect that letting it go will not be easy.

Since I'm only considering binary neural networks, either a neuron is active (1) or it is not (0). A *state* (activation pattern) of the net is just a set of neurons that happen to be active. Additionally, I assume there is a special bias node that is active in every state. This bias node provides provides persistent input to the neural network; its purpose is to avoid the edge case where the input S_0 is empty, and so the net doesn't have any "fuel" to activate any new nodes. (Since bias is always active, we do not need any edges going into it. So I assume bias has no predecessors.) Putting all this together, the states of the net are defined as follows.

$$\mathsf{State}_{\mathcal{N}} = \{S \mid S \subseteq N \text{ and } \mathsf{bias} \in S\}$$

Usually $\mathcal{N}$ is understood from context, and I'll just write State without the subscript.

Example 2.2. [Give an example of a neural network, with weights and bias, and valuation function, and give examples of states (including the bias node!) Does the valuation function necessarily have to be a state? i.e. does it necessarily have to include bias when mapping propositions? I believe it shouldn't, and this example could illustrate that (e.g. we map p to $\{a, b, c\}$, but despite the mapping note that this set is not a *state* of the net, since bias is not in it. We can extend it to a state by adding bias.)]

[**TODO:** Add a note here where I discuss the difference between neural network models and standard graph models for logic. The key thing here is that neural network models are *connectionist* / represent their inferences in a distributed way. But I have to make that somewhat precise.]

2.1 Clos: The Network Closure Operator

We can describe the input-output behavior of neural networks in terms of their state. Say we are given an input state A consisting of input-layer nodes, and a possible classification state B consisting of output-layer nodes. Active neurons in A subsequently activate new neurons, which

activate yet more neurons, until eventually the state of the net stabilizes. If this final state includes the output B, we say "the net *classifies* A as B".

Consider for example the neural network in [EXAMPLE where we classify Tweety as flying. Make the connection clearer—for binary nets, this is exactly what (binary) classification means!]

The state at the fixed point of this process is called the *closure* or *forward propagation* of the signal A through the net, $\mathsf{Clos}(A)$. This closure operator is central to the semantics, since it captures the underlying dynamics involved in neural network inference. Formally, each choice of E, W, A specifies a transition function from state $S \in \mathsf{State}$ to the next state. Given an initial state S_0, this transition function $F_{S_0} \colon \mathsf{State}_{\mathcal{N}} \to \mathsf{State}_{\mathcal{N}}$ is given by

$$F_{S_0}(S) = S_0 \cup \left\{ n \mid A\left(\sum_{m \in \mathrm{preds}(n)} W(m,n) \cdot \chi_S(m) \right) = 1 \right\}$$

where $\chi_S(m) = 1$ iff $m \in S$ is the indicator function. In other words, $F_{S_0}(S)$ consists of the initial state S_0, along with all those nodes that are activated by their predecessors in S. Notice that $F_{S_0}(S)$ is extensive: all nodes in the initial state will stay activated in successive states.

These neural network models have one final constraint: This transition function F_{S_0} eventually gives a unique *fixed point* (or stable state) under the input S_0, i.e. a unique state S such that $F_{S_0}(S) = S$. This guarantees that the closure $\mathsf{Clos}(S)$ exists.

Postulate 2.3. I assume that for all states S_0, F_{S_0} applied repeatedly to S_0, i.e.

$$S_0, F_{S_0}(S_0), F_{S_0}(F_{S_0}(S_0)), \ldots, F^k_{S_0}(S_0), \ldots$$

eventually reaches a finite fixed point, and moreover this state is the *only* fixed point under S_0. Formally, this means that for all $S_0 \in \mathsf{State}_{\mathcal{N}}$ there is some $k \in \mathbb{N}$ such that:

1. $F_{S_0}(F^k_{S_0}(S_0)) = F^k_{S_0}(S_0)$. That is, the activation pattern under F_{S_0} will eventually stabilize.
2. $F^k_{S_0}(S_0)$ is the only state $S \in \mathsf{State}_{\mathcal{N}}$ such that $F_{S_0}(S) = S$. In other words, the final state is unique for each initial state S_0.

Let the closure $\mathsf{Clos} \colon \mathsf{State}_{\mathcal{N}} \to \mathsf{State}_{\mathcal{N}}$ be the function that produces that least fixed point: $\mathsf{Clos}(S) = F^k_{S_0}(S_0)$ for that $k \in \mathbb{N}$ above. Finally, let **Net** be the class of all binary neural network models that

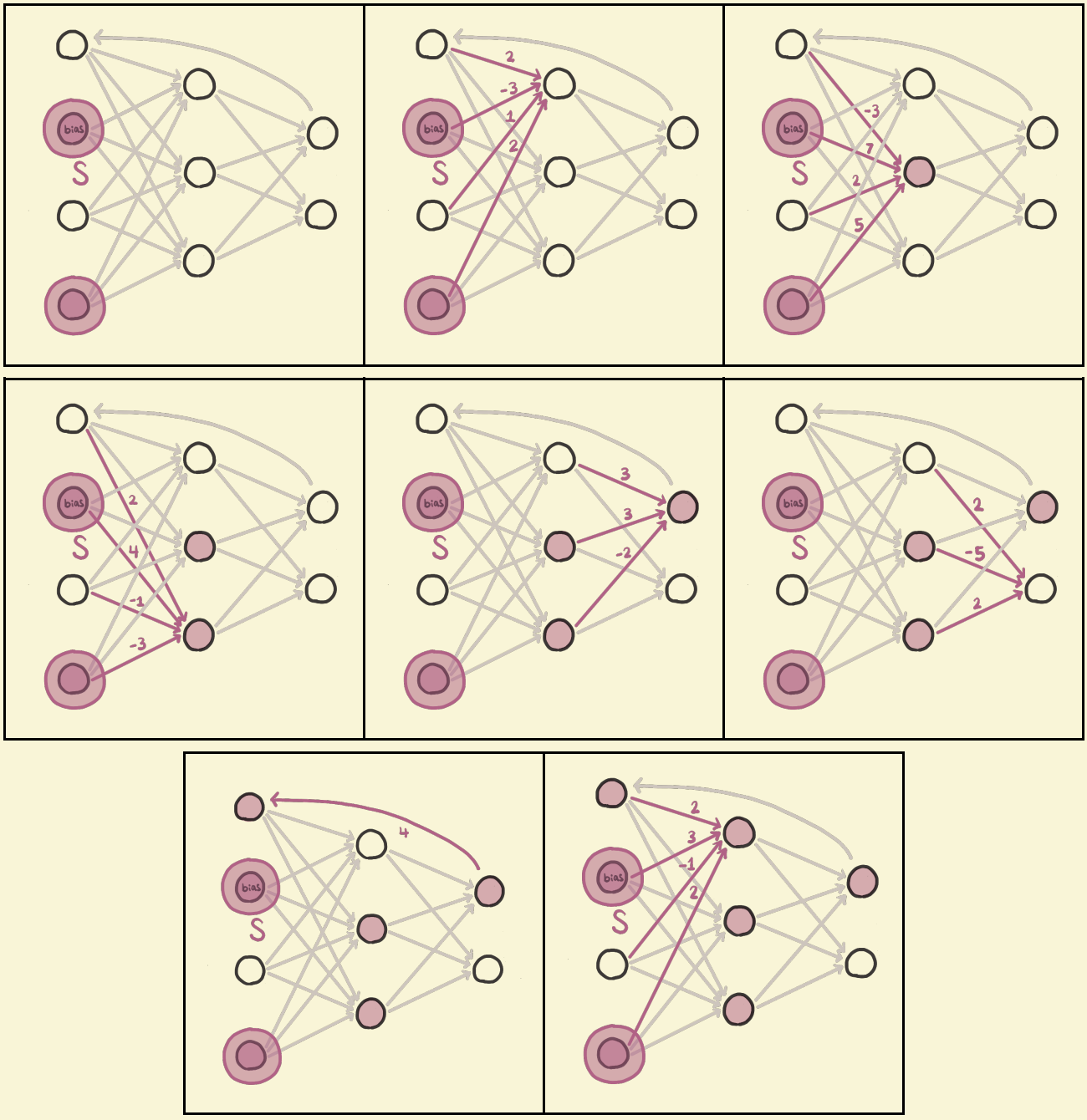

Figure 2.1. [caption!]

satisfy this postulate.

Example 2.4. At this point, let's walk through an example run of the closure of a set $\mathsf{Clos}(S)$. Consider the neural network shown in Figure 2.1, and take the activation function to be $A(x) = 1$ iff $x \geqslant 1$. The first cell shows the neural network at initial state S, where only the bias node and

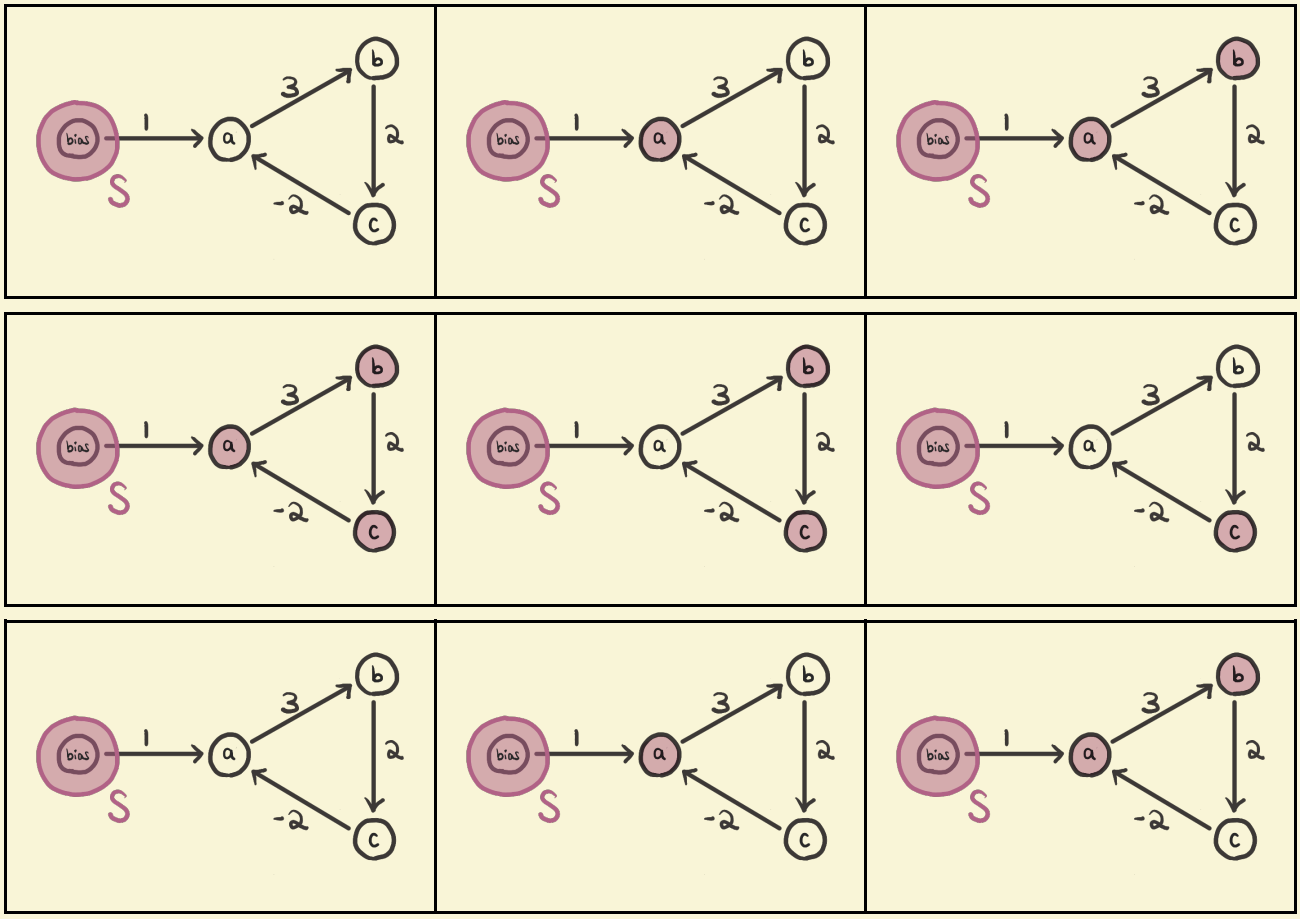

Figure 2.2. [Separate into (a) left, and (b) right. TODO—caption!]

one other node are active. For each neuron n, we activate that neuron if the weighted sum of its predecessors is ≥ 1. (Note that currently inactive neurons do not contribute to this sum.)

Now repeat this process for multiple rounds k, until no new neurons activate in the current round. The first two rows of Figure 2.1 display the $k = 1$ round, and the final row shows an abbreviated run of the $k = 2$ round. The final cell gives exactly those neurons activated in the closure $\mathsf{Clos}(S)$. Take a moment to check that this is in fact a stable state, i.e., the remaining nodes in the final cell do not subsequently activate. (Hint: You can look at the previous cells for the weights of those nodes.)

Note. Not every neural network has a unique fixed point $\mathsf{Clos}(S)$ for every $S \in \mathsf{State}_{\mathcal{N}}$. Consider the recurrent neural network $\mathcal{N}$ and the state S shown in Figure 2.2. Take the activation function to be $A(x) = 1$ iff $x \geqslant 0$. The active bias neuron in S subsequently activates a, then b, then c. But once c

is active, a is no longer active (the weighted sum of its predecessors is now $-2+1=-1$). And so the net continues to oscillate between these states, never stabilizing.

Characterizing those nets that have a unique fixed point for every state (i.e., characterizing **Net**) is a big open problem. Clearly, feed-forward networks do, since without recurrent edges the subsequent activations will eventually terminate. But the difference between Clos being well-defined and not is often subtle. In the net $\mathcal{N}$ from before, if we change the weight $W(\mathsf{bias}, a)$ to 3 instead of 1, the resulting net will no longer oscillate!

Open Question 1. Is there an algebraic or topological characterization of the class of recurrent nets whose closure $\mathsf{Clos}(S)$ reaches a unique fixed point (i.e., does not diverge or oscillate)?

2.2 Reach and Reach$^{\downarrow}$: The Graph Reachability Operators

[As I mentioned before] [TODO I no longer mention this! I need a different segue], a key feature of Clos is that it is not monotonic. Let's consider two closure operators over neural networks that *are* monotonic—graph reachability Reach, and "reverse" graph reachability $\mathsf{Reach}^{\downarrow}$. $\mathsf{Reach}(S)$ just returns the set of all neurons graph-reachable from S, i.e. $\mathsf{Reach}\colon \mathsf{State}_{\mathcal{N}} \to \mathsf{State}_{\mathcal{N}}$ is given by $n \in \mathsf{Reach}(S)$ iff there exists $m \in S$ with an E-path from $\boldsymbol{m}$ **to** $\boldsymbol{n}$. Similarly, $\mathsf{Reach}^{\downarrow}(S)$ returns the set of all neurons that graph-reach some node in S, i.e. $\mathsf{Reach}^{\downarrow}\colon \mathsf{State}_{\mathcal{N}} \to \mathsf{State}_{\mathcal{N}}$ is given by $n \in \mathsf{Reach}^{\downarrow}(S)$ iff there exists $m \in S$ with an E-path from $\boldsymbol{n}$ **to** $\boldsymbol{m}$.

Reach and $\mathsf{Reach}^{\downarrow}$ are not very interesting operators in their own right, but I consider them in this discussion for two reasons. First, graph reachability is necessary for reasoning about Hebbian learning (see Chapter ?). Second, in Chapter ? I would like to compare the expressive power of neural networks against many classes of models, including relational (graph) models.

powerp

3 Properties of Clos, Reach, and Reach$^{\downarrow}$

The following theorem, due to [38], says that we can neatly characterize the algebraic structure of Clos a closure operator. Note that Leitgeb proves this for *inhibition nets*, i.e. weightless neural

networks with both excitatory and inhibitory connections. But inhibition nets and our nets $\mathcal{N} \in \mathbf{Net}$ are equivalent with respect to their propagation structure—I prove this result again for **Net** as a kind of "sanity check" that my definitions are correct.

Proposition 3.1. (Leitgeb, [38; 39]) For all $S, S_1, \ldots, S_k \in \mathsf{State}_{\mathcal{N}}$,

Inclusion. $S \subseteq \mathsf{Clos}(S)$

Idempotence. $\mathsf{Clos}(\mathsf{Clos}(S)) = \mathsf{Clos}(S)$

Cumulative. If $S_1 \subseteq S_2 \subseteq \mathsf{Clos}(S_1)$, then $\mathsf{Clos}(S_1) = \mathsf{Clos}(S_2)$

Proof. I'll prove each in turn:

Inclusion. By definition, $\mathsf{Clos}(S) = F_S^k(S)$ for some $k \in \mathbb{N}$, where F_S^k is the transition function from Definition [which?]. By induction on this k (the number of iterations needed to construct the closure):

Base Step. $k = 0$, and so $\mathsf{Clos}(S) = S$. So if $n \in S$, then $n \in \mathsf{Clos}(S)$.

Inductive Step. Let $k \geq 0$, and suppose $n \in S$. We have $\mathsf{Clos}(S) = F_S^k(S) = F_S(F_S^{k-1}(S))$. Expanding this term out, we have

$$F_S(F_S^{k-1}(S)) = S \cup \left\{ n \mid A\left(\sum_{m \in \mathrm{preds}(n)} W(m,n) \cdot \chi_{F_S^{k-1}(S)}(m) \right) = 1 \right\}$$

Since $n \in S$, n is in the left-hand side of this union. And so $n \in \mathsf{Clos}(S)$.

Idempotence. I will prove a stronger claim: For all $k \in \mathbb{N}$,

$$F_S^k(\mathsf{Clos}(S)) = \mathsf{Clos}(S)$$

In other words, applying the transition function F_S any number of times to $\mathsf{Clos}(S)$ has no effect. Since $\mathsf{Clos}(\mathsf{Clos}(S)) = F_S^k(\mathsf{Clos}(S))$ for some $k \in \mathbb{N}$, the Idempotence property follows from this. Let's proceed by induction on k.

Base Step. $k = 0$, and so the goal simplifies to $\mathsf{Clos}(S) = \mathsf{Clos}(S)$, which is true.

Inductive Step. Let $k \geq 0$. We have $F_S^k(\mathsf{Clos}(S)) = F_S(F_S^{k-1}(\mathsf{Clos}(S)))$. Our inductive hypothesis here is that, for $k-1$, $F_S^{k-1}(\mathsf{Clos}(S)) = \mathsf{Clos}(S)$. So we can subsitute that to get $F_S^k(\mathsf{Clos}(S)) = F_S(\mathsf{Clos}(S)) = \mathsf{Clos}(S)$, which is what we wanted to show.

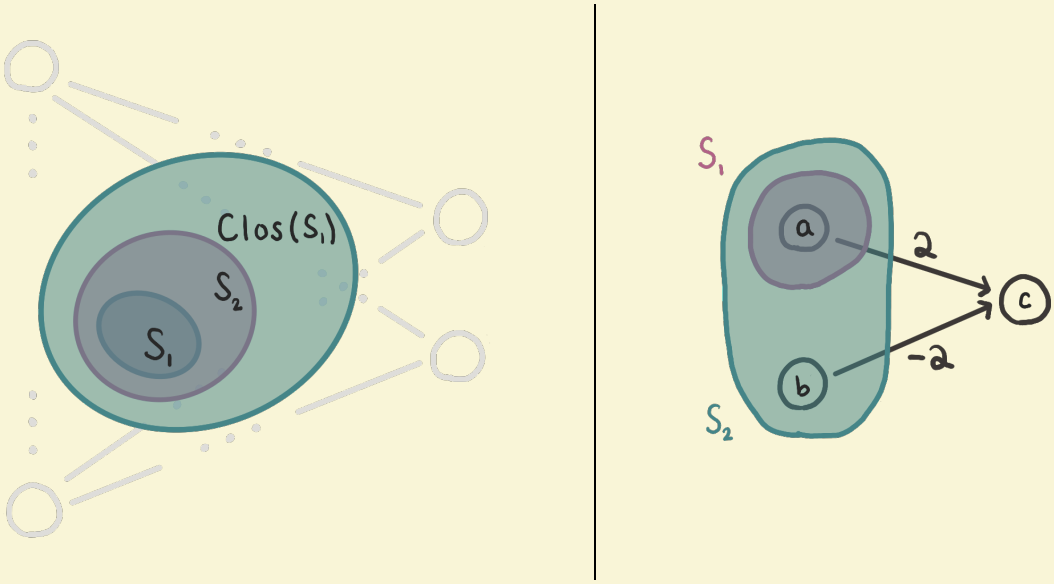

Figure 3.1. [Separate into (a) left, and (b) right. TODO—caption!]

Cumulative. Suppose $S_1 \subseteq S_2 \subseteq \mathsf{Clos}(S_1)$, as illustrated in Figure 3.1 (a). First, since $\mathsf{Clos}(S_2)$ is a fixed point under S_2, we have $F_{S_2}(\mathsf{Clos}(S_2)) = \mathsf{Clos}(S_2)$. But I will show that $\mathsf{Clos}(S_1)$ is *also* a fixed point under S_2, i.e. $F_{S_2}(\mathsf{Clos}(S_1)) = \mathsf{Clos}(S_1)$ (notice the difference in the subscripts). Since we assumed that there is a *unique* fixed point under S_2, it will follow that these two states must be the same. In other words, $\mathsf{Clos}(S_1) = \mathsf{Clos}(S_2)$.

Let's expand $F_{S_2}(\mathsf{Clos}(S_1))$. By definition of F,

$$F_{S_2}(\mathsf{Clos}(S_1)) = S_2 \cup \left\{ n \mid A\left(\sum_{m \in \mathrm{preds}(n)} W(m,n) \cdot \chi_{\mathsf{Clos}(S_2)}(m) \right) = 1 \right\}$$

Compare this to $F_{S_1}(\mathsf{Clos}(S_1))$:

$$F_{S_1}(\mathsf{Clos}(S_1)) = S_1 \cup \left\{ n \mid A\left(\sum_{m \in \mathrm{preds}(n)} W(m,n) \cdot \chi_{\mathsf{Clos}(S_2)}(m) \right) = 1 \right\}$$

Putting the two together, we can see that

$$\begin{aligned} F_{S_2}(\mathsf{Clos}(S_1)) &= (F_{S_1}(\mathsf{Clos}(S_1)) - S_1) \cup S_2 \\ &= (\mathsf{Clos}(S_1) - S_1) \cup S_2 && \text{(since } \mathsf{Clos}(S_1) \text{ is fixed under } S_1) \\ &= \mathsf{Clos}(S_1) && \text{(since } S_1 \subseteq S_2 \subseteq \mathsf{Clos}(S_1)) \end{aligned}$$ □

In the terminology of [37], Prop is a *cumulative* closure operator (it satisfies the cumulative property). An important feature of Clos is that it is nonmonotonic: it is not the case that for all

$A, B \in \mathsf{State}$, if $A \subseteq B$ then $\mathsf{Clos}(A) \subseteq \mathsf{Clos}(B)$. Intuitively, this is because our net's weights can be negative, and so $\mathsf{Clos}(B)$ can inhibit the activation of new neurons that would otherwise be activated by $\mathsf{Clos}(A)$.

[**TODO:** Add a note here where I talk about this proof strategy used in the proof of the Cumulative property. This proof trick is going to show up everywhere I want to prove something important about neural network closure patterns. I borrow the trick from Hannes, but my applications of the trick (from completeness to Hebbian update to first-order properties) is the biggest technical contribution of this dissertation, and is important for the reader to internalize.]

Proposition 3.2. (Leitgeb, [38; 39]) It is not the case that for all $S_1, S_2 \in \mathsf{State}_{\mathcal{N}}$, if $S_1 \subseteq S_2$, then $\mathsf{Clos}(S_1) \subseteq \mathsf{Clos}(S_2)$.

Proof. Consider the BFNN $\mathcal{N}$ in Figure 3.1 (b). Let the activation function A be $A(x) = 1$ iff $x > 0$. We have $S_1 = \{a\} \subseteq \{a, b\} = S_2$, and so the hypothesis holds. But $\mathsf{Clos}(S_1) = \{a, c\} \not\subseteq \{a, b\} = \mathsf{Clos}(S_2)$. Observe that c does not get activated in $\mathsf{Clos}(S_2)$ because the weights cancel each other out. □

Next, let's check that Reach is in fact a monotonic closure operator.

Proposition 3.3. For all $S, A, B \in \mathsf{State}_{\mathcal{N}}$,

Inclusion. $S \subseteq \mathsf{Reach}(S)$

Idempotent. $\mathsf{Reach}(\mathsf{Reach}(S)) = \mathsf{Reach}(S)$

Monotonic. If $A \subseteq B$ then $\mathsf{Reach}(A) \subseteq \mathsf{Reach}(B)$

Closed under ∪. $\mathsf{Reach}(A \cup B) = \mathsf{Reach}(A) \cup \mathsf{Reach}(B)$

Proof. I'll prove each in turn:

Inclusion. Suppose $n \in S$. We have the trivial path from $n \in S$ to itself.

Idempotent. The $(\leftarrow)$ direction follows from inclusion. As for $(\rightarrow)$, suppose $n \in \mathsf{Reach}(\mathsf{Reach}(S))$, i.e. there is a path from some $m \in \mathsf{Reach}(S)$ to n. By definition of Reach again, there is a path from some $x \in S$ to m. But we can put these together to obtain a path from $x \in S$ to n.

Monotonic. Suppose $A \subseteq B$, and let $n \in \mathsf{Reach}(A)$. By definition of Reach, we have a path from some $m \in A$ to n. But since $A \subseteq B$, $m \in B$. So we have a path from $m \in B$ to n, i.e. $n \in \mathsf{Reach}(B)$.

Closed under $\cup$. For the $(\rightarrow)$ direction, suppose $n \in \mathsf{Reach}(A \cup B)$. So there is a path from some $m \in A \cup B$ to n. We have two cases: $m \in A$, in which case we have a path from $m \in A$ to n; or $m \in B$, in which case we have a path from $m \in B$ to n.

As for the $(\leftarrow)$ direction, suppose $n \in \mathsf{Reach}(A) \cup \mathsf{Reach}(B)$. Similarly, we have two cases, and in either case we have a path from $n \in A \cup B$. So $n \in \mathsf{Reach}(A \cup B)$. □

$\mathsf{Reach}^{\downarrow}$ is as well, and the proof for this is the same, mutatis mutandis the direction of the path.

Proposition 3.4. For all $S, A, B \in \mathsf{State}_{\mathcal{N}}$,

Inclusion. $S \subseteq \mathsf{Reach}^{\downarrow}(S)$

Idempotent. $\mathsf{Reach}^{\downarrow}(\mathsf{Reach}^{\downarrow}(S)) = \mathsf{Reach}^{\downarrow}(S)$

Monotonic. If $A \subseteq B$ then $\mathsf{Reach}^{\downarrow}(A) \subseteq \mathsf{Reach}^{\downarrow}(B)$

Closed under $\cup$. $\mathsf{Reach}^{\downarrow}(A \cup B) = \mathsf{Reach}^{\downarrow}(A) \cup \mathsf{Reach}^{\downarrow}(B)$

[What interaction property do Clos, Reach and $\mathsf{Reach}^{\downarrow}$ share?]

4 Neural Network Semantics for Conditional Logic

[Rewrite this section so that I'm basically mentioning Hannes' semantics and results, and not proving anything new myself! Give an illustrative example, which I will be borrowing later for the logic of **C**]

[Introduce this all slowly. I will now explain how we can use neural networks as models to interpret formulas in logic. First, I will give Hannes' semantics for conditional logic. Then I will introduce my own semantics for modal logic based on his.]

Definition 4.1. Formulas in our conditional language $\mathcal{L}_{\Rightarrow}$ are given by [todo]

Definition 4.2. The semantics for $\mathcal{L}_{\Rightarrow}$ is given as follows. [todo]

Definition 4.3. We write $\models \alpha \Rightarrow \beta$ to mean all nets $\mathcal{N} \models \alpha \Rightarrow \beta$, and $\Gamma \models \alpha \Rightarrow \beta$ to mean every model $\mathcal{N}$ of Γ, i.e. $\mathcal{N} \models \gamma \Rightarrow \delta$ for all $\gamma \Rightarrow \delta \in \Gamma$ also models $\alpha \Rightarrow \beta$.

[Prove a few key properties for forward propagation, we can read the axioms directly off of these, then the proofs for the axioms' soundness follows]

Definition 4.4. The proof system for the conditional logic over $\mathcal{L}_{\Rightarrow}$ is given as follows: $\vdash \varphi$ iff [todo]

Definition 4.5. [Definition of $\Gamma \vdash \alpha \Rightarrow \beta$]

Theorem 4.6. (Hannes Leitgeb, [38; 39]) This proof system is sound; for all $\Gamma \subseteq \mathcal{L}_{\Rightarrow}$ and $\varphi \in \mathcal{L}_{\Rightarrow}$, if $\Gamma \vdash \varphi$ then $\Gamma \models \varphi$. [Rephrase for conditionals $\alpha \Rightarrow \beta$!]

Proof. [todo] □

Lemma 4.7. Let $\mathcal{M} = \langle W, \prec, V \rangle$ be a plausibility model, and $\mathcal{N}$ be given by the NAND construction above. For all conditional *terms* $\alpha \in$ [I need a symbol for this…], $[\![\alpha]\!]_{\mathcal{N}} = [\![\alpha]\!]^{C}_{\mathcal{M}}$ (the *complement* of $[\![\alpha]\!]_{\mathcal{M}}$!)

Proof. We proceed by induction on α.

[TODO] □

Lemma 4.8. Let $\mathcal{M} = \langle W, \prec, V \rangle$ be a plausibility model, and $\mathcal{N}$ be given by the NAND construction above. For all conditional formulas $\alpha \Rightarrow \beta \in \mathcal{L}^{\Rightarrow}$, where $\alpha, \beta \in$ [todo],

$$\mathcal{N} \models \alpha \Rightarrow \beta \text{ iff } \mathcal{M} \models \alpha \Rightarrow \beta$$

Proof. Combining the previous two lemmas, we have

$$\begin{array}{lll}
\mathcal{N} \models \alpha \Rightarrow \beta & \text{iff} & [\![\beta]\!]_{\mathcal{N}} \subseteq \mathsf{Clos}([\![\alpha]\!]_{\mathcal{N}}) \quad \text{(by definition)} \\
& \text{iff} & [\![\beta]\!]^{C}_{\mathcal{M}} \subseteq \mathsf{Clos}([\![\alpha]\!]^{C}_{\mathcal{M}}) \quad \text{(by Lemma 5.17)} \\
& \text{iff} & [\![\beta]\!]^{C}_{\mathcal{M}} \subseteq \mathsf{best}_{\prec}([\![\alpha]\!]_{\mathcal{M}})^{C} \quad \text{(by Lemma 5.17)} \\
& \text{iff} & \mathsf{best}_{\prec}([\![\alpha]\!]_{\mathcal{M}}) \subseteq [\![\beta]\!]_{\mathcal{M}} \quad \text{(flipping } \subseteq \text{ and complementing both sides)} \\
& \text{iff} & \mathcal{M} \models \alpha \Rightarrow \beta \quad \text{(by definition)}
\end{array}$$

□

Theorem 4.9. (Model Building for $\mathcal{L}^{\Rightarrow}$) For all consistent $\Gamma \subseteq \mathcal{L}^{\Rightarrow}$, there is finite $\mathcal{N}$ such that $\mathcal{N} \models \Gamma$.

Proof. [] □

Corollary 4.10. (Completeness for $\mathcal{L}^{\Rightarrow}$) For all consistent $\Gamma \subseteq \mathcal{L}^{\Rightarrow}$ and all conditionals $\alpha \Rightarrow \beta \in \mathcal{L}^{\Rightarrow}$,

$$\text{if } \Gamma \models \alpha \Rightarrow \beta \text{ then } \Gamma \vdash \alpha \Rightarrow \beta$$

Proof. [] □

- Need to also have modal logic semantics for plausibility models. The big thing here is that I have to prove we can still build a finite plausibility model in this setting!!!—completeness on the plausibility model end is going to be the hard part, and will involve temporal logic tricks.
- We will get a lot more mileage out of Lemma [todo]!

5 Neural Network Semantics for the Modal Logic of C

I can now state the specific logic and neural network semantics that I will consider. Let $p, q, \ldots$ be finitely many atomic propositions. 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 5.1. Formulas in the base modal language $\mathcal{L}_{\mathbf{C}}$ are given by

$$\varphi, \psi := p \mid \neg\varphi \mid \varphi \wedge \psi \mid \mathbf{A}\varphi \mid \Box\varphi \mid \mathbf{C}\varphi$$

$\top, \bot, \vee, \rightarrow, \leftrightarrow$ and the dual modal operators $\mathbf{E}, \Diamond, \langle\mathbf{C}\rangle$ are defined in the usual way.

The operator $\mathbf{A}$ is just the modal universal quantifier; $\mathbf{A}\varphi$ reads "φ holds everywhere." The $\Box$ operator is the standard relational modal one, with intended reading [todo]. Recall that we read conditionals $\varphi \Rightarrow \psi$ classically as "the best φ-states are ψ-states." [**UPDATE READING** The modal operator $\mathbf{C}$ is intended to be the "best" modality that occurs in this reading. In other words, $\mathbf{C}\varphi$ is meant to read "the current state (neuron) n is a best (most normal, plausible, typical) one."]

[Move to plausibility model discussion: Note that the semantics for $\mathbf{C}\varphi$ here are not the most general possible; [cite Johan, Sonja + Alexandru] consider $\mathsf{best}_{\prec,i,w}$ relations that depend on the agent i, as well as the current state w where $\mathbf{C}\varphi$ is being evaluated. [Define $\models_{\mathbf{Plaus}}$!!!]]

It is not immediately clear how these readings are justified; in my dissertation, I will justify these readings by connecting the neural network semantics I give here to more traditional semantics for $\mathbf{A}, \Box, \Box^{\downarrow}$, and $\mathbf{C}$.

At last, here are the neural network semantics for $\mathcal{L}_{\mathbf{C}}$. I will define the semantics for the dual operators $\mathbf{E}, \Diamond, \langle \mathbf{C} \rangle$ instead of $\mathbf{A}, \Box, \mathbf{C}$; either choice is equivalent, but this choice is somewhat more intuitive, since $\Diamond$ and $\langle \mathbf{C} \rangle$ directly map to the neural network operators Reach and Clos.

Definition 5.2. For all $\mathcal{N} \in \mathbf{Net}$, $n \in N$:

$$
\begin{array}{lll}
\mathcal{N}, n \Vdash p & \text{iff} & n \in V(p) \\
\mathcal{N}, n \Vdash \neg\varphi & \text{iff} & \mathcal{N}, n \nVdash \varphi \\
\mathcal{N}, n \Vdash \varphi \wedge \psi & \text{iff} & \mathcal{N}, n \Vdash \varphi \text{ and } \mathcal{N}, n \Vdash \psi \\
\mathcal{N}, n \Vdash \mathbf{E}\varphi & \text{iff} & [\![\varphi]\!] \neq \emptyset \\
\mathcal{N}, n \Vdash \Diamond\varphi & \text{iff} & n \in \mathsf{Reach}([\![\varphi]\!] \cup \{\mathsf{bias}\}) \\
\mathcal{N}, n \Vdash \langle \mathbf{C} \rangle \varphi & \text{iff} & n \in \mathsf{Clos}([\![\varphi]\!] \cup \{\mathsf{bias}\})
\end{array}
$$

where $[\![\varphi]\!] = \{n \in N \mid \mathcal{N}, n \Vdash \varphi\}$.

Proposition 5.3. By the definition of the duals $\mathbf{E}, \Diamond, \langle \mathbf{C} \rangle$, we can instead define the semantics for $\mathbf{A}, \Box, \mathbf{C}$ as follows. For all $\mathcal{N} \in \mathbf{Net}$, $n \in N$:

$$
\begin{array}{lll}
\mathcal{N}, n \Vdash \mathbf{A}\varphi & \text{iff} & [\![\varphi]\!] = N \\
\mathcal{N}, n \Vdash \Box\varphi & \text{iff} & n \in (\mathsf{Reach}([\![\varphi]\!]^{\mathsf{C}} \cup \{\mathsf{bias}\}))^{\mathsf{C}} \\
\mathcal{N}, n \Vdash \mathbf{C}\varphi & \text{iff} & n \in (\mathsf{Clos}([\![\varphi]\!]^{\mathsf{C}} \cup \{\mathsf{bias}\}))^{\mathsf{C}}
\end{array}
$$

Definition 5.4. We write $\mathcal{N} \models_{\mathbf{Net}} \varphi$ ("the net *models* φ") to mean $\mathcal{N}, n \Vdash \varphi$ for all $n \in N$; $\models_{\mathbf{Net}} \varphi$ to mean all nets $\mathcal{N} \models \varphi$; and finally, $\Gamma \models_{\mathbf{Net}} \varphi$ to mean every model $\mathcal{N}$ of Γ, i.e. $\mathcal{N} \models_{\mathbf{Net}} \psi$ for all $\psi \in \Gamma$ also models φ.

The following fact says that these neural semantics for $\mathbf{C}$ match up with Leitgeb's neural semantics for conditionals $\varphi \Rightarrow \psi$. Formally, conditionals $\varphi \Rightarrow \psi$ are expressible in $\mathcal{L}_{\mathbf{C}}$ by $\mathbf{A}(\mathbf{C}\varphi \to \psi)$. This justifies our reading of $\mathbf{C}\varphi$ as "the current state (neuron) n is a best (most normal, plau-

sible, typical) one."

Proposition 5.5. Let $\varphi \Rightarrow \psi \in \mathcal{L}_{\Rightarrow}$ (that is, $\varphi, \psi \in \mathcal{L}_{\text{prop}}$). Then for all $\mathcal{N} \in \mathbf{Net}$,

$$\mathcal{N} \models_{\mathbf{Net}} \varphi \Rightarrow \psi \text{ iff } \mathcal{N} \models_{\mathbf{Net}} \mathbf{A}(\mathbf{C}\varphi \rightarrow \psi)$$

Proof. [Todo] □

Example 5.6. [Give an example of a neural network model from before, but this time with our semantics in action. Give a number of constraints involving bird, penguin, and fly, and evaluate the truth of these at particular states]

[Discussion of semantics]

[Note that the syntax is backwards/dualed from the syntax for neural network semantics. But because they're duals of each other, it doesn't matter. (It's a known modal logic trick that you can do induction on either version/form.)]

[Talk about the possible worlds interpretation of these, & the local evaluation at particular neurons!]

[talk about $\mathcal{N} \models \varphi$, which in this context means "φ activates *all* of the neurons in $\mathcal{N}$", or "φ holds across the entire net." Also define $\Gamma \models \varphi$.]

[A reader might be confused about the "swaps" from $\mathsf{Reach}^{\downarrow}$ to $\Diamond$, as well as the choice to use diamond operators instead of the normal box ones.] [How can we justify the readings we had above? (at least intuitively)]

[I now have space to say these points in more detail (there's lots to say!)] Although these semantics are based on Leitgeb's [40], they differ in a few key ways. First, his semantics uses conditionals $\varphi \Rightarrow \psi$ to capture neural network inference, whereas mine instead centers on the modal operator $\langle \mathbf{C} \rangle$. Second, I include these additional operators $\Box$ and $\Box^{\downarrow}$ that are not mentioned in his work. Finally, Leitgeb battles with the issue of how to correctly interpret negation; I sidestep this issue by using neural networks for interpreting $\langle \mathbf{C} \rangle \varphi$ (where the "action" happens), but not for $\neg$ and $\wedge$. The bottom line is this: proving completeness for this logic is not necessarily just a matter of importing the proof from [40].

As with possible-worlds semantics for modal logic, we evaluate the truth of formulas *locally* at particular neurons.

I have tried to keep these semantics as close as possible to the ordinary presentation of semantics for modal logic. Basically, $\neg, \wedge, \vee, \rightarrow, \mathbf{A}, \mathbf{E}$ are all treated classically. But we defer to the neural network when faced with evaluating $\langle \mathbf{C} \rangle, \Diamond$, and $\Diamond^{\downarrow}$. Since $[\![\varphi]\!]$ can be any arbitrary set of neurons, it may not include the bias node. This means $[\![\varphi]\!]$ is not necessarily be a valid state of the neural network, since the bias might not be in it. So in order to apply the neural network functions $\mathsf{Reach}^{\downarrow}$, Reach, and Clos, we provide the bias on input by giving $[\![\varphi]\!] \cup \{\mathsf{bias}\} \in \mathsf{State}_{\mathcal{N}}$. [TODO: I will have to do this for the conditional logic semantics too!]

5.1 Why Consider this *Modal* Logic?

[The conditional logic came first, and I could have just re-used it in order to save myself a lot of work. But I think it's important to explain my aesthetic choices in moving to the modal logic, rather than just using Hannes' conditional logic. But this is such a technical detail that I should at least advise the reader to skip this section if they're not interested.]

5.2 What Makes these Semantics "Neural"

What makes these semantics "neural" or "connectionistic"? Easy answer: the key operators for inference are implemented in a neural network. Better answer: no single neuron/node holds the information for $\langle \mathbf{C} \rangle \varphi$ How is this different from relational or neighborhood semantics? Is this a meaningful difference? What does Hannes have to say about it in his dissertation?

Tbh maybe it just means "loosely inspired by real neuron and synapse behavior", but even then there are probably properties we can write down and check.

6 Proof System and Soundness

For **C** alone, [38] proves that the properties in Proposition [which?] are complete for Clos over

Axioms for $\Box$:

(Dual) $\Diamond\varphi \leftrightarrow \neg\Box\neg\varphi$
(Distr) $\Box(\varphi \wedge \psi) \leftrightarrow (\Box\varphi \wedge \Box\psi)$
(Refl) $\Box\varphi \rightarrow \varphi$
(Trans) $\Box\varphi \rightarrow \Box\Box\varphi$

Axioms for A:

(Dual) $\mathbf{E}\varphi \leftrightarrow \neg\mathbf{A}\neg\varphi$
(Distr) $\mathbf{A}(\varphi \rightarrow \psi) \rightarrow (\mathbf{A}\varphi \rightarrow \mathbf{A}\psi)$
(Refl) $\mathbf{A}\varphi \rightarrow \varphi$
(5) $\mathbf{E}\varphi \rightarrow \mathbf{A}(\mathbf{E}\varphi)$
(Interact) $\mathbf{A}\varphi \rightarrow \Box\varphi$

Axioms for C:

(Dual) $\langle\mathbf{C}\rangle\varphi \leftrightarrow \neg\mathbf{C}\neg\varphi$
(Refl) $\mathbf{C}\varphi \rightarrow \varphi$
(Trans) $\mathbf{C}\varphi \rightarrow \mathbf{C}\mathbf{C}\varphi$
(CM) $\mathbf{A}(\mathbf{C}\varphi \rightarrow \psi) \rightarrow (\mathbf{C}(\varphi \wedge \psi) \rightarrow \mathbf{C}\varphi)$
(Interact) $\Box\varphi \rightarrow \mathbf{C}\varphi$

Rules of Inference:

(MP) From $\vdash \varphi \rightarrow \psi$ and $\vdash \varphi$ we can infer $\vdash \psi$
(A-Nec) From $\vdash \varphi$, we can infer $\vdash \mathbf{A}\varphi$
($\Box$-Rep) From $\vdash \varphi \leftrightarrow \psi$, infer $\vdash \Box\varphi \leftrightarrow \Box\psi$
(C-Rep) From $\vdash \varphi \leftrightarrow \psi$, infer $\vdash \mathbf{C}\varphi \leftrightarrow \mathbf{C}\psi$

Figure 6.1. Axioms and rules of inference for [todo]

binary, feed-forward nets. We transcribe these into our modal language.

Definition 6.1. The proof system $\vdash$ for neural network semantics $\models_{\mathbf{Net}}$ over $\mathcal{L}_{\mathbf{C}}$ is given as follows: $\vdash\varphi$ iff either φ is valid in propositional logic, or φ is one of the axioms listed in Figure 6.1, or φ follows from some previously obtained formulas by one of the inference rules.

The axioms for $\Box$ and $\mathbf{A}$ form "*MLU*" (modal logic with the universal quantifier), and this is *almost* the standard complete axiomatization of this logic [Cite! Johan van Benthem mentions it in Modal Logic for Open Minds]; $\Box$ does not satisfy a **(Nec)** rule, so instead we have the rule of replacement **(Rep)** from classical modal logic. Aside from this difference, you can think of $\Box$ as the modal logic *S4*. $\mathbf{A}$ is just the modal logic *S5* with an additional interaction axiom $\mathbf{A}\varphi \rightarrow \Box\varphi$ stating that if φ holds everywhere, then it holds everywhere above the current world.

The axioms and rules for $\mathbf{C}$ come from the rules for cumulative conditionals $\varphi \Rightarrow \psi$. Let me explain how I derived them. First, from [cite Giordano's paper], if in conditional logic we allow for *propositional* nesting of formulas (we still cannot nest $\varphi \Rightarrow \psi$), we can express the axioms for cumulative logic as follows. Let $\vdash_{\text{Prop}}\varphi$ mean that φ is provable in the underlying propositional

system.

(Reflexivity). $\varphi \Rightarrow \varphi$

(Left Equivalence). If $\vdash_{\text{Prop}} \varphi \leftrightarrow \psi$ then $(\varphi \Rightarrow \rho) \rightarrow (\psi \rightarrow \rho)$

(Right Weakening). If $\vdash_{\text{Prop}} \varphi \rightarrow \psi$ then $(\rho \Rightarrow \varphi) \rightarrow (\rho \rightarrow \psi)$

(Cautious Monotonicity). $(\varphi \Rightarrow \psi) \wedge (\varphi \Rightarrow \rho) \rightarrow (\varphi \wedge \psi \rightarrow \rho)$

(Cautious Cut). $(\varphi \wedge \psi \Rightarrow \rho) \wedge (\varphi \Rightarrow \psi) \rightarrow (\varphi \Rightarrow \rho)$

We can straightforwardly translate these into the language of $\mathcal{L}_{\mathbf{C}}$ by replacing each $\varphi \Rightarrow \psi$ with $\mathbf{A}(\mathbf{C}\varphi \rightarrow \psi)$. From here, each of the above cumulative logic axioms follows from modal ones: **(Reflexivity)** follows from **(Refl)**; **(Left Equivalence)** follows from propositional axioms plus **(Rep)**; **(Right Weakening)** follows from propositional axioms plus modus ponens **(MP)**; **(Cautious Monotonicity)** follows from the modal axiom **(CM)**. [TODO: What about **(Cautious Cut)** now? Is it sound for **Net**? Does it need to be replaced by something else?]

Additionally, I have added the axioms **(Trans)** and **(Interact)** to reflect the fact that Clos is idempotent, and for all $S \in \mathsf{State}_{\mathcal{N}}$, $\mathsf{Clos}(S) \subseteq \mathsf{Reach}(S)$.

As a modal operator, **C** is non-normal and classical. Instead of a **(Distr)** axiom, **C** satisfies the weaker **(CM)**. The **C** operator also satisfies the replacement rule **(Rep)** from classical modal logic in lieu of **(Nec)** [cite **(Rep)** rule].

Definition 6.2. If $\Gamma \subseteq \mathcal{L}_{\mathbf{C}}$ is a set of formulas and $\varphi \in \mathcal{L}_{\mathbf{C}}$ a formula, then $\Gamma \vdash \varphi$ whenever there are finitely many $\psi_1, \ldots, \psi_k \in \Gamma$ such that $\vdash \psi_1 \wedge \ldots \wedge \psi_k \rightarrow \varphi$.

Theorem 6.3. (☞ Soundness for $\mathcal{L}_{\mathbf{C}}$ over $\models_{\mathbf{Net}}$) These rules and axioms are sound for neural network models; for all consistent $\Gamma \subseteq \mathcal{L}_{\mathbf{C}}$ and $\varphi \in \mathcal{L}_{\mathbf{C}}$, if $\Gamma \vdash \varphi$ then $\Gamma \models_{\mathbf{Net}} \varphi$.

Proof. Suppose $\Gamma \vdash \varphi$. That is, there are finitely many $\psi_1, \ldots, \psi_k \in \Gamma$ such that $\vdash \psi_1 \wedge \ldots \wedge \psi_k \rightarrow \varphi$, which in turn means (by **(MP)**) that if $\vdash \psi_1, \ldots, \vdash \psi_k$, then $\vdash \varphi$. Now let $\mathcal{N} \models \Gamma$. In particular, this means $\mathcal{N} \models \psi_1, \ldots, \psi_k$. I now need to show that $\mathcal{N} \models \varphi$. Since $\vdash \varphi$, φ is itself an axiom or follows from previously obtained formulas by the inference rules. In order to prove $\mathcal{N} \models \varphi$, it's enough to show that the axioms and rules of inference are valid (hold for all $\mathcal{N} \in \mathbf{Net}$ at all neurons $n \in N$ whatsoever).

To check:	Use:	To check:	Use:
(Distr) for $\Box$	Monotonicity of Reach	**(Refl)** for $\mathbf{C}$	Inclusion of Clos
(Refl) for $\Box$	Inclusion of Reach	**(Trans)** for $\mathbf{C}$	Idempotence of Clos
(Trans) for $\Box$	Idempotence of Reach	**(CM)** for $\mathbf{C}$	Cumulativity of Clos
(Interact), $\mathbf{A}+\Box$	$\mathsf{Reach}(S) \subseteq W$	**(Interact)**, $\Box + \mathbf{C}$	Reach contains Clos
(Rep) for $\Box$	Reach is a function	**(Rep)** for $\mathbf{C}$	Clos is a function

Figure 6.2. [caption!]

The propositional axioms and **(MP)** are totally classical, and are known to be sound. The **(Dual)** axioms for $\mathbf{A}$, $\Box$, and $\mathbf{C}$ are also sound (by definition of their duals). The semantics for $\mathbf{A}$ do not make use of the neural network at all, so the axioms **(Distr)**, **(Refl)**, **(5)** and the **(Nec)** rule for $\mathbf{A}$ are classically sound as well. Let $\mathcal{N} \in \mathbf{Net}, n \in N$. I will now check the remaining rules and axioms in detail. This is a bit tedious, so see Figure 6.2 for a summary.

(Distr) for $\Box$.

(Refl) for $\Box$. Suppose contrapositively $\mathcal{N}, n \not\Vdash \varphi$. So $n \in [\![\varphi]\!]^{\mathsf{C}}_{\mathcal{N}}$. Note that in particular $n \in [\![\varphi]\!]^{\mathsf{C}}_{\mathcal{N}} \cup \{\mathsf{bias}\}$ (we can simply add the bias node). By Inclusion of Reach, $n \in \mathsf{Reach}([\![\varphi]\!]^{\mathsf{C}}_{\mathcal{N}} \cup \{\mathsf{bias}\})$. By the semantics, $\mathcal{N}, n \Vdash \Diamond\neg\varphi$. By **(Dual)** for $\Box$, $\mathcal{N}, n \Vdash \neg\Box\varphi$, i.e., $\mathcal{N}, n \not\Vdash \Box\varphi$, and we are done.

(Trans for $\Box$). Suppose contrapositively $\mathcal{N}, n \not\Vdash \Box\Box\varphi$. This gives us $\mathcal{N}, n \Vdash \neg\Box\Box\varphi$. Applying **(Dual)** twice, we have $\mathcal{N}, n \Vdash \Diamond\Diamond\neg\varphi$. By the semantics, $n \in \mathsf{Reach}(\mathsf{Reach}([\![\varphi]\!]^{\mathsf{C}}_{\mathcal{N}} \cup \{\mathsf{bias}\}) \cup \{\mathsf{bias}\})$. Note that bias is always in the Reach of a set already containing bias, so we can simplify this to $n \in \mathsf{Reach}(\mathsf{Reach}([\![\varphi]\!]^{\mathsf{C}}_{\mathcal{N}} \cup \{\mathsf{bias}\}))$. By Idempotence of Reach, $n \in \mathsf{Reach}([\![\varphi]\!]^{\mathsf{C}}_{\mathcal{N}} \cup \{\mathsf{bias}\})$. But this means $\mathcal{N}, n \Vdash \Diamond\neg\varphi$, and by **(Dual)** we conclude that $\mathcal{N}, n \not\Vdash \Box\varphi$.

(Interact) for A and $\Box$. Suppose contrapositively $\mathcal{N}, n \not\Vdash \Box\varphi$. By **(Dual)** for $\Box$, $\mathcal{N}, n \Vdash \Diamond\neg\varphi$. By the semantics, $n \in \mathsf{Reach}([\![\varphi]\!]^{\mathsf{C}}_{\mathcal{N}} \cup \{\mathsf{bias}\}) \subseteq N$. [Todo]

(Rep) for $\Box$. Suppose $\models_{\mathbf{Net}} \varphi \leftrightarrow \psi$, i.e., $\varphi \leftrightarrow \psi$ holds for all nets, for all neurons. So for all $\mathcal{N} \in \mathbf{Net}$, $[\![\varphi]\!]_{\mathcal{N}} = [\![\psi]\!]_{\mathcal{N}}$. Since Reach is a function, $\mathsf{Reach}([\![\varphi]\!]_{\mathcal{N}} \cup \{\mathsf{bias}\}) = \mathsf{Reach}([\![\psi]\!]_{\mathcal{N}} \cup \{\mathsf{bias}\})$ for all $\mathcal{N}$. So in particular, $\mathsf{Reach}([\![\varphi]\!]^{\mathsf{C}}_{\mathcal{N}} \cup \{\mathsf{bias}\}) = (\mathsf{Reach}([\![\psi]\!]^{\mathsf{C}}_{\mathcal{N}}) \cup \{\mathsf{bias}\})^{\mathsf{C}}$ for all $\mathcal{N}$, which means $\models_{\mathbf{Net}} \Box\varphi \leftrightarrow \Box\psi$.

(Refl) for C. Suppose contrapositively $\mathcal{N}, n \not\Vdash \varphi$. So $n \in [\![\varphi]\!]^{\mathsf{C}}_{\mathcal{N}}$. Note that in particular $n \in [\![\varphi]\!]^{\mathsf{C}}_{\mathcal{N}} \cup \{\mathsf{bias}\}$ (we can simply add the bias node). By Inclusion of Clos, $n \in \mathsf{Clos}([\![\varphi]\!]^{\mathsf{C}}_{\mathcal{N}} \cup \{\mathsf{bias}\})$. By the semantics, $\mathcal{N}, n \Vdash \langle \mathbf{C} \rangle \neg \varphi$. By **(Dual)** for $\mathbf{C}$, $\mathcal{N}, n \Vdash \neg \mathbf{C}\varphi$, i.e., $\mathcal{N}, n \not\Vdash \mathbf{C}\varphi$, and we are done.

(Trans) for C. Suppose contrapositively $\mathcal{N}, n \not\Vdash \mathbf{CC}\varphi$. This gives us $\mathcal{N}, n \Vdash \neg \mathbf{CC}\varphi$. Applying **(Dual)** twice, we have $\mathcal{N}, n \Vdash \langle \mathbf{C} \rangle \langle \mathbf{C} \rangle \neg \varphi$. By the semantics, $n \in \mathsf{Clos}(\mathsf{Clos}([\![\varphi]\!]^{\mathsf{C}}_{\mathcal{N}} \cup \{\mathsf{bias}\}) \cup \{\mathsf{bias}\})$. Note that bias is always in the closure of a set, so we can simplify this to $n \in \mathsf{Clos}(\mathsf{Clos}([\![\varphi]\!]^{\mathsf{C}}_{\mathcal{N}} \cup \{\mathsf{bias}\}))$. By Idempotence of Clos, $n \in \mathsf{Clos}([\![\varphi]\!]^{\mathsf{C}}_{\mathcal{N}} \cup \{\mathsf{bias}\})$. But this means $\mathcal{N}, n \Vdash \langle \mathbf{C} \rangle \neg \varphi$, and by **(Dual)** we conclude that $\mathcal{N}, n \not\Vdash \mathbf{C}\varphi$.

(CM) for C. [Todo]

(Interact) for $\Box$ and C.

(Rep) for C. Suppose $\models_{\mathbf{Net}} \varphi \leftrightarrow \psi$, i.e., $\varphi \leftrightarrow \psi$ holds for all nets, for all neurons. So for all $\mathcal{N} \in \mathbf{Net}$, $[\![\varphi]\!]_{\mathcal{N}} = [\![\psi]\!]_{\mathcal{N}}$. Since Clos is a function, $\mathsf{Clos}([\![\varphi]\!]_{\mathcal{N}} \cup \{\mathsf{bias}\}) = \mathsf{Clos}([\![\psi]\!]_{\mathcal{N}} \cup \{\mathsf{bias}\})$ for all $\mathcal{N}$. So in particular, $\mathsf{Clos}([\![\varphi]\!]^{\mathsf{C}}_{\mathcal{N}} \cup \{\mathsf{bias}\}) = (\mathsf{Clos}([\![\psi]\!]^{\mathsf{C}}_{\mathcal{N}}) \cup \{\mathsf{bias}\})^{\mathsf{C}}$ for all $\mathcal{N}$, which means $\models_{\mathbf{Net}} \mathbf{C}\varphi \leftrightarrow \mathbf{C}\psi$. □

7 Reflections on Neural Network Verification

[Here's where I can discuss things like property verification vs verifying a particular neural network vs constraints + model building (alignment), extraction, “valuation search”, and Thomas Icards' method]

[say something about neuro-symbolic systems more generally in this framework] The basis for many neuro-symbolic systems is that they encode logical information into neural networks, or conversely, encode neural networks as models in logic (see [53]). Given these translations, certain neural networks and logic models are able to represent the same information.

Interlude: A History and Survey of Neural Network Semantics

[Rewrite this to flow better, and to relate neuro-symbolic systems to the work in this section. I'm trying to get across a broader perspective, not simply a single new system!] My thesis work builds on existing logics that use neural network semantics, and shares similarities with logics for modeling social networks. Additionally, my approach to modeling learning takes inspiration from work on learning in Dynamic Epistemic Logic (DEL). Here I'll take a moment to situate my thesis in this broader context and clarify what my contribution is in each case.

[Make it clear that this point of view is not my own original idea; but applying it broadly and developing it as far as I do *is* novel. Give a history of this perspective, and also make an effort to cast other neuro-symbolic systems within this framework (including the FLaNN work, etc.) This is the place for a proper literature review.]

The core idea behind neural network semantics—that neural networks can be treated as models for logic—actually dates back to the very first paper on neural networks. In McCulloch and Pitts [44], logical formulas are mapped directly to individual neurons in the net. This approach suffers from the well-known "grandmother cell" problem [30]: it is cognitively implausible that an individual neuron could represent a complex concept such as "grandmother". Instead, concepts in brain networks are distributed across multiple neurons at once.

Neural network semantics is based on a recent reimagining of this approach [7; 40], where logical formulas are mapped to sets of neurons rather than to individual neurons. Early work established the correspondence between inference in a neural network and nonmonotonic conditionals [7; 17; 38; 39]. More recently, [28; 29] proved soundness for inference over fuzzy neural networks. In my thesis work so far [34; 35], I applied ideas from Dynamic Epistemic Logic to model a simple update policy, Hebbian learning, in neural network semantics. The key results of this work are the first ever soundness and completeness theorems for any learning policy on neural networks.

[Go through popular neuro-symbolic proposals, and explain their relationship with neural network semantics. What order???]

Semantic Encodings.

Logics with Social Network Semantics.

Logic Tensor Networks.

Distributed Alignment Search.

DeepProbLog.

Logic Explained Networks.

Neural Networks as Automata.

Neural Network Fibring.

Logics with Social Network Semantics. [Make this a single point]It has recently come to my attention that a similar approach is being used to model group behavior in social networks. In these social network logics [4; 8; 18], nodes in the graph represent individual agents, and each formula is mapped to the set of agents that adopt a certain social attitude. Agents influence each other, and the spread of their attitudes is modeled much in the same way as forward propagation of a signal in a neural network.

This work shares essentially the same premise and techniques as neural network semantics; I personally view this as a case of parallel discovery. But the two approaches still differ in interesting ways. First, in some sense the two semantics are operating on different "levels": social networks model interactions between multiple agents, whereas neural networks model interactions between components of the same (single) agent. Second, the two differ in subject matter. Social network semantics focuses on different social links between agents, and how these links change [4]. Neural network semantics, my own work included, instead focuses on inferences and updates inspired by artificial and natural neural networks.

Chapter 4

Neural Network Model Constructions

1 Introduction

- model-building techniques
- applications: completeness + expressivity for neural networks viewed as logic models
- expressivity, including how to build neural network models from other types of models, and vice-versa

I will now show that these are all the axioms for neural network semantics, i.e., these axioms are complete. This is philosophically significant for the theory of neuro-symbolic AI—at a high level, completeness says that this logic exactly matches the closure behavior of binary neural networks. At the crux of this proof is the task of neural network model building: Given any set of constraints Γ consistent in this logic, I will need to construct a neural network model $\mathcal{N} \in \mathbf{Net}$ satisfying Γ. This is just as interesting, but is more practical: it means we can custom-design neural networks that model exactly the constraints over $\mathcal{L}_{\mathbf{C}}$ we would like them to have.

In previous work by Leitgeb [38; 39], completeness for the cumulative conditional logic over $\mathcal{L}_{\Rightarrow}$ was shown by way of plausibility models **Plaus**. Leitgeb showed that any plausibility model $\mathcal{M} \in \mathbf{Plaus}$ can be transformed into an equivalent neural network $\mathcal{N}' \in \mathbf{Net}$: For all $\varphi \Rightarrow \psi \in \mathcal{L}_{\Rightarrow}$,

$$\mathcal{N}' \models \varphi \text{ iff } \mathcal{M} \models \varphi$$

It would be nice if this technique lifted to the more expressive language $\mathcal{L}_{\mathbf{C}}$. But we have no such luck. In Section [todo], I will show that the language $\mathcal{L}_{\mathbf{C}}$ can express formulas that hold for all $\mathcal{M} \in \mathbf{Plaus}$, yet do not hold in every $\mathcal{N} \in \mathbf{Net}$. In a precise sense, plausibility models are *not* the correct analogy for explaining the closure behavior of neural networks.

Instead, I will directly build a neural network model using a modified version of the standard canonical model construction in modal logic. At first, I will build a net with infinitely many neu-

rons. Then, in section [todo] I will use the standard modal logic technique of filtration to transform this network into a finite one.

The proof I give here is non-constructive, since it first requires building an infinite neural network. Unfortunately, I do not know of a constructive algorithm for building a neural network model from Γ. For now, I will leave it as an open question.

Open Question 2. For a given finite set of constraints $\Gamma \subseteq \mathcal{L}_{\mathbf{C}}$, is there a constructive algorithm that produces a model $\mathcal{M} \in \mathbf{Plaus}$ and state $w \in W$ such that $\mathcal{M}, w \models \Gamma$?

2 The Canonical Neural Network Construction

Lemma 2.1. (Lindenbaum's Lemma [cite!]) We can extend any consistent set Γ to a maximally consistent set $\Delta \supseteq \Gamma$.

First, lets start by constructing a relational model inspired by the standard Kripke-style canonical model. I will use this model as a base for the canonical neural network model.

Definition 2.2. Let the canonical neural network model be $\mathcal{N}^c = \langle N^c, \mathsf{bias}^c, E^c, W^c, A^c, \eta^c, V^c \rangle$, where

- $N^c = \{\Delta \mid \Delta \text{ is maximally consistent over } \mathcal{L}_{\mathbf{C}}\}$. There are countably many; let's fix an order on these nodes $\Delta_0, \Delta_1, \ldots$
- $\Delta' E^c \Delta$ iff for all $\varphi \in \mathcal{L}_{\mathbf{C}}$, if $\Box\varphi \in \Delta$ then $\varphi \in \Delta'$.
- bias^c is [todo]
- Suppose $\Delta_i \in N$ has predecessors $\Delta'_{i1}, \Delta'_{i2}, \ldots, \Delta'_{ik}$. For each $\Delta'_{ij} E^c \Delta_i$, let

$$W^c(\Delta'_{ij}, \Delta_i) = (p_i)^j$$

where p_i is the i^{th} prime number. The idea is that each prime p_i uniquely codes for the node Δ_i, and the weight between Δ_i and its predecessor Δ'_{ij} is a power of p_i that uniquely codes for Δ'_{ij}. Consequently, any activation value x we care about is going to be a sum of powers of p_i, from which we can reconstruct precisely the Δ_i being activated and the predecessors Δ'_{ij} that were involved in activating it.

- Let $x \in \mathbb{Q}$, and suppose that $x = \sum_{j \in X} (p_i)^j$ for some $i \in \{1,...,|N^c|\}$ and some subset $X \subseteq \{1,..., k\}$, where k is the number of predecessors of node Δ_i. In other words, x uniquely codes for a subset $\{\Delta'_{ij} \mid \Delta'_{ij} E^c \Delta_i \text{ and } j \in X\}$ of predecessors of Δ_i, as explained just above. Let the activation function $A^c(x)$ be defined as follows:

$$A^c(x) = 1 \text{ iff for all } \psi, \langle \mathbf{C} \rangle \psi \in \bigcap_{\Delta'_{ij} E^c \Delta_i \text{ and } j \in X} \Delta'_{ij} \quad \text{implies } \langle \mathbf{C} \rangle \psi \in \Delta_i$$

  (If x does not code for any valid subset X, then simply set $A^c(x) = 0$.)

  [This is my current best guess for how to define A^c! It's tricky!]

- $\Delta \in V^c(p)$ iff $p \in \Delta$

This construction is still a bit opaque. First, the choices for N^c and V^c are totally standard fare when building a canonical model. The basic idea is that the nodes are maximally consistent sets Δ, and we can control what must be true at the current world by moving to the right Δ. The choice for E^c is the reverse of the canonical Kripke relation R^c, since neural networks look "downwards" rather than "upwards" for a set that caused the activation of n.

All of the action is in the choice for the activation function A^c. Let Δ_i have predecessors Δ'_{i1}, $\Delta'_{i2}, \ldots, \Delta'_{ik}$ as before, and let X denote the subset of them that are currently active. Intuitively, we want $A^c(x) = 1$ precisely whenever Δ_i "says" that the subset X "should" activate Δ_i. But since we don't yet have access to the semantics—we are currently *defining* the semantics!—we have to say this syntactically. [This is difficult to say syntactically, but what's written here is my current best guess!]

Proposition 2.3. (☞) The canonical neural network model $\mathcal{N}^c$ is a well-defined (infinite) neural network model.

Proof. [TODO] I need to show is that, for all $S \in \mathsf{State}_{\mathcal{N}^c}$, $\mathsf{Clos}_{\mathcal{N}^c}(S)$ is in fact the unique fixed point under S. [how the hell do I prove that…?] [What constraints to I need on E^c??] □

Lemma 2.4. (☞ Truth Lemma for $\mathcal{L}_{\mathbf{C}}$) We have, for all $\Delta \in N^c$, $\varphi \in \mathcal{L}_{\mathbf{C}}$,

$$\mathcal{N}^c, \Delta \Vdash \varphi \text{ iff } \varphi \in \Delta$$

Proof. [WARNING: I do not have this result yet! It's a pain in the ass to get right.]

By induction on φ. The propositional and boolean cases are straightforward. The $\mathbf{A}\varphi$ case is totally standard, and follows from the usual lemmas about maximally consistent sets [cite]. I'll skip to the most relevant cases. It's enough to consider the $\Diamond$ and $\langle \mathbf{C} \rangle$ cases:

$\Diamond$ Case.

$\langle \mathbf{C} \rangle$ Case. Observe that the Truth Lemma in this case, $\mathcal{N}^c, \Delta \Vdash \langle \mathbf{C} \rangle \varphi$ iff $\langle \mathbf{C} \rangle \varphi \in \Delta$, is equivalent to the claim:

$$\mathsf{Clos}_{\mathcal{N}^c}([\![\varphi]\!]) = \{\Delta \mid \langle \mathbf{C} \rangle \varphi \in \Delta\}$$

First, by construction $\mathsf{Clos}_{\mathcal{N}^c}([\![\varphi]\!])$ is the unique fixed point of the transition function $F_{[\![\varphi]\!]}$ under $[\![\varphi]\!]$. I will show here that the set $\{\Delta \mid \langle \mathbf{C} \rangle \varphi \in \Delta\}$ is *also* such a fixed point, i.e., $F_{[\![\varphi]\!]}(\{\Delta \mid \langle \mathbf{C} \rangle \varphi \in \Delta\}) = \{\Delta \mid \langle \mathbf{C} \rangle \varphi \in \Delta\}$. Since $\mathsf{Clos}_{\mathcal{N}^c}([\![\varphi]\!])$ is the *unique* fixed point under $[\![\varphi]\!]$, it will follow that these two states are the same. In other words, $\mathsf{Clos}_{\mathcal{N}^c}([\![\varphi]\!]) = \{\Delta \mid \langle \mathbf{C} \rangle \varphi \in \Delta\}$.

For the $(\rightarrow)$ direction, suppose $\Delta \in F_{[\![\varphi]\!]}(\{\Delta \mid \langle \mathbf{C} \rangle \varphi \in \Delta\})$. By definition of $F_{[\![\varphi]\!]}$, we have two cases:

Case 1. $\Delta \in [\![\varphi]\!]$. So $\mathcal{N}^c, \Delta \Vdash \varphi$, and so by the inductive hypothesis $\varphi \in \Delta$. By **(Refl)** and **(Dual)**, $\langle \mathbf{C} \rangle \varphi \in \Delta$.

Case 2. Δ is activated by its predecessors. For concreteness, let's say $\Delta = \Delta_i$ for some $i \in \mathbb{N}$, and let $\Delta'_{i1}, \Delta'_{i2}, \ldots, \Delta'_{ik}$ be its E^c-predecessors. Formally, we have

$$A^c\left(\sum_{\Delta'_{ij} E^c \Delta_i} W^c(\Delta'_{ij}, \Delta_i) \cdot \chi_{\{\Delta \mid \langle \mathbf{C} \rangle \varphi \in \Delta\}}(\Delta'_{ij})\right) = 1$$

Let this inner sum be denoted by x (that is, $A^c(x) = 1$). Since $A^c(x) = 1$, we must have $x = \sum_{j \in X} (p_i)^j$ for some subset $X \subseteq \{1, \ldots, k\}$. As explained above, from this sum of prime powers we can reconstruct the subset of predecessors that are active: $\{\Delta'_{ij} \mid \Delta'_{ij} E^c \Delta_i \text{ and } j \in X\}$. By construction of A^c, we have:

$$\text{For all } \psi, \langle \mathbf{C} \rangle \psi \in \bigcap_{\Delta'_{ij} E^c \Delta_i \text{ and } j \in X} \Delta'_{ij} \quad \text{implies } \langle \mathbf{C} \rangle \psi \in \Delta_i$$

Notice that in the sum above, Δ'_{ij} is active ($j \in X$) exactly when $\langle \mathbf{C} \rangle \varphi \in \Delta'_{ij}$. It follows that $\langle \mathbf{C} \rangle \varphi \in \bigcap_{\Delta'_{ij} E^c \Delta_i \text{ and } j \in X} \Delta'_{ij}$ holds. And so we conclude that $\langle \mathbf{C} \rangle \varphi \in \Delta_i = \Delta$, which is what I wanted to show.

As for the ($\leftarrow$) direction, suppose $\langle \mathbf{C} \rangle \varphi \in \Delta$. Let's say $\Delta = \Delta_i$ for some $i \in \mathbb{N}$, and let $X = \{j \mid \Delta'_{ij} E^c \Delta_i \text{ and } \langle \mathbf{C} \rangle \varphi \in \Delta'_{ij}\}$ be a set that selects active predecessors of Δ. I want to show that $\Delta \in F_{[\![\varphi]\!]}(\{\Delta \mid \langle \mathbf{C} \rangle \varphi \in \Delta\})$, i.e., these predecessors activate Δ:

$$A^c\left(\sum_{\Delta'_{ij} E^c \Delta_i} W^c(\Delta'_{ij}, \Delta_i) \cdot \chi_{\{\Delta \mid \langle \mathbf{C} \rangle \varphi \in \Delta\}}(\Delta'_{ij}) \right) = 1$$

By construction of A^c it's enough to show:

$$\text{For all } \psi, \langle \mathbf{C} \rangle \psi \in \bigcap_{\Delta'_{ij} E^c \Delta_i \text{ and } \langle \mathbf{C} \rangle \varphi \in X} \Delta'_{ij} \quad \text{implies } \langle \mathbf{C} \rangle \psi \in \Delta_i$$

Let $\psi \in \mathcal{L}_{\mathbf{C}}$ be a formula, and suppose $\langle \mathbf{C} \rangle \psi \in \bigcap_{\Delta'_{ij} E^c \Delta_i \text{ and } j \in X} \Delta'_{ij}$. I will now drop the subscripts and re-phrase this to be as general as possible: For all maximally consistent sets Δ', if $\Delta' E^c \Delta$ and $\langle \mathbf{C} \rangle \varphi \in \Delta'$, then $\langle \mathbf{C} \rangle \psi \in \Delta'$.

[todo, stuck!] □

3 Filtration: Building a Finite Neural Network

The fact that we can build this canonical neural network is not enough. The canonical net $\mathcal{N}^c$ is huge—it has infinitely many nodes, each of which are infinite sets of formulas! In this section, I will transform $\mathcal{N}^c$ into an equivalent finite neural network model $\mathcal{N} \in \mathbf{Net}$. [Relate this to the finite model property: every formula φ that has a model has a finite countermodel.] (Polynomial size would be nice, but I will leave this to future work. We probably won't be so lucky.)

The canonical neural network construction is not totally bizarre from a modal logic standpoint, so we can apply the standard technique of *filtration* to construct the finite model.

Definition 3.1. [Define filtration!]

[For computer science reader: this is similar to the Myhill-Nerode theorem about finding a

minimal DFA equivalent to a given DFA,]

Theorem 3.2. [A filtration of $\mathcal{M}$ preserves the formulas true in $\mathcal{M}$, at every world.]

Proposition 3.3. [A filtration of a (possibly infinite) $\mathcal{N}$ is still a well-defined neural network model.]

Theorem 3.4. [If Γ is finite, then any filtration of a (possibly infinite) neural network model $\mathcal{N}$ through Γ is also finite.]

Definition 3.5. [Define the fine filtration (I need to pick a specific filtration to see that one exists. In order to pick one that may prove to be more useful as a neural network, I pick the fine filtration (the most densely connected filtration) over the coarse filtration (the least densely connected filtration))]

Proposition 3.6. [The fine filtration is in fact a filtration]

Corollary 3.7. (Finite Model Property) [State the finite model property, putting it all together. We can build a finite]

Corollary 3.8. (Finite Model Building) [For any finite set of constraints Γ, we can construct a *finite* $\mathcal{N} \in$ **Net** satisfying Γ]

[Note that this doesn't actually give us a constructive *algorithm* for building the finite model, since we have to build the infinite canonical model first.] [After talking to Saul and Larry about this, I should just leave it as an open question and point to Larry's work on constructively building finite models from formulas. It is likely this trick will work here too.]

Open Question 3. For a given finite set of constraints $\Gamma \subseteq \mathcal{L}_{\mathbf{C}}$, is there a constructive algorithm that produces a model $\mathcal{M} \in$ **Plaus** and state $w \in W$ such that $\mathcal{M}, w \models \Gamma$?

4 Model Building and Completeness Theorems

[I can only state + prove these after doing filtration!!]

Theorem 4.1. (☞ Model Building for $\models_{\mathbf{Net}}$) For all consistent $\Gamma \subseteq \mathcal{L}_{\mathbf{C}}$, there is some finite $\mathcal{N} \in \mathbf{Net}$ and neuron $n \in N$ such that $\mathcal{N}, n \Vdash_{\mathbf{Net}} \Gamma$.

Proof. Let Γ be consistent, and let $\mathcal{M} \in \mathbf{Plaus}$, $w \in W$ be some finite plausibility model and world such that $\mathcal{M}, w \Vdash_{\mathbf{Plaus}} \varphi$ (which we know exists by Theorems [model building] and [filtration], see Appendix [] for details). Now let $\mathcal{N} \in \mathbf{Net}$ be given by the NAND construction above. We have $w \in W \subset N$, and so we can evaluate $\Vdash_{\mathbf{Net}}$ at $w \in N$. By Lemma 5.19, $\mathcal{N}, w \Vdash_{\mathbf{Net}} \varphi$, and we are done. □

Corollary 4.2. (☞ Completeness for $\models_{\mathbf{Net}}$) For all consistent $\Gamma \subseteq \mathcal{L}$ and all formulas $\varphi \in \mathcal{L}$,

$$\text{if } \Gamma \models_{\mathbf{Net}} \varphi \text{ then } \Gamma \vdash \varphi$$

Proof. Since the language $\mathcal{L}_{\mathbf{C}}$ has negation, completeness follows from model building in the usual way; this proof is entirely standard. Suppose contrapositively that $\Gamma \nvdash \varphi$. It follows that $\Gamma \vdash \neg\varphi$. So $\Gamma \cup \{\neg\varphi\}$ is consistent, and by Theorem 4.1 we have $\mathcal{N} \in \mathbf{Net}$ and $n \in N$ such that $\mathcal{N}, n \models_{\mathbf{Net}} \Gamma \cup \{\neg\varphi\}$. But then $\mathcal{N}, n \models_{\mathbf{Net}} \Gamma$ yet $\mathcal{N}, n \not\models \varphi$ [fix notation here], which is what we wanted to show. □

Example 4.3. (Building a Neural Network Model I) The discussion above is all very abstract. Let's see how the neural network model building is done by example. Consider our birds example from before, where $\mathcal{L}_{\text{prop}} = \{\text{bird}, \text{penguin}, \text{fly}\}$. Say we want to build a model satisfying the non-monotonic constraints from before, i.e.,

$$\Gamma = \{\text{bird} \Rightarrow \text{fly}, \text{penguin} \rightarrow \text{bird}, \text{penguin} \rightarrow \neg\text{fly}\}$$

That is, birds typically fly, penguins are birds, but penguins do not fly.

First, Theorem 4.1 says that we can construct a neural network *and a neuron* at which these constraints are true. But say we want these constraints to be true everywhere. This is simple enough: just add the universal quantifier $\mathbf{A}$ in front of each constraint. Additionally, bird⇒fly is not in the language of $\mathcal{L}_{\mathbf{C}}$, so we need to translate it. Recall that the conditional bird⇒fly is equivalent to $\mathbf{A}(\mathbf{C}\text{bird} \rightarrow \text{fly})$. Our set of constraints is now:

$$\Gamma = \{\mathbf{A}(\mathbf{C}\text{bird} \rightarrow \text{fly}), \mathbf{A}(\text{penguin} \rightarrow \text{bird}), \mathbf{A}(\text{penguin} \rightarrow \neg\text{fly})\}$$

First, we need to construct a finite plausibility model $\mathcal{M} \in \mathbf{Plaus}$. It is always possible to construct

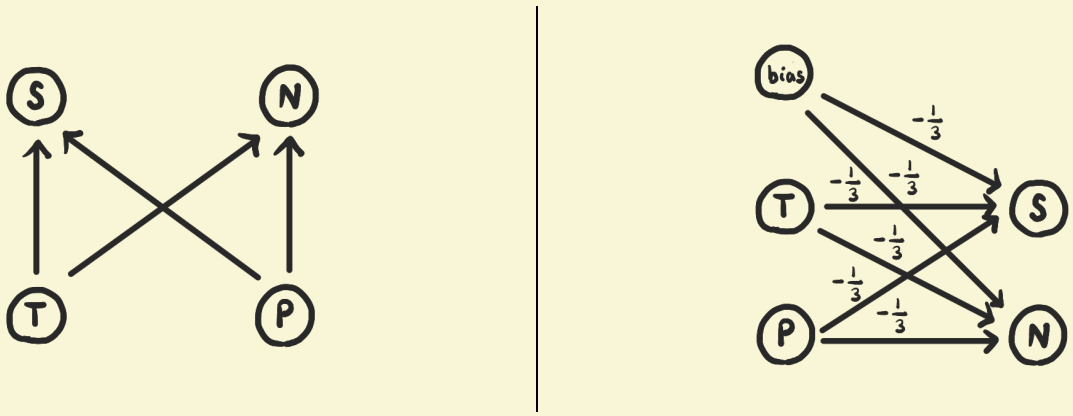

Figure 4.1. [TODO—caption! Separate into (a) left, and (b) right] T, P, S, N denote Tweety, Piper, Skipper, and Nils, respectively. For (b), mention that the activation function is $A(x) = 1$ iff $x > -1$.

a plausibility model with constraints in $\mathcal{L}_C$—see Appendix [TODO] for the details. However, the proof in Appendix [TODO] is not constructive, so I do not currently have an algorithm for doing this. For small examples like this one, it's not too difficult to hand-craft a model.

Take the following model, illustrated by Figure 4.1 (a). Let $W = \{\text{Tweety}, \text{Piper}, \text{Skipper}, \text{Nils}\}$, and let the propositional assignment V be given by $[\![\text{bird}]\!] = \{\text{Tweety}, \text{Piper}, \text{Skipper}, \text{Nils}\}$, $[\![\text{penguin}]\!] = \{\text{Skipper}, \text{Nils}\}$, and $[\![\text{fly}]\!] = \{\text{Tweety}, \text{Piper}\}$. That is, Skipper and Nils are the only penguins, and Tweety and Piper are the only birds that fly. Finally, let the plausibility order $(\prec) = \{(\text{Tweety}, \text{Skipper}), (\text{Tweety}, \text{Nils}), (\text{Piper}, \text{Skipper}), (\text{Piper}, \text{Nils})\}$.

In order to transform this model into a neural network, we apply the NAND construction. Let $N = W$, $V = V$, $E = \prec$, and add a new node bias. Connect the bias node to Skipper and Nils (since they are not $\prec$-minimal). Then set all weights to $W(m, n) = -\frac{1}{3}$. Finally, pick $A(x) = 1$ iff $x > -1$, and take an arbitrary learning rate η. The resulting neural network $\mathcal{N}$ is shown in Figure 4.1 (b).

Let's check that $\mathcal{N}$ does in fact satisfy the constraints Γ, i.e., $\mathcal{N} \models \Gamma$. Well,

- $\mathcal{N} \models \mathbf{A}(\text{penguin} \to \text{bird})$, since $[\![\text{penguin}]\!] = \{\text{Skipper}, \text{Nils}\} \subseteq [\![\text{bird}]\!]$ (for all $w \in N$, $w \Vdash \text{penguin}$ implies $w \Vdash \text{bird}$).
- Similarly, $\mathcal{N} \models \mathbf{A}(\text{penguin} \to \neg\text{fly})$, since $[\![\text{penguin}]\!] = \{\text{Skipper}, \text{Nils}\} \subseteq [\![\text{flies}]\!]^C$.
- Finally, let's check $\mathcal{N} \models \mathbf{A}(\mathbf{C}\text{bird} \to \text{fly})$. In terms of neural network semantics, this says that

for all $w \in N$, if $w \in (\mathsf{Clos}(\llbracket \mathsf{bird} \rrbracket^{\mathsf{C}} \cup \{\mathsf{bias}\}))^{\mathsf{C}}$ then $w \in \llbracket \mathsf{fly} \rrbracket$. Observe that

$$\begin{aligned}(\mathsf{Clos}(\llbracket \mathsf{bird} \rrbracket^{\mathsf{C}} \cup \{\mathsf{bias}\}))^{\mathsf{C}} &= (\mathsf{Clos}(\{\mathsf{bias}\}))^{\mathsf{C}} \\ &= \{\mathsf{bias}, \mathrm{Skipper}, \mathrm{Nils}\}^{\mathsf{C}} \\ &= \{\mathrm{Tweety}, \mathrm{Piper}\}\end{aligned}$$

From here we see that all w in this set $\{\mathrm{Tweety}, \mathrm{Piper}\}$ fly.

Example 4.4. (Building a Neural Network Model II) [TODO]

5 The Modelling Power of Neural Networks

So far, I have presented one important neural network model construction: the canonical construction. The canonical construction shows us how a neural network model $\mathcal{N}$ can simulate any finite choice of formulas Γ over the logic $\mathcal{L}_{\mathbf{C}}$. This gives us a sense of how powerful **Net** is—of what sorts of constraints neural networks (as logical models) are capable of representing. In this section, I will give a richer sense of how powerful neural networks are by comparing **Net** against other model classes. Along the way, I will catalog specialized neural network constructions that we can use to have **Net** simulate (or be simulated by) these other model classes. I aim to answer the questions:

- What kinds of models can **Net** simulate?
- What kind of models can be simulated by **Net**?

I will introduce a number of models for modal logics in the literature: relational models **Rel**, neighborhood models **Nbhd**, plausibility models **Plaus**, and a special case of social network models $\mathbf{SocialNet}^{\mathrm{MAJ}}$. I will include in the mix:

- *Pointed* neural network models $\mathbf{Net}^{*}$, briefly discussed in Section []
- "Universal" models **Univ**, which is degenerate case of **Rel** where all modalities are interpreted as the universal modality **A**.

To make the comparison fair, all models will share a generalized multi-modal language $\mathcal{L}_{\Box}$ given by

$$\varphi, \psi \coloneqq p \mid \neg\varphi \mid \varphi \wedge \psi \mid \{\Box_i\}_{i \in \boldsymbol{I}}\varphi$$

where $\boldsymbol{I}$ is some fixed set of indices. Notice that $\mathcal{L}_{\mathbf{C}}$ is a special case of $\mathcal{L}_{\Box}$, where the operators $\mathbf{A}, \Box, \mathbf{C}$ are different choices for $\Box_i$ that come with additional interaction axioms. More generally, you can think of each $\Box_i$ as representing a different modality (e.g., belief vs knowledge). Or, you could instead think of each $i \in \boldsymbol{I}$ as an agent in a multi-agent setting, and $\Box_i$ as agent i's belief (or knowledge, etc.).

Generalized Neural Network Models. This change in language requires a somewhat generalized neural network semantics. In this work so far, I have only defined the neural network closure operators Clos and Reach. But in principle we could define other closure operators, each reflecting a different kind of modality or conditional. I want to characterize what $\Box_i$-formulas a neural network can *in principle model*, and for that I need a more general definition.

Definition 5.1. A neural network model is $\mathcal{N} = \langle N, \{E_i\}_{i \in \boldsymbol{I}}, \{W_i\}_{i \in \boldsymbol{I}}, \{A_i\}_{i \in \boldsymbol{I}}, \eta, V \rangle$, where we now have a choice of E_i, W_i, A_i for each $i \in \boldsymbol{I}$.

Definition 5.2. A *pointed* neural network model is $\mathcal{N} = \langle N, \mathsf{bias}, \{E_i\}_{i \in \boldsymbol{I}}, \{W_i\}_{i \in \boldsymbol{I}}, \{A_i\}_{i \in \boldsymbol{I}}, \eta, V \rangle$, whose states are given by $\mathsf{State}_{\mathcal{N}} = \{S \mid S \subseteq N \text{ and } \mathsf{bias} \in S\}$.

Everything from Section [] generalizes: Each choice E_i, W_i, A_i specifies a state transition function $F_{i,S_0} \colon \mathsf{State}_{\mathcal{N}} \to \mathsf{State}_{\mathcal{N}}$, which in turn defines a closure function $\mathsf{Clos}_i \colon \mathsf{State}_{\mathcal{N}} \to \mathsf{State}_{\mathcal{N}}$. For the purposes of this section, let **Net** refer to the class of this generalized neural network model, and **Net*** refer to the pointed variation.

The semantics for $\Diamond_i$ generalizes as follows.

$$\mathcal{N}, n \Vdash \Diamond_i \varphi \text{ iff } n \in \mathsf{Clos}_i(\llbracket \varphi \rrbracket)$$

Similarly, for pointed models I define

$$\mathcal{N}, n \Vdash \Diamond_i \varphi \text{ iff } n \in \mathsf{Clos}_i(\llbracket \varphi \rrbracket \cup \{\mathsf{bias}\})$$

As expected, $\Box_i$ is defined to be the dual of $\Diamond_i$.

Note. I claimed above that the operators $\mathbf{A}, \Box, \mathbf{C}$ can be viewed as instances of $\Box_i$. For $\mathbf{C}$, this is fairly obvious, since the semantics for $\langle \mathbf{C} \rangle$ are exactly the same as $\Diamond_i$. To see how $\mathbf{A}$ can be

simulated via $\Box_i$, see the subsection on simulating **Univ** (where $\Box_i$ is defined to be exactly **A**). Finally, the $\Box$ operator was originally given semantics in terms of graph-reachability Reach. To see how it can be simulated via $\Box_i$, see the subsection on simulating **Rel**, and then reverse the edges in that construction. (The reason for this is that relational models "look upward" for some world u where φ holds, whereas Reach "looks downward.")

An Outline and Summary. I will now get in the weeds and formally compare these model classes. First, I will formally define simulations that will allow us to compare the modeling power of **Net** against the rest. Then, for each model class, I will prove simulation results that position them around **Net**. Figure [] illustrates a summary of these results. The figure displays a lattice where each class can simulate the models in the classes above it.

5.1 Measuring Modeling Power via Simulations

To compare the modeling power of neural networks with these other classes, I need to measure their ability to simulate each other. Formally,

Definition 5.3. Let $\mathscr{C}_1, \mathscr{C}_2$ be two model classes, with semantics defined over $\mathcal{L}_\Box$. A *simulation* f: $\mathscr{C}_1 \to \mathscr{C}_2$ is an injective function such that for all models $\mathcal{M} \in \mathscr{C}_1$ and formulas $\varphi \in \mathcal{L}_\Box$,

$$\mathcal{M} \models \varphi \text{ iff } f(\mathcal{M}) \models \varphi$$

If a simulation exists, we say that $\mathscr{C}_2$ *simulates* $\mathscr{C}_1$. If $\mathscr{C}_2$ simulates $\mathscr{C}_1$, but not conversely, we say that the simulation is *strict*.

In model theory, the *theory* of a class of models $\mathrm{Th}(\mathscr{C})$ is the set of validities, i.e., formulas that *must* be true for every $\mathcal{M} \in \mathscr{C}$. This is often used as a measure of the modeling power of $\mathscr{C}$: if $\mathrm{Th}(\mathscr{C}_1) \subseteq \mathrm{Th}(\mathscr{C}_2)$, then $\mathscr{C}_1$ is more general than $\mathscr{C}_2$ in the sense that it requires fewer axioms. (In other words, $\mathscr{C}_1$ can satisfy formulas that $\mathscr{C}_2$ cannot.)

Definition 5.4. Let $\mathscr{C}$ be a class of models. The *theory* of $\mathscr{C}$ (over $\mathcal{L}_\Box$) is

$$\mathrm{Th}(\mathscr{C}) = \{\varphi \in \mathcal{L}_\Box \mid \text{for all } \mathcal{M} \in \mathscr{C} \text{ we have } \mathcal{M} \models \varphi\}$$

The following proposition states the relationship between simulations and $\mathrm{Th}(\mathscr{C})$. In words, if $\mathscr{C}_2$ simulates $\mathscr{C}_1$, then $\mathscr{C}_2$ is the more general one. This proposition also gives us an easy test for showing that $\mathscr{C}_2$ does not simulate $\mathscr{C}_1$: just try to show $\mathrm{Th}(\mathscr{C}_2) \not\subseteq \mathrm{Th}(\mathscr{C}_1)$ by finding a formula $\varphi \in \mathrm{Th}(\mathscr{C}_1)$ such that $\varphi \notin \mathrm{Th}(\mathscr{C}_2)$.

Proposition 5.5. Let $\mathscr{C}_1$, $\mathscr{C}_2$ be two model classes. If $\mathscr{C}_2$ simulates $\mathscr{C}_1$, then $\mathrm{Th}(\mathscr{C}_2) \subseteq \mathrm{Th}(\mathscr{C}_1)$.

Proof. Suppose $\mathscr{C}_2$ simulates $\mathscr{C}_1$. This means there is an injection $f: \mathscr{C}_1 \to \mathscr{C}_2$ such that $\mathcal{M} \models \varphi$ iff $f(\mathcal{M}) \models \varphi$. Now let $\varphi \in \mathrm{Th}(\mathscr{C}_2)$, and let $\mathcal{M} \in \mathscr{C}_1$. Well, $f(\mathcal{M}) \in \mathscr{C}_2$, and so by the definition of $\mathrm{Th}(\mathscr{C}_2)$, $f(\mathcal{M}) \models \varphi$. By our simulation, this gives us $\mathcal{M} \models \varphi$, and we are done. □

Note. The notion of simulation that I've defined here only tells the semantic side of the story. If we want to compare different *languages* in addition to model classes, we need to define *infomorphisms* between two logics $(\mathcal{L}, \mathscr{C})$. (Pairs of languages, along with their model class, are also known as *institutions* in Institution Theory [cite institution theory, "Information Flow: The Logic of Distributed Systems", "Categories, Allegories", "Institution Theory"]). Infomorphisms are defined as follows.

Definition 5.6. Let $\mathscr{C}_1, \mathscr{C}_2$ be model classes, and let $\mathcal{L}_1, \mathcal{L}_2$ be languages. There is an *infomorphism* (aka *translation*) from $(\mathcal{L}_1, \mathscr{C}_1)$ into $(\mathcal{L}_2, \mathscr{C}_2)$ if there exist $f: \mathscr{C}_2 \to \mathscr{C}_1$, $\tau: \mathcal{L}_1 \to \mathcal{L}_2$ such that for all $\varphi \in \mathcal{L}_1, \mathcal{M} \in \mathscr{C}_2$

$$f(\mathcal{M}) \models \varphi \text{ iff } \mathcal{M} \models \tau(\varphi)$$

Since I have fixed the language $\mathcal{L}_\Box$, I won't necessarily need to use this idea. But I mention it here in order to set readers who are curious about the expressivity of $(\mathcal{L}_\Box, \mathbf{Net})$ on the right foot.

5.2 Relational Models (Rel)

Relational models are just Kripke models, which I briefly introduced in Section [].

A relational model is $\mathcal{M} = \langle W, \{R\}_{i \in I}, V \rangle$, where

- W is some finite set of worlds (or states)
- Each $R_i \subseteq W \times W$ (the accessibility relations)
- V: Proposition$\to \mathcal{P}(W)$ (the valuation function)

Define **Rel** to be the class of all such models, and define $\mathbf{Rel}_{S4}$ to be the class of all such models where R is additionally reflexive and transitive. The semantics for both classes is given by: [Todo, extend to the full language of $\mathcal{L}_C$!]

$$\begin{array}{lll} \mathcal{M}, w \Vdash p & \text{iff} & w \in V(p) \\ \mathcal{M}, w \Vdash \neg\varphi & \text{iff} & \mathcal{M}, w \nVdash \varphi \\ \mathcal{M}, w \Vdash \varphi \wedge \psi & \text{iff} & \mathcal{M}, w \Vdash \varphi \text{ and } \mathcal{M}, w \Vdash \psi \\ \mathcal{M}, w \Vdash \Box_i\varphi & \text{iff} & \text{for all } u \text{ with } wR_iu, \mathcal{M}, u \Vdash \varphi \end{array}$$

[Mention axioms, soundness, completeness (refer to the appendix!)]

5.3 Universal Models (Univ)

5.4 Neighborhood Models (Nbhd)

5.5 Plausibility Models (Plaus)

[(this is also where I can talk about $\mathbf{Net}^*$)]

5.6 Majority-Vote Social Networks ($\mathbf{SocialNet}^{\mathbf{MAJ}}$)

- Let's now do model translations to get at the expressivity of neural networks (over Modal and Conditional logic). Here's a hierarchy of the models above, to start:

- To make the comparison with neural networks fair, I will only consider the reflexive and transitive variants $\mathbf{Rel}_{S4}$, $\mathbf{Nbhd}_{S4}$ of relational and neighborhood models.
- [DIAGRAM]
- The translations from $\mathbf{Nbhd}_{S4}$ to **Plaus** and from **Plaus** to $\mathbf{Rel}_{S4}$ are folklore. Instead of giving these translations, I will instead translate from **Net** to $\mathbf{Rel}_{S4}$ and from $\mathbf{Nbhd}_{S4}$ to **Net** in order to explicilty show how to translations involving neural networks (neural network model constructions). The equivalence between **Plaus** and **Net** is already known, but the backwards direction has never been proven with an explicit model construction. So although the results in this section are already known, the proofs I give here are totally new.
- First, show translation from **Net** to $\mathbf{Rel}_{S4}$, and show there is no translation the other way around (axiom in $\mathbf{Rel}_{S4}$ that is not an axiom in **Net**)
- Next, show translation from $\mathbf{Nbhd}_{S4}$ to **Net**, and show there is no translation the other way around (axiom in **Net** that is not an axiom in $\mathbf{Nbhd}_{S4}$)
- Next, from the model-building construction in the completeness chapter, we have a translation from **Net** to **Plaus**.
- Explain that completeness implies that *in principle* we have a translation the other way (**Plaus** to **Net**), but it doesn't actually give the explicit model building procedure! Here is where I will give my own.
- Make a note here about the social majority logic above: There is a strict translation from **Net** to the social majority logic (**Net** is more general, in the sense that it requires fewer axioms).
- Give a brief note here on the expressive power of the conditional logic **Net** vs the modal logic **Net**.

Plausibility Models. A plausibility model, first introduced in [37], is $\mathcal{M} = \langle W, \{R\}_{i \in \mathbf{I}}, V \rangle$, i.e. the models themselves are just relational models. As before, I assume that W is finite, and as with $\mathbf{Rel}_{S4}$, each R_i is reflexive and transitive. The key difference is that we interpret $\Box_i \varphi$ to hold in the best (or most plausible) states satisfying φ. Formally, let $\mathsf{best}_{R_i}(S) = \{w \in S \mid \text{for all } u \in S, \neg u R_i w\}$

(the R_i-minimal states over S). We additionally impose the following "smoothness condition" [37] on best_{R_i}:

Postulate 5.7. For all models $\mathcal{M}$, $i \in \boldsymbol{I}$, sets S, and all $w \in W$, if $w \in S$ then either $w \in \mathsf{best}_{R_i}(S)$, or there is some vR_iw better than w that *is* the best, i.e. $v \in \mathsf{best}_{R_i}(S)$.

The new semantics for $\Box_i$ is

$$\mathcal{M}, w \Vdash \Box_i \varphi \quad \text{iff} \quad w \in \mathsf{best}_{R_i}(\llbracket \varphi \rrbracket)$$

where $\llbracket \varphi \rrbracket = \{u \mid \mathcal{M}, u \Vdash \varphi\}$. In practice, plausibility semantics coexist alongside relational semantics, so I allow some $\Box_i\varphi$ to be given relational semantics instead. Let **Plaus** be the class of all such models. Since we include relational operators, note that $\mathbf{Rel}_{S4} \subseteq \mathbf{Plaus}$.

Any plausibility operator $\Box_i$ picks out a corresponding conditional: $\Box_i\varphi \to \psi$ reads "the best φ are ψ," which in the KLM tradition is the semantics for the conditional $\varphi \Rightarrow \psi$.

[Mention axioms, soundness, completeness (refer to the appendix!)]

Social Network Models.

- Introduce social network models, an example with a [DIAGRAM] would be nice!
- In these social network logics [4; 8; 18], nodes in the graph represent individual agents, and each formula is mapped to the set of agents that adopt a certain social attitude. Agents influence each other, and the spread of their attitudes is modeled much in the same way as forward propagation of a signal in a neural network.
- Give a concrete social network logic: Social majority. Make sure to emphasize that social majority is one of many (!) choices, and is a relatively simple choice to model.
- Both kinds of models use fundamentally the same approach ("This work shares essentially the same premise and techniques as neural network semantics"): distributed information over several connected nodes, modal operator interpreted as the fixed-point of some diffusion
- But the two approaches still differ in interesting ways. First, in some sense the two semantics are operating on different "levels": social networks model interactions between multiple

agents, whereas neural networks model interactions between components of the same (single) agent. Second, the two differ in subject matter. Social network semantics focuses on different social links between agents, and how these links change [4]. Neural network semantics, my own work included, instead focuses on inferences and updates inspired by artificial and natural neural networks.

Neighborhood Models.

A neighborhood model is $\mathcal{M} = \langle W, \{N\}_{i\in\mathbf{I}}, V\rangle$, where W and V are as before and each $N_i\colon W \to \mathcal{P}(\mathcal{P}(W))$ is an accessibility *function*. The intuition is that N_i maps each state w to the "formulas" (sets of states) that hold at w. Define **Nbhd** to be the class of all neighborhood models.

Moreover, the *core* of N is $\cap N(x) = \bigcap_{X\in N(w)} X$. As with **Rel**, let $\mathbf{Nbhd}_{S4}$ be the class of all neighborhood models that are additionally reflexive ($\forall w, w \in \cap N(w)$) and transitive ($\forall w$, if $X \in N(w)$ then $\{v \mid X \in N(v)\} \in N(w)$).

The semantics for both classes is the same as the previous classes, except the $\Box_i$ case is now:

$$\mathcal{M}, w \Vdash \Box_i\varphi \text{ iff } [\![\varphi]\!] \in N_i(w)$$

where again $[\![\varphi]\!] = \{u \mid \mathcal{M}, u \Vdash \varphi\}$.

Proposition 5.8. (☞) There is a strict translation from (Modal, **Net**) to (Modal, $\mathbf{Rel}_{S4}$).

Proof. []

Proposition 5.9. (☞) There is a strict translation from (Modal, $\mathbf{Nbhd}_{S4}$) to (Modal, **Net**).

Proof. []

Proposition 5.10. There is a translation from ($\mathcal{L}_{\mathbf{C}}$, **Net**) to ($\mathcal{L}_{\mathbf{C}}$, **Plaus**).

Proof. [This is just a corollary of completeness: Mention that we just use the NAND construction from the Completeness proof.]

An easy corollary of completeness is that exactly the same axioms hold over **Plaus** and **Net**, i.e. [This is no longer true!!! Rewrite this whole story here!]

Corollary 5.11. Th(**Plaus**) = Th(**Net**).

Proof. We have:

$$\begin{array}{lll} \varphi \in \mathrm{Th}(\mathbf{Plaus}) & \text{iff} \ \models_{\mathbf{Plaus}} \varphi & \text{(By definition)} \\ & \text{iff} \ \vdash_{\mathbf{C}} \varphi & \text{(By weak completeness for } \mathbf{Plaus}\text{)} \\ & \text{iff} \ \models_{\mathbf{Net}} \varphi & \text{(By weak completeness for } \mathbf{Net}\text{)} \\ & \text{iff} \ \varphi \in \mathrm{Th}(\mathbf{Plaus}) & \text{(By definition)} \end{array}$$

□

Based on this, and the completeness result, we might naturally expect that there is a translation from $(\mathcal{L}_{\mathbf{C}}, \mathbf{Plaus})$ to $(\mathcal{L}_{\mathbf{C}}, \mathbf{Net})$. But surprisingly, there is *no* such translation!

Theorem 5.12. (☞) There is no translation from $(\mathcal{L}_{\mathbf{C}}, \mathbf{Plaus})$ to $(\mathcal{L}_{\mathbf{C}}, \mathbf{Net})$.

Proof. [First, I can prove it if $\tau(\varphi) = \varphi$! I should try to generalize this for any τ!] For $\tau(\varphi) = \varphi$, let $f : \mathbf{Net} \to \mathbf{Plaus}$ be any arbitrary way to transform a neural net into a plausibility model. I need to show that there is some net $\mathcal{N} \in \mathbf{Net}$ and some formula $\varphi \in \mathcal{L}_{\mathbf{C}}$ such that one of $f(\mathcal{N}) \models \varphi$ or $\mathcal{N} \models \varphi$ holds and the other does not. Let $\varphi = \mathbf{C}p$ for some proposition p, and let $\mathcal{N}$ be the net in Figure [DIAGRAM] (a). Since $[\![\mathbf{C}p]\!]_{\mathcal{M}} = \mathsf{best}_{\prec_{f(\mathcal{N})}}([\![p]\!])$ and $[\![\mathbf{C}p]\!]_{\mathcal{N}} = (\mathsf{Clos}_{\mathcal{N}}([\![p]\!]^{\mathsf{C}}))^{\mathsf{C}}$, it's enough to show that

$$\text{There is some } [\![p]\!] \in \mathsf{State}_{\mathcal{N}} \text{ such that } \mathsf{best}_{\prec_{f(\mathcal{N})}}([\![p]\!]) \neq (\mathsf{Clos}_{\mathcal{N}}([\![p]\!]^{\mathsf{C}}))^{\mathsf{C}}$$

Well, for this particular net we have $\mathsf{State}_{\mathcal{N}} = \{\{\mathsf{bias}\}, \{\mathsf{bias}, a\}, \{\mathsf{bias}, b\}, \{\mathsf{bias}, a, b\}\}$ (all subsets of the nodes that contain the bias node). Any $[\![p]\!] \in \mathsf{State}_{\mathcal{N}}$ must be one of these four sets.

Suppose for contradiction that for all such $[\![p]\!]$, $\mathsf{best}_{\prec_{f(\mathcal{N})}}([\![p]\!]) = (\mathsf{Clos}_{\mathcal{N}}([\![p]\!]^{\mathsf{C}}))^{\mathsf{C}}$. In particular, $\mathsf{best}_{\prec_{f(\mathcal{N})}}(\{\mathsf{bias}, a, b\}) = (\mathsf{Clos}_{\mathcal{N}}(\{\mathsf{bias}, a, b\}^{\mathsf{C}}))^{\mathsf{C}} = \{\mathsf{bias}, a, b\}$. (See the calculation in Figure [DIAGRAM] (b).) But this implies that bias, a, and b must all be mutually incomparable via $\prec_{f(\mathcal{N})}$. But then

$$\mathsf{best}_{\prec_{f(\mathcal{N})}}(\{\mathsf{bias}, b\}) = \{\mathsf{bias}, b\} \neq \{\mathsf{bias}\} = (\mathsf{Clos}_{\mathcal{N}}(\{\mathsf{bias}, b\}^{\mathsf{C}}))^{\mathsf{C}}$$

(The first equality comes from bias, a, b incomparable; the second follows from the calculation in Figure [DIAGRAM] (b).) □

In particular, the neural network used in the proof [refer to it] cannot be transformed into an equivalent plausibility model. Moreover, this net isn't a particularly strange one—it's just an ordinary feed-forward weighted net! So we have an intuition that neural networks are more general in

some sense. But Corollary 5.11 says that no axiom $\varphi \in \mathcal{L}_{\mathbf{C}}$ witnesses this difference. This means that there *is* some difference between them, but $\mathcal{L}_{\mathbf{C}}$ is not expressive enough to point to it. [the interesting conclusion of this reasoning—what axiom can give us the difference??]

Theorem 5.13. (☞) There is a strict translation from (Modal, **Net**) to the social majority logic [give it a name]

Proof. [] □

Theorem 5.14. (☞) There is a strict translation from (Modal, **Net**) to (Conditional, **Net**). [does it go this way, or the other way??]

Proof. [] □

[Integrate the following into this chapter!]

The Graph-Reachability Construction. I will show here how the more general Clos operator can be used to simulate Reach and $\mathsf{Reach}^{\downarrow}$. Suppose we are given a graph $\langle N, E\rangle$ and a valuation function V. How can we build a neural network whose closure Clos is graph-reachability Reach? We want to build a net:

$$\mathcal{N} = \langle N, \mathsf{bias}, E, W, A, \eta, V\rangle$$

[what to do about bias and η?] For the weights, pick

$$W(m,n) = \begin{cases} 1 & \text{if } mEn \\ 0 & \text{otherwise} \end{cases}$$

Then pick the activation function $A(x) = 1$ iff $x > 0$. Recall that $n \in F_{S_0}(S)$ iff $n \in S_0$ or is activated by its predecessors in S. In this case, $n \in F_{S_0}(S)$ whenever $n \in S_0$ or at least one E-predecessor m of n is in S. I call this the *graph-reachability construction* because the closure $\mathsf{Clos}(S)$ produces exactly those nodes graph-reachable from S:

Proposition 5.15. For all states $S \in \mathsf{State}$, $\mathsf{Clos}(S) = \mathsf{Reach}(S)$.

Proof. First, the $(\subseteq)$ direction. Let $n \in \mathsf{Clos}(S) = F_S^k(S)$ for some $k \in \mathbb{N}$. By induction on k.

Base Step. $n \in F_S^0(S) = S$. So there is a trivial E_i-path (length=0) from $n \in S$ to itself.

Inductive Step. Let $k \geq 0$. We have $n \in F_S^k(S) = F_S(F_S^{k-1}(S))$. By construction of F_S, we have two cases: Either $n \in F_S^{k-1}(S)$ or at least one E-predecessor x of n is in $F_S^{k-1}(S)$. In the first case, our inductive hypothesis gives a path from some $m \in S$ to n. In the second case, our inductive hypothesis gives a path from some $m \in S$ to x. But since xEn, we can extend this path to be from m to n.

As for the $(\supseteq)$ direction, suppose there is an E-path from some $m \in S$ to n. We proceed by induction on the length of that path.

Base Step. The path is trivial, i.e. has length 0. So $n \in S$. But $S = F_i^0(S) \subseteq \mathsf{Clos}_i(S)$, and so $n \in \mathsf{Clos}_i(S)$.

Inductive Step. Say the path is of length $l \geq 0$. Let x be some immediate E_i-predecessor of n. By the inductive hypothesis, $x \in \mathsf{Clos}_i(S)$, and so $x \in F_i^k(S)$ for some natural k. But since x is an E_i-predecessor of n, by construction of F_i, $n \in F_i(F_i^k(S)) = F_i^{k+1}(S)$. Since $\mathsf{Clos}_i(S)$ is a closure, it includes $F_i^{k+1}(S)$. So $n \in \mathsf{Clos}_i(S)$. □

As for $\mathsf{Reach}^{\downarrow}$, we first *reverse* the edges E, and then do the graph-reachability construction. In other words, let $mE'n$ iff nEm,

$$W(m,n) = \begin{cases} 1 & \text{if } mE'n \\ 0 & \text{otherwise} \end{cases}$$

and pick A, η, V the same as above. For this construction, the closure $\mathsf{Clos}(S)$ produces exactly those nodes which reach some node in S. The proof is similar to the proof for Reach. [Check that that's actually true!!]

Proposition 5.16. For all states $S \in \mathsf{State}$, $\mathsf{Clos}(S) = \mathsf{Reach}^{\downarrow}(S)$.

The Social Majority Construction. [Introduce the idea of social networks here if I haven't already, and mention the "social majority" propagation/diffusion (tell it slowly, like a story). I will show that our neural networks can simulate this simple social majority operator (make the social majority operator a bit more formal).]

As before, we want to build a neural network $\mathcal{N}$ where the graph $\langle N, E_i \rangle$, bias, and evaluation V are given. This time, pick $W_i(m,n) = \frac{1}{|\mathrm{preds}(n)|}$, and then pick $A_i(x) = 1$ iff $x \geq \frac{1}{2}$. Visually, for each

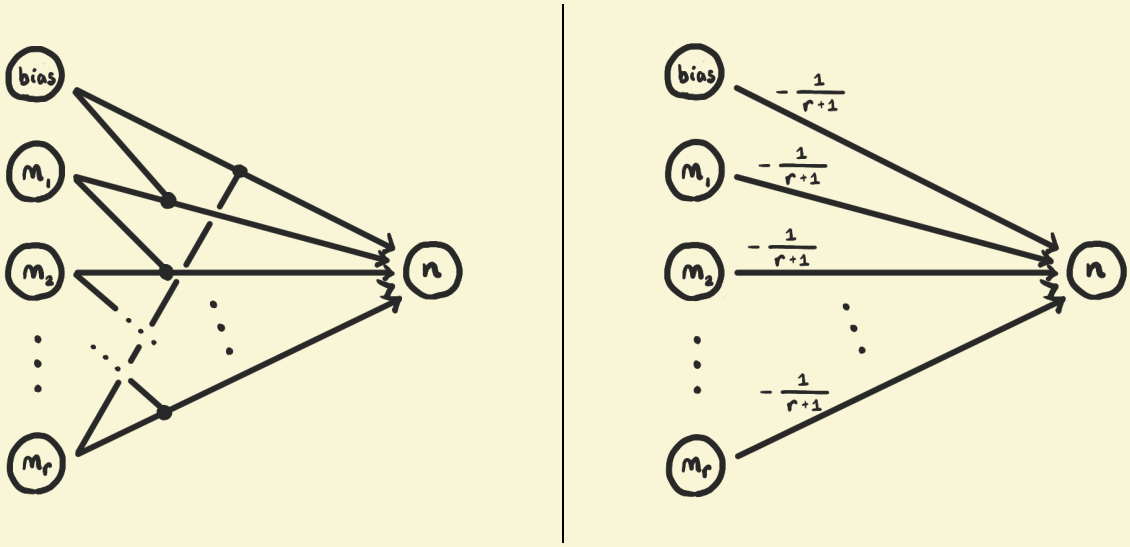

Figure 5.1. [TODO—caption! Separate into (a) left, and (b) right]

node n and its predecessors $m_1, \ldots, m_r$ we have

[DIAGRAM!]

This gives us $n \in F_{S_0}(S)$ if $n \in S_0$ or if the majority (more than half) of E-predecessors are in S. In this case, the closure Clos can be interpreted as the diffusion of an opinion or attitude through a social network. This is one of the choices that [8] consider for modelling influence in social networks. [this paragraph is a bit terse now that I'm writing a longer version of this.]

The NAND Construction. Suppose we have plausibility model $\mathcal{M} = \langle W, R, \prec, V \rangle$ and we want to construct an equivalent neural network $\mathcal{N}$. In [38], Hannes first does this for *inhibition nets*, i.e., nets with inhibitory edges that block excitatory edges. (He handles weighted nets later.) I will first consider his construction, and then modify it for weighted nets.

Here's the inhibition net construction: First, create a fresh node bias. Take $N = W \cup \{\mathsf{bias}\}$ (so $\mathcal{N}$ is still finite), $V = V$, let the excitatory edges be exactly uEv iff vRu (E is the reverse of R). Next, create an edge from bias to every n that is not E-minimal (in other words, if n has any predecessors at all, then bias is one of them). Then for each node n and its predecessors $\mathsf{bias} = m_0, m_1, \ldots, m_r$, connect inhibition edges as illustrated in Figure 5.1 (a).

That is, each node m_i is inhibited by m_{i-1} ($\mathsf{bias} = m_0$ inhibited by m_r). This has the following effect: if all m_i activate, they each inhibit each other, and so n does not activate. If only *some* m_i activate, then there is some m_i that is uninhibited, and so n activates. And finally, since bias is always active we cannot have *no* m_i active. In other words, $n \in F_{S_0}(S)$ iff $n \in S_0$, or *not all non-bias predecessors m are in S*. (Since bias is always active, this results in a NAND-like output.)

We can simulate this effect with weighted neural networks. Let $W(m,n) = -\left(\frac{1}{r+1}\right)$ (the extra $+1$ accounts for the bias), and let pick $A(x) = 1$ iff $x > -1$. For now, the choice of learning rate η is arbitrary (see Section [todo]). This construction is illustrated in Figure 5.1 (b). Take a moment to check that $n \in F_{S_0}(S)$ iff $n \in S_0$, or at least one non-bias predecessor $m \notin S$. [also, the negative values ensure A is nondecreasing!]

What is the relationship between this neural network's fixed points $\mathsf{Clos}(S)$ and the plausibility model's minimal states $\mathsf{best}_{\prec}(S)$? It turns out that for this NAND construction, Clos is precisely the *dual* of $\mathsf{best}_{\prec}$:

Lemma 5.17. Let $\mathcal{M} = \langle W, R, \prec, V\rangle$ be a plausibility model such that $R = (\prec_{\text{reverse}})$. Let $\mathcal{N}$ be given by the NAND construction above. For all $S \subseteq \mathsf{State}_{\mathcal{N}}$, $\mathsf{Clos}_{\mathcal{N}}(S) = (\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}}$. In a slogan:

$$\mathsf{Clos}_{\mathcal{N}}(S) \textit{ is the dual of } \mathsf{best}_{\prec}(S).$$

Note. There is some ambiguity over what universe the complements in $(\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}}$ are taken. The bias node of course occurs in S, since S is a state of the net. But we should also allow for bias to occur in the final complement of $(\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}}$, in order for this term to be exactly $\mathsf{Clos}_{\mathcal{N}}(S)$. So I interpret all of these complements over universe N (rather than W). Explicitly, the claim is:

$$\text{For all } S \subseteq \mathsf{State}_{\mathcal{N}}, \mathsf{Clos}_{\mathcal{N}}(S) = N \setminus (\mathsf{best}_{\prec}(N \setminus S))$$

Proof. Once again, I will take advantage of the fact that fixed points of the transition function F_S are unique. First, since $\mathsf{Clos}(S)$ is a fixed point under S, we have $F_S(\mathsf{Clos}(S)) = \mathsf{Clos}(S)$. But I will show that $(\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}}$ is *also* a fixed point under S, i.e. $F_S((\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}}) = (\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}}$. Since we assumed that there is a *unique* fixed point under S, it will follow that these two sets must be the same. In other words, $\mathsf{Clos}(S) = (\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}}$.

Recall that the type of F_S is F_S: $\mathsf{State}_{\mathcal{N}} \to \mathsf{State}_{\mathcal{N}}$. We need to make sure that $(\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}} \in \mathsf{State}_{\mathcal{N}}$, i.e., it is in fact a state of the neural network and we can apply F_S to it. In particular, we need to check that $\mathsf{bias} \in (\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}}$. Well, $S \subseteq \mathsf{State}_{\mathcal{N}}$, and so $\mathsf{bias} \in S$. This means $\mathsf{bias} \notin S^{\mathsf{C}}$, and by the contrapositive of best-inclusion, $\mathsf{bias} \notin (\mathsf{best}_{\prec}(S^{\mathsf{C}}))$, which is what I wanted to show. So the term $F_S((\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}})$ is well-defined.

We are now ready to show that $F_S((\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}}) = (\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}}$. For the $(\to)$ direction, suppose $n \in F_S((\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}})$. By construction of F_S, we have two cases:

Case 1. $n \in S$. In this case, we trivially have $n \notin \mathsf{best}_{\prec}(S^{\mathsf{C}})$, since n is not even in S^{C}. And so $n \in (\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}}$.

Case 2. At least one non-bias predecessor mEn is $m \notin (\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}}$. So $m \in \mathsf{best}_{\prec}(S^{\mathsf{C}})$. Since m is a predecessor of n (mEn), nRm (reverse it). But then since $R = (\prec_{\text{reverse}})$, $m \prec n$. So m is $\prec$-better than n (that is, $m \prec n$), which implies that n cannot be a best S^{C}-element: $n \notin \mathsf{best}_{\prec}(S^{\mathsf{C}})$. So $n \in (\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}}$.

As for the $(\leftarrow)$ direction, suppose contrapositively that $n \notin F_S((\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}})$. By construction, this means that $n \notin S$ and for all m with $m \prec n$ (including bias of course), $m \in (\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}}$.

[all predecessors m with mEn m of n (including bias of course) are in $(\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}}$.]

We already have $n \in S^{\mathsf{C}}$; from here I'd like to show that n is the *best* S^{C}-element. Suppose for contradiction that $n \notin \mathsf{best}_{\prec}(S^{\mathsf{C}})$. By the smoothness condition (see Appendix [todo]) there must be some $m \in S^{\mathsf{C}}, m \prec n$ that is the best, i.e., $m \in \mathsf{best}_{\prec}(S^{\mathsf{C}})$. Since $m \prec n$ and $R = (\prec_{\text{reverse}})$, we have nRm, and so m is an E-predecessor of n (mEn). Note that we always have $\mathsf{bias} \in S$, and in particular this means $\mathsf{bias} \notin \mathsf{best}_{\prec}(S^{\mathsf{C}})$ (by best-inclusion, since $\mathsf{best}_{\prec}(S^{\mathsf{C}}) \subseteq S^{\mathsf{C}}$). So m cannot be the bias node. Complementing, we see that $m \notin (\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}}$. In other words, some non-bias E-predecessor of n is not in $(\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}}$. By construction of F_S, this means $n \in F_S((\mathsf{best}_{\prec}(S^{\mathsf{C}}))^{\mathsf{C}})$, which contradicts our initial hypothesis. □

Lemma 5.18. Let $\mathcal{M} = \langle W, R, \prec, V \rangle$ be a plausibility model such that $R = (\prec_{\text{reverse}})$. Let $\mathcal{N}$ be given by the NAND construction above. For all $S \subseteq \mathsf{State}_{\mathcal{N}}$:

- $\mathsf{Reach}_{\mathcal{N}}(S) = \{w \mid \text{there is some } u \in S \text{ with } uRw\}$

- $\mathsf{Reach}^{\downarrow}_{\mathcal{N}}(S) = \{w \mid \text{there is some } u \in S \text{ with } wRu\}$

Proof. I will prove it for Reach; the proof for $\mathsf{Reach}^{\downarrow}$ is similar. Let $w \in W \subseteq N$.

$$\begin{array}{lll} w \in \mathsf{Reach}_{\mathcal{N}}(S) & \text{iff} & \text{there exists } u \in S \text{ with an } E\text{-path from } w \text{ to } u \\ & \text{iff} & \text{there exists } u \in S \text{ with an } R\text{-path from } u \text{ to } w \quad \text{(By constr of } \mathcal{N}) \\ & \text{iff} & u \in S \text{ and } uRw \quad \text{(Since } R \text{ is transitive)} \end{array}$$

□

The following lemma says that the constructed net $\mathcal{N}$ is in fact equivalent to $\mathcal{M}$, i.e., the two satisfy exactly the same formulas at exactly the same worlds over the modal language of $\mathcal{L}_{\mathbf{C}}$. Notice that the claim is only made for worlds $w \in W$ and not for the bias node, since we cannot evaluate $\Vdash_{\mathbf{Plaus}}$ at the bias node.

Lemma 5.19. (☞) Let $\mathcal{M} = \langle W, \prec, V \rangle$ be a plausibility model such that $R = (\prec_{\text{reverse}})$, and let $\mathcal{N}$ be given by the NAND construction above. For all modal formulas $\varphi \in \mathcal{L}_{\mathbf{C}}$ and all $w \in W$,

$$\mathcal{N}, w \Vdash_{\mathbf{Net}} \varphi \text{ iff } \mathcal{M}, w \Vdash_{\mathbf{Plaus}} \varphi$$

Proof. By induction on φ. I will show the key inductive cases $\Diamond^{\downarrow}\varphi$, and $\langle \mathbf{C} \rangle \varphi$. The inductive case for $\mathbf{E}\varphi$ is straightforward, and the $\Diamond\varphi$ case is similar to the case for $\Diamond^{\downarrow}\varphi$ (substitute $\mathsf{Reach}^{\downarrow}$ for Reach).

Case $\Box\varphi$.

$$\begin{array}{lll} \mathcal{N}, w \Vdash_{\mathbf{Net}} \Box\varphi & \text{iff} & w \in \mathsf{Reach}^{\downarrow}(\llbracket \varphi \rrbracket_{\mathcal{N}} \cup \{\mathsf{bias}\}) \quad \text{(By neural net semantics)} \\ & \text{iff} & \exists u \in \llbracket \varphi \rrbracket_{\mathcal{N}} \cup \{\mathsf{bias}\} \text{ with } wRu \quad \text{(By Lemma 5.18)} \\ & & \\ & \text{iff} & \text{[TODO]} \\ & \text{iff} & \\ & \text{iff} & \exists u \in W \text{ with } wRu \text{ and } \mathcal{M}, u \Vdash \varphi \\ & \text{iff} & \mathcal{M}, w \Vdash_{\mathbf{Plaus}} \Box\varphi \quad \text{(By plausibility semantics)} \end{array}$$

Case $\Diamond^{\downarrow}\varphi$.

$$\begin{array}{lll} \mathcal{N}, w \Vdash_{\mathbf{Net}} \Diamond^{\downarrow}\varphi & \text{iff} & w \in \mathsf{Reach}(\llbracket \varphi \rrbracket_{\mathcal{N}} \cup \{\mathsf{bias}\}) \quad \text{(By neural net semantics)} \\ & & \exists u \in \llbracket \varphi \rrbracket_{\mathcal{N}} \cup \{\mathsf{bias}\} \text{ with } uRw \quad \text{(By Lemma 5.18)} \\ & & \\ & \text{iff} & \exists u \in W \text{ with } uRw \text{ and } \mathcal{M}, u \Vdash \varphi \\ & \text{iff} & \mathcal{M}, w \Vdash_{\mathbf{Plaus}} \Diamond^{\downarrow}\varphi \quad \text{(By plausibility semantics)} \end{array}$$

Case $\langle \mathbf{C} \rangle \varphi$. In this case, we need to take care to be very explicit about what universe our complements are taken over. So I will write $N \backslash S$ and $W \backslash S$ in place of S^{C}.

$$
\begin{array}{lll}
\mathcal{N}, w \Vdash_{\mathbf{Net}} \langle \mathbf{C} \rangle \varphi & \text{iff} & w \in \mathsf{Clos}(\llbracket \varphi \rrbracket_{\mathcal{N}} \cup \{\mathsf{bias}\}) \qquad \text{(By neural net semantics)} \\
& \text{iff} & w \in N \backslash (\mathsf{best}_{\prec}(N \backslash (\llbracket \varphi \rrbracket_{\mathcal{N}} \cup \{\mathsf{bias}\}))) \qquad \text{(By Lemma 5.17)} \\
& \text{iff} & w \in N \backslash (\mathsf{best}_{\prec}(\\
& & \qquad (N \backslash \llbracket \varphi \rrbracket_{\mathcal{N}}) \cap (N \backslash \{\mathsf{bias}\}))) \qquad \text{(Grouping terms)} \\
& \text{iff} & w \in N \backslash (\mathsf{best}_{\prec}((N \backslash \llbracket \varphi \rrbracket_{\mathcal{N}}) \cap W)) \qquad \text{(Since } W = N \backslash \{\mathsf{bias}\}\text{)} \\
& \text{iff} & w \in N \backslash (\mathsf{best}_{\prec}(W \backslash \llbracket \varphi \rrbracket_{\mathcal{N}})) \\
& \text{iff} & w \in W \backslash (\mathsf{best}_{\prec}(W \backslash \llbracket \varphi \rrbracket_{\mathcal{N}})) \qquad \text{(Since } w \in W \text{ and } w \neq \mathsf{bias}\text{)} \\
& \text{iff} & w \in W \backslash (\mathsf{best}_{\prec}(W \backslash \llbracket \varphi \rrbracket_{\mathcal{M}})) \qquad \text{(By inductive hypothesis)} \\
& \text{iff} & \mathcal{M}, w \Vdash_{\mathbf{Plaus}} \neg \mathbf{C} \neg \varphi \qquad \text{(By plausibility semantics)} \\
& \text{iff} & \mathcal{M}, w \Vdash_{\mathbf{Plaus}} \langle \mathbf{C} \rangle \varphi
\end{array}
$$

□

6 Reflections on Model Building and Interpretability

[Todo]

Chapter 5

Dynamic Update in Neural Network Semantics

1 Introduction

[I now have space to express these points more slowly, in detail.] The neural network semantics presented so far shows us how we can use neural networks as models for modal logic. Neural network inference can be expressed in this logic using $\langle \mathbf{C} \rangle \varphi$, which denotes the forward propagation of the signal $[\![\varphi]\!]$ through the net. However, as discussed in the introduction, the mystery about neural networks is how their inference interacts with their *learning*. In this section, I will show how to extend these semantics to model learning and update in a neural net.

As previously mentioned, I formalize neural network update using the methodology of Dynamic Epistemic Logic. Our static operators $\Diamond$, $\Diamond^{\downarrow}$, and $\langle \mathbf{C} \rangle$ are interpreted by examining the state of the neural net. The DEL trick is to introduce a new "dynamic" operator $[P]$ which *changes* the net in response to some observed formula P. First, we extend the language $\mathcal{L}_{\mathbf{C}}$ to $\mathcal{L}_{\mathsf{Update}}$, which includes these dynamic operators:

$$\varphi, \psi := p \mid \neg\varphi \mid \varphi \wedge \psi \mid \mathbf{E}\varphi \mid \Diamond\varphi \mid \ \Diamond^{\downarrow}\varphi \mid \langle \mathbf{C} \rangle \varphi \mid [P]\varphi$$

Here, $[P]\varphi$ reads "after the agent observes P, φ is true".

Let $\mathsf{Update}\colon \mathbf{Net} \times \mathsf{State} \to \mathbf{Net}$ be any function which takes a neural network, some state S, and "updates" the net somehow in response to S. We can interpret $[P]$ as performing this update by adding the following line to the semantics:

$$\mathcal{N}, n \Vdash [P]\varphi \quad \text{iff} \quad \mathsf{Update}(\mathcal{N}, [\![P]\!]), n \Vdash \varphi$$

In other words, in order to evaluate $[P]\varphi$, we simply evaluate φ in the updated net $\mathsf{Update}(\mathcal{N}, [\![P]\!])$.

From a DEL perspective, this is a standard move to make. But from a machine learning perspective, there are a couple caveats that I should mention. First, $[P]$ does not model learning in the sense of "iterated update until convergence", but rather only models a single step of update.

Second, we should think of $[P]$ as modeling *unsupervised learning*—the model updates in response to an input P, but no "expected answer" y is given alongside P. It is an open problem to formalize supervised learning (in this machine learning sense) in DEL in a non-trivial way.

2 Hebbian Learning: A Simple Neural Network Update Policy

[I now have space to express these points more slowly, in detail.] So far, I've discussed learning and update in very general terms. For my thesis, I will model a simple update policy over neural networks: Hebbian learning. The point in starting with Hebbian learning is to get the details right on a simpler example before lifting these ideas to, say, gradient descent through backpropagation [58].

Hebb's classic learning rule [32] states that when two adjacent neurons are simultaneously and persistently active, the connection between them strengthens ("neurons that fire together wire together"). In contrast with backpropagation, Hebbian learning is errorless and unsupervised. Another key feature 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 I will 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. This is the *unstable* variation of Hebb's rule; repeatedly applying the rule will make the weights arbitrarily large. I will not consider stabilizing variants such as Oja's rule [54].

Single-Step Hebbian Update. First, consider what happens in a single step of Hebbian update. Given a net $\mathcal{N}$ and a state S, we first propagate S forward through $\mathcal{N}$. Any edges that are involved in this propagated activation pattern $\mathsf{Clos}(S)$ simply have their weights strengthened. Formally,

Definition 2.1. Let $\mathsf{Hebb}\colon \mathbf{Net} \times \mathsf{State} \to \mathbf{Net}$ be given by

$$\mathsf{Hebb}(\langle N, \mathsf{bias}, E, W, A, \eta, V\rangle,\ S) = \langle N, \mathsf{bias}, E, W', A, \eta, V\rangle$$

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

Note that Hebb does not affect the edges, activation function, or evaluation of propositions. This means the resulting net is still binary, and closures $\mathsf{Clos}(S)$ still exist and are unique. Therefore Hebb is well-defined. This also means that Hebb does not affect the Reach or $\mathsf{Reach}^{\downarrow}$ operators.

Proposition 2.2. $\mathsf{Reach}_{\mathsf{Hebb}(\mathcal{N},A)}(B) = \mathsf{Reach}_{\mathcal{N}}(B)$

Proof. A single step of Hebbian update $\mathsf{Hebb}(\mathcal{N}, A)$ doesn't change the edge relation E of the graph. So if $n \in N$, any path from $m \in B$ to n in $\mathsf{Hebb}(\mathcal{N}, A)$ is the same path in $\mathcal{N}$. □

And similarly:

Proposition 2.3. $\mathsf{Reach}^{\downarrow}_{\mathsf{Hebb}(\mathcal{N},A)}(B) = \mathsf{Reach}^{\downarrow}_{\mathcal{N}}(B)$

The following is easy to see [I now have space to explain] (since $\eta \geq 0$).

Proposition 2.4. Let $m, n \in N$. We have:

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

Proof. For the first part, observe:

$$\begin{aligned} W_{\mathcal{N}}(m,n) &\leq W_{\mathcal{N}}(m,n) + \eta && \text{(since } \eta \geq 0\text{)} \\ &\leq W_{\mathcal{N}}(m,n) + \eta \cdot \chi_{\mathsf{Clos}(S)}(m) \cdot \chi_{\mathsf{Clos}(S)}(n) && \text{(since for all } S, n,\ \chi_S(n) \geq 0\text{)} \\ &= W_{\mathsf{Hebb}(\mathcal{N},S)}(m,n) \end{aligned}$$

As for the second part, if either $m \notin \mathsf{Clos}(S)$ or $n \notin \mathsf{Clos}(S)$, then by definition of Hebb,

$$\begin{aligned} &W_{\mathsf{Hebb}(\mathcal{N},S)}(m,n) \\ &\quad = W_{\mathcal{N}}(m,n) + \eta \cdot \chi_{\mathsf{Clos}(S)}(m) \cdot \chi_{\mathsf{Clos}(S)}(n) \\ &\quad = W_{\mathcal{N}}(m,n) + \eta \cdot 0 \\ &\quad = W_{\mathcal{N}}(m,n) + \eta \\ &\quad = W_{\mathcal{N}}(m,n) \end{aligned}$$

□

Iterated Hebbian Update. In addition to the single-step Hebb operator, in my thesis work I have also modelled *iterated* Hebbian update Hebb^*. The idea is this: what happens when we propagate a signal S through the net, and then *repeatedly* strengthen the weights of the edges that are involved? Recall that our single-step Hebb is unstable; if we repeat Hebb on a single input state S, the net's

weights within $\mathsf{Clos}(S)$ will be so high that *any* activation pattern that makes contact with $\mathsf{Clos}(S)$ will "rip through" it entirely. Repeating Hebb on S further will not change the $\mathsf{Clos}(S)$-structure, i.e., the update has reached a fixed point. Hebb^* returns the net at this fixed point.

Instead of reasoning abstractly about this fixed point, I formalize it by explicitly defining the number of iterations iter needed to reach it. The idea is to set iter to be so high, all updated weights $W'(m,n)$ overpower any negative weights that would otherwise cancel their effect. The following definitions might look like black magic, but they are set up to capture this intuition (I verified in Lean that this is the right choice for iter, see [35]).

Definition 2.5. Let $\mathcal{N}$ be a net, $n \in N$, and let $m_1, \ldots, m_k$ 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.,

$$\mathsf{nws}(n) = \sum_{m \in \mathrm{preds}(n)} \begin{cases} W(m,n) & \text{if } W(m,n) < 0 \\ 0 & \text{otherwise} \end{cases}$$

Definition 2.6. The *minimum negative weight score* is simply

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

Proposition 2.7. For all $S \in \mathsf{State}$, $m, n \in N$, we have $\mathsf{mnws} \leq W(m,n) \cdot \chi_S(m)$.

Proof. Let m, n be any nodes in N. We have:

$$\begin{aligned}
\mathsf{mnws} &\leq \mathrm{nws}(n) \\
&= \sum_{m \in \mathrm{preds}(n)} \begin{cases} W(m,n) & \text{if } W(m,n) < 0 \\ 0 & \text{otherwise} \end{cases} && \text{(by definition)} \\
&= \sum_{m \in \mathrm{preds}(n)} \begin{cases} W(m,n) \cdot \chi_S(m) & \text{if } W(m,n) < 0 \\ 0 & \text{otherwise} \end{cases} && \text{(since each } W(m,n) < 0 \\
& && \quad \text{and } \chi_S(m) \in \{0,1\}) \\
&\leq W(m,n) \cdot \chi_S(m) && \text{(the sum of negative terms is} \leq \\
& && \quad \text{any particular term)}
\end{aligned}$$

□

Definition 2.8. Recall that the activation function A is nonzero, i.e. there is some $t \in \mathbb{Q}$ such that $A(t) = 1$. We set the number of iterations iter to be exactly

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

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 [which] why it guarantees that the updated weights overpower competing edge weights.

Definition 2.9. Let $\mathsf{Hebb}^*\colon \mathbf{Net} \times \mathsf{State} \to \mathbf{Net}$ be given by

$$\mathsf{Hebb}^*(\langle N, \mathsf{bias}, E, W, A, \eta, V\rangle,\ S) = \langle N, \mathsf{bias}, E, W', A, \eta, V\rangle$$

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

As with Hebb, Hebb* does not affect the edges, activation function, or evaluation of propositions. Therefore Hebb* is well-defined. This also means that Hebb* does not affect the Reach or $\mathsf{Reach}^{\downarrow}$ operators.

Proposition 2.10. $\mathsf{Reach}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B) = \mathsf{Reach}_{\mathcal{N}}(B)$

Proposition 2.11. $\mathsf{Reach}^{\downarrow}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B) = \mathsf{Reach}^{\downarrow}_{\mathcal{N}}(B)$

Similar to Proposition [todo], we have the following:

Proposition 2.12. Let $m, n \in N$. We have:

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

Proof. [] □

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 Hebb* then n must be as well.

Lemma 2.13. (☞) Let $A, B \in \mathsf{State}, m, n \in N$. If $m \in \mathrm{preds}$, $m, n \in \mathsf{Clos}(A)$, and $m \in \mathsf{Clos}(B)$, then

$$A\Big(\sum_{m \in \mathrm{preds}(n)} W_{\mathsf{Hebb}^*(\mathcal{N},A)}(m_i, n) \cdot \chi_{\mathsf{Clos}(B)}(m)\Big) = 1$$

(Take care to notice the different subscripts for W and χ!)

Proof. A is a binary step function, which in particular means it is binary, has a threshold, some $t \in \mathbb{Q}$ with $A(t) = 1$, and is nondecreasing. Since A is nondecreasing, it's enough for us to show

$$t \leq \sum_{m \in \mathrm{preds}(n)} W_{\mathsf{Hebb}^*(\mathcal{N},A)}(m_i, n) \cdot \chi_{\mathsf{Clos}(B)}(m)$$

Well, we have

$$\begin{aligned}
&\sum_{m_i \in \text{preds}(n)} W_{\text{Hebb}^*(\mathcal{N},A)}(m_i, n) \cdot \chi_{\text{Clos}(B)}(m_i) \\
&\quad = \sum_{m_i \in \text{preds}(n), \text{ and } m_i \neq m} W_{\text{Hebb}^*(\mathcal{N},A)}(m_i, n) \cdot \chi_{\text{Clos}(B)}(m_i) \\
&\qquad + W_{\text{Hebb}^*(\mathcal{N},A)}(m, n) \cdot \chi_{\text{Clos}_{\text{Hebb}^*(\mathcal{N},A)}(B)}(m) \\
&\quad \geq (|N| - 1) \cdot \text{mnws} + W_{\text{Hebb}^*(\mathcal{N},A)}(m, n) \cdot \chi_{\text{Clos}(B)}(m) \\
&\qquad \text{(by Proposition [todo], since we are adding } |N| - 1 \text{ terms)} \\
&\quad = (|N| - 1) \cdot \text{mnws} + W_{\text{Hebb}^*(\mathcal{N},A)}(m, n) \cdot 1 \\
&\qquad \text{(since } m \in \text{Clos}(B)\text{)} \\
&\quad = (|N| - 1) \cdot \text{mnws} + W_{\mathcal{N}}(m, n) + \text{iter} \cdot \chi_{\text{Clos}(A)}(m) \cdot \chi_{\text{Clos}(A)}(n) \\
&\qquad \text{(by definition of Hebb}^*\text{)} \\
&\quad = (|N| - 1) \cdot \text{mnws} + W_{\mathcal{N}}(m, n) + \text{iter} \\
&\qquad \text{(since } m, n \in \text{Clos}(A)\text{)} \\
&\quad \geq (|N| - 1) \cdot \text{mnws} + \text{mnws} + \text{iter} \cdot \eta \cdot \chi_{\text{Clos}(A)}(m) \cdot \chi_{\text{Clos}(A)}(n) \\
&\qquad \text{(the sum of negative weights is} \leq \text{any particular weight)} \\
&\quad = |N| \cdot \text{mnws} + \text{iter} \cdot \eta \\
&\qquad \text{(grouping like terms)}
\end{aligned}$$

So at this point we need to show:

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

Rearranging this to solve for iter, it suffices to show:

$$\frac{t - |N| \cdot \text{mnws}}{\eta} \leq \text{iter}$$

But we defined iter to be exactly the integer ceiling of this expression on the left (and 1 if the expression on the left is negative)! □

3 Properties of Hebb and Hebb*

We have the following algebraic properties for Hebb.

Theorem 3.1. (☞) [revise properties from FLAIRS paper…]

One worry we might have is that, in each iteration, we always update by $\text{Clos}(S)$ in the *original* net. But it turns out that this $\text{Clos}(S)$ doesn't change with each iteration, i.e.

Proposition 3.2. (☞) $\mathsf{Clos}_{\mathsf{Hebb}(\mathcal{N},S)}(S) = \mathsf{Clos}_{\mathcal{N}}(S)$

Proof. Let F_S be the state transition function for $\mathcal{N}$ under S, and F'_S be the state transition function for $\mathsf{Hebb}(\mathcal{N},S)$ under S. First, since $\mathsf{Clos}_{\mathsf{Hebb}(\mathcal{N},S)}(S)$ is a fixed point under S in $\mathsf{Hebb}(\mathcal{N},S)$, we have $F'_S(\mathsf{Clos}_{\mathsf{Hebb}(\mathcal{N},S)}(S)) = \mathsf{Clos}_{\mathsf{Hebb}(\mathcal{N},S)}(S)$. But I will show that $\mathsf{Clos}_{\mathcal{N}}(S)$ is *also* a fixed point under S in $\mathsf{Hebb}(\mathcal{N},S)$, i.e. $F'_S(\mathsf{Clos}_{\mathcal{N}}(S)) = \mathsf{Clos}_{\mathcal{N}}(S)$. Since we assumed there is a *unique* fixed point under S in $\mathsf{Hebb}(\mathcal{N},S)$, it will follow that these two states must be the same. In other words, $\mathsf{Clos}_{\mathsf{Hebb}(\mathcal{N},S)}(S) = \mathsf{Clos}_{\mathcal{N}}(S)$.

For the $(\leftarrow)$ direction, suppose $n \in \mathsf{Clos}_{\mathcal{N}}(S)$. Since $\mathsf{Clos}_{\mathcal{N}}(S)$ is a fixed point under S in $\mathcal{N}$, $\mathsf{Clos}_{\mathcal{N}}(S) = F_S(\mathsf{Clos}_{\mathcal{N}}(S))$. By definition of F, either $n \in S$ (in which case we are done), or n is activated by its predecessors m in $\mathsf{Clos}_{\mathcal{N}}(S)$ over $\mathcal{N}$, i.e.

$$A\left(\sum_{m\in\mathrm{preds}(n)} W_{\mathcal{N}}(m,n)\cdot\chi_{\mathsf{Clos}_{\mathcal{N}}(S)}(m)\right)=1$$

By the first part of Proposition [todo], each $W_{\mathcal{N}}(m,n) \leq W_{\mathsf{Hebb}(\mathcal{N},S)}(m,n)$. So the inner sum using the former is $\leq$ the inner sum using the latter. Since A is nondecreasing, we have

$$A\left(\sum_{m\in\mathrm{preds}(n)} W_{\mathsf{Hebb}(\mathcal{N},S)}(m,n)\cdot\chi_{\mathsf{Clos}_{\mathcal{N}}(S)}(m)\right)=1$$

But this implies that $n \in F'_S(\mathsf{Clos}_{\mathcal{N}}(S))$.

As for the $(\rightarrow)$ direction, suppose $n \in F'_S(\mathsf{Clos}_{\mathcal{N}}(S))$. By definition of F', either $n \in S$ (in which case we are done), or n is activated by its predecessors m in $\mathsf{Clos}_{\mathcal{N}}(S)$ over $\mathsf{Hebb}(\mathcal{N},S)$, i.e.

$$A\left(\sum_{m\in\mathrm{preds}(n)} W_{\mathsf{Hebb}(\mathcal{N},S)}(m,n)\cdot\chi_{\mathsf{Clos}_{\mathcal{N}}(S)}(m)\right)=1$$

Suppose for contradiction that $n \notin \mathsf{Clos}_{\mathcal{N}}(S)$. By the second part of Proposition [todo], each $W_{\mathsf{Hebb}(\mathcal{N},S)}(m,n) = W_{\mathcal{N}}(m,n)$, and so we have

$$A\left(\sum_{m\in\mathrm{preds}(n)} W_{\mathcal{N}}(m,n)\cdot\chi_{\mathsf{Clos}_{\mathcal{N}}(S)}(m)\right)=1$$

but this implies that $n \in F_S(\mathsf{Clos}_{\mathcal{N}}(S)) = \mathsf{Clos}_{\mathcal{N}}(S)$, which contradicts $n \notin \mathsf{Clos}_{\mathcal{N}}(S)$. □

and so Hebb^* is equivalent to repeatedly applying Hebb until we reach a fixed point [35]. [Elaborate on this point, it's said a little too quickly for the reader to internalize it! (maybe a picture would help?)]

We have the following algebraic properties for Hebb^*. Before proving these, I'll give some intuition for what these properties say about Hebb^*. [(1) is just used to show (2)] Part (2) expresses a lower bound for $\mathsf{Clos}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$, whereas (3) gives an upper bound within $\mathsf{Clos}(A)$. [There's got to be a better way to explain what these mean… this isn't clear or intuitive at all.]

Proposition 3.3. (☞) $\mathsf{Clos}_{\mathsf{Hebb}^*(\mathcal{N},S)}(S) = \mathsf{Clos}_{\mathcal{N}}(S)$

Proof. The proof is the same as the proof for Hebb (see Proposition [todo]). In place of Propositions [todo] and [todo], we instead apply Propositions [todo] and [todo], respectively. □

Theorem 3.4. (☞) Let $\mathcal{N} \in \mathbf{Net}$, and suppose $\mathcal{N}$ is fully connected. For all $A, B \in \mathsf{State}$,

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

Proof. Let F_B be the state transition function for $\mathcal{N}$ under B, and F_B^* be the state transition function for $\mathsf{Hebb}^*(\mathcal{N}, A)$ under B. For notational convenience, let T be the set inside Clos on the right-hand side, i.e.

$$T = B \cup (\mathsf{Clos}(A) \cap \mathsf{Reach}(\mathsf{Clos}(A) \cap \mathsf{Clos}(B)))$$

This proof follows the major plot beats of the proof for Theorem [TODO]. First, since $\mathsf{Clos}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$ is a fixed point under B in $\mathsf{Hebb}^*(\mathcal{N}, A)$, we have $F_B^*(\mathsf{Clos}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)) = \mathsf{Clos}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)$. But I will show that $\mathsf{Clos}(T)$ is *also* a fixed point under B in $\mathsf{Hebb}(\mathcal{N}, A)$, i.e. $F_B^*(\mathsf{Clos}(T)) = \mathsf{Clos}(T)$. Since we postulated that there is a *unique* fixed point under B in $\mathsf{Hebb}^*(\mathcal{N}, A)$, it will follow that these two states must be the same: $\mathsf{Clos}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B) = \mathsf{Clos}(T)$.

Let's show that $F_B^*(\mathsf{Clos}(T)) = \mathsf{Clos}(T)$. For the ($\leftarrow$) direction, suppose $n \in \mathsf{Clos}(T)$. Since $\mathsf{Clos}(T)$ is a fixed point under T in $\mathcal{N}$, $\mathsf{Clos}(T) = F_T(\mathsf{Clos}(T))$. By definition of F, we have two cases:

Case 1. $n \in T$, i.e. $n \in B \cup (\mathsf{Clos}(A) \cap \mathsf{Reach}(\mathsf{Clos}(A) \cap \mathsf{Clos}(B)))$. If $n \in B$, then we're done by the definition of F_B^* (the next state includes all nodes in B). Otherwise, we have $n \in \mathsf{Clos}(A)$ and a path from some $m \in \mathsf{Clos}(A) \cap \mathsf{Clos}(B)$ to n in $\mathcal{N}$. Since $\mathcal{N}$ is fully connected, m is in fact a predecessor of n. Moreover, $m \in \mathsf{Clos}(T)$, since $m \in \mathsf{Clos}(A) \cap \mathsf{Clos}(B)$ and by inclusion of Reach and Clos. So we have $m \in \mathsf{preds}(n)$, $m, n \in \mathsf{Clos}(A)$, and $m \in \mathsf{Clos}(T)$. But

these are exactly the conditions of Lemma [todo]! This means we have

$$A\left(\sum_{m\in\text{preds}(n)} W_{\mathsf{Hebb}^*(\mathcal{N},A)}(m,n)\cdot\chi_{\mathsf{Clos}(T)}(m)\right)=1$$

which implies that $n\in F_B^*(\mathsf{Clos}(T))$.

Case 2. n is activated by its predecessors m in $\mathsf{Clos}(T)$ over $\mathcal{N}$, i.e.

$$A\left(\sum_{m\in\text{preds}(n)} W_{\mathcal{N}}(m,n)\cdot\chi_{\mathsf{Clos}(T)}(m)\right)=1$$

By the first part of Proposition [todo], each $W_{\mathcal{N}}(m,n)\leq W_{\mathsf{Hebb}^*(\mathcal{N},S)}(m,n)$. So the inner sum using the former is $\leq$ the inner sum using the latter. Since A is nondecreasing, we have

$$A\left(\sum_{m\in\text{preds}(n)} W_{\mathsf{Hebb}^*(\mathcal{N},A)}(m,n)\cdot\chi_{\mathsf{Clos}(T)}(m)\right)=1$$

But this immediately implies that $n\in F_B^*(\mathsf{Clos}(T))$.

As for the $(\rightarrow)$ direction, suppose $n\in F_B^*(\mathsf{Clos}(T))$. By definition of F^*, we have two cases:

Case 1. $n\in B$. So $n\in B\cup(\mathsf{Clos}(A)\cap\mathsf{Reach}(\mathsf{Clos}(A)\cap\mathsf{Clos}(B)))=T$. By inclusion of Clos, we have $n\in\mathsf{Clos}(T)$.

Case 2. n is activated by its predecessors m in $\mathsf{Clos}(T)$ over $\mathsf{Hebb}^*(\mathcal{N},A)$, i.e.

$$A\left(\sum_{m\in\text{preds}(n)} W_{\mathsf{Hebb}^*(\mathcal{N},A)}(m,n)\cdot\chi_{\mathsf{Clos}(T)}(m)\right)=1$$

Suppose for contradiction that $n\notin\mathsf{Clos}(T)$. By the second part of Proposition [todo], each $W_{\mathsf{Hebb}^*(\mathcal{N},A)}(m,n)=W_{\mathcal{N}}(m,n)$, and so we have

$$A\left(\sum_{m\in\text{preds}(n)} W_{\mathcal{N}}(m,n)\cdot\chi_{\mathsf{Clos}(T)}(m)\right)=1$$

but this implies that $n\in F_T(\mathsf{Clos}(T))=\mathsf{Clos}(T)$, which contradicts $n\notin\mathsf{Clos}(T)$. So we must have $n\in\mathsf{Clos}(T)$. □

Corollary 3.5. If $\mathsf{Clos}(A)\cap\mathsf{Clos}(B)=\emptyset$, then $\mathsf{Clos}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B)=\mathsf{Clos}(B)$.

Proof. Suppose $\mathsf{Clos}(A)\cap\mathsf{Clos}(B)=\emptyset$. We have

$$\begin{aligned}
\mathsf{Clos}_{\mathsf{Hebb}^*(\mathcal{N},A)}(B) &= \mathsf{Clos}(B\cup(\mathsf{Clos}(A)\cap\mathsf{Reach}(\mathsf{Clos}(A)\cap\mathsf{Clos}(B))))\\
&= \mathsf{Clos}(B\cup(\mathsf{Clos}(A)\cap\mathsf{Reach}(\emptyset)))\\
&= \mathsf{Clos}(B\cup\emptyset)\\
&= \mathsf{Clos}(B)
\end{aligned}$$

□

4 Neural Network Semantics for Hebbian Update

[TODO] Give official languages $\mathcal{L}_{\mathsf{Hebb}}$ and $\mathcal{L}_{\mathsf{Hebb}^\star}$ for the logics of Hebb and Hebb*, i.e. using operators $[P]_{\mathsf{Hebb}}\varphi$ and $[P]_{\mathsf{Hebb}^\star}\varphi$ and their semantics!!! And also just say that the definitions $\mathcal{N} \models_{\mathsf{Hebb}} \varphi$ (same for Hebb*), $\Gamma \models_{\mathsf{Hebb}} \varphi$ (same for Hebb*) are what you'd expect.

5 A Complete Logic of Iterated Hebbian Update

- In the previous section, I gave sound axioms for Hebb*. It turns out those axioms are nearly complete! I'll show this here, and give the complete set of "reduction axioms"
- I already proved soundness, but I have to do it again for these new axioms!

Definition 5.1. [Reduction axioms for Hebb*]

Theorem 5.2. (👉) These reduction axioms are sound; for all $\Gamma \subseteq \mathcal{L}^*$ and $\varphi \in \mathcal{L}^*$, if $\Gamma \vdash \varphi$ then $\Gamma \models \varphi$.

Proof. [todo] □

Proof. [todo] □

5.1 Example: Verifying a Neural Networks Behavior After Learning

do example using single-step Hebbian learning, since iterated is a bit more abstract…

Definition 5.3. [Term rewriting translation system]

Proposition 5.4. (👉) Each $\mathrm{tr}(\varphi)$ is update-operator free [todo, fix this statement!]

Proof. [todo] □

Proposition 5.5. (👉) Each $\mathrm{tr}(\varphi)$ actually terminates [todo, fix this statement!]

Proof. [todo] □

Proposition 5.6. (☞) $\vdash \varphi \leftrightarrow \mathrm{tr}(\varphi)$ [todo, fix this statement, which $\vdash$ is this?]

Proof. [todo] □

Theorem 5.7. (☞ Model Building for $\mathcal{L}^*$) For all consistent $\Gamma \subseteq \mathcal{L}^*$, there is finite $\mathcal{N}$ such that $\mathcal{N} \models \Gamma$.

Proof. [todo] □

Theorem 5.8. (☞ Completeness for $\mathcal{L}^*$) For all consistent $\Gamma \subseteq \mathcal{L}^*$ and all formulas $\varphi \in \mathcal{L}^*$,

$$\text{if } \Gamma \models \varphi \text{ then } \Gamma \vdash \varphi$$

Theorem 5.9. (☞) For all $\Gamma^* \subseteq \mathcal{L}^*$, there is $\mathcal{N}$ such that $\mathcal{N} \models \Gamma^*$.

Proof. Let $\Gamma^* \subseteq \mathcal{L}^*$. As outlined in the paper, our plan is to define rewrite rules based on our reduction axioms that "translate away" all of the dynamic formulas $\langle \varphi \rangle_{\mathsf{Hebb}} \psi$ in Γ^*, resulting in $\Gamma^{\mathrm{tr}} \subseteq \mathcal{L}$. By our assumption, we have a net $\mathcal{N} \models \Gamma^{\mathrm{tr}}$, and we show that this very same net $\mathcal{N} \models \Gamma^*$.

It's easy to see intuitively how this translation should go. For example, given the formula

$$\langle p \rangle_{\mathsf{Hebb}} (\langle p \rangle_{\mathsf{Hebb}} \langle \mathbf{B} \rangle \wedge \Diamond) \in \Gamma^*$$

we would recursively apply our reduction axioms, pushing $\langle p \rangle_{\mathsf{Hebb}}$ further into the expression until we can eliminate the propositional cases $\langle p \rangle_{\mathsf{Hebb}} q$.

We define the term-rewriting system that does the translation $\tau(\varphi)$ for all φ as follows.

- $\tau(p) = p$
- $\tau(\neg \varphi) = \neg \tau(\varphi)$
- $\tau(\varphi \wedge \psi) = \tau(\varphi) \wedge \tau(\psi)$
- $\tau(\mathbf{K} \varphi) = \mathrm{box}(\tau(\varphi))$
- $tr(\langle \varphi \rangle_{\mathsf{Hebb}} p) = tr(p)$
- $tr(\langle \varphi \rangle_{\mathsf{Hebb}} \neg \psi) = tr(\neg \langle \varphi \rangle_{\mathsf{Hebb}} \psi)$
- $tr(\langle \varphi \rangle_{\mathsf{Hebb}} (\psi \wedge \rho)) = tr(\langle \varphi \rangle_{\mathsf{Hebb}} \psi \wedge \langle \varphi \rangle_{\mathsf{Hebb}} \rho)$
- $tr(\langle \varphi \rangle_{\mathsf{Hebb}} \Diamond) = tr(\Diamond)$

- $tr(\langle\varphi\rangle_{\text{Hebb}}\langle\mathbf{B}\rangle) = tr(\langle\mathbf{B}\rangle)$
- $tr(\langle\varphi\rangle_{\text{Hebb}}\langle\psi\rangle_{\text{Hebb}}\rho) = tr(\langle\varphi\rangle_{\text{Hebb}}(tr(\langle\psi\rangle_{\text{Hebb}}\rho)))$

Formally, the term-rewriting system takes a formula φ and recursively applies these equational rules to φ (from left-to-right). We just need to check that

1. For all ψ, $tr(\psi)$ is update-operator-free
2. This term rewriting actually terminates

The work involved in showing termination is long and tedious. The usual approach is to define a measure on formulas $c(\varphi)$ that *decreases* with each application of our reduction axioms (from left-to-right). In particular, we need c to satisfy

- If ψ is a subexpression of φ, $c(\varphi) > c(\psi)$
- $c(\langle\varphi\rangle_{\text{Hebb}}p) > c(p)$
- $c(\langle\varphi\rangle_{\text{Hebb}}\neg\psi) > c(\neg\langle\varphi\rangle_{\text{Hebb}}\psi)$
- $c(\langle\varphi\rangle_{\text{Hebb}}(\psi \wedge \rho)) > c(\langle\varphi\rangle_{\text{Hebb}}\psi \wedge \langle\varphi\rangle_{\text{Hebb}}\rho)$
- $c(\langle\varphi\rangle_{\text{Hebb}}\Diamond) > c(\Diamond)$
- $c(\langle\varphi\rangle_{\text{Hebb}}\langle\mathbf{B}\rangle) > c(\langle\mathbf{B}\rangle)$
- $c(\langle\varphi\rangle_{\text{Hebb}}\langle\psi\rangle_{\text{Hebb}}\rho) > c(\langle\varphi\rangle_{\text{Hebb}}(tr(\langle\psi\rangle_{\text{Hebb}}\rho)))$

But coming up with a measure c that works is tricky, and is dependent on the specific reduction axioms. For the gritty details involved in coming up with this measure, as well as proving termination for the term rewriting system, see [12].

From here, we assume we have this measure c. We now have two things left to show:

Proposition. (☞) For all $\varphi \in \Gamma^*$, we have $\vdash \varphi \leftrightarrow \tau(\varphi)$

Proof. By induction on $c(\varphi)$.

Base Step. If φ is a proposition p, then we (trivially) have $\vdash p \leftrightarrow p$.

Inductive Step. We consider each possible inductive case, and suppose the claim holds for formulas ψ with smaller $c(\psi)$. The $\neg\varphi$, $\varphi \wedge \psi$, **K**, and **B** cases all follow from applying the translation, and then applying inductive hypothesis on the subexpression that results from this.

Here are the rest of the cases. Notice that we apply the inductive hypothesis to terms whose c-cost is smaller (this is why we needed the decreasing properties of c before).

$\langle\varphi\rangle_{\mathsf{Hebb}}p$ case. We have

$$tr(\langle\varphi\rangle_{\mathsf{Hebb}}p) = tr(p) = p$$

and so we need to show that

$$\vdash \langle\varphi\rangle_{\mathsf{Hebb}}p \leftrightarrow p$$

but this holds by our propositional reduction axiom.

$\langle\varphi\rangle_{\mathsf{Hebb}}\neg\psi$ case. We have:

$$\begin{array}{lll}
\vdash \langle\varphi\rangle_{\mathsf{Hebb}}\neg\psi & & \\
\quad \leftrightarrow \neg\langle\varphi\rangle_{\mathsf{Hebb}}\psi & & \text{(by the reduction axiom)} \\
\quad \leftrightarrow tr(\neg\langle\varphi\rangle_{\mathsf{Hebb}}\psi) & & \text{(inductive hypothesis)} \\
\quad = tr(\langle\varphi\rangle_{\mathsf{Hebb}}\neg\psi) & & \text{(by our translation)}
\end{array}$$

$\langle\varphi\rangle_{\mathsf{Hebb}}\psi \wedge \rho$ case.. We have:

$$\begin{array}{lll}
\vdash \langle\varphi\rangle_{\mathsf{Hebb}}(\psi \wedge \rho) & & \\
\quad \leftrightarrow \langle\varphi\rangle_{\mathsf{Hebb}}\psi \wedge \langle\varphi\rangle_{\mathsf{Hebb}}\rho & & \text{(by the reduction axiom)} \\
\quad \leftrightarrow tr(\langle\varphi\rangle_{\mathsf{Hebb}}\psi \wedge \langle\varphi\rangle_{\mathsf{Hebb}}\rho) & & \text{(inductive hypothesis)} \\
\quad = tr(\langle\varphi\rangle_{\mathsf{Hebb}}(\psi \wedge \rho)) & & \text{(by our translation)}
\end{array}$$

$\langle\varphi\rangle_{\mathsf{Hebb}}\mathbf{K}$ case.. We have:

$$\begin{array}{lll}
\vdash \langle\varphi\rangle_{\mathsf{Hebb}}\mathbf{K} & & \\
\quad \leftrightarrow \mathbf{K} & & \text{(by the reduction axiom)} \\
\quad \leftrightarrow tr(\mathbf{K}) & & \text{(inductive hypothesis)} \\
\quad = tr(\langle\varphi\rangle_{\mathsf{Hebb}}\mathbf{K}) & & \text{(by our translation)}
\end{array}$$

$\langle\varphi\rangle_{\mathsf{Hebb}}\mathbf{B}$ case.. We have:

$$\begin{array}{l}
\vdash \langle\varphi\rangle_{\mathsf{Hebb}}\mathbf{B} \\
\quad \leftrightarrow \langle\mathbf{B}\rangle \\
\qquad \text{(by the reduction axiom)} \\
\quad \leftrightarrow tr(\langle\mathbf{B}\rangle) \\
\qquad \text{(inductive hypothesis)} \\
\quad = tr(\langle\varphi\rangle_{\mathsf{Hebb}}\mathbf{B}) \\
\qquad \text{(by our translation)}
\end{array}$$

$\langle\varphi\rangle_{\mathsf{Hebb}}\langle\psi\rangle_{\mathsf{Hebb}}\rho$ case.. This case is more interesting. First, notice our translation for this case:

$$tr(\langle\varphi\rangle_{\mathsf{Hebb}}\langle\psi\rangle_{\mathsf{Hebb}}\rho) = tr(\langle\varphi\rangle_{\mathsf{Hebb}}tr(\langle\psi\rangle_{\mathsf{Hebb}}\rho))$$

That is, we translate the inner expression first, then translate the outer expression. This inner $tr(\langle\psi\rangle_{\mathsf{Hebb}}\rho)$ is equivalent to some update-operator-free formula χ:

$$\vdash \chi \leftrightarrow tr(\langle\psi\rangle_{\mathsf{Hebb}}\rho) \leftrightarrow \langle\psi\rangle_{\mathsf{Hebb}}\rho \tag{5.1}$$

(This last equivalence follows from our inductive hypothesis, which we can apply because $\langle\psi\rangle_{\mathsf{Hebb}}\rho$ is a subexpression of $\langle\varphi\rangle_{\mathsf{Hebb}}\langle\psi\rangle_{\mathsf{Hebb}}\rho$.)

What about $tr(\langle\varphi\rangle_{\mathsf{Hebb}}\chi)$? Well, since χ is update-operator-free, this reduces to our previous inductive cases. So we have

$$\vdash tr(\langle\varphi\rangle_{\mathsf{Hebb}}\chi) \leftrightarrow \langle\varphi\rangle_{\mathsf{Hebb}}\chi \tag{5.2}$$

Putting this all together, we have:

$$\begin{array}{lll}
\vdash \langle\varphi\rangle_{\mathsf{Hebb}}\langle\psi\rangle_{\mathsf{Hebb}}\rho & & \\
\quad \leftrightarrow \langle\varphi\rangle_{\mathsf{Hebb}}\chi & \text{(by (5.1))} & \\
\quad \leftrightarrow tr(\langle\varphi\rangle_{\mathsf{Hebb}}\chi) & \text{(by (5.2))} & \square\ \square \\
\quad \leftrightarrow tr(\langle\varphi\rangle_{\mathsf{Hebb}}(tr(\langle\psi\rangle_{\mathsf{Hebb}}\rho))) & \text{(by (5.1))} & \\
\quad \leftrightarrow tr(\langle\varphi\rangle_{\mathsf{Hebb}}\langle\psi\rangle_{\mathsf{Hebb}}\rho) & \text{(by our translation)} &
\end{array}$$

Theorem 5.10. (Completeness) The logic of Hebbian Learning is completely axiomatized by the base axioms [ref!], along with the above reduction axioms. That is, for all consistent $\Gamma^* \subseteq \mathcal{L}^*$, if $\Gamma^* \models \varphi$ then $\Gamma^* \vdash \varphi$.

Proof. Since our language $\mathcal{L}^*$ has negation, completeness follows from model building in the usual way; this proof is entirely standard. Suppose contrapositively that $\Gamma^* \not\models \varphi$. It follows that $\Gamma^* \vdash \neg\varphi$. So $\Gamma^* \cup \{\neg\varphi\}$ is consistent, and by Theorem [todo—need to modify the construction to make sure the net is fully connected. Quote: "But remember that our nets are also fully connected! So we need to modify the model construction from [38] 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."], we have $\mathcal{N} \in \mathsf{Net}$ such that $\mathcal{N} \models \Gamma^* \cup \{\neg\varphi\}$. But then $\mathcal{N} \models \Gamma^*$ yet $\mathcal{N} \not\models \varphi$, which is what we wanted to show. $\square$

5.2 Example: Building a Neural Network with Learning Constraints

6 Reflections on Neural Network Alignment

[Prove soundness of axioms for both single-step and iterated Hebbian update!]

Chapter 6

Neural Network Semantics for First-Order Logic

1 Introduction

2 Lifting a Modal Logic to First-Order Logic

- A relatively new technique in modal logic allows us to "lift" the modal semantics to first-order logic. In this section, I describe this technique, and explain how it is used to give semantics for classical first-order logic.
- Explain where this technique comes from: A way to explain how first-order logic can be derived as a special kind of modal logic. This is widely known within certain logic circles, but is not widely known among those who use logics in AI. I am using it in a totally non-controversial way: I follow it as instructions for how to get a predicate logic from a modal one.
- Basic idea: Possible worlds are variable assignments
- This is done in two stages: Stage 1 is to treat each quantifier per variable $\forall x$ like a dynamic modal operator $[x]$. Stage 2 is to consider substitution $[y/x]$ like a 2-variable dynamic modal operator as well.
- This is first-order logic *without* equality. We can optionally introduce equality as a predicate $=(x, y)$ [I don't know the consequences of that choice…].
- In Section 3, I will follow Stage 1 and interpret $\exists x$ as the closure of a signal in a neural network (following closely to the modal logic interpretation). In Section 4, I will consider *classical* substitution and explore how it interacts with the neural network closure variant

FOL Axiom:	Frame Property:	FOL Axiom:	Frame Property:

Figure 2.1.

of $\exists x$. (I need to define free variables and "y is free for x in φ")

- Here I can make a table of first-order logic axioms and the frame properties they correspond to. Figure 2.1.

3 Neural Network Semantics for First-Order Logic

3.1 Variable-Assignment Networks

- I should first say that instead of an underlying set of propositions, there is an underlying set of predicates $p(x_1, \ldots, x_k)$

Definition 3.1. A variable-assignment network is $\mathcal{N} = \langle N, \{E_x\}_{x \in \mathrm{VAR}}, \{W_x\}_{x \in \mathrm{VAR}}, \{A_x\}_{x \in \mathrm{VAR}}, V \rangle$, where

- N [type it correctly] is a finite nonempty set of *variable assignments*, which we treat as nodes of the network. I will use lowercase greek symbols $\alpha, \beta, \ldots$ to denote these special variable assignment neurons, in order to distinguish them from ordinary neurons $n, m, \ldots$
- The edges, weights, and activation function are defined exactly as before, but now I define one for each variable: E_x, W_x, A_x for $x \in \mathrm{VAR}$.
- V: [] $\to \mathcal{P}(N)$ is a *valuation* that maps each atomic predicate $p(x_1, \ldots, x_k)$ to the set of variable assignments that make p true (equivalently, the set of variable assignments that are active in state p).

[define State$_x$ for variable-assignment models ($\mathcal{N}$ subscript will be hidden)]

Think of a variable-assignment network like an ensemble of neural networks $\mathcal{N}_x = \langle N, E_x, W_x, A_x, V \rangle$ for each variable x. Each of these nets $\mathcal{N}_x$ in the ensemble specifies a particular accessibility

structure for assignments $\alpha \in N$ under variable x. Unlike classical assignment models however, this accessibility structure is determined by a neural network transition function F_{x,S_0} from state $S \in \mathsf{State}_x$ to the next state (for a particular variable x). This function F_{x,S_0} is defined exactly as before; the only difference is that there is now one per $x \in \mathrm{VAR}$.

$$F_{x,S_0}(S) = S_0 \cup \left\{ \eta \mid A_x\left(\sum_{m \in \mathrm{preds}_x(n)} W_x(m,n) \cdot \chi_S(m) \right) = 1 \right\}$$

where $\chi_S(m) = 1$ iff $m \in S$ is the indicator function.

The closure for each $x \in \mathrm{VAR}$ is defined similarly. I postulate that for all states S_0, F_{x,S_0} applied repeatedly to S_0 has a unique fixed point under S_0 (it is the only state S such that $F_{x,S_0}(S) = S$). Let $\mathsf{Clos}_x \colon \mathsf{State}_x \to \mathsf{State}_x$ be the function that produces that least fixed point, for network $\mathcal{N}_x$. [define class of VANs]

3.2 Semantics for Quantifiers and Predicates

Definition 3.2. Formulas in the [extended first-order language] are given by

$$\varphi, \psi := p(x_1, \ldots, x_k) \mid \neg\varphi \mid \varphi \wedge \psi \mid U\varphi \mid \forall x \varphi \mid \blacktriangledown x \varphi$$

$\top, \bot, \vee, \to, \leftrightarrow$ and the dual quantifiers $\exists x, \blacktriangle x$ are defined in the usual way.

$\forall x \varphi$ is intended to be the minimal universal quantifier, which says "φ is true in all variable assignments that are accessible from the current one." The special operator $\blacktriangledown x \varphi$ is intended to be the universal quantifier we get by interpreting $\blacktriangle x \varphi$ as the closure of signal φ in the neural network $\mathcal{N}_x$.

Definition 3.3. For all $\mathcal{N} \in \mathbf{Net}$, and variable assignments $\alpha \in N$:

$\mathcal{N}, \alpha \models p(x_1, \ldots, x_k)$	iff	$\alpha \in V(p(x_1, \ldots, x_k))$
$\mathcal{N}, \alpha \models \neg\varphi$	iff	$\mathcal{N}, \alpha \not\models \varphi$
$\mathcal{N}, \alpha \models \varphi \wedge \psi$	iff	$\mathcal{N}, \alpha \models \varphi$ and $\mathcal{N}, \alpha \models \psi$
$\mathcal{N}, \alpha \models U\varphi$	iff	[todo]
$\mathcal{N}, \alpha \models \exists x \varphi$	iff	there is an E_x-path from $[\![\varphi]\!]$ to α
$\mathcal{N}, \alpha \models \blacktriangle x \varphi$	iff	$\alpha \in \mathsf{Clos}_x([\![\varphi]\!])$

where $[\![\varphi]\!] = \{\alpha \in N \mid \mathcal{N}, \alpha \models \varphi\}$.

Proposition 3.4. By the definition of the duals $\exists x, \exists x$, we can instead define the semantics for $\forall x$, $\forall x$ as follows. For all $\mathcal{N} \in [\,]$, $\alpha \in N$:

$$\begin{array}{lll} \mathcal{N}, \alpha \models \forall x\varphi & \text{iff} & \text{there is } \textit{no } E_x\text{-path from } [\![\varphi]\!]^C \text{ to } \alpha \\ \mathcal{N}, \alpha \models \forall x\varphi & \text{iff} & \alpha \in (\mathsf{Clos}([\![\varphi]\!]^C))^C \end{array}$$

3.3 Semantics for Variable Substitution

The story up to here is incomplete; variable substitution is at the heart of FOL, and I have not yet said anything about substitution in variable-assignment networks. Following [cite van Benthem] closely, I consider a substitution relation $\rightsquigarrow_{x,y}$. The intended meaning of $\alpha \rightsquigarrow_{x,y} \beta$ is "β is the result of assigning $x := y$ in α." Formally,

$$\alpha \rightsquigarrow_{x,y} \beta \text{ iff } \beta(x) = \alpha(y) \text{ and } \alpha(z) = \beta(z) \text{ for all } z \neq x$$

For convenience, I assume that $\rightsquigarrow_{x,y}$ is *functional*, i.e. for each $\alpha \in N$ there is exactly one $\beta \in N$ with $\alpha \rightsquigarrow_{x,y} \beta$.

Van Benthem also considers a generalized notion of substitution, where $\rightsquigarrow_{x,y}$ is just an abstract relation on assignments. For my purposes I will opt for the concrete definition, since substitution is a syntactic notion and we don't gain much by interpreting it in a neural network structure.

From here, I will extend the language [] to [], which has substitution operators $[y/x]$ for x, $y \in \mathrm{VAR}$. Formulas are given by

$$\varphi, \psi := p(x_1, \ldots, x_k) \mid \neg\varphi \mid \varphi \wedge \psi \mid U\varphi \mid \forall x\varphi \mid \forall x\varphi \mid [y/x]\varphi$$

The semantics for $[y/x]$ is given by

$$\mathcal{N}, \alpha \models [y/x]\varphi \text{ iff for that unique } \beta \text{ such that } \alpha \rightsquigarrow_{x,y} \beta, \text{ we have } \mathcal{N}, \beta \models \varphi$$

4 Axioms, Soundness, and Frame Conditions

4.1 Axioms for Quantifiers and Predicates

- In this section, I will explore which first-order logic axioms are sound for $\forall x$ interpreted in a variable-assignment neural network.
- Sound ones: **(Dual)**, **(Rep)**, **(CM)**, **(Refl)**, and **(Trans)** get inherited by the underlying modal logic, and are sound for the same reasons. (e.g., **(CM)** holds because each Clos_x is cumulative.)
- Are there any sound variable interactions (for two variables x, y)? All the ones listed appear to not necessarily be sound, but is there something we can say that's true?
- The unsound ones. For each of them, I should (1) prove that it isn't sound with a counterexample, and (2) think about *what property of* Clos_x *would make it true?* (Think of each of these axioms as somewhat negotiable; it's perfectly fine for neural networks to model dependent quantifiers $\forall x$ rather than the classical *independent* quantifiers.)

  (Nec).

  (Distr).

  (Eucl).

  (???).

  (Exch). [Give the neural network that is a countermodel here] [Also, talk about what this means—neural networks are capable of modeling *dependent* quantifiers!]

  Proposition 4.1. The exchange axiom **(Exch)** is sound if and only if for all nets $\mathcal{N} \in [\,]$, $\varphi \in [\,]$ and $x, y \in \mathrm{VAR}$, $\mathsf{Clos}_x(\mathsf{Clos}_y([\![\varphi]\!])) = \mathsf{Clos}_y(\mathsf{Clos}_x([\![\psi]\!]))$.

  Proof. [todo] □

  (PR).

  (Confl).

		Semantics for First-Order Logic		
		Neural	**Generalized**	**Classical**
(Dual)	$\exists x\varphi \leftrightarrow \neg\forall x\neg\varphi$	✓	✓	✓
(Nec)	If $\vdash\varphi$ then $\vdash\forall x\varphi$	✓	✓	✓
(CM)	$U(\forall x\varphi \to \psi) \to (\forall x(\varphi \wedge \psi) \to \forall x\varphi)$	✓	✓	✓
(Distr)	$\forall x(\varphi \to \psi) \to (\forall x\varphi \to \forall x\psi)$	✗	✓	✓
(Refl)	$\forall x\varphi \to \varphi$	✓	✗	○
(Trans)	$\forall x\varphi \leftrightarrow \forall x\forall x\varphi$	✓	✗	✓
(Eucl)	$\exists x\varphi \to \forall x\exists x\varphi$	✗	✗	✓
(???)	$\varphi \to \forall x\varphi$ for x not free in φ	✗	✗	✓
(Exch)	$\forall x\forall y\varphi \leftrightarrow \forall y\forall x\varphi$	✗	✗	✓
(Confl)	$\exists x\forall y\varphi \to \forall y\exists x\varphi$	✗	✗	✓

Figure 4.1. Soundness for various FOL axioms, under three different semantics: neural semantics using variable-assignment networks [], the generalized semantics [], and classical semantics for FOL []. For the generalized and classical semantics, $\forall x$ and $\exists x$ should be interpreted as the usual $\forall x$ and $\exists x$. ✓ indicates that the axiom is sound, i.e., holds for all models in that class without introducing any new frame properties. ✗ indicates that the axiom does not hold for all models in that class, and may require additional frame properties to make it hold.

Abbreviations: **(CM)** is Cautious Monotonicity (from conditional logic); **(Eucl)** is Euclidean; **(Exch)** is the standard FOL exchange principle; **(Confl)** is Confluence.

4.2 Axioms for Variable Substitution

Let's now consider soundness for this extended language with $[y/x]$. Since the operator $[y/x]$ is defined classically, it satisfies all the usual FOL substitution axioms over predicates, $\wedge$, and $\neg$:

- $[y/x]p(x) \leftrightarrow p(y)$, and $[y/x]p(z) \leftrightarrow p(z)$ for $z \neq x$.
- $[y/x](\varphi \wedge \psi) \leftrightarrow ([y/x]\varphi \wedge [y/x]\psi)$
- $[y/x]\neg\varphi \leftrightarrow \neg[y/x]\varphi$ (since substitution is *functional*)

[It satisfies the usual interaction axioms with the classical/minimal quantifier $\forall x$ so long as I add in the right frame conditions, which I should do above.]

I will now explore the different possible interactions between classical substitution $[y/x]$ and

neural quantifiers $\forall x$, i.e., interactions between the substitution relation $\rightsquigarrow_{x,y}$ and Clos_x. In contrast to the axioms with quantifiers alone, I consider the $[y/x]$ interactions to be relatively non-negotiable; in order to interpret a neural network as a FOL reasoner, it's important for the net's activation function Clos_x to get along with substitution (the core mechanic of FOL). Here is a list of the standard interactions for classical FOL (these versions with the $[y/x]$ operator come from [cite van Benthem], who drew the standard axioms from [cite Enderton]):

1. $[y/x]\forall x\varphi \leftrightarrow \forall x\varphi$
2. $[y/x]\forall z\varphi \leftrightarrow \forall z[y/x]\varphi$ for $z \neq x$
3. $\forall x\varphi \rightarrow [y/x]\varphi$, if y free for x in φ

[The second axiom includes the case $[y/x]\forall y\varphi \leftrightarrow \forall y[y/x]\varphi$. I'm not listing existential interactions such as $[y/x]\exists x\varphi \leftrightarrow \exists x\varphi$, $[y/x]\exists y\varphi \leftrightarrow \exists y[y/x]\varphi$, and $[y/x]\exists z\varphi \leftrightarrow \exists z[y/x]\varphi$ for $z \neq x, y$, since for functional $[y/x]$ they are each equivalent to their duals.]

[I'm also only interested in the ($\rightarrow$) direction of (1) and (2) above,]

Note that these are *all* sound in classical FOL, and in generalized FOL they are each only sound alongside frame conditions. For variable-assignment network semantics, it's not necessary for Clos_x to interact with $\rightsquigarrow_{x,y}$ at all—by default, none of these interaction axioms are sound. But we can now ask: what neural network frame conditions are sufficient to make these axioms hold? In other words, what class of variable-assignment nets $\mathcal{N}$ interact in the expected way with substitution?

Consider the following three properties of variable-assignment nets $\mathcal{N}$. The first, that $\mathcal{N}$ respects substitutions, says that if β is the result of assigning $x := y$ in α, and α is activated by signal $[\![\varphi]\!]$, then β is activated as well.

Definition 4.2. A variable-assignment net $\mathcal{N} \in [\,]$ *respects substitutions* iff for all $\alpha, \beta \in N$, x, $y \in \mathrm{VAR}$, $\varphi \in [\,]$, if $\alpha \rightsquigarrow_{x,y} \beta$ and $\alpha \in \mathsf{Clos}_x([\![\varphi]\!])$, then $\beta \in \mathsf{Clos}_x([\![\varphi]\!])$.

Example 4.3. [Give an example of a neural network that respects substitutions (but does not necessarily reflect them)]

The second property, $\mathcal{N}$ reflects substitutions, says that if β is the result of assigning $x := y$ in α, and substituting y for x in φ does not result in any binding issues, if β is activated by signal $[\![\varphi]\!]$,

then α is activated as well.

Definition 4.4. A variable-assignment net $\mathcal{N} \in [\,]$ *reflects substitutions* iff for all $\alpha, \beta \in N$, $x, y \in$ VAR, $\varphi \in [\,]$, if $\alpha \rightsquigarrow_{x,y} \beta$, $\beta \in \mathsf{Clos}_x([\![\varphi]\!])$, and and y is free for x in φ, we have $\alpha \in \mathsf{Clos}_x([\![\varphi]\!])$.

Example 4.5. [Give an example of a neural network that reflects substitutions (but does not necessarily respect them)]

A neural network both *respects and reflects substitutions* iff whenever $\alpha \rightsquigarrow_{x,y} \beta$, α and β are activated by exactly the same signals $[\![\varphi]\!]$ (provided y is free for x in φ).

Definition 4.6. [A third condition for the other axiom]

Example 4.7. [Give an example of a neural network with this third condition, but not the others]

The main result of this section is that these three properties are sufficient for the neural network interpreted $\forall x$ to interact normally with $[y/x]$. [Answer: what kinds of neural network ensembles are these ones like?]

Example 4.8. [Give an example of a neural network with all three!]

I will now prove that these three properties are sufficient:

Proposition 4.9. Suppose a variable-assignment net $\mathcal{N}$ *respects* substitutions. Then the $(\rightarrow)$ direction of the axiom $[y/x]\forall x\varphi \leftrightarrow \forall x\varphi$ is sound.

Proof. Let $\mathcal{N} \in [\,]$ and let $\alpha \in N$ be any assignment. Let β be that unique assignment such that $\alpha \rightsquigarrow_{x,y} \beta$. Suppose $\mathcal{N}, \alpha \models [y/x]\forall x\varphi$. By the semantics for substitution, $\mathcal{N}, \beta \models \forall x\varphi$, and then by the semantics for $\forall x$ we have $\beta \notin \mathsf{Clos}_x([\![\varphi]\!]^{\mathsf{C}})$. Since $\mathcal{N}$ respects substitutions, $\alpha \notin \mathsf{Clos}_x([\![\varphi]\!]^{\mathsf{C}})$. But this means $\mathcal{N}, \alpha \models \forall x\varphi$, and we are done. □

Proposition 4.10. Suppose a variable-assignment net $\mathcal{N}$ *reflects* substitutions. Then the axiom "$\forall x\varphi \rightarrow [y/x]\varphi$, if y free for x in φ" is sound.

Proof. Let $\mathcal{N} \in [\,]$ and let $\alpha \in N$ be any assignment. Let β be that unique assignment such that $\alpha \rightsquigarrow_{x,y} \beta$. Suppose $\mathcal{N}, \alpha \models \forall x\varphi$, and suppose y is free for x in φ. By the semantics for $\forall x$ we have

$\alpha \notin \mathsf{Clos}_x([\![\varphi]\!]^{\mathsf{C}}) = \mathsf{Clos}_x([\![\neg\varphi]\!])$.

Now, suppose for contradiction that $\mathcal{N}, \alpha \not\models [y/x]\varphi$, i.e., $\beta \in [\![\varphi]\!]^{\mathsf{C}}$. By Inclusion of Clos_x, $\beta \in \mathsf{Clos}_x([\![\varphi]\!]^{\mathsf{C}}) = \mathsf{Clos}_x([\![\neg\varphi]\!])$. Since y is free for x in φ, y is free for x in $\neg\varphi$. So we can apply the fact that $\mathcal{N}$ reflects substitutions, which gives us $\alpha \in \mathsf{Clos}_x([\![\varphi]\!]^{\mathsf{C}})$. But this directly contradicts our hypothesis. So we must conclude that $\mathcal{N}, \alpha \models [y/x]\varphi$. □

Corollary 4.11. Suppose a variable-assignment net $\mathcal{N}$ respects substitutions. Then the ($\leftarrow$) direction of the axiom $[y/x]\forall x\varphi \leftrightarrow \forall x\varphi$ is sound.

Proof. If $\mathcal{N}$ respects substitutions, then "$\forall x\varphi \to [y/x]\varphi$, if y free for x in φ" is sound. I will show that the formula $\forall x\varphi \to [y/x]\forall x\varphi$ follows. Let $\alpha \in N$ and suppose $\mathcal{N}, \alpha \models \forall x\varphi$. By the soundness of **(Trans)**, $\mathcal{N}, \alpha \models \forall x\forall x\varphi$. Now, notice that y is free for x in the expression $\forall x\varphi$ (since x is not free at all, it is safe to substitute y for x). So by hypothesis, $\mathcal{N}, \alpha \models [y/x]\forall x\varphi$, which was the goal. □

Proposition 4.12. Suppose a variable-assignment net $\mathcal{N}$ is [todo]. Then the following substitution interaction axioms are sound:

- [todo]

Proof. [todo] □

5 Reflections on First-Order Reasoning using Neural Networks

5.1 How to Interpret Variable-Assignment Networks

- Part of my conclusion is this: In order for a variable-assignment net to behave like full FOL reasoner, it must have [list of properties]
- In addition to whatever else I do here, I should probably sanity check and make sure that there *is* actually a variable-assignment net that satisfies all of the properties necessary to make it act like a full FOL reasoner. (If there isn't one that has all the properties, then one of the properties is saying too much / together they are trivial.)

5.2 Are variable-assignment networks cognitively plausible?

Working list of papers that might bear on this question:

- Corey, *Neural basis for generalized quantifier comprehension*, 2005
- Kiela, *Variable Binding in Biologically Plausible Neural Networks*, 2011

The questions here are (1) whether there really are distinct **types** of edges in neural networks, and (2) whether neurons can be thought of as carrying variable-assignment information.

5.3 Related Work: Other Neuro-Symbolic Systems for First-Order Reasoning

Working list of neuro-symbolic systems that interpret quantifiers or variable substitution in any formal way:

- Logic Tensor Networks (over "real logic")—see *Badreddine, Logic Tensor Networks, 2022*
- Logical Neural Networks—see *Riegel, Gray, Logical neural networks, 2020*
- Kiela, *Variable Binding in Biologically Plausible Neural Networks*, 2011
- What about DeepProbLog?

Chapter 7

Conclusions

In the Introduction, I said that this dissertation serves to promote the point of view that **neural networks can be thought of as models for formal logic**. Let's now take a step back and see how far this perpective has taken us.

[I like the way Levin Hornischer wrote his: A summary, followed by a list of results, followed by a list of open questions]

I could also give a fun sampler of some future applications of neural network semantics. [For these 4 directions, I already have some basic ideas for how to proceed. So I will finish out by outlining what I think the next steps are. I don't necessarily have to prove anything difficult, but I think it's important to show that I've been thinking about these deeply, and I have a start for how to proceed (and it's nice to be able to claim the ideas!)] [Maybe it's better in this section to pivot to an "open questions" format and keep it brief…]

Neural Network Updates from Epistemic Ones. My original goal was to explore what "classical" updates correspond to neural network updates such as Hebb*, and conversely what neural updates correspond to plausibility updates such as Cond, Lex, and Consr. This theorem doesn't seem to clarify that; it's constructive (we *do* in fact build the corresponding updates), but by translating back and forth at the static level we don't define the update in the original updates "native environment." [Elaborate/clarify this point]. For the remainder of this section, I will put in the extra work to see what Hebb*, Cond, Lex, and Consr "look like" on the other side.

Definition 0.1. [Define the plausibility update that simulates Hebb*]

Definition 0.2. [Define the neural update that simulates Cond]

Definition 0.3. [Define the plausibility update that simulates Lex]

Definition 0.4. [Define the plausibility update that simulates Consr]

The Learning Power of Hebbian Learning. [Differentiate between (single-step) update and (in the limit) learning. Cite papers on learning power of different epistemic updates] [Go ahead and give the mathematical setup needed to ask the question formally.] [An interesting question, but broad: what can we say about neural network learning in the context of computational learning theory?]

A Dynamic Logic of Backpropagation. [Of course I don't have any results about this (I'm focusing on giving a complete story about the unsupervised upgrade), but in principle we can do the same thing for gradient descent (implemented as backpropagation over a neural network). Formulas $[P;Q]\varphi$ need to be used. I can go ahead and give what the semantics would look like over neural network models. And it would be nice if I could give at least one sound property of backpropagation here!]

1 Open Questions

[First, list all the open questions! Then talk in detail about a few!]

1. Give a characterization of those (recurrent) nets whose closure $\mathsf{Clos}(S)$ reaches a unique fixed point (i.e. does not oscillate)
2. Extend the semantics to account for recurrent neural networks whose $\mathsf{Clos}(S)$ *does* oscillate!
3. Extend the semantics for fuzzy (and possibly probabilistic) activation patterns $\mathsf{Clos}(S)$
4. Give semantics and sound axioms for other neural network architectures that use new ideas, e.g. transformers & attention mechanisms. (Brand new ideas are needed, this isn't just a simple extension!)
5. It's an open problem to prove formal soundness results for neuro-symbolic systems like LTNs and DeepProbLog? Do the techniques from this work shed light on how to do this, or what axioms are needed?
6. Extend the semantics to give sound (and possibly complete) axioms for other neural network updates, especially gradient descent implemented as backpropagation! Can we use

this to give a "classical" belief revision operator equivalent to backpropagation?

7. Extend the semantics to give sound (and possibly complete) neural network semantics for first-order logic! Proving soundness & completeness for *any* neural network interpretation of first-order logic is a huge open problem in neuro-symbolic AI.
8. Extend the semantics to *stabilized* Hebbian update, using Oja's rule
9. What concrete neural network learning policy corresponds to Lex? To Consr? What about Rott's 21 different belief revision operators? What classical belief revision policies correspond to Hebbian update? Oja's rule? Backpropagation?
10. Compare the learning power of neural network updates (Hebbian update, backprop) in the limit against other kinds of updates, e.g. epistemic updates, or against each other.
11. It's clear that every unsupervised belief revision operator is a supervised one. But is every supervised belief revision operator equivalent to an unsupervised one? Does this result carry over to neural networks?
12. Characterize different belief revision operators & learning policies using *frame correspondences*. Modal logic is really the perfect environment for asking this question in this way.

Bibliography

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat et al. GPT-4 technical report. *ArXiv preprint arXiv:2303.08774*, 2023.

[2] Alfred V. Aho, Michael R Garey, and Jeffrey D. Ullman. The transitive reduction of a directed graph. *SIAM Journal on Computing*, 1(2):131–137, 1972.

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

[4] Edoardo Baccini, Zoé Christoff, and Rineke Verbrugge. Dynamic logics of diffusion and link changes on social networks. *Studia Logica*, pages 1–71, 2024.

[5] Sebastian Bader and Pascal Hitzler. Dimensions of neural-symbolic integration – A structured survey. In *We Will Show Them! Essays in Honour of Dov Gabbay, Volume 1*, pages

167–194. College Publications, 2005.

[6] Samy Badreddine, Artur d'Avila Garcez, Luciano Serafini, and Michael Spranger. Logic Tensor Networks. *Artificial Intelligence*, 303:103649, 2022.

[7] Christian Balkenius and Peter Gärdenfors. Nonmonotonic inferences in neural networks. In *KR*, pages 32–39. Morgan Kaufmann, 1991.

[8] Alexandru Baltag, Zoé Christoff, Rasmus K Rendsvig, and Sonja Smets. Dynamic epistemic logics of diffusion and prediction in social networks. *Studia Logica*, 107:489–531, 2019.

[9] Alexandru Baltag, Nina Gierasimczuk, Aybüke Özgün, Ana Lucia Vargas Sandoval, and Sonja Smets. A dynamic logic for learning theory. *Journal of Logical and Algebraic Methods in Programming*, 109:100485, 2019.

[10] Alexandru Baltag, Nina Gierasimczuk, and Sonja Smets. Truth-tracking by belief revision. *Studia Logica*, 107:917–947, 2019.

[11] Alexandru Baltag, Dazhu Li, and Mina Young Pedersen. On the right path: A modal logic for supervised learning. In *International Workshop on Logic, Rationality and Interaction*, pages 1–14. Springer, 2019.

[12] Alexandru Baltag, Lawrence S Moss, and Sławomir Solecki. Logics for epistemic actions: completeness, decidability, expressivity. *Logics*, 1(2):97–147, 2023.

[13] Alexandru Baltag, Lawrence S Moss, and Slawomir Solecki. The logic of public announcements, common knowledge, and private suspicions. In *Proceedings of the 7th conference on Theoretical aspects of rationality and knowledge*, pages 43–56. 1998.

[14] Alexandru Baltag and Sonja Smets. 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*, pages 41–50. 2009.

[15] Tarek R Besold, Artur d'Avila Garcez, Sebastian Bader, Howard Bowman, Pedro Domingos, Pascal Hitzler, Kai-Uwe Kühnberger, Luis C Lamb, Priscila Machado Vieira Lima, Leo de Penning et al. Neural-symbolic learning and reasoning: A survey and interpretation. In *Neuro-Symbolic Artificial Intelligence: The State of the Art*, pages 1–51. IOS press, 2021.

[16] Patrick Blackburn, Maarten de Rijke, and Yde Venema. *Modal Logic*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2001.

[17] Reinhard Blutner. Nonmonotonic inferences and neural networks. *Synthese*, 142:143–174, 2004.

[18] Zoé Christoff and Jens Ulrik Hansen. A logic for diffusion in social networks. *Journal of Applied Logic*, 13(1):48–77, 2015.

[19] Gabriele Ciravegna, Pietro Barbiero, Francesco Giannini, Marco Gori, Pietro Lió, Marco Maggini, and Stefano Melacci. Logic Explained Networks. *Artificial Intelligence*, 314:103822, 2023.

[20] Artur d'Avila Garcez, Krysia Broda, and Dov M Gabbay. Symbolic knowledge extraction from trained neural networks: A sound approach. *Artificial Intelligence*, 125(1-2):155–207, 2001.

[21] Walter Dean. Computational complexity theory. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, 2021.

[22] Daniel C Dennett. Cognitive wheels: the frame problem of ai. *The philosophy of artificial intelligence*, 147:170, 1990.

[23] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan et al. The Llama 3 herd of models. *ArXiv preprint arXiv:2407.21783*, 2024.

[24] Herbert B Enderton. *A mathematical introduction to logic*. Elsevier, 2001.

[25] Isabel O Gallegos, Ryan A Rossi, Joe Barrow, Md Mehrab Tanjim, Sungchul Kim, Franck Dernoncourt, Tong Yu, Ruiyi Zhang, and Nesreen K Ahmed. Bias and fairness in Large Language Models: A survey. *Computational Linguistics*, pages 1–79, 2024.

[26] Artur SD'Avila Garcez, Luis C Lamb, and Dov M Gabbay. *Neural-Symbolic Cognitive Reasoning*. Springer Science & Business Media, 2008.

[27] Atticus Geiger, Zhengxuan Wu, Christopher Potts, Thomas Icard, and Noah Goodman. Finding alignments between interpretable causal variables and distributed neural represen-

tations. In *Causal Learning and Reasoning*, pages 160–187. PMLR, 2024.

[28] Laura Giordano, Valentina Gliozzi, and Daniele Theseider Dupré. A conditional, a fuzzy and a probabilistic interpretation of self-organizing maps. *Journal of Logic and Computation*, 32(2):178–205, 2022.

[29] Laura Giordano and Daniele Theseider Dupré. 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*, pages 225–242. Springer, 2021.

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

[31] Frankvan Harmelen. Preface: The 3rd AI wave is coming, and it needs a theory. In *Neuro-Symbolic Artificial Intelligence: The State of the Art*, page 0. IOS Press BV, 2022.

[32] Donald Hebb. *The Organization of Behavior*. Psychology Press, apr 1949.

[33] Neil Immerman. *Descriptive Complexity*. Springer Science & Business Media, 1998.

[34] Caleb Kisby, Saúl Blanco, and Lawrence Moss. The logic of Hebbian learning. In *The International FLAIRS Conference Proceedings*, volume 35. 2022.

[35] Caleb Schultz Kisby, Saúl A Blanco, and Lawrence S Moss. What do Hebbian learners learn? Reduction axioms for iterated Hebbian learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 14894–14901. 2024.

[36] Dexter Kozen and Rohit Parikh. An elementary proof of the completeness of PDL. *Theoretical Computer Science*, 14(1):113–118, 1981.

[37] Sarit Kraus, Daniel Lehmann, and Menachem Magidor. Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial intelligence*, 44(1-2):167–207, 1990.

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

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

[40] Hannes Leitgeb. Neural network models of conditionals. In *Introduction to Formal Philosophy*, pages 147–176. Springer, 2018.

[41] Hector J Levesque. *Common sense, the Turing test, and the quest for real AI*. MIT press, 2017.

[42] Leonid Libkin. *Elements of Finite Model Theory*, volume 41. Springer, 2004.

[43] Robin Manhaeve, Sebastijan Dumančić, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Neural probabilistic logic programming in DeepProbLog. *Artificial Intelligence*, 298:103504, 2021.

[44] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133, dec 1943.

[45] Drew McDermott. A critique of pure reason. *Computational intelligence*, 3(3):151–160, 1987.

[46] William Merrill. Sequential neural networks as automata. *ArXiv preprint arXiv:1906.01615*, 2019.

[47] William Merrill and Ashish Sabharwal. The expresssive power of transformers with chain of thought. *ArXiv preprint arXiv:2310.07923*, 2023.

[48] William Merrill, Gail Weiss, Yoav Goldberg, Roy Schwartz, Noah A Smith, and Eran Yahav. A formal hierarchy of RNN architectures. *ArXiv preprint arXiv:2004.08500*, 2020.

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

[50] Leonardo de Moura and Sebastian Ullrich. 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*, pages 625–635. Springer, 2021.

[51] Gregory Murphy. *The Big Book of Concepts*. MIT press, 2004.

[52] Cathy O'neil. *Weapons of math destruction: How big data increases inequality and threatens democracy*. Crown, 2017.

[53] Simon Odense and Artur d'Avila Garcez. A semantic framework for neurosymbolic compu-

tation. *Artificial Intelligence*, 340:104273, 2025.

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

[55] Eric Pacuit. *Neighborhood Semantics for Modal Logic*. Springer, 2017.

[56] Jan A. Plaza. Logics of public communications. *Synthese*, 158:165–179, 2007.

[57] George Polya. *Mathematics and Plausible Reasoning: Induction and Analogy in Mathematics*, volume 2. Princeton University Press, 1954.

[58] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. *Biometrika*, 71(599-607):6, 1986.

[59] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[60] Md Kamruzzaman Sarker, Lu Zhou, Aaron Eberhart, and Pascal Hitzler. Neuro-Symbolic Artificial Intelligence: Current Trends. *AI Communications*, 34, 2022 2022.

[61] Murray Shanahan. The frame problem. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, 2016.

[62] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton et al. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359, 2017.

[63] Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. Training very deep networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28, page 0. Curran Associates, Inc., 2015.

[64] Lena Strobl, William Merrill, Gail Weiss, David Chiang, and Dana Angluin. What formal languages can transformers express? A survey. *Transactions of the Association for Computational Linguistics*, 12:543–561, 2024.

[65] Alex Tamkin, Miles Brundage, Jack Clark, and Deep Ganguli. Understanding the capabilities, limitations, and societal impact of Large Language Models. *ArXiv preprint arXiv:2102.02503*, 2021.

[66] Johan van Benthem. Dynamic logic for belief revision. *Journal of applied non-classical logics*, 17(2):129–155, 2007.

[67] Johan van Benthem. *Modal logic for open minds*. Number 199. Center for the Study of Language and Information Stanford, 2010.

[68] Johan van Benthem. *Logical Dynamics of Information and Interaction*. Cambridge University Press, 2011.

[69] Johan van Benthem and Fenrong Liu. Dynamic logic of preference upgrade. *Journal of Applied Non-Classical Logics*, 17(2):157–182, 2007.

[70] Johan van Benthem and Sonja Smets. Dynamic logics of belief change. In H. Van Ditmarsch, J. Halpern, W. van der Hoek, and B. Kooi, editors, *Handbook of Epistemic Logic*, pages 313–393. College Publications, London, UK, 2015.

[71] Hans Van Ditmarsch, Wiebe van Der Hoek, and Barteld Kooi. *Dynamic Epistemic Logic*, volume 337. Springer, 2007.

[72] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[73] Gail Weiss, Yoav Goldberg, and Eran Yahav. On the practical computational power of finite precision RNNs for language recognition. *ArXiv preprint arXiv:1805.04908*, 2018.

Appendix A

A.1 Appendix for Section 3.

Theorem 4.1. blah blah

Proof. □

Proof Sketch. dfadsl;fjal;sdfj □

A.2 Appendix A2

A.3 Appendix A3