Semantics for Non-symbolic Computation

Including Neural Networks and Other Analog Computers

Levin Hornischer

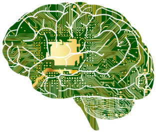
LMU Munich / MCMP

https://levinhornischer.github.io/

Logic and AI workshop

Amsterdam, 16 July 2024

Introduction: Foundational theory of AI?



symbolic	non-symbolic
Semantics	?
→ interpret	?
→ explain	?
→ verify	?
Computability theory	...
Complexity theory	...

Outline

Outline

Outline

Outline

Outline

The problem

- Classical ‘symbolic’ computation is specified by program code P in a programming language.
  - What does this code mean?
  - How to interpret and explain it?
  - How can we verify that it computes what it should?

The problem

- Classical ‘symbolic’ computation is specified by program code P in a programming language.
  - What does this code mean?
  - How to interpret and explain it?
  - How can we verify that it computes what it should?

Answer: semantics!

1. Operational semantics (systems) describes P by the steps a machine would take to implement P.
   - transition system.
2. Denotational semantics (domains) describes P by the computed function $[\![P]\!]$ and its finite approximations.
   - elements of domain D.
3. Logical semantics (logic) describes P by its properties
   - Hoare triple, e.g., $\{\text{is } 1\}P\{\text{is even}\}$.

Answer: semantics!

1. Operational semantics (systems) describes P by the steps a machine would take to implement P.
   - transition system.
2. Denotational semantics (domains) describes P by the computed function $[\![P]\!]$ and its finite approximations.
   - elements of domain D.
3. Logical semantics (logic) describes P by its properties
   - Hoare triple, e.g., $\{\mathsf{is}\ 1\}P\{\mathsf{is\ even}\}$.

Answer: semantics!

1. Operational semantics (systems) describes P by the steps a machine would take to implement P.
   - transition system.
2. Denotational semantics (domains) describes P by the computed function $[\![P]\!]$ and its finite approximations.
   - elements of domain D.
3. Logical semantics (logic) describes P by its properties
   - Hoare triple, e.g., $\{\textsf{is } 1\}P\{\textsf{is even}\}$.

Harmony

- Operational vs logical: partial correctness [Hoa69]
  - if $\{\varphi\}P\{\psi\}$ is provable, then running program P in a φ-state results in a ψ-state (if P terminates).
- Operational vs denotational: full abstraction [Ong95]
  - two programs have the same denotation iff the implementing machines show the same behavior.
- Denotational vs logical: Stone duality [Abr91]
  - the properties of P jointly determine the denotation $[\![P]\!]$, and vice versa.

Harmony

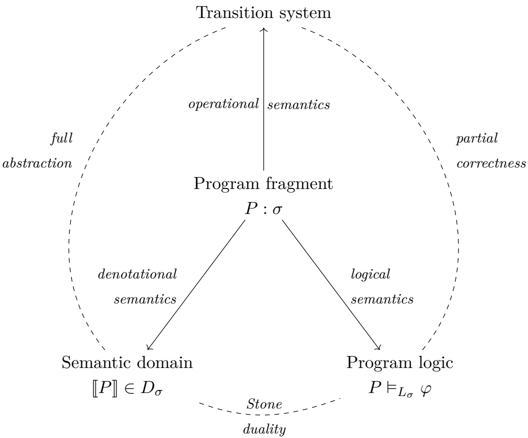
- Operational vs logical: partial correctness [Hoa69]
  - if $\{\varphi\}P\{\psi\}$ is provable, then running program P in a φ-state results in a ψ-state (if P terminates).
- Operational vs denotational: full abstraction [Ong95]
  - two programs have the same denotation iff the implementing machines show the same behavior.
- Denotational vs logical: Stone duality [Abr91]
  - the properties of P jointly determine the denotation $[\![P]\!]$, and vice versa.

Harmony

- Operational vs logical: partial correctness [Hoa69]
  - if $\{\varphi\}P\{\psi\}$ is provable, then running program P in a φ-state results in a ψ-state (if P terminates).
- Operational vs denotational: full abstraction [Ong95]
  - two programs have the same denotation iff the implementing machines show the same behavior.
- Denotational vs logical: Stone duality [Abr91]
  - the properties of P jointly determine the denotation $[\![P]\!]$, and vice versa.

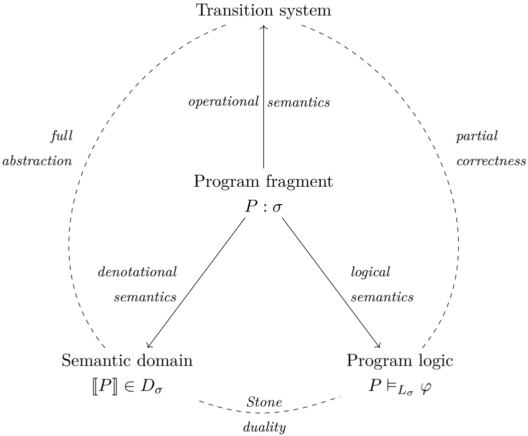
Summary

(extended from [Jun13])



Summary

(extended from [Jun13])



Goal: Analogous semantics for non-symbolic computation!

Outline

Handwritten digit classification

- Standard machine learning task (MNIST dataset)

   $\mapsto 0$  $\mapsto 5$  $\mapsto 3$  $\mapsto 3$

- Use neural network:

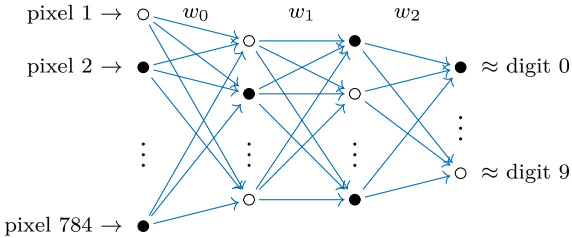
  - First, randomly initialize weights.
  - Then, go through batches of training data and update weights by backprop for better classification.

Handwritten digit classification

- Standard machine learning task (MNIST dataset)

 $\mapsto 0$  $\mapsto 5$  $\mapsto 3$  $\mapsto 3$

- Use neural network:



  - First, randomly initialize weights.
  - Then, go through batches of training data and update weights by backprop for better classification.

Training dynamics

- Backprop and dataset specify
  - training dynamics $T : X \to X$ on weight space X
  - probability distribution μ on X for initialization.
- Understanding this nonlinear dynamics (X, μ, T) is a crucial open problem in the theory of machine learning! [SMG14]

Training dynamics

- Backprop and dataset specify
  - training dynamics $T : X \to X$ on weight space X
  - probability distribution μ on X for initialization.
- Understanding this nonlinear dynamics (X, μ, T) is a crucial open problem in the theory of machine learning! [SMG14]

Outline

Semantics for non-symbolic computation (idea)

- How is non-symbolic computation specified? By dynamical systems! [BP21]
  - Neural networks
  - Cellular automata
  - Analog computation
- Denotation of a dynamical system
  - as the limit of finite approximations
  - via interpretable observations.
- Logic of a dynamical system as Hoare triple $\{\varphi\}T\{\psi\}$
  - if in φ-state now,
  - then in ψ-state next.

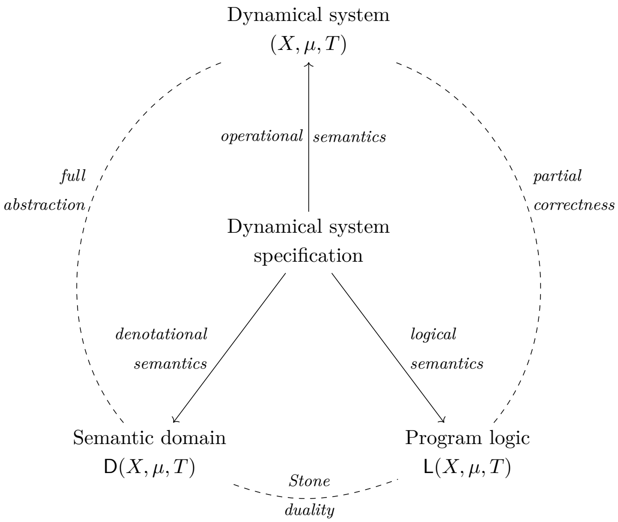
Semantics for non-symbolic computation (idea)

- How is non-symbolic computation specified? By dynamical systems! [BP21]
  - Neural networks
  - Cellular automata
  - Analog computation
- Denotation of a dynamical system
  - as the limit of finite approximations
  - via interpretable observations.
- Logic of a dynamical system as Hoare triple $\{\varphi\}T\{\psi\}$
  - if in φ-state now,
  - then in ψ-state next.

Semantics for non-symbolic computation (idea)

- How is non-symbolic computation specified? By dynamical systems! [BP21]
  - Neural networks
  - Cellular automata
  - Analog computation
- Denotation of a dynamical system
  - as the limit of finite approximations
  - via interpretable observations.
- Logic of a dynamical system as Hoare triple $\{\varphi\}T\{\psi\}$
  - if in φ-state now,
  - then in ψ-state next.

Semantics for non-symbolic computation (idea)



Outline

Dynamical systems I

Definition

A dynamical system $\mathfrak{X}$ is a structure (X, τ, μ, T) where

- (X, τ) compact zero-dimensional Polish space
- μ is Borel probability measure
- $T : X \to X$ is continuous.

- Captures measurable dynamics on standard probability spaces.
- Includes ergodic theory.

Dynamical systems I

Definition

A dynamical system $\mathfrak{X}$ is a structure (X, τ, μ, T) where

- (X, τ) compact zero-dimensional Polish space
- μ is Borel probability measure
- $T : X \to X$ is continuous.

- Captures measurable dynamics on standard probability spaces.
- Includes ergodic theory.

Dynamical systems I

Definition

A dynamical system $\mathfrak{X}$ is a structure (X, τ, μ, T) where

- (X, τ) compact zero-dimensional Polish space
- μ is Borel probability measure
- $T : X \to X$ is continuous.

- Captures measurable dynamics on standard probability spaces.
- Includes ergodic theory.

Dynamical systems II

Definition

A system morphism $\varphi : (X, \tau, \mu, T) \to (Y, \sigma, \nu, S)$ is a continuous, measure-preserving, and equivariant function $\varphi : X \to Y$.

$$\begin{array}{ccc} X & \xrightarrow{\varphi} & Y \\ {\scriptstyle T}\downarrow & & \downarrow{\scriptstyle S} \\ X & \xrightarrow{\varphi} & Y \end{array}$$

Definition

The category of dynamical systems is denoted DS.

Outline

From systems to domains: the idea

- Given $\mathfrak{X} = (X, \tau, \mu, T)$, construct denotation $\mathfrak{D} = (D, v, f)$, the dynamical domain of $\mathfrak{X}$.
- $\mathfrak{D}$ as the limit of finite approximations, given by interpretable observations about the system.

From systems to domains: the idea

- Given $\mathfrak{X} = (X, \tau, \mu, T)$, construct denotation $\mathfrak{D} = (D, v, f)$, the dynamical domain of $\mathfrak{X}$.
- $\mathfrak{D}$ as the limit of finite approximations, given by interpretable observations about the system.

Observed system I

- A clopen set $U \subset X$ as observation/measurement:
  - if system in a state $x \in U$, measurement U is positive.
- Observe system through a finite clopen cover $\mathcal{C}$ of X for time n:
  - An observation sequence that a state x gives rise to:

$$\begin{array}{cccccc} U_0 & U_1 & U_2 & U_3 & \cdots & U_{n-1} \\ \rotatebox{90}{$\in$} & \rotatebox{90}{$\in$} & \rotatebox{90}{$\in$} & \rotatebox{90}{$\in$} & \cdots & \rotatebox{90}{$\in$} \\ x & T(x) & T^2(x) & T^3(x) & \cdots & T^{n-1}(x) \end{array}$$

- Formally: For observation parameter $i = (n, \mathcal{C})$ and state $x \in X$, the observation sequences are:

$$\mathcal{O}_i(x) := \left\{ t \in \mathcal{C}^n : T^k(x) \in t_k \text{ for } k = 0, \ldots, n-1 \right\}$$

Observed system I

- A clopen set $U \subset X$ as observation/measurement:
  - if system in a state $x \in U$, measurement U is positive.
- Observe system through a finite clopen cover $\mathcal{C}$ of X for time n:
  - An observation sequence that a state x gives rise to:

$$
\begin{array}{cccccc}
U_0 & U_1 & U_2 & U_3 & \cdots & U_{n-1} \\
\rotatebox{90}{$\in$} & \rotatebox{90}{$\in$} & \rotatebox{90}{$\in$} & \rotatebox{90}{$\in$} & \cdots & \rotatebox{90}{$\in$} \\
x & T(x) & T^2(x) & T^3(x) & \cdots & T^{n-1}(x)
\end{array}
$$

- Formally: For observation parameter $i = (n, \mathcal{C})$ and state $x \in X$, the observation sequences are:

$$
\mathcal{O}_i(x) := \left\{ t \in \mathcal{C}^n : T^k(x) \in t_k \text{ for } k = 0, \ldots, n-1 \right\}
$$

Observed system I

- A clopen set $U \subset X$ as observation/measurement:
  - if system in a state $x \in U$, measurement U is positive.
- Observe system through a finite clopen cover $\mathcal{C}$ of X for time n:
  - An observation sequence that a state x gives rise to:

$$\begin{array}{cccccc} U_0 & U_1 & U_2 & U_3 & \cdots & U_{n-1} \\ \Cup & \Cup & \Cup & \Cup & \cdots & \Cup \\ x & T(x) & T^2(x) & T^3(x) & \cdots & T^{n-1}(x) \end{array}$$

- Formally: For observation parameter $i = (n, \mathcal{C})$ and state $x \in X$, the observation sequences are:

$$\mathcal{O}_i(x) := \Big\{ t \in \mathcal{C}^n : T^k(x) \in t_k \text{ for } k = 0, \dots, n-1 \Big\}$$

Observed system I

- A clopen set $U \subset X$ as observation/measurement:
  - if system in a state $x \in U$, measurement U is positive.
- Observe system through a finite clopen cover $\mathcal{C}$ of X for time n:
  - An observation sequence that a state x gives rise to:

$$\begin{array}{cccccc} U_0 & U_1 & U_2 & U_3 & \cdots & U_{n-1} \\ \Cup & \Cup & \Cup & \Cup & \cdots & \Cup \\ x & T(x) & T^2(x) & T^3(x) & \cdots & T^{n-1}(x) \end{array}$$

- Formally: For observation parameter $i = (n, \mathcal{C})$ and state $x \in X$, the observation sequences are:

$$\mathcal{O}_i(x) := \left\{ t \in \mathcal{C}^n : T^k(x) \in t_k \text{ for } k = 0, \dots, n-1 \right\}$$

Observed system II

- All the observable behaviors:

$$\mathsf{H}_i := \{\mathcal{O}_i(x) : x \in X\}.$$

- Has dynamics: $\mathcal{O}_i(x) \to_i \mathcal{O}_i(y)$ iff

$$\exists x' : \mathcal{O}_i(x) = \mathcal{O}_i(x') \text{ and } \mathcal{O}_i(y) = \mathcal{O}_i(T(x')).$$

- And probability:

$$v_i(\mathcal{O}_i(x)) := \mu\{x' \in X : \mathcal{O}_i(x') = \mathcal{O}_i(x)\}.$$

Observed system II

- All the observable behaviors:

$$\mathsf{H}_i := \{\mathcal{O}_i(x) : x \in X\}.$$

- Has dynamics: $\mathcal{O}_i(x) \rightarrow_i \mathcal{O}_i(y)$ iff

$$\exists x' : \mathcal{O}_i(x) = \mathcal{O}_i(x') \text{ and } \mathcal{O}_i(y) = \mathcal{O}_i(T(x')).$$

- And probability:

$$v_i(\mathcal{O}_i(x)) := \mu\{x' \in X : \mathcal{O}_i(x') = \mathcal{O}_i(x)\}.$$

Observed system II

- All the observable behaviors:

$$\mathsf{H}_i := \{\mathcal{O}_i(x) : x \in X\}.$$

- Has dynamics: $\mathcal{O}_i(x) \to_i \mathcal{O}_i(y)$ iff

$$\exists x' : \mathcal{O}_i(x) = \mathcal{O}_i(x') \text{ and } \mathcal{O}_i(y) = \mathcal{O}_i(T(x')).$$

- And probability:

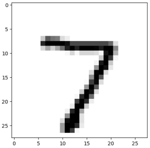
$$v_i(\mathcal{O}_i(x)) := \mu\{x' \in X : \mathcal{O}_i(x') = \mathcal{O}_i(x)\}.$$

Example: Training dynamics I

- Observation U (human-interpretable!):

  weight state w is in U

  iff the neural network with weights w correctly classifies the following image as a 7



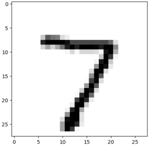
- Cover $\mathcal{C} = \{U, U^c\}$, time $n = 2$. Set $i := (n, \mathcal{C})$.
- Python implementation to compute the observed system (D_i, v_i, f_i):

Example: Training dynamics I

- Observation U (human-interpretable!):

  weight state w is in U

  iff the neural network with weights w correctly classifies the following image as a 7



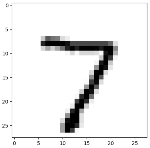
- Cover $\mathcal{C} = \{U, U^c\}$, time $n = 2$. Set $i := (n, \mathcal{C})$.
- Python implementation to compute the observed system (D_i, v_i, f_i):

Example: Training dynamics I

- Observation U (human-interpretable!):

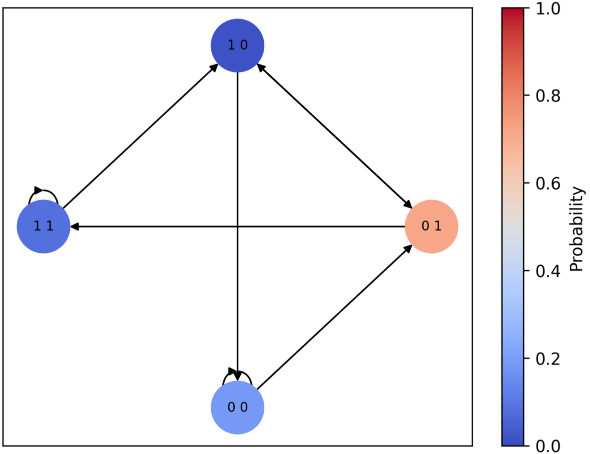
  weight state w is in U

  iff the neural network with weights w correctly classifies the following image as a 7



- Cover $\mathcal{C} = \{U, U^c\}$, time $n = 2$. Set $i := (n, \mathcal{C})$.
- Python implementation to compute the observed system (D_i, v_i, f_i):

Example: Training dynamics II



Legend: `01` for $\mathcal{O}_i(x) = \{(U^c, U)\}$, `11` for $\mathcal{O}_i(y) = \{(U, U)\}$, etc.

`01` $\rightarrow$ `11` for $\mathcal{O}_i(x) \rightarrow_i \mathcal{O}_i(y)$

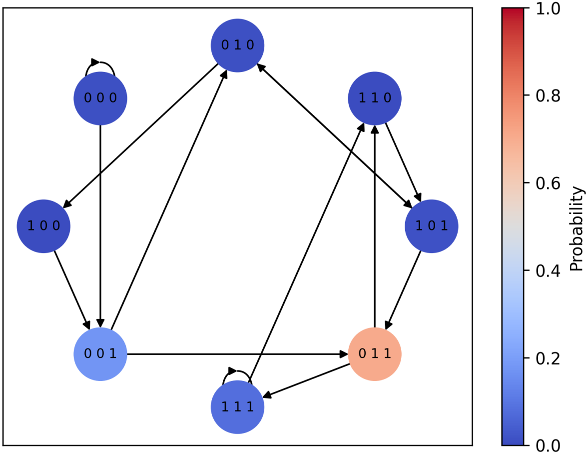
Example: Training dynamics I

Refine observation parameters: longer observation time n and finer cover $\mathcal{C}$.

Example: Training dynamics I

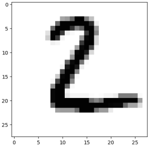
Refine observation parameters: longer observation time n and finer cover $\mathcal{C}$.

Refine time from $n = 2$ to $n = 3$:



Example: Training dynamics II

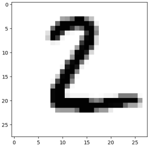
Refine cover: Also consider observation V whether this image is classified correctly



From $\mathcal{C} = \{U, U^c\}$ to $\mathcal{D} = \{U \cap V, U \cap V^c, U^c \cap V, U^c \cap V^c\}$

Example: Training dynamics II

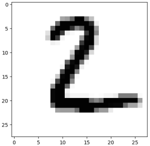
Refine cover: Also consider observation V whether this image is classified correctly



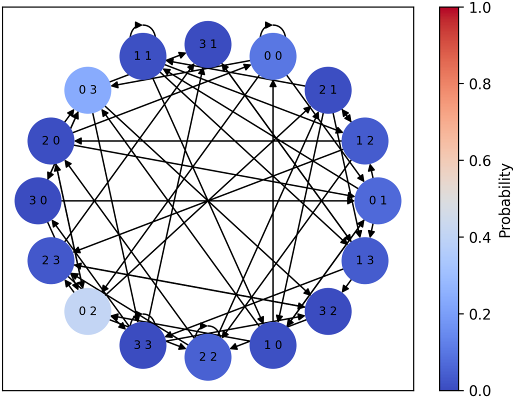
From $\mathcal{C} = \{U, U^c\}$ to $\mathcal{D} = \{U \cap V, U \cap V^c, U^c \cap V, U^c \cap V^c\}$

Example: Training dynamics II

Refine cover: Also consider observation V whether this image is classified correctly



From $\mathcal{C} = \{U, U^c\}$ to $\mathcal{D} = \{U \cap V, U \cap V^c, U^c \cap V, U^c \cap V^c\}$



Taking stock

- Have:
  - Finite interpretable approximations $(\mathsf{H}_i, \rightarrow_i, v_i)$ to system (X, τ, μ, T).
  - Can refine them by refining the observation parameter
- Want:
  - Take the limit of the approximations under refinement: to get denotation of the system.
  - Relate to domain theory: to get an analog of denotational semantics for non-symbolic computation.
- Plan:
  - Turn each $(\mathsf{H}_i, \rightarrow_i, v_i)$ into domain $\mathfrak{D}_i = (D_i, f_i, v_i)$.
  - and construct limit of these domains.
  - Cf. Čech/sheaf cohomology (presheaf of 'domains')

Taking stock

- Have:
  - Finite interpretable approximations $(\mathsf{H}_i, \to_i, v_i)$ to system (X, τ, μ, T).
  - Can refine them by refining the observation parameter
- Want:
  - Take the limit of the approximations under refinement: to get denotation of the system.
  - Relate to domain theory: to get an analog of denotational semantics for non-symbolic computation.
- Plan:
  - Turn each $(\mathsf{H}_i, \to_i, v_i)$ into domain $\mathfrak{D}_i = (D_i, f_i, v_i)$.
  - and construct limit of these domains.
  - Cf. Čech/sheaf cohomology (presheaf of 'domains')

Taking stock

- Have:
  - Finite interpretable approximations $(\mathsf{H}_i, \to_i, v_i)$ to system (X, τ, μ, T).
  - Can refine them by refining the observation parameter
- Want:
  - Take the limit of the approximations under refinement: to get denotation of the system.
  - Relate to domain theory: to get an analog of denotational semantics for non-symbolic computation.
- Plan:
  - Turn each $(\mathsf{H}_i, \to_i, v_i)$ into domain $\mathfrak{D}_i = (D_i, f_i, v_i)$.
  - and construct limit of these domains.
  - Cf. Čech/sheaf cohomology (presheaf of 'domains')

Outline

Background domain theory I

- Domain theory provides denotational semantics for symbolic computation.
- Domains are certain partial orders.
- Intuitively, elements are outputs of computational processes
- and the order describes information containment.
- Example: finite and infinite binary strings ordered by extension.

Background domain theory II

- A dcpo is a partial order $(D, \leq)$ where every directed subset A has a least upper bound $\bigvee A$ (aka join).
- Scott topology: $U \subseteq D$ is Scott-open if
  - $a \in U, a \leq b \Rightarrow b \in U$
  - $A \subseteq D$ directed, $\bigvee A \in U \Rightarrow \exists a \in A : a \in U$
- Function $f : D \to E$ between dcpos Scott-continuous iff monotone and preserves directed joins.
- Scott domain: non-empty, 'ω-algebraic', 'bounded complete' dcpo.

Background domain theory II

- A dcpo is a partial order $(D, \leq)$ where every directed subset A has a least upper bound $\bigvee A$ (aka join).
- Scott topology: $U \subseteq D$ is Scott-open if
  - $a \in U, a \leq b \Rightarrow b \in U$
  - $A \subseteq D$ directed, $\bigvee A \in U \Rightarrow \exists a \in A : a \in U$
- Function $f : D \to E$ between dcpos Scott-continuous iff monotone and preserves directed joins.
- Scott domain: non-empty, 'ω-algebraic', 'bounded complete' dcpo.

Background domain theory II

- A dcpo is a partial order $(D, \leq)$ where every directed subset A has a least upper bound $\bigvee A$ (aka join).
- Scott topology: $U \subseteq D$ is Scott-open if
  - $a \in U, a \leq b \Rightarrow b \in U$
  - $A \subseteq D$ directed, $\bigvee A \in U \Rightarrow \exists a \in A : a \in U$
- Function $f : D \to E$ between dcpos Scott-continuous iff monotone and preserves directed joins.
- Scott domain: non-empty, 'ω-algebraic', 'bounded complete' dcpo.

Background domain theory II

- A dcpo is a partial order $(D, \leq)$ where every directed subset A has a least upper bound $\bigvee A$ (aka join).
- Scott topology: $U \subseteq D$ is Scott-open if
  - $a \in U, a \leq b \Rightarrow b \in U$
  - $A \subseteq D$ directed, $\bigvee A \in U \Rightarrow \exists a \in A : a \in U$
- Function $f : D \to E$ between dcpos Scott-continuous iff monotone and preserves directed joins.
- Scott domain: non-empty, 'ω-algebraic', 'bounded complete' dcpo.

Turn observed system into domain

Given observed system $(\mathsf{H}_i, \rightarrow_i, v_i)$, trick: Smyth powerdomain!

- $D_i := \{M : \emptyset \neq M \subseteq \mathsf{H}_i\}$, ordered by $\supseteq$
  Finite Scott domain
- $f_i : D_i \to D_i$ by $f_i(M) := \{\mathcal{O}_i(T(y)) : \mathcal{O}_i(y) \in M\}$
  Scott-continuous function
- $v_i(\mathcal{O}_i(x)) := \mu\{y \in X : \mathcal{O}_i(y) = \mathcal{O}_i(x)\}$.
  Valuation

Turn observed system into domain

Given observed system $(\mathsf{H}_i, \to_i, v_i)$, trick: Smyth powerdomain!

- $D_i := \{M : \emptyset \neq M \subseteq \mathsf{H}_i\}$, ordered by $\supseteq$
  Finite Scott domain
- $f_i : D_i \to D_i$ by $f_i(M) := \{\mathcal{O}_i(T(y)) : \mathcal{O}_i(y) \in M\}$
  Scott-continuous function
- $v_i(\mathcal{O}_i(x)) := \mu\{y \in X : \mathcal{O}_i(y) = \mathcal{O}_i(x)\}$.
  Valuation

Turn observed system into domain

Given observed system $(\mathsf{H}_i, \rightarrow_i, v_i)$, trick: Smyth powerdomain!

- $D_i := \{M : \emptyset \neq M \subseteq \mathsf{H}_i\}$, ordered by $\supseteq$
  Finite Scott domain
- $f_i : D_i \rightarrow D_i$ by $f_i(M) := \{\mathcal{O}_i(T(y)) : \mathcal{O}_i(y) \in M\}$
  Scott-continuous function
- $v_i(\mathcal{O}_i(x)) := \mu\{y \in X : \mathcal{O}_i(y) = \mathcal{O}_i(x)\}$.
  Valuation

Turn observed system into domain

Given observed system $(\mathsf{H}_i, \rightarrow_i, v_i)$, trick: Smyth powerdomain!

- $D_i := \{M : \emptyset \neq M \subseteq \mathsf{H}_i\}$, ordered by $\supseteq$
  Finite Scott domain
- $f_i : D_i \to D_i$ by $f_i(M) := \{\mathcal{O}_i(T(y)) : \mathcal{O}_i(y) \in M\}$
  Scott-continuous function
- $v_i\big(\mathcal{O}_i(x)\big) := \mu\{y \in X : \mathcal{O}_i(y) = \mathcal{O}_i(x)\}$.
  Valuation

Turn observed system into domain

Given observed system $(\mathsf{H}_i, \to_i, v_i)$, trick: Smyth powerdomain!

- $D_i := \{M : \emptyset \neq M \subseteq \mathsf{H}_i\}$, ordered by $\supseteq$
  Finite Scott domain
- $f_i : D_i \to D_i$ by $f_i(M) := \{\mathcal{O}_i(T(y)) : \mathcal{O}_i(y) \in M\}$
  Scott-continuous function
- $v_i(\mathcal{O}_i(x)) := \mu\{y \in X : \mathcal{O}_i(y) = \mathcal{O}_i(x)\}$.
  Valuation

Write $\mathfrak{D}_i := (D_i, v_i, f_i)$.

Taking limits of these domains?

In domain theory,

- one takes limits
- of expanding systems
- to define interesting categories of domains.

Expanding systems

Definition (sketch)

An expanding system $(\mathfrak{D}_i, p_{ij})_I$ consists of

- I countable directed
- $\mathfrak{D}_i = (D_i, v_i, f_i)$ with D_i finite Scott domain, v_i 'max-normalized' valuation, $f_i : D_i \to D_i$ continuous.
- $p_{ij} : D_j \to D_i$ commuting projections that are 'max-bisimulative', 'valuation-preserving', and 'max-semi-equivariant' ($p_{ij}(f_j(a)) \geq f_i(p_{ij}(a))$)
- 'upward deterministic': if no unique maximal element about $f_i(a_i)$, then remedied at some later j.

Dynamical domains

Theorem (The limit theorem (sketch))

Let $(\mathfrak{D}_i, p_{ij})_I$ be an expanding system. Then it has a limit (D, v, f), subject to f preserving maximal elements: i.e., and other such (E, w, g) uniquely factors through (D, v, f).

Definition (Dynamical domains)

The category dDOM of dynamical domains consists of:

- objects: those (D, v, f) that arise as limits in the above sense
- morphisms: Scott continuous functions that are 'max-bisimulative', valuation-preserving, and 'max-semi-equivariant'.

Outline

From systems to domains

Theorem

- *Let* $\mathfrak{X} = (X, \tau, \mu, T)$ *be a dynamical system.*
- *Let* $I(\tau)$ *be the observation parameters* $(n, \mathcal{C})$ *ordered by refinement.*
- *Build* $\mathfrak{D}_i = (D_i, v_i, f_i)$ *and* $p_{ij} : D_j \to D_i$ *as described.*
- *Then* $(\mathfrak{D}_i, p_{ij})_{I(\tau)}$ *is an expanding system.*
- *Then the limit* $\mathsf{D}(\mathfrak{X}) := (D, v, f)$*, the denotation of* $\mathfrak{X}$*, is in* dDOM*.*

- $\mathsf{D}(\mathfrak{X})$ can be seen as an element of

$$\underbrace{[D \to D]}_{\text{function space}} \times \underbrace{\mathsf{P}^1(D)}_{\text{norm. prob. powerdomain}}$$

From systems to domains

Theorem

- *Let $\mathfrak{X} = (X, \tau, \mu, T)$ be a dynamical system.*
- *Let $I(\tau)$ be the observation parameters $(n, \mathcal{C})$ ordered by refinement.*
- *Build $\mathfrak{D}_i = (D_i, v_i, f_i)$ and $p_{ij} : D_j \to D_i$ as described.*
- *Then $(\mathfrak{D}_i, p_{ij})_{I(\tau)}$ is an expanding system.*
- *Then the limit $\mathsf{D}(\mathfrak{X}) := (D, v, f)$, the denotation of $\mathfrak{X}$, is in* dDOM.

- $\mathsf{D}(\mathfrak{X})$ can be seen as an element of

$$\underbrace{[D \to D]}_{\text{function space}} \times \underbrace{\mathsf{P}^1(D)}_{\text{norm. prob. powerdomain}}$$

From systems to domains

Theorem

- *Let* $\mathfrak{X} = (X, \tau, \mu, T)$ *be a dynamical system.*
- *Let* $I(\tau)$ *be the observation parameters* $(n, \mathcal{C})$ *ordered by refinement.*
- *Build* $\mathfrak{D}_i = (D_i, v_i, f_i)$ *and* $p_{ij} : D_j \to D_i$ *as described.*
- *Then* $(\mathfrak{D}_i, p_{ij})_{I(\tau)}$ *is an expanding system.*
- *Then the limit* $\mathsf{D}(\mathfrak{X}) := (D, v, f)$*, the denotation of* $\mathfrak{X}$*, is in* dDOM.

- $\mathsf{D}(\mathfrak{X})$ can be seen as an element of

$$\underbrace{[D \to D]}_{\text{function space}} \times \underbrace{\mathsf{P}^1(D)}_{\text{norm. prob. powerdomain}}$$

From systems to domains

Theorem

- *Let* $\mathfrak{X} = (X, \tau, \mu, T)$ *be a dynamical system.*
- *Let* $I(\tau)$ *be the observation parameters* $(n, \mathcal{C})$ *ordered by refinement.*
- *Build* $\mathfrak{D}_i = (D_i, v_i, f_i)$ *and* $p_{ij} : D_j \to D_i$ *as described.*
- *Then* $(\mathfrak{D}_i, p_{ij})_{I(\tau)}$ *is an expanding system.*
- *Then the limit* $\mathsf{D}(\mathfrak{X}) := (D, v, f)$*, the denotation of* $\mathfrak{X}$*, is in* dDOM*.*

- $\mathsf{D}(\mathfrak{X})$ can be seen as an element of

$$\underbrace{[D \to D]}_{\text{function space}} \times \underbrace{\mathsf{P}^1(D)}_{\text{norm. prob. powerdomain}}$$

From systems to domains

Theorem

- *Let $\mathfrak{X} = (X, \tau, \mu, T)$ be a dynamical system.*
- *Let $I(\tau)$ be the observation parameters $(n, \mathcal{C})$ ordered by refinement.*
- *Build $\mathfrak{D}_i = (D_i, v_i, f_i)$ and $p_{ij} : D_j \to D_i$ as described.*
- *Then $(\mathfrak{D}_i, p_{ij})_{I(\tau)}$ is an expanding system.*
- *Then the limit $\mathsf{D}(\mathfrak{X}) := (D, v, f)$, the denotation of $\mathfrak{X}$, is in* dDOM.

- $\mathsf{D}(\mathfrak{X})$ can be seen as an element of

$$\underbrace{[D \to D]}_{\text{function space}} \times \underbrace{\mathsf{P}^1(D)}_{\text{norm. prob. powerdomain}}$$

From systems to domains

Theorem

- *Let* $\mathfrak{X} = (X, \tau, \mu, T)$ *be a dynamical system.*
- *Let* $I(\tau)$ *be the observation parameters* $(n, \mathcal{C})$ *ordered by refinement.*
- *Build* $\mathfrak{D}_i = (D_i, v_i, f_i)$ *and* $p_{ij} : D_j \to D_i$ *as described.*
- *Then* $(\mathfrak{D}_i, p_{ij})_{I(\tau)}$ *is an expanding system.*
- *Then the limit* $\mathsf{D}(\mathfrak{X}) := (D, v, f)$*, the denotation of* $\mathfrak{X}$*, is in* dDOM.

- $\mathsf{D}(\mathfrak{X})$ can be seen as an element of

$$\underbrace{[D \to D]}_{\text{function space}} \times \underbrace{\mathsf{P}^1(D)}_{\text{norm. prob. powerdomain}}$$

From domains to systems

Theorem

Let $\mathfrak{D} = (D, v, f)$ be a dynamical domain. Then

$$\mathsf{S}(\mathfrak{D}) := \big(\max D, \mathcal{B}(\tau), \mu_v, f \restriction \max D\big).$$

is the dynamical system modeled by $\mathfrak{D}$.

Equivalence: Full abstraction

Theorem

$$
\begin{tikzcd}
\mathsf{DS} \arrow[rr, bend left, "\mathsf{D}"] \arrow[rr, phantom, "\top"] & & \mathsf{dDOM} \arrow[ll, bend left, "\mathsf{S}"]
\end{tikzcd}
$$

equivalence when restricting to 'max-reflective' domains.
In particular, isomorphism of systems $\varphi : \mathfrak{X} \to \mathsf{S}(\mathsf{D}(\mathfrak{X}))$

$$x \mapsto \left\langle \{\mathcal{O}_i(x)\} : i \in I(\tau) \right\rangle.$$

Outline

Program logic?

Given system $\mathfrak{X} = (X, \tau, \mu, T)$,

- Clopens $\mathsf{Clp}(X)$ as observations
  - logic of observations of the state space.
- Measure restricts to $m := \mu \restriction \mathsf{Clp}(X)$
  - measure algebra
- Dynamics operator $\lozenge := T^{-1} : \mathsf{Clp}(X) \to \mathsf{Clp}(X)$.
  - $\lozenge a$ is the observation that the system will have property a next.
  - Hoare triple $\{a\}T\{b\} := a \to \lozenge b$ valid iff for any a-state, applying T yields b-state.

Program logic?

Given system $\mathfrak{X} = (X, \tau, \mu, T)$,

- Clopens $\mathsf{Clp}(X)$ as observations
  - logic of observations of the state space.
- Measure restricts to $m := \mu \upharpoonright \mathsf{Clp}(X)$
  - measure algebra
- Dynamics operator $\Diamond := T^{-1} : \mathsf{Clp}(X) \to \mathsf{Clp}(X)$.
  - $\Diamond a$ is the observation that the system will have property a next.
  - Hoare triple $\{a\}T\{b\} := a \to \Diamond b$ valid iff for any a-state, applying T yields b-state.

Program logic?

Given system $\mathfrak{X} = (X, \tau, \mu, T)$,

- Clopens $\mathsf{Clp}(X)$ as observations
  - logic of observations of the state space.
- Measure restricts to $m := \mu \restriction \mathsf{Clp}(X)$
  - measure algebra
- Dynamics operator $\Diamond := T^{-1} : \mathsf{Clp}(X) \to \mathsf{Clp}(X)$.
  - $\Diamond a$ is the observation that the system will have property a next.
  - Hoare triple $\{a\}T\{b\} := a \to \Diamond b$ valid iff for any a-state, applying T yields b-state.

Measure algebra with operator

Definition

A *measure algebra with operator* (MAO) is a structure $(A, m, \Diamond)$ where

- A is a countable Boolean algebra,
- $m : A \to [0,1]$ such that $m(1) = 1$ and $a \wedge b = 0 \Rightarrow m(a \vee b) = m(a) + m(b)$
- $\Diamond : A \to A$ is a Boolean algebra homomorphism.

If $\mathfrak{X} = (X, \tau, \mu, T)$ is a system, get MAO

$$\mathsf{L}(\mathfrak{X}) := (\mathsf{Clp}(X), \mu \restriction \mathsf{Clp}(X), T^{-1})$$

Measure algebra with operator

Definition

A *measure algebra with operator* (MAO) is a structure $(A, m, \Diamond)$ where

- A is a countable Boolean algebra,
- $m : A \to [0,1]$ such that $m(1) = 1$ and $a \wedge b = 0 \Rightarrow m(a \vee b) = m(a) + m(b)$
- $\Diamond : A \to A$ is a Boolean algebra homomorphism.

If $\mathfrak{X} = (X, \tau, \mu, T)$ is a system, get MAO

$$\mathsf{L}(\mathfrak{X}) := (\mathsf{Clp}(X), \mu \restriction \mathsf{Clp}(X), T^{-1})$$

The logic

- The logic of just the $\Diamond$ (i.e., without the measure):
  - As modal logic: functional modal logic aka $\mathsf{KD!}$, KDDc, $\mathsf{DAlt1}$ [Sta18]
  - As linear temporal logic: the next-fragment [GMS24]
- The logic with the measure but non-deterministic $\Diamond$:
  - Markovian logics [KMP13; Koz+13]
  - Reasoning about knowledge and probability [FH94]
  - Modal logic of $\mathcal{D}$-coalgebras [KP11]
- Further logics of dynamical systems:
  - [Lei05], [KM07], [Pla12], [Fer12], …

The logic

- The logic of just the $\Diamond$ (i.e., without the measure):
  - As modal logic: functional modal logic aka KD!, KDDc, DAlt1 [Sta18]
  - As linear temporal logic: the next-fragment [GMS24]
- The logic with the measure but non-deterministic $\Diamond$:
  - Markovian logics [KMP13; Koz+13]
  - Reasoning about knowledge and probability [FH94]
  - Modal logic of $\mathcal{D}$-coalgebras [KP11]
- Further logics of dynamical systems:
  - [Lei05], [KM07], [Pla12], [Fer12], …

The logic

- The logic of just the $\Diamond$ (i.e., without the measure):
  - As modal logic: functional modal logic aka $\mathsf{KD!}$, KDDc, $\mathsf{DAlt1}$ [Sta18]
  - As linear temporal logic: the next-fragment [GMS24]
- The logic with the measure but non-deterministic $\Diamond$:
  - Markovian logics [KMP13; Koz+13]
  - Reasoning about knowledge and probability [FH94]
  - Modal logic of $\mathcal{D}$-coalgebras [KP11]
- Further logics of dynamical systems:
  - [Lei05], [KM07], [Pla12], [Fer12], …

Category of MAOs

Definition

A MAO-homomorphism $h : (A, m, \Diamond) \to (A', m', \Diamond')$ is a Boolean algebra homomorphism $h : A \to A'$ such that

- $m'(h(a)) = m(a)$
- $\Diamond'(h(a)) = h(\Diamond(a))$

Definition

Let MAO be the category of measure algebras with operator and their homomorphisms.

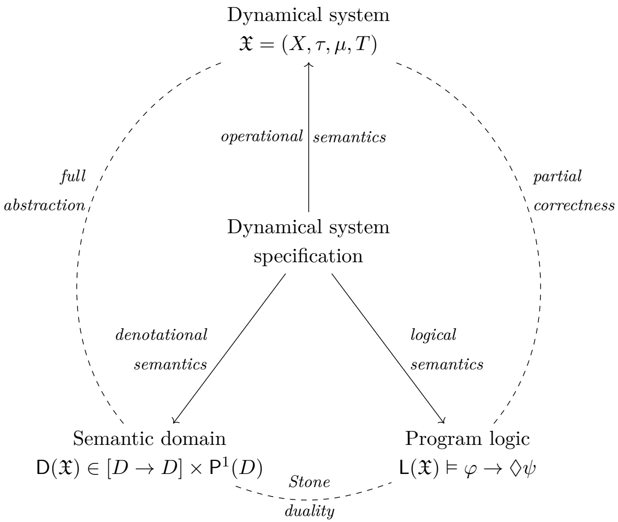
Equivalence: partial correctness

Corollary

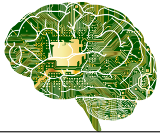
Stone duality (plus Carathéodory) extends to equivalence

$$\begin{tikzcd} \mathsf{DS} \arrow[rr, bend left, "\mathsf{L}"] \arrow[rr, phantom, "\wr"] & & \mathsf{MAO}^{\mathsf{op}} \arrow[ll, bend left, "\mathsf{I}"] \end{tikzcd}$$

Summary I


Dynamical system
$\mathfrak{X} = (X, \tau, \mu, T)$
operational semantics
full abstraction
partial correctness
Dynamical system specification
denotational semantics
logical semantics
Semantic domain
$\mathsf{D}(\mathfrak{X}) \in [D \to D] \times \mathsf{P}^1(D)$
Program logic
$\mathsf{L}(\mathfrak{X}) \vDash \varphi \to \Diamond\psi$
Stone duality

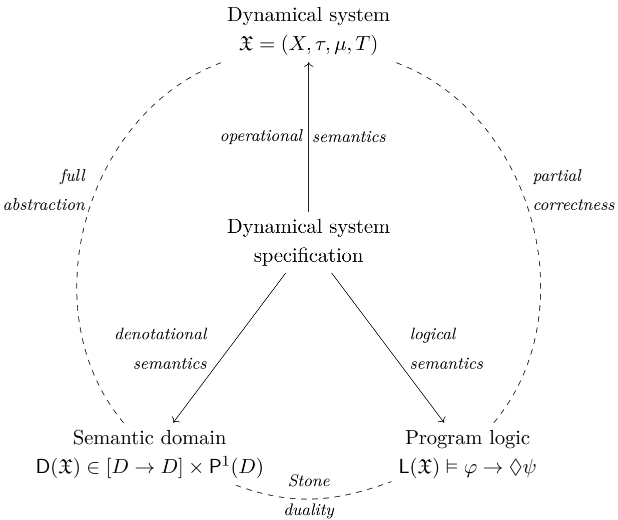
Summary II

		
Algorithm	Program P of type σ	Specification of dynamics
Operational semantics	Transition system $s \to s'$	Realization as dynamical system $\mathfrak{X} = (X, \tau, \mu, T)$
Denotational semantics	$[\![P]\!]$ an element of domain D_σ	$\mathsf{D}(\mathfrak{X})$ an element of $[D \to D] \times \mathsf{P}^1(D)$
Logical semantics	$\{\varphi\}P\{\psi\}$ Hoare triple	$\mathsf{L}(\mathfrak{X}) \vDash \varphi \to \Diamond\psi$ If φ is observed now, ψ is observed next.

Some further questions

- Metalanguage of machine learning reverse-engineered from category of domains? [Wei23]
- Computability theory for machine learning via effective domain theory?
- What does generalizing to Markov kernels on the logic side correspond to on the domain-theoretic side?

Thank you!


Dynamical system
$\mathfrak{X} = (X, \tau, \mu, T)$
operational semantics
full abstraction
partial correctness
Dynamical system specification
denotational semantics
logical semantics
Semantic domain
$\mathsf{D}(\mathfrak{X}) \in [D \to D] \times \mathsf{P}^1(D)$
Program logic
$\mathsf{L}(\mathfrak{X}) \vDash \varphi \to \Diamond\psi$
Stone duality

References I

[Abr91] S. Abramsky. "Domain theory in logical form". In: *Annals of pure and applied logic* 51.1-2 (1991), pp. 1–77.

[BP21] O. Bournez and A. Pouly. "A Survey on Analog Models of Computation". In: *Handbook of Computability and Complexity in Analysis*. Ed. by V. Brattka and P. Hertling. Cham: Springer International Publishing, 2021, pp. 173–226.

[Fer12] D. Fernández-Duque. "Dynamic Topological Logic of Metric Spaces". In: *The Journal of Symbolic Logic* 77.1 (2012), pp. 308–328.

[FH94] R. Fagin and J. Y. Halpern. "Reasoning about knowledge and probability". In: *Journal of the ACM (JACM)* 41.2 (1994), pp. 340–367.

[GMS24] S. van Gool, J. Marti, and M. Sweering. *Solving temporal equationsby folding de Bruijn graphs*. Workshop on Automata, Transducers, Logic, Algebras and Semigroups (ATLAS). 2024.

[Hoa69] C. A. R. Hoare. "An Axiomatic Basis for Computer Programming". In: *Communications of the ACM* 12.10 (1969), pp. 576–580.

[Jun13] A. Jung. "Continuous Domain Theory in Logical Form". In: *Computation, Logic, Games, and Quantum Foundations. The Many Facets of Samson Abramsky: Essays Dedicated to Samson Abramsky on the Occasion of His 60th Birthday*. Ed. by B. Coecke, L. Ong, and P. Panangaden. Berlin, Heidelberg: Springer, 2013, pp. 166–177.

[KM07] P. Kremer and G. Mints. "Dynamic Topological Logic". In: *Handbook of Spatial Logics*. Ed. by M. Aiello, I. Pratt-Hartmann, and J. van Benthem. Dordrecht: Springer, 2007, pp. 565–606.

References II

[KMP13] D. Kozen, R. Mardare, and P. Panangaden. "Strong completeness for Markovian logics". In: *Mathematical Foundations of Computer Science 2013*. Berlin Heidelberg: Springer, 2013.

[Koz+13] D. Kozen et al. "Stone duality for markov processes". In: *28th Annual ACM/IEEE Symposium on Logic in Computer Science*. 2013.

[KP11] C. Kupke and D. Pattinson. "Coalgebraic semantics of modal logics: An overview". In: *Theoretical Computer Science* 412.38 (2011). CMCS Tenth Anniversary Meeting, pp. 5070–5094.

[Lei05] H. Leitgeb. "Hodges' Theorem Does not Account for Determinacy of Translation. A Reply to Werning". In: *Erkenntnis* 62.3 (2005), pp. 411–425.

[Ong95] C.-H. L. Ong. "Correspondence between operational and denotational semantics: the full abstraction problem for PCF". In: *Handbook of Logic in Computer Science*. Ed. by S. Abramsky, D. M. Gabbay, and T. S. E. Maibaum. Vol. 4. Oxford: Clarendon Press, 1995, pp. 269–356.

[Pla12] A. Platzer. "Logics of Dynamical Systems". In: *Proceedings of the 2012 27th Annual IEEE/ACM Symposium on Logic in Computer Science*. LICS '12. IEEE Computer Society, 2012, pp. 13–24.

[SMG14] A. M. Saxe, J. L. McClelland, and S. Ganguli. *Exact solutions to the nonlinear dynamics of learning in deep linear neural networks*. 2014. arXiv: `1312.6120`.

[Sta18] S. Standefer. "Proof Theory for Functional Modal Logic". In: *Studia Logica* 106 (2018), pp. 49–84.

References III

[Wei23] P. Weigert. “A Program Calculus for Dynamical Systems”. MA thesis. University of Amsterdam, Institute for Logic, Language and Computation, 2023.

Image sources

- Personal computer: By Rama & Musée Bolo - CC BY-SA 2.0 fr, https://commons.wikimedia.org/w/index.php?curid=94784371
- Stylized brain: By GDJ - CC0, https://commons.wikimedia.org/w/index.php?curid=55392646
- Handwritten digits from the MNIST database: Excerpt from a picture by Josef Steppan - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=64810040
- Handwritten number 7 and 2: From MNIST database (first and second element of the test set, respectively).
- All others were created by the author.