To my beloved

Benedetta

Introduction

Machine learning literature is exploding in size and complexity, but most solutions found are ad hoc, there is little communication between different subfields, and there is a large research debt. Category theory can solve these problems. [SGW].

Talk about the origins of category theory and its ”rise to power” as a common language that aims to unite different fields of knowledge.

Discuss the purpose of this work: a beginner-friendly survey of categorical approaches to neural networks, causal models, and interpretability.

Introduzione

Traduzione italiana dell’introduzione.

Contents

List of Figures

List of Tables

Chapter 1

Categorical Toolkit

Brief introduction to category theory and the categorical toolkit used in the following sections. As each kind of category is introduced we shall also introduce appropriate string diagrams.

1.1 Basics of Category Theory

Definition of category. Definition of functor. Definition of natural transformation. Definition of 2-category.

1.2 Various Families of Categories

1.2.1 Monoidal Categories

Definition of monoidal category. Definition of Cartesian category. Expand on the properties of Cartesian categories. In particular, expand on the existence of copy maps. Definition of left-additive category. Definition of Cartesian left-additive category.

1.2.2 Differential Categories

The recent rise in AI techniques was only possible because of advancements in automatic differentiation (AD) techniques. Differentiation in machine learning is usually carried out in Euclidean spaces $\mathbb{R}^n$, but is worth considering more abstract settings because the techniques used in gradient-based learning can be greatly generalized. For instance, [WZ] demonstrated that gradient-based learning can take place in the context of Boolean circuits.

We consider two abstract settings for differentiation: Cartesian differential categories (first introduced in [BSC]) and Cartesian reverse differential categories (first introduced by [CCG$^+$]). The former is a setting where forward derivatives of morphisms can be taken, while the latter is a setting where reverse derivatives can be taken. We shall only give intuitive definitions for the sake of brevity; rigorous definition which account of all the necessary axioms can be found in [CCG$^+$].

(Intuitive) Definition 1 (Cartesian differential category). *A Cartesian differential category (CDR) $\mathcal{C}$ is a Cartesian left-additive category where a differential combinator* D *is defined. Such differential combinator must take a morphism $f : A \to B$ and return a morphism* $\mathrm{D}[f] : A \times A \to B$*, which is known as the derivative of f. The combinator* D *must satisfy a number of axioms.*

(Intuitive) Definition 2 (Cartesian reverse differential category). *A Cartesian reverse differential category (CRDC) $\mathcal{C}$ is a Cartesian left-additive category where a reverse differential combinator* R *is defined. Such reverse differential combinator must take a morphism $f : A \to B$ and return a morphism* $\mathrm{R}[f] : A \times B \to A$*, which is known as the reverse derivative of f. The combinator* R *must satisfy a number of axioms.*

The aforementioned axioms make sure that R and D satisfy the properties expected from derivative and reverse derivative. In particular, a differential combinator satisfies a chain rule (see figure ADDIM), whereas a reverse differential combinator satisfies a reverse chain rule (see figure ADDIM).

Example 3. **Smooth** is a both a CDC and a CRDC. In fact, if $\mathcal{J}_f$ is the Jacobian matrix of a smooth morphism f,

$$\mathrm{D}[f] : (x, v) \mapsto \mathcal{J}_f(x)v$$

and

$$\mathrm{R}[f] : (x, y) \mapsto \mathcal{J}_f(x)^T y$$

induce well-defined combinators D and R. This is only a partial coincidence, as it is shown in [CCG$^+$] that CRDCs are always CDCs under a canonical choice of differential combinator. The converse, however, is generally false.

1.2.3 Lenses

Lenses are a mathematical construct used to model bidirectional flows of information[1]. Such flows are extremely important in machine learning as a machine learning model needs both to carry out a computation and to update its parameters based on the training data.

Definition 4 (Lenses)**.** *Let $\mathcal{C}$ be a Cartesian category. We define* **Lens**$(\mathcal{C})$ *as the category constituted by the following objects and morphisms. An object of* **Lens**$(\mathcal{C})$ *is a pair* $\begin{pmatrix} A \\ A' \end{pmatrix}$ *of objects in $\mathcal{C}$; A* $\begin{pmatrix} A \\ A' \end{pmatrix} \to \begin{pmatrix} B \\ B' \end{pmatrix}$ *morphism (or lens) is a pair* $\begin{pmatrix} f \\ f^* \end{pmatrix}$ *of morphisms of $\mathcal{C}$ such that $f : A \to B$ and $f^* : A \times A'$. f is known as get part of the lens* $\begin{pmatrix} f \\ f^* \end{pmatrix}$*, whereas f^* is known as get part. Given a pair* $\begin{pmatrix} A \\ A' \end{pmatrix}$*, the associated identical lens is* $\begin{pmatrix} 1_A \\ \pi_1 \end{pmatrix}$*. Lens composition is illustrated by ADDIM.*

Lenses can be represented using the language string diagrams (see [CGG$^+$]), both in compact form and in expanded form.

[1]The theory of optics generalizes lenses to a much wider family of constructs that model these same bidirectional flows. See [Ril].

It is important to note the following (see [CGG+]):

Proposition 5. *If $\mathcal{C}$ is a Cartesian category,* $\mathbf{Lens}(\mathcal{C})$ *is a monoidal category under the monoidal product* $\begin{pmatrix} A \\ A' \end{pmatrix} \otimes \begin{pmatrix} B \\ B' \end{pmatrix} = \begin{pmatrix} A \times B \\ A' \times B' \end{pmatrix}$.

Another important result from [CGG+] is (according the formulation found in [SGW]):

Proposition 6. *If $\mathcal{C}$ is a CRDC, there exists a canonical Cartesian left-additive functor $R_{\mathcal{C}}$ which embeds $\mathcal{C} \to \mathbf{Lens}(\mathcal{C})$. Such functor maps objects as* $A \mapsto \begin{pmatrix} A \\ A \end{pmatrix}$ *and maps morphisms as* $f \mapsto \begin{pmatrix} f \\ \mathrm{R}[f] \end{pmatrix}$.

Remark 7. Consider the put map of the composition of two lenses, as in figure ADDIM. As [SGW] notes, there is a striking resemblance between the form that such put map takes and the reverse chain rule that holds in a CRDC. This similarity will be important for modelling neural networks as lenses.

1.2.4 The Para Construction

In machine learning, it is often necessary to work with parameters. Although lenses can model bidirectional flow, they don't afford us the machinery needed to work with such parameters. The **Para** construction allows us to overcome such limit. For the sake of simplicity, we shall only examine the case where parameters and values are taken from the same category $\mathcal{C}$ (as in [CGG+]). A more general **Para** construction is described in [SGW], where actegories are used to handle a much wider choice of possible parameters.

(Intuitive) Definition 8 ($\mathbf{Para}(\mathcal{C})$)**.** *Let $(\mathcal{C}, I, \otimes)$ be a symmetric monoidal category. Then, we define* $\mathbf{Para}(\mathcal{C})$ *as the 2-category whose components are as follows.*

- *The 0-cells are the objects of $\mathcal{C}$.*

- *The* 1*-cells are pairs* $(P, f) : A \to B$*, where* $P \in \mathcal{C}$ *and* $f : P \otimes A \to B$*.*
- *The* 2*-cells come in the form* $r : (P, f) \to (Q, g)$*, where* $r : P \to Q$ *is a morphism in* $\mathcal{C}$*.* r *must also satisfy a naturality condition.*
- *The* 1*-cell identity on* A *in* $\mathbf{Para}(\mathcal{C})$ *is* $(I, 1_A)$*.*
- *The* 2*-cell identity on* (P, f) *in* $\mathbf{Para}(\mathcal{C})$ *is* 1_P*.*
- *The* 1*-cell composition law is*

$$(P, f); (Q, g) = (Q \otimes P, Q \otimes f; g.$$

- *The* 2*-cell composition law is the same as the* $\mathcal{C}$ *composition law.*

The intuition behind the definition above is the following: the 1 cells are parametric maps, whereas the 2 cells are reparametrisations. It is quite handy to represent such cells using the string diagram notation illustrated in figure ADDIM.

It is shown in [CGG$^+$] that $\mathbf{Para}(\mathcal{C})$ is natural with respect to $\mathcal{C}$. In other words, as emphasized in [SGW],

Proposition 9. *If* $\mathcal{C}$ *and* $\mathcal{D}$ *are symmetric monoidal categories and* $F : \mathcal{C} \to \mathcal{D}$ *is a symmetric monoidal functor, then there is a canonical* 2*-functor*

$$\mathbf{Para}(F) : \mathbf{Para}(\mathcal{C}) \to \mathbf{Para}(\mathcal{D}).$$

Chapter 2

Categorical Approaches to Neural Networks

Brief summary of the chapter.

2.1 Neural Networks as Parametric Lenses

Parametric lenses allow us to represent neural networks so that we can model both forward and backward propagation at the same time. This representation is compositional, and the learning iteration can also be represented [CGG+].

2.1.1 Neural Networks as Parametric Maps

2.1.2 Neural Networks as Parametric Lenses

2.1.3 Learning with Parametric Lenses

Show that loss functions and optimizers can be represented as parametric lenses. Show that such components can be combined to actually model learning. Show that the learning iteration itself can be handled like this [CGG+].

Parametric lenses can be successfully applied to software development [CGG$^+$].

Conclusions

Bibliography

[BSC] R.F. Blute, R.A.G. Seely, and J.R.B. Cockett. Differential Categories.

[CCG$^+$] Robin Cockett, Geoffrey Cruttwell, Jonathan Gallagher, Jean-Simon Pacaud Lemay, Benjamin MacAdam, Gordon Plotkin, and Dorette Pronk. Reverse derivative categories.

[CGG$^+$] Geoffrey S H Cruttwell, Bruno Gavranovic, Neil Ghani, Paul Wilson, and Fabio Zanasi. Deep Learning with Parametric Lenses.

[Ril] Mitchell Riley. Categories of Optics.

[SGW] Dan Shiebler, Bruno Gavranović, and Paul Wilson. Category Theory in Machine Learning.

[WZ] Paul Wilson and Fabio Zanasi. Reverse Derivative Ascent: A Categorical Approach to Learning Boolean Circuits. 333:247–260.

Acknowledgements

Placeholder