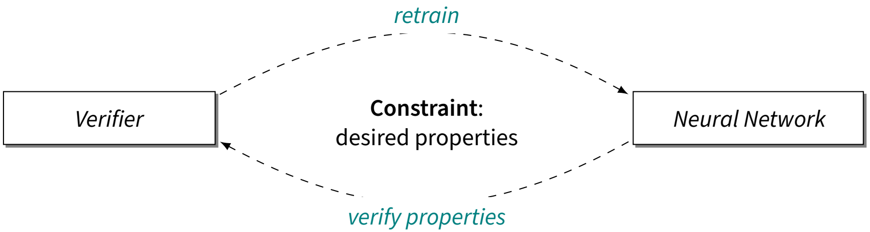
Logic of Differentiable Logics: how to implement property-driven training

Natalia Ślusarz Ekaterina Komendantskaya Matthew L. Daggitt Robert Stewart Kathrin Stark

October 27, 2023

Where do differentiable logics come in?

(and what are they?)



Motivation

Neural Networks

- A **neural network** is a function $f : \mathbb{R}^n \to \mathbb{R}^m$ parametrised by a set of weights $\mathbf{w}$
- A **training dataset** $\mathcal{X}$ is a set of pairs $(\mathbf{x}, \mathbf{y})$ consisting of an input $\mathbf{x} \in \mathbb{R}^n$ and the desired output $\mathbf{y} \in \mathbb{R}^m$
- Outputs $\mathbf{y}$ are generated from $\mathbf{x}$ by some function $\mathcal{H} : \mathbb{R}^n \to \mathbb{R}^m$
- $\mathbf{x}$ is drawn from some probability distribution over $\mathbb{R}^n$

Motivation

Neural Networks

- A **neural network** is a function $f : \mathbb{R}^n \to \mathbb{R}^m$ parametrised by a set of weights $\mathbf{w}$
- A **training dataset** $\mathcal{X}$ is a set of pairs $(\mathbf{x}, \mathbf{y})$ consisting of an input $\mathbf{x} \in \mathbb{R}^n$ and the desired output $\mathbf{y} \in \mathbb{R}^m$
- Outputs $\mathbf{y}$ are generated from $\mathbf{x}$ by some function $\mathcal{H} : \mathbb{R}^n \to \mathbb{R}^m$
- $\mathbf{x}$ is drawn from some probability distribution over $\mathbb{R}^n$

The goal of training is to use the dataset $\mathcal{X}$ to find weights $\mathbf{w}$ such that f approximates $\mathcal{H}$. This is done using a *loss function* $\mathcal{L}$, that given a pair $(\mathbf{x}, \mathbf{y})$ calculates how much $f(\mathbf{x})$ differs from the desired output $\mathbf{y}$.

Motivation

Neural Networks - Loss functions

A loss function $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ computes a penalty proportional to the difference between the output of f on a training input x and a desired output y.

Motivation

Neural Networks - Loss functions

A loss function $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}$ computes a penalty proportional to the difference between the output of f on a training input x and a desired output y.

Example (Cross-entropy loss)

The cross-entropy loss $\mathcal{L}_{ce}$ is defined as

$$\mathcal{L}_{ce}(\mathbf{x}, \mathbf{y}) = -\sum_{i=1}^{m} \mathbf{y}_i \log(f(\mathbf{x})_i) \quad (1)$$

Example - constraint

This is the type of proposition we could use:

Definition

Given constants $\epsilon, \delta \in \mathbb{R}$, a function f is ϵ-δ-*robust* around a point $\hat{\mathbf{x}} \in \mathbb{R}^n$ if:

$$\forall \mathbf{x} \in \mathbb{R}^n : ||\mathbf{x} - \hat{\mathbf{x}}|| \leq \epsilon \Rightarrow ||f(\mathbf{x}) - f(\hat{\mathbf{x}})|| \leq \delta$$

Example - differentiable logic

We define a very simple differentiable logic on a toy propositional language

$$a := a \,|p \ \leq \ p|\, a \wedge a \,|\, a \Rightarrow a$$

based on Godel fuzzy logic [van Krieken 2022].

$$\mathbf{T}(a_1 \leq a_2) := 1 - \max\left(\tanh\left|\mathbf{T}(a_1) - \mathbf{T}(a_2)\right|, 0\right)$$

$$\mathbf{T}(a_1 \wedge a_2) := \min\left(\mathbf{T}(a_1), \mathbf{T}(a_2)\right)$$

$$\mathbf{T}(a_1 \Rightarrow a_2) := \max\left(1 - \mathbf{T}(a_1), (\mathbf{T})\right)$$

Example - translation

$$\mathbf{T}(\forall \mathbf{x} \in \mathbb{R}^n : ||\mathbf{x} - \hat{\mathbf{x}}|| \leq \epsilon \Rightarrow ||f(\mathbf{x}) - f(\hat{\mathbf{x}})|| \leq \delta)$$

Example - translation

$$\mathbf{T}(\forall \mathbf{x} \in \mathbb{R}^n : ||\mathbf{x} - \hat{\mathbf{x}}|| \leq \epsilon \Rightarrow ||f(\mathbf{x}) - f(\hat{\mathbf{x}})|| \leq \delta)$$

$$\mathbf{T}(||\mathbf{x} - \hat{\mathbf{x}}|| \leq \epsilon \Rightarrow ||f(\mathbf{x}) - f(\hat{\mathbf{x}})|| \leq \delta)$$

Example - translation

$$\mathbf{T}(\forall \mathbf{x} \in \mathbb{R}^n : ||\mathbf{x} - \hat{\mathbf{x}}|| \leq \epsilon \Rightarrow ||f(\mathbf{x}) - f(\hat{\mathbf{x}})|| \leq \delta)$$

$$\mathbf{T}(||\mathbf{x} - \hat{\mathbf{x}}|| \leq \epsilon \Rightarrow ||f(\mathbf{x}) - f(\hat{\mathbf{x}})|| \leq \delta)$$

$$\max(1 - \mathbf{T}(||\mathbf{x} - \hat{\mathbf{x}}|| \leq \epsilon), ||\mathbf{T}(f(\mathbf{x}) - f(\hat{\mathbf{x}})|| \leq \delta))$$

Example - translation

$$\mathbf{T}(\forall \mathbf{x} \in \mathbb{R}^n : ||\mathbf{x} - \hat{\mathbf{x}}|| \leq \epsilon \Rightarrow ||f(\mathbf{x}) - f(\hat{\mathbf{x}})|| \leq \delta)$$

$$\mathbf{T}(||\mathbf{x} - \hat{\mathbf{x}}|| \leq \epsilon \Rightarrow ||f(\mathbf{x}) - f(\hat{\mathbf{x}})|| \leq \delta)$$

$$\max(1 - \mathbf{T}(||\mathbf{x} - \hat{\mathbf{x}}|| \leq \epsilon), ||\mathbf{T}(f(\mathbf{x}) - f(\hat{\mathbf{x}})|| \leq \delta))$$

$$\max(\max(\tanh|\mathbf{T}(||\mathbf{x} - \hat{\mathbf{x}}|| - \epsilon|, 0), 1 - \max(\tanh|\mathbf{T}(||f(\mathbf{x}) - f(\hat{\mathbf{x}})|| - \delta|, 0)$$

Different existing DLs

- DL2 [Fisher et al. 2019]
- fuzzy DLs such as: Godel, Łukasiewicz, Yager, product and others [van Krieken et al. 2022]
- Signal Temporal Logic based DL [Varnai et al. 2020]

Property language

This is only an example of the type of properties we may have to deal with. It underlined some of the key features that need to be present in the DL for it to handle such constraints such as:

- vectors

Example (Robustness)

$$\forall \mathbf{x} \in \mathbb{R}^n : ||\mathbf{x} - \hat{\mathbf{x}}|| \leq \epsilon \Rightarrow ||f(\mathbf{x}) - f(\hat{\mathbf{x}})|| \leq \delta$$

Property language

This is only an example of the type of properties we may have to deal with. It underlined some of the key features that need to be present in the DL for it to handle such constraints such as:

- vectors
- quantifiers

Example (Robustness)

$$\forall \mathbf{x} \in \mathbb{R}^n : ||\mathbf{x} - \hat{\mathbf{x}}|| \leq \epsilon \Rightarrow ||f(\mathbf{x}) - f(\hat{\mathbf{x}})|| \leq \delta$$

Property language

This is only an example of the type of properties we may have to deal with. It underlined some of the key features that need to be present in the DL for it to handle such constraints such as:

- vectors
- quantifiers
- random variables

Example (Robustness)

$$\forall \mathbf{x} \in \mathbb{R}^n : ||\mathbf{x} - \hat{\mathbf{x}}|| \leq \epsilon \Rightarrow ||f(\mathbf{x}) - f(\hat{\mathbf{x}})|| \leq \delta$$

Summary of the problems

1. Expressivity

Summary of the problems

1. Expressivity
2. Quantifiers (and reasoning about random variables)

Summary of the problems

1. Expressivity
2. Quantifiers (and reasoning about random variables)
3. Effect of DLs

Summary of the problems

1. Expressivity
2. Quantifiers (and reasoning about random variables)
3. Effect of DLs
4. Modular implementation

Summary of the problems

1. Expressivity
2. Quantifiers (and reasoning about random variables)
3. Effect of DLs
4. Modular implementation

What do we want then?

1. Well-separated syntax and semantics.

What do we want then?

1. Well-separated syntax and semantics.
2. DL general enough to cover all elements of syntax needed for machine learning problems (such as vectors, quantifiers, random variables)

What do we want then?

1. Well-separated syntax and semantics.
2. DL general enough to cover all elements of syntax needed for machine learning problems (such as vectors, quantifiers, random variables)
3. Language in which we can formalise and compare the DLs.

Logic of Differentiable Logics (LDL)

Syntax

LDL - Syntax

$\langle expr \rangle \ni e \ ::= x \mid f \mid r \in \mathbb{R} \mid i \in \mathbb{N} \mid b \in \mathbb{B}$

$\mid \quad e \ e$

$\mid \quad \text{lam } (x : \tau) \ . \ e$

$\mid \quad \text{let } (x : \tau) = e \text{ in } e$

$\mid \quad \wedge \mid \vee \mid \neg \mid \Rightarrow \mid + \mid - \mid \times$

$\mid \quad \neq \mid \leq \mid \geq \mid < \mid > \mid ==$

$\mid \quad [e, \ldots, e] \mid !$

$\mid \quad \forall (x : \tau) \ . \ e \mid \exists (x : \tau) \ . \ e$

$\langle type \rangle \ni \tau \ ::= s \rightarrow \tau \mid \text{s}$

$\langle simple\ type \rangle \ni s \ ::=$

$\mid \quad \text{Bool}$

$\mid \quad \text{Real}$

$\mid \quad \text{Vec } n \mid \text{Index } n \qquad \text{for } n \in \mathbb{N}$

Syntax - Example

$$\forall \mathbf{x} \in \mathbb{R}^n : ||\mathbf{x} - \hat{\mathbf{x}}|| \leq \epsilon \Rightarrow ||f(\mathbf{x}) - f(\hat{\mathbf{x}})|| \leq \delta$$

Example (Encoding of robustness property in LDL)

Assuming a network f with input of size 784 (28×28 pixel images).

$$
\begin{aligned}
e_{robust} = \quad & \mathsf{let}\ (bounded : \mathsf{Vec}\ 784 \rightarrow \mathsf{Vec}\ 784 \rightarrow \mathsf{Real} \rightarrow \mathsf{Bool}) \ = \\
& \quad \mathsf{lam}\ (v : \mathsf{Vec}\ 784)\ .\ \mathsf{lam}\ (u : \mathsf{Vec}\ 784)\ .\ \mathsf{lam}\ (a : \mathsf{Real})\ . \\
& \quad\quad \forall\ (i : \mathsf{Index}\ 784)\ .\ \mathsf{let}\ (d : \mathsf{Real}) = v\ !\ i - u\ !\ i\ \mathrm{in} - a \leq d\ \wedge\ d \leq a \\
& \mathrm{in} \\
& \quad \mathsf{lam}\ (\epsilon : \mathsf{Real})\ .\ \mathsf{lam}\ (\delta : \mathsf{Real})\ .\ \mathsf{lam}\ (\hat{x} : \mathsf{Vec}\ 784)\ . \\
& \quad\quad \forall\ (x : \mathsf{Vec}\ 784)\ .\ (bounded\ x\ \hat{x}\ \epsilon) \ \Rightarrow\ (bounded\ (f\ x)\ (f\ \hat{x})\ \delta)
\end{aligned}
$$

LDL - Syntax

Typing Context

- A *bound variable context*, Δ
- A *network context*, Ξ

We can now move on to the typing relation to see how they are relevant.

LDL - Typing relation

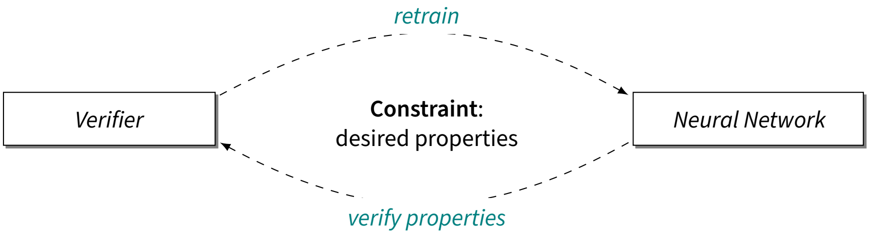
$$\frac{\Xi[f] = (m, n)}{\Xi, \Delta \Vdash f : \text{Vec}\ m \rightarrow \text{Vec}\ n}\ (\textit{networkVar}) \qquad \frac{\Delta[x] = \tau}{\Xi, \Delta \Vdash x : \tau}\ (\textit{boundVar})$$

$$\frac{}{\Xi, \Delta \Vdash \wedge, \vee, \Rightarrow : \text{Bool} \rightarrow \text{Bool} \rightarrow \text{Bool}}\ (\textit{and})(\textit{or})(\textit{implies}) \qquad \frac{}{\Xi, \Delta \Vdash +, \times\ : \text{Real} \rightarrow \text{Real} \rightarrow \text{Real}}\ (\textit{add})(\textit{mul})$$

$$\frac{\Xi, \Delta \Vdash e_1 : \text{Real} \quad \ldots \quad \Xi, \Delta \Vdash e_n : \text{Real}}{\Xi, \Delta \Vdash [e_1, ..., e_n] : \text{Vec}\ n}\ (\textit{vec}) \qquad \frac{}{\Xi, \Delta \Vdash\ !\ : \text{Vec}\ n \rightarrow \text{Index}\ n \rightarrow \text{Real}}\ (\textit{lookup})$$

$$\frac{\Xi, \Delta[x \rightarrow \tau] \Vdash e : \text{Bool} \quad \tau \not\equiv \tau_1 \rightarrow \tau_2}{\Xi, \Delta \Vdash \forall (x : \tau)\ .\ e : \text{Bool}}\ (\textit{forall}) \qquad \frac{\Xi, \Delta[x \rightarrow \tau] \Vdash e : \text{Bool} \quad \tau \not\equiv \tau_1 \rightarrow \tau_2}{\Xi, \Delta \Vdash \exists (x : \tau)\ .\ e : \text{Bool}}\ (\textit{exists})$$

A REMINDER


retrain
Verifier
Constraint:
desired properties
Neural Network
verify properties

Semantics

Semantics - overview

Modular semantics

We have semantics **modular** on the choice of DL - part that is indeptndent of the DL and a part depending on DL.

Semantics - overview

Modular semantics

We have semantics **modular** on the choice of DL - part that is indeptndent of the DL and a part depending on DL.

What do we need in them?

- semantics of types

$$\langle\!\langle \text{Real} \rangle\!\rangle = \mathbb{R} \qquad \langle\!\langle \text{Vec } n \rangle\!\rangle = \langle\!\langle \text{Real} \rangle\!\rangle^n$$

$$\langle\!\langle \text{Index } n \rangle\!\rangle = \{0, \ldots, n-1\} \qquad \langle\!\langle \tau_1 \to \tau_2 \rangle\!\rangle = \langle\!\langle \tau_2 \rangle\!\rangle^{\langle\!\langle \tau_1 \rangle\!\rangle}$$

The semantics of Bool is dependant on the DL!

SEMANTICS - OVERVIEW

Modular semantics

We have semantics **modular** on the choice of DL - part that is independent of the DL and a part depending on DL.

What do we need in them?

- semantic context that allow us to work with neural networks (**network context**, **bound variable context** and **quantifier context**)

$$[\![e]\!]_L^{N,\Gamma,Q}$$

Semantic Context

- A **semantic network context** N is a function that maps each network variable $f \in \Xi$, such that $\Xi[f] = (m, n)$, to a function $\mathtt{f} : \mathbb{R}^m \rightarrow \mathbb{R}^n$, the actual (external) implementation of the neural network.
- A **semantic bound context** Γ is a partial function that assigns each bound variable $x \in \Delta$ a semantic value in $\langle\langle \Delta(x) \rangle\rangle$.
- Let $q(e)$ be the set of infinitely quantified syntactic variables within expression e. A **semantic quantifier context** Q is a function that maps each variable x in $q(e)$ to a random variable X, from which values for x are sampled from.

Semantics of LDL independent of the choice of DL.

$$[\![x]\!]_L^{N,Q,\Gamma} = \Gamma[x] \qquad [\![f]\!]_L^{N,Q,\Gamma} = N[f] \qquad [\![r]\!]_L^{N,Q,\Gamma} = r \qquad [\![i]\!]_L^{N,Q,\Gamma} = i$$

$$[\![\text{lam } (x : \tau) \,.\, e]\!]_L^{N,Q,\Gamma} = \lambda(y : \langle\!\langle\tau\rangle\!\rangle).[\![e]\!]_L^{N,Q,\Gamma[x\rightarrow y]} \qquad [\![e_1 \; e_2]\!]_L^{N,Q,\Gamma} = [\![e_1]\!]_L^{N,Q,\Gamma}([\![e_2]\!]_L^{N,Q,\Gamma})$$

$$[\![\text{let } (x : \tau) = e_1 \text{ in } e_2]\!]_L^{N,Q,\Gamma} = [\![e_2]\!]_L^{N,Q,\Gamma[x\rightarrow [\![e_1]\!]_L^{N,Q,\Gamma}]} \qquad [\![[e_1, ..., e_n]]\!]_L^{N,Q,\Gamma} = < [\![e_1]\!]_L^{N,Q,\Gamma}, \ldots, [\![e_n]\!]_L^{N,Q,\Gamma} >$$

$$[\![!]\!]_L^{N,Q,\Gamma} = \lambda(a_1 : \langle\!\langle\text{Vec } n\rangle\!\rangle), (a_2 : \langle\!\langle\text{Index } n\rangle\!\rangle) \,.a_{1_{a_2}}$$

$$[\![+]\!]_L^{N,Q,\Gamma} = \lambda(a_1, a_2 : \langle\!\langle\text{Real}\rangle\!\rangle) \,.a_1 + a_2 \qquad [\![\times]\!]_L^{N,Q,\Gamma} = \lambda(a_1, a_2 : \langle\!\langle\text{Real}\rangle\!\rangle) \,.a_1 \times a_2$$

Semantics dependent on the choice of DL.

Syntax	Gödel interpretation
$\langle\langle \text{Bool} \rangle\rangle_G$	$[0, 1]$
$[\![\top]\!]_G$	1
$[\![\bot]\!]_G$	0
$[\![\neg]\!]_G$	$\lambda a\,.1 - a$
$[\![\wedge]\!]_G$	$\lambda a_1, a_2\,.\,\min(a_1, a_2)$
$[\![\vee]\!]_G$	$\lambda a_1, a_2\,.\,\max(a_1, a_2)$
$[\![\Rightarrow]\!]_G$	$\lambda a_1, a_2\,.\,\max(1 - a_1, a_2)$
$[\![==]\!]_G$	$\lambda a_1, a_2 .1 - \tanh \lvert a_1 - a_2 \rvert$
$[\![\leq]\!]_G$	$\lambda a_1, a_2\,.1 - \max(\tanh \lvert a_1 - a_2 \rvert, 0)$

Random Variables, Quantifiers and Vectors

Quantifier semantics - background

A random variable X **ranges over some domain** D

Given a continuous random variable X with domain D, *the probability distribution function (PDF)* $p_X : D \rightarrow [0,1]$ is a function that satisfies: $\int_D p_X(x)dx = 1$.

The probability distribution p_X for X characterises how probable it is for a sample from it to take a given value in the domain D.

Random vector variables X range over the domain D $= D_1 \times \ldots \times D_n$

Each element of the vector is assumed to be drawn from a random variable X_i over domain D_i. The *joint PDF* is a function $p_{X_1,\ldots,X_n} : D_1 \times \ldots \times D_n \rightarrow [0,1]$ that satisfies:

$$\int_{D_1} \ldots \int_{D_n} p_{X_1,\ldots,X_n}(x_1,\ldots,x_n)dx_n \ldots dx_1 = 1$$

Quantifier semantics - background

Given a function $g : \mathbb{R}^n \rightarrow \mathbb{R}$, a probability distribution $p_{\mathbf{X}}$ over the random variable $\mathbf{X}$ with domain $\mathbf{D}$:

$\mathbb{E}[g(\mathbf{X})]$, the *expected value* for $g : \mathbb{R}^n \rightarrow \mathbb{R}$ over the random variable $\mathbf{X}$, is defined as:

$$\mathbb{E}[g(\mathbf{X})] = \int_{\mathbf{D}} p_{\mathbf{X}}(\mathbf{x}) g(\mathbf{x}) d\mathbf{x}. \tag{2}$$

Quantifier semantics

For a function $g : \mathbb{R}^n \rightarrow \mathbb{R}$, we say that $\mathbf{x}_{\text{min}}$ is the *global minimum* if $g(\mathbf{x}_{\text{min}}) \leq g(\mathbf{y})$ for any $\mathbf{y}$ on which g is defined. We define a γ-ball around a point $\mathbf{x}$ as follows: $\mathbb{B}^\gamma_{\mathbf{x}} = \{\mathbf{y} \mid ||\mathbf{x} - \mathbf{y}|| \leq \gamma\}$. We call the expectation

$$\mathbb{E}_{\text{min}}[g(\mathbf{X})] = \lim_{\gamma \rightarrow 0} \int_{\mathbf{x} \in \mathbb{B}^\gamma_{\mathbf{x}_{\text{min}}}} p_{\mathbf{X}}(\mathbf{x}) g(\mathbf{x}) d\mathbf{x}$$

minimised expected value for g (over the random variable $\mathbf{X}$*).*

Quantifier semantics

This gives us interpretation of universal and existential quantifiers as minimised (or maximised) expected values for the interpretation of their body:

$$[\![\forall x : \tau .\ e]\!]_L^{N,Q,\Gamma} = \mathbb{E}_{\mathrm{min}}[(\lambda y.[\![e]\!]_L^{N,Q,\Gamma[x \to y]})(Q[x])]$$

$$[\![\exists x : \tau .\ e]\!]_L^{N,Q,\Gamma} = \mathbb{E}_{\mathrm{max}}[(\lambda y.[\![e]\!]_L^{N,Q,\Gamma[x \to y]})(Q[x])]$$

Example - robustness

Robustness:

$$\forall \mathbf{x} \in \mathbb{R}^n : ||\mathbf{x} - \hat{\mathbf{x}}|| \leq \epsilon \Rightarrow ||f(\mathbf{x}) - f(\hat{\mathbf{x}})|| \leq \delta$$

In LDL syntax:

$$[\![\text{lam } (\epsilon : \text{Real}) \text{ . lam } (\delta : \text{Real}) \text{ . lam } (\hat{x} : \text{Vec } 784) \text{ .}$$

$$\forall\ (x : \text{Vec } 784) \text{ . } (\textit{bounded}\ x\ \hat{x}\ \epsilon) \ \Rightarrow\ (\textit{bounded}\ (f\ x)\ (f\ \hat{x})\ \delta)]\!]_G$$

Let us now show the process of interpreting a property using LDL. For simplicity instead of defining the function *bounded* in a let expression we will assume it is already defined.

Example - robustness

Before:

$$[\![\text{lam } (\epsilon : \text{Real}) \text{ . lam } (\delta : \text{Real}) \text{ . lam } (\hat{x} : \text{Vec } 784) \text{ .}$$

$$\forall\, (x : \text{Vec } 784) \text{ . } (\mathit{bounded}\; x\; \hat{x}\; \epsilon) \;\Rightarrow\; (\mathit{bounded}\; (f\; x)\; (f\; \hat{x})\; \delta)]\!]_G$$

Now:

$$\lambda\, (\epsilon : \mathbb{R}) \text{ . } \lambda\, (\delta : \mathbb{R}) \text{ . } \lambda\, (\hat{x} : \mathbb{R}^{784}) \text{ .}$$

$$[\![\forall\, (x : \text{Vec } 784) \text{ . } (\mathit{bounded}\; x\; \hat{x}\; \epsilon) \;\Rightarrow\; (\mathit{bounded}\; (f\; x)\; (f\; \hat{x})\; \delta)]\!]_G$$

Example - robustness

Before:

$$\lambda\,(\epsilon : \mathbb{R})\,.\,\lambda\,(\delta : \mathbb{R})\,.\,\lambda\,(\hat{x} : \mathbb{R}^{784})\,.$$

$$[\![\forall\,(x : \text{Vec } 784)\,.\,(\textit{bounded}\; x\; \hat{x}\; \epsilon) \;\Rightarrow\; (\textit{bounded}\;(f\;x)\;(f\;\hat{x})\;\delta)]\!]_G$$

Now:

$$\lambda\,(\epsilon : \mathbb{R})\,.\,\lambda\,(\delta : \mathbb{R})\,.\,\lambda\,(\hat{x} : \mathbb{R}^{784})\,.$$

$$\mathbb{E}_{min}(\,[\lambda x : \mathbb{R}^{784}\,.[\![(\textit{bounded}\; x\; \hat{x}\; \epsilon) \;\Rightarrow\; (\textit{bounded}\;(f\;x)\;(f\;\hat{x})\;\delta)]\!]_G)(Q[x])]$$

Example - robustness

Before:

$$\lambda\,(\epsilon : \mathbb{R})\,.\,\lambda\,(\delta : \mathbb{R})\,.\,\lambda\,(\hat{x} : \mathbb{R}^{784})\,.$$
$$\mathbb{E}_{min}\,[(\lambda x : \mathbb{R}^{784}\,.[\![(bounded\; x\; \hat{x}\; \epsilon) \Rightarrow (bounded\; (f\; x)\; (f\; \hat{x})\; \delta)]\!]_G)(Q[x])]$$

Now:

$$\lambda\,(\epsilon : \mathbb{R})\,.\,\lambda\,(\delta : \mathbb{R})\,.\,\lambda\,(\hat{x} : \mathbb{R}^{784})\,.$$
$$\mathbb{E}_{min}\,[(\lambda x : \mathbb{R}^{784}\,.\,\max(1 - [\![bounded\; x\; \hat{x}\; \epsilon]\!]_G, [\![bounded\; (f\; x)\; (f\; \hat{x})\; \delta]\!]_G))(Q[x])]$$

Example - robustness

Before:

$$\lambda\,(\epsilon : \mathbb{R})\,.\,\lambda\,(\delta : \mathbb{R})\,.\,\lambda\,(\hat{x} : \mathbb{R}^{784})\,.$$

$$\mathbb{E}_{min}\,[(\lambda x : \mathbb{R}^{784}\,.[\![(bounded\ x\ \hat{x}\ \epsilon) \Rightarrow (bounded\ (f\ x)\ (f\ \hat{x})\ \delta)]\!]_G)(Q[x])]$$

Now:

$$\lambda\,(\epsilon : \mathbb{R})\,.\,\lambda\,(\delta : \mathbb{R})\,.\,\lambda\,(\hat{x} : \mathbb{R}^{784})\,.$$

$$\mathbb{E}_{min}\,[(\lambda x : \mathbb{R}^{784}\,.\ \max(1 - [\![bounded\ x\ \hat{x}\ \epsilon]\!]_G, [\![bounded\ (f\ x)\ (f\ \hat{x})\ \delta]\!]_G))(Q[x])]$$

The next step would be to interpret the *bounded* function analogously.

Properties

Motivation

Now that we have a way to express all the DLs we can **reason about their properties** - both logical and geometric.

The most important logical properties are **soundness** and ~~**completness**~~.

Motivation

Now that we have a way to express all the DLs we can **reason about their properties** - both logical and geometric.

The most important logical properties are **soundness** and ~~**completness**~~.

Soundness

We will say that a DL is *sound*, if given a formula that it interprets as $[\![\top]\!]_L^{N,Q,\Gamma}$ this formula is provable in **LJ**[Gentzen 1969].

Motivation

Now that we have a way to express all the DLs we can **reason about their properties** - both logical and geometric.

The most important logical properties are soundness and ~~completness~~.

Soundness

We will say that a DL is *sound*, if given a formula that it interprets as $[\![\top]\!]_L^{N,Q,\Gamma}$ this formula is provable in **LJ**[Gentzen 1969].

Theorem (Soundness of Gödel DL)

Given a formula e, for any contexts N, Γ, Q *if* $[\![e]\!]_G^{N,Q,\Gamma} = [\![\top]\!]_G^{N,Q,\Gamma}$ *then* $\vdash e$.

What else is in the paper?

(but did not fit here)

- more DLs expressible in LDL - and the analysis of the differences between them

What else is in the paper?

(but did not fit here)

- more DLs expressible in LDL - and the analysis of the differences between them
- comparison of geometric and logical properties of all the LDLs

What else is in the paper?

(but did not fit here)

- more DLs expressible in LDL - and the analysis of the differences between them
- comparison of geometric and logical properties of all the LDLs
- proofs of soundness (or lack thereof) for other DLs

Comparison

Properties:	DL2	Gödel	Łukasiewicz	Yager	Product	STL
Weak Smoothness	yes*	no	no	no	yes*	**yes**
Shadow-lifting	yes	no	no	no	yes	**yes**
Scale invariance	yes	yes	no	no	no	**yes**
Idempotence	no	**yes**	**no**	**no**	**no**	**yes**
Commutativity	yes	**yes**	**yes**	**yes**	**yes**	**yes**
Associativity	yes	**yes**	**yes**	**yes**	**yes**	**no**
Quantifier commutativity	no	yes	no	no	no	no
Soundness	yes	yes	no	no	yes	no

Conclusions

Logic for Differentiable Logics

We created a meta-DL - **Logic for Differentiable Logics** (LDL).

Logic for Differentiable Logics

We created a meta-DL - **Logic for Differentiable Logics** (LDL).

- On the syntactic side it is a **typed first-order language** with negation and universal and existential quantification that can express properties of functions and vectors.

Logic for Differentiable Logics

We created a meta-DL - **Logic for Differentiable Logics** (LDL).

- On the syntactic side it is a **typed first-order language** with negation and universal and existential quantification that can express properties of functions and vectors.
- On the semantic side LDL is defined to be parametric on the choice of domain and interpretation of logical connectives.

Logic for Differentiable Logics

We created a meta-DL - **Logic for Differentiable Logics** (LDL).

- On the syntactic side it is a **typed first-order language** with negation and universal and existential quantification that can express properties of functions and vectors.
- On the semantic side LDL is defined to be parametric on the choice of domain and interpretation of logical connectives.
- This structure will allow for a modular implementation of LDL as an extension of Vehicle [Kokke et al. 2023].

Future work

- Finding novel ways of defining quantifiers that commute with DL connectives.
- Investigating other ways of stating soundness that correspond better to DLs.
- Defining a new DL that has better properties - and is both adequate and complete (possibly using equality-up-to-epsilon or some ideas from Lawvere Quantale [Bacci et al. 2023])

Questions?