Generalized Gradient Descent is a Hypergraph Functor

Tyler Hanks James Fairbanks Matthew Klawonn
Department of Computer and Information Science and Engineering
University of Florida
Gainesville, Florida
Information Directorate
Air Force Research Lab
Rome, New York
t.hanks@ufl.edu fairbanksj@ufl.edu matthew.klawonn.2@us.af.mil

Cartesian reverse derivative categories (CRDCs) provide an axiomatic generalization of the reverse derivative, which allows generalized analogues of classic optimization algorithms such as gradient descent to be applied to a broad class of problems. In this paper, we show that generalized gradient descent with respect to a given CRDC induces a hypergraph functor from a hypergraph category of optimization problems to a hypergraph category of dynamical systems. The domain of this functor consists of objective functions that are 1) general in the sense that they are defined with respect to an arbitrary CRDC, and 2) open in that they are decorated spans that can be composed with other such objective functions via variable sharing. The codomain is specified analogously as a category of general and open dynamical systems for the underlying CRDC. We describe how the hypergraph functor induces a distributed optimization algorithm for arbitrary composite problems specified in the domain. To illustrate the kinds of problems our framework can model, we show that parameter sharing models in multitask learning, a prevalent machine learning paradigm, yield a composite optimization problem for a given choice of CRDC. We then apply the gradient descent functor to this composite problem and describe the resulting distributed gradient descent algorithm for training parameter sharing models.

1 Introduction

Recently, Cartesian reverse derivative categories (CRDCs) were used to define generalized analogues of classic optimization algorithms, such as gradient descent, that minimize generalized objective functions [8, 17]. This has allowed techniques from machine learning such as backpropagation to be applied not only to artificial neural networks, but to a broad class of models including boolean circuits. The generality of the framework makes future expansion to other problem types likely. Central to the definition of a CRDC is the Cartesian reverse derivative combinator **R**, which must obey axioms mirroring the behavior of the standard directional derivative in the Euclidean domain. This combinator is *compositional* in that it satisfies a chain rule, allowing one to define a general version of the backpropagation algorithm [13, 15].

In prior work, CRDCs have been used primarily to compose parameterized morphisms, i.e to build a learning model, all the while leveraging **R** to ensure the parameters can be updated with respect to the generalized gradient of an objective. In this way the induced optimization problem of maximizing/minimizing the given objective with respect to the parameters has a compositional structure by virtue of the *model* being compositional. We propose that in addition to compositional structure in the *model*, compositional structure in the *objective* is also of interest. In particular, in machine learning it is common for parameters to be optimized for more than one objective. Simple examples of compositional objectives are those arising from various regularization methods, for example ℓ_1 or ℓ_2 penalties [3], wherein one term in the objective incentivizes "good performance" on the given task while the other enforces some generically desirable property like sparseness. Other examples with more sophisticated composition patterns include parameter sharing methods wherein subsets of parameters may be optimized for

Submitted to:
ACT 2024

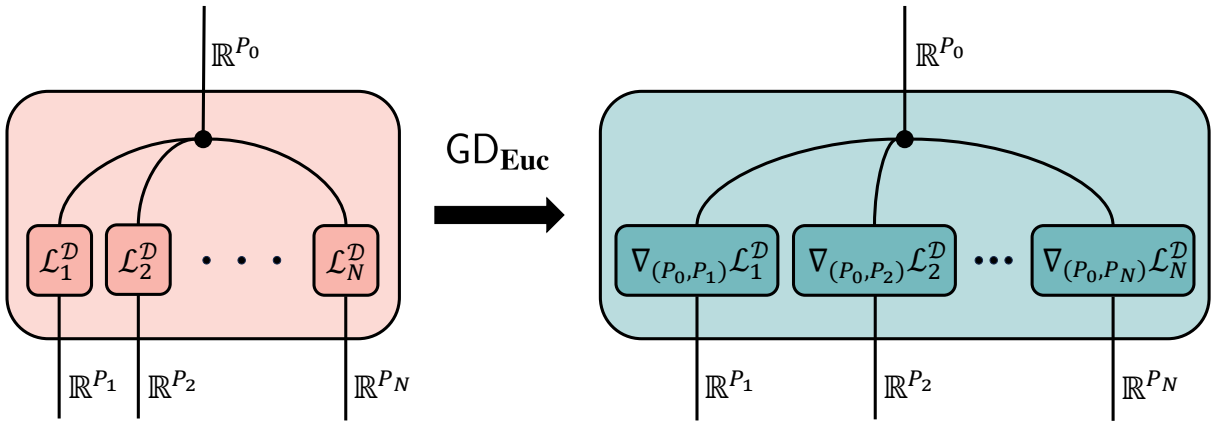


Figure 1: An illustration of multitask learning in our framework. The left string diagram shows the composite objective function representing the multitask learning objective. The right string diagram is the result of applying the gradient descent functor to the morphism on the left, resulting in a distributed optimization algorithm.

one or more objective functions. Parameter sharing arises within the setting of multitask learning, which we will discuss in this paper. Figure 1 illustrates this example.

To capture composition of objective functions that may have shared parameters, we turn to decorated spans [10]. Our approach closely follows that of [11], where algebras of undirected wiring diagrams are used to compose objective functions with shared decision variables, and a variety of first-order methods including (sub)gradient descent and primal-dual methods are shown to be functorial in the standard Euclidean domain. This paper extends the results on gradient descent to the general setting of Cartesian reverse derivative categories and generalized optimization introduced in [17]. Specifically, our contributions are as follows.

- We show how to produce hypergraph categories of generalized open objectives and optimizers over a given optimization domain.
- We prove that generalized gradient descent yields a hypergraph functor from open objectives to open optimizers.
- We highlight how functoriality allows the gradient descent solution algorithms for composite problems to be implemented in a distributed fashion.
- We show that multitask learning [4] with hard parameter sharing is an example of objective composition occurring in the machine learning literature, and we leverage our functorial formulation of gradient descent to recover a distributed optimization scheme.

We will begin by setting up the categories of open generalized objectives and open dynamical systems. We then specify the generalized gradient descent functor between them before turning to the multitask learning example.

2 Generalized Gradient Descent as a Hypergraph Functor

In order to proceed we need to recall many definitions from prior work. For convenience, they have been included in Appendix A. The reader who is unfamiliar with Cartesian reverse derivative categories, linear maps in CRDCs, optimization domains, and generalized gradients may wish to peruse the appendix before proceeding.

The fundamental component for generalized optimization is the notion of an *optimization domain*, which generalizes the role of the real numbers in defining the "cost" or "value" of a given configuration of decision variables. Specifically, an **optimization domain** consists of a pair $(\mathcal{C}, R)$ where $\mathcal{C}$ is a Cartesian reverse derivative category and $R \in \mathcal{C}$ has the structure of an ordered commutative ring (for full details, see Definition A.5). To construct hypergraph categories of objectives and optimizers, we will also require that the subcategory of linear maps in $\mathcal{C}$, denoted $\mathsf{Lin}(\mathcal{C})$, has finite limits. Going forward, we use $\times$ to denote the product and $*$ to denote the terminal object in an arbitrary CRDC.

2.1 The Hypergraph Category of Open Objectives

The goal of this section is to define a hypergraph category of generalized optimization problems. In standard optimization, a generic unconstrained minimization problem has the form

$$\text{minimize } f(x),$$

where $f\colon \mathbb{R}^n \to \mathbb{R}$ is known as the objective function, and $x \in \mathbb{R}^n$ is referred to as the decision or optimization variable. When $f(x)$ has the form $\sum_{i=1}^{N} f_i(x_i)$, where the x_i are disjoint subvectors of x, f is referred to as *separable*. Separable objectives can be solved entirely in parallel as each optimal x_i has no effect on the rest of the objective. However, in many important problems, the objective function is *partially separable*. For example, the problem

$$\text{minimize } f(w,x) + g(u,w,y) + h(u,w,z)$$

consists of a sum of objectives with *complicating* variables u and w. Such a sum of objectives with shared variables is the type of compositional structure that we wish to capture and generalize in our hypergraph category of optimization problems.

A standard method for defining such a hypergraph category is to leverage the theory of decorated spans[1] [9, 10]. Decorated spans turn a closed system into an open system by specifying a boundary for the system in addition to morphisms that relate the domain of the system to its boundary. We now leverage this general pattern to define a notion of *open objectives*.

Definition 2.1. Given an optimization domain $(\mathcal{C}, R)$, an **open objective** with domain boundary X and codomain boundary Y consists of a span $X \leftarrow N \rightarrow Y$ in $\mathsf{Lin}(\mathcal{C})$ together with a choice of objective on N, i.e., a map $f\colon N \to R$ in $\mathcal{C}$.

There is a straightforward interpretation of open objectives when the legs are both projections: the boundary objects are subobjects of the overall decision space which can be shared with other objectives. This intuition of sharing parts of the decision space is still helpful even when the legs are general linear maps. We now arrange open objectives into a hypergraph category by specifying the following decorating functor.

[1] The literature typically uses *covariant* decorating functors to define decorated cospan categories. However, all the same results hold for the dual construction of *contravariant* decorating functors and decorated spans, which we leverage in this paper.

Theorem 2.2. *Given an optimization domain $(\mathcal{C}, R)$, there is a contravariant lax symmetric monoidal functor $O^R_{\mathcal{C}}\colon (\mathsf{Lin}(\mathcal{C})^{\mathrm{op}}, \times) \to (\mathbf{Set}, \times)$ defined by the following maps (where we let $O \coloneqq O^R_{\mathcal{C}}$ for the remainder of the theorem and proof).*

- *Given an object X, $O(X)$ is the set of generalized objectives with domain X, i.e., the hom-set $\mathcal{C}(X, R)$.*
- *Given a linear map $\phi\colon X \to Y$ and objective $f\colon Y \to R$, $O(\phi)(f)\colon X \to R$ is the objective $f \circ \phi$ obtained by precomposition.*
- *Given objects X, Y, the product comparison $\varphi_{X,Y}\colon O(X) \times O(Y) \to O(X \times Y)$ is defined by*

$$\varphi_{X,Y}(f, g) \coloneqq f \circ \pi_0 + g \circ \pi_1, \tag{1}$$

  where $\pi_0\colon X \times Y \to X$ and $\pi_1\colon X \times Y \to Y$ are the natural projections.
- *The unit comparison $\varphi_0\colon \{*\} \to O(*)$ selects the additive identity 0_R for the hom-set $\mathcal{C}(*, R)$ supplied by the left additive structure.*

Proof. O is plainly functorial as it is just the contravariant hom functor $\mathcal{C}(-, R)$ restricted to act only on the linear maps of $\mathcal{C}$. We still need to verify the symmetric lax monoidal axioms. For naturality of the product comparison, we need the following diagram to commute for all objects X, X', Y, Y' and linear maps $\phi\colon X' \to X$ and $\psi\colon Y' \to Y$:

$$\begin{array}{ccc} O(X) \times O(Y) & \xrightarrow{O(\phi) \times O(\psi)} & O(X') \times O(Y') \\ {\scriptstyle \varphi_{X,Y}}\downarrow & & \downarrow{\scriptstyle \varphi_{X',Y'}} \\ O(X \times Y) & \xrightarrow[O(\phi \times \psi)]{} & O(X' \times Y') \end{array}$$

Letting $f\colon X \to R$ and $g\colon Y \to R$ be arbitrary, following the top path yields the morphism

$$f \circ \phi \circ \pi_0 + g \circ \psi \circ \pi_1 \tag{2}$$

while following the bottom path yields

$$(f \circ \pi_0 + g \circ \pi_1) \circ (\phi \times \psi) = f \circ \pi_0 \circ (\phi \times \psi) + g \circ \pi_1 \circ (\phi \times \psi), \tag{3}$$

where the equality comes from **LA.1**. Now let $x \in X'$ and $y \in Y'$ be generalized elements. Then applying (2) to (x, y) gives $f(\phi(x)) + g(\psi(y))$ while applying (3) to (x, y) gives

$$f(\pi_0(\phi(x), \psi(y))) + g(\pi_1(\phi(x), \psi(y))) = f(\phi(x)) + g(\psi(y)), \tag{4}$$

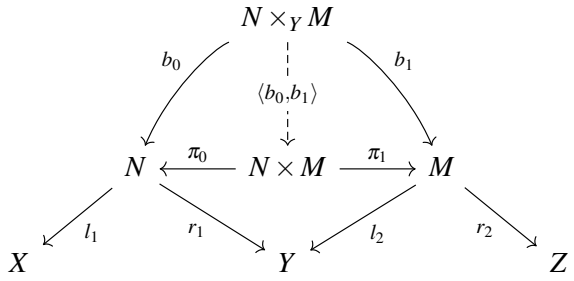
as desired. Commutativity of the symmetry, associativity, and unitality diagrams all follow from the fact that the hom-set $\mathcal{C}(X' \times Y', R)$ is a commutative monoid. □

We can now use our decorated spans of open objectives to construct a hypergraph category.

Corollary 2.2.1 (Open Generalized Optimization Problems)**.** *Given an optimization domain $(\mathcal{C}, R)$ such that $\mathsf{Lin}(\mathcal{C})$ has finite limits, there is a hypergraph category $\mathbf{Opt}^R_{\mathcal{C}}$ defined by the following data.*

- *Objects are the same as those of $\mathcal{C}$.*
- *Morphisms are (isomorphism classes of) open generalized objectives.*

- *Given two open objectives* $(X \xleftarrow{l_1} N \xrightarrow{r_1} Y, f \in O^R_{\mathcal{C}}(N))$ *and* $(Y \xleftarrow{l_2} M \xrightarrow{r_2} Z, g \in O^R_{\mathcal{C}}(M))$*, their composite consists of the following span computed by pullback*



(5)

together with the objective obtained from the composite morphism

$$\{*\} \cong \{*\} \times \{*\} \xrightarrow{f \times g} O^R_{\mathcal{C}}(N) \times O^R_{\mathcal{C}}(M) \xrightarrow{\varphi_{N,M}} O^R_{\mathcal{C}}(N \times M) \xrightarrow{O^R_{\mathcal{C}}\langle b_0, b_1 \rangle} O^R_{\mathcal{C}}(N \times_Y M). \tag{6}$$

- *The identity on* X *is the pair* $(X = X = X, \{*\} \xrightarrow{\varphi_0} O^R_{\mathcal{C}}(*) \xrightarrow{O^R_{\mathcal{C}}(!)} O^R_{\mathcal{C}}(X))$.
- *The monoidal product on objects is their product in* $\mathcal{C}$ *while the monoidal product of morphisms* $(X \leftarrow N \rightarrow Y, f)$ *and* $(W \leftarrow M \rightarrow Z, g)$ *is*

$$(X \times W \leftarrow N \times M \rightarrow Y \times Z, \varphi_{N,M}(f,g)). \tag{7}$$

- *The hypergraph maps are inherited from those of* $\mathsf{Span}(\mathsf{Lin}(\mathcal{C}))$ *together with the identity decoration.*

Proof. This follows by applying Proposition 3.2 and Theorem 3.4 in [9] to Theorem 2.2. □

2.2 The Hypergraph Category of Open Optimizers

Having defined a hypergraph category of open objective functions, we will now use the same machinery of decorated spans to define a hypergraph category of open dynamical systems over a given CRDC.

Definition 2.3. Given a CRDC $\mathcal{C}$, an **open dynamical system** with domain boundary X and codomain boundary Y consists of a span $X \leftarrow N \rightarrow Y$ in $\mathsf{Lin}(\mathcal{C})$ together with a choice of system on N, i.e., a map $v \colon N \rightarrow N$ in $\mathcal{C}$.

Endomaps in a CRDC are also referred to as optimizers in [17], thus we use the terms optimizer and dynamical system interchangeably. Similar to open objectives, the boundaries of open dynamical systems specify which parts of a system's state space can be influenced by other systems. The decorating functor is given as follows.

Theorem 2.4. *Given a CRDC* $\mathcal{C}$*, there is a contravariant lax symmetric monoidal functor* $D_{\mathcal{C}} \colon (\mathsf{Lin}(\mathcal{C})^{\mathrm{op}}, \times) \rightarrow (\mathbf{Set}, \times)$ *defined by the following maps (where we let* $D := D_{\mathcal{C}}$ *for the remainder of the theorem and proof).*

- *Given an object* X*,* $D(X)$ *is the set of generalized dynamical systems with state space* X*, i.e., the hom-set* $\mathcal{C}(X,X)$.
- *Given a linear map* $\phi \colon X \rightarrow Y$ *and a system* $v \colon Y \rightarrow Y$*,* $D(\phi)(v)$ *is the system* $\phi^\dagger \circ v \circ \phi \colon X \rightarrow X$.

- *Given objects X, Y, the product comparison $\varphi_{X,Y} : D(X) \times D(Y) \to D(X \times Y)$ is defined by*

$$\varphi_{X,Y}(v,w) \coloneqq \pi_0^\dagger \circ v \circ \pi_0 + \pi_1^\dagger \circ w \circ \pi_1. \tag{8}$$

- *The unit comparison $\varphi_0 : \{*\} \to D(*)$ is uniquely determined as $\mathcal{C}(*,*)$ is singleton.*

Proof. For functoriality of D, let $\phi : X \to Y$, $\psi : Y \to Z$, and $v : Z \to Z$ be arbitrary. For preservation of composition, we have

$$\begin{aligned} D(\phi \circ \psi)(v) \coloneqq (\phi \circ \psi)^\dagger \circ v \circ (\phi \circ \psi) &= (\psi^\dagger \circ \phi^\dagger) \circ v \circ (\phi \circ \psi) \\ &= \psi^\dagger \circ (\phi^\dagger \circ v \circ \phi) \circ \psi = (D(\psi) \circ D(\phi))(v), \end{aligned} \tag{9}$$

where the first equality comes from contravariant functoriality of $(-)^\dagger$ and the second equality comes from associativity of composition. Similarly, for preservation of identities, we have

$$D(\mathrm{id}_Z)(v) \coloneqq \mathrm{id}_Z^\dagger \circ v \circ \mathrm{id}_Z = \mathrm{id}_Z \circ v \circ \mathrm{id}_Z = v, \tag{10}$$

where the first equality again comes from functoriality of $(-)^\dagger$.

To verify the naturality of the product comparison, we need the following diagram to commute for all $\phi : X' \to X$ and $\psi : Y' \to Y$:

$$\begin{array}{ccc} D(X) \times D(Y) & \xrightarrow{D(\phi) \times D(\psi)} & D(X') \times D(Y') \\ {\scriptstyle \varphi_{X,Y}}\downarrow & & \downarrow{\scriptstyle \varphi_{X',Y'}} \\ D(X \times Y) & \xrightarrow[D(\phi \times \psi)]{} & D(X' \times Y') \end{array}$$

For this, let $(v,w) \in D(X) \times D(Y)$ be arbitrary. Then, following the top path yields

$$\pi_0^\dagger \circ \phi^\dagger \circ v \circ \phi \circ \pi_0 + \pi_1^\dagger \circ \psi^\dagger \circ w \circ \psi \circ \pi_1, \tag{11}$$

while following the bottom path yields

$$\begin{array}{ll} (\phi \times \psi)^\dagger \circ (\pi_0^\dagger \circ v \circ \pi_0 + \pi_1^\dagger \circ w \circ \pi_1) \circ (\phi \times \psi) & \\ = (\phi \times \psi)^\dagger \circ (\pi_0^\dagger \circ v \circ \pi_0 \circ (\phi \times \psi) + \pi_1^\dagger \circ w \circ \pi_1 \circ (\phi \times \psi)) & \text{Left additivity} \\ = (\phi \times \psi)^\dagger \circ \pi_0^\dagger \circ v \circ \pi_0 \circ (\phi \times \psi) + (\phi \times \psi)^\dagger \circ \pi_1^\dagger \circ w \circ \pi_1 \circ (\phi \times \psi) & \text{Additivity of linear maps} \\ = (\phi \times \psi)^\dagger \circ \pi_0^\dagger \circ v \circ \phi \circ \pi_0 + (\phi \times \psi)^\dagger \circ \pi_1^\dagger \circ w \circ \psi \circ \pi_1 & \text{Lemma A.7} \\ = (\pi_0 \circ \phi \times \psi)^\dagger \circ v \circ \phi \circ \pi_0 + (\pi_1 \circ \phi \times \psi)^\dagger \circ w \circ \psi \circ \pi_1 & \text{Contravariant functoriality} \\ = (\phi \circ \pi_0)^\dagger \circ v \circ \phi \circ \pi_0 + (\psi \circ \pi_1)^\dagger \circ w \circ \psi \circ \pi_1 & \text{Lemma A.7} \\ = \pi_0^\dagger \circ \phi^\dagger \circ v \circ \phi \circ \pi_0 + \pi_1^\dagger \circ \psi^\dagger \circ w \circ \psi \circ \pi_1, & \end{array} \tag{12}$$

as desired. The symmetric monoidal coherence axioms again follow from the commutative monoid structure of hom-sets. □

This is a generalization of the dynamics decorating functor defined in [1]. We can now define an analogous hypergraph category of open generalized dynamical systems.

Corollary 2.4.1 (Open Generalized Dynamics)**.** *Given a CRDC $\mathcal{C}$ such that $\mathsf{Lin}(\mathcal{C})$ has finite limits, there is a hypergraph category $\mathbf{Dynam}_{\mathcal{C}}$ defined by the following data.*

- *Objects are the same as those of* $\mathcal{C}$.
- *Morphisms are (isomorphism classes of) open dynamical systems.*
- *Given two open systems* $(X \xleftarrow{l_1} N \xrightarrow{r_1} Y, v \in D_{\mathcal{C}}(N))$ *and* $(Y \xleftarrow{l_2} M \xrightarrow{r_2} Z, w \in D_{\mathcal{C}}(M))$*, their composite consists of the pullback span as in* (5) *together with the system obtained from the composite morphism*

$$\{*\} \cong \{*\} \times \{*\} \xrightarrow{v \times w} D_{\mathcal{C}}(N) \times D_{\mathcal{C}}(M) \xrightarrow{\varphi_{N,M}} D_{\mathcal{C}}(N \times M) \xrightarrow{D_{\mathcal{C}}\langle b_0, b_1 \rangle} D_{\mathcal{C}}(N \times_Y M). \tag{13}$$

- *The identity on* X *is the pair* $(X = X = X, \{*\} \xrightarrow{\varphi_0} D_{\mathcal{C}}(*) \xrightarrow{D_{\mathcal{C}}(!)} D_{\mathcal{C}}(X))$.
- *The monoidal product on objects is their product in* $\mathcal{C}$ *while the monoidal product of morphisms* $(X \leftarrow N \rightarrow Y, v)$ *and* $(W \leftarrow M \rightarrow Z, w)$ *is*

$$(X \times W \leftarrow N \times M \rightarrow Y \times Z, \varphi_{N,M}(v, w)). \tag{14}$$

- *The hypergraph maps are inherited from those of* $\mathrm{Span}(\mathsf{Lin}(\mathcal{C}))$ *together with the identity decoration.*

Proof. This follows by applying Proposition 3.2 and Theorem 3.4 in [9] to Theorem 2.4. □

2.3 Functoriality of Generalized Gradient Descent

The pieces are now in place for our main result. We would like to show that one can functorially relate a composite objective function defined in $\mathbf{Opt}_{\mathcal{C}}^{R}$ to a composite optimizer in $\mathbf{Dynam}_{\mathcal{C}}$ that performs gradient descent on the given objective function. The framework of decorated spans allows the construction of such a hypergraph functor by specifying a monoidal natural transformation between the underlying decorating functors defining the domain and codomain hypergraph categories. Note that we work with continuous gradient flow systems; however, these results easily extend to discrete gradient descent systems as Euler's method is known to be functorial for resource sharing dynamical systems [12].

Theorem 2.5. *Given an optimization domain* $(\mathcal{C}, R)$*, there is a monoidal natural transformation* $\mathsf{gd}(\mathcal{C})\colon O_{\mathcal{C}}^{R} \Rightarrow D_{\mathcal{C}}$ *with components* $\mathsf{gd}(\mathcal{C})_X\colon O_{\mathcal{C}}^{R}(X) \rightarrow D_{\mathcal{C}}(X)$ *defined by the generalized gradient descent optimization scheme, i.e.,*

$$f \mapsto -\mathbf{R}[f]_1. \tag{15}$$

Proof. Let $\mathsf{gd} := \mathsf{gd}_{\mathcal{C}}, O := O_{\mathcal{C}}^{R}$, and $D := D_{\mathcal{C}}$. For naturality, we need to verify that the following diagram commutes for all objects $X, Y \in \mathsf{Lin}(\mathcal{C})$ and linear maps $\phi\colon Y \rightarrow X$:

$$\begin{array}{ccc} O(X) & \xrightarrow{O(\phi)} & O(Y) \\ {\scriptstyle \mathsf{gd}_X}\downarrow & & \downarrow{\scriptstyle \mathsf{gd}_Y} \\ D(X) & \xrightarrow[D(\phi)]{} & D(Y) \end{array}$$

Let $f\colon X \rightarrow R$ be an arbitrary objective. Then, following the top path yields the optimizer

$$-\mathbf{R}[f \circ \phi]_1 := -\mathbf{R}[f \circ \phi] \circ \langle \mathrm{id}_Y, 1_{YR} \rangle. \tag{16}$$

Likewise, following the bottom path yields

$$-\phi^\dagger \circ \mathbf{R}[f]_1 \circ \phi := -\phi^\dagger \circ \mathbf{R}[f] \circ \langle \mathrm{id}_X, 1_{XR} \rangle \circ \phi. \tag{17}$$

Now consider applying (16) to a generalized element $y \in Y$:

$$\begin{aligned} -\mathbf{R}[f \circ \phi] \circ \langle \mathrm{id}_Y, 1_{YR} \rangle(y) &= -\mathbf{R}[f \circ \phi](y, 1_R) \\ &= -\mathbf{R}[\phi](y, \mathbf{R}[f](\phi(y), 1_R)) \text{ Chain rule } \mathbf{RD.5} \\ &= -\phi^\dagger(\mathbf{R}[f](\phi(y), 1_R)) \quad \text{Linearity.} \end{aligned} \tag{18}$$

Finally, applying (17) to the same element gives

$$-\phi^\dagger(\mathbf{R}[f](\langle \mathrm{id}_X, 1_{XR} \rangle(\phi(y)))) = -\phi^\dagger(\mathbf{R}[f](\phi(y), 1_R)) \tag{19}$$

as desired. To verify the transformation is monoidal, we must show that the following diagrams commute:

$$\begin{array}{ccc} O(X) \times O(Y) & \xrightarrow{\mathsf{gd}_X \times \mathsf{gd}_Y} & D(X) \times D(Y) \\ \downarrow{\scriptstyle \varphi_{X,Y}} & & \downarrow{\scriptstyle \varphi_{X,Y}} \\ O(X \times Y) & \xrightarrow[\mathsf{gd}_{X \times Y}]{} & D(X \times Y) \end{array} \qquad \begin{array}{ccc} * & \xrightarrow{\varphi_0} & O(*) \\ & \searrow^{\varphi_0} & \downarrow{\scriptstyle \mathsf{gd}_*} \\ & & D(*) \end{array}$$

Let $g \colon Y \to R$ be another arbitrary objective. Following the top path of the product comparison diagram yields the system

$$-\pi_0^\dagger \circ \mathbf{R}[f]_1 \circ \pi_0 - \pi_1^\dagger \circ \mathbf{R}[g]_1 \circ \pi_1 \tag{20}$$

while following the bottom path gives

$$\begin{aligned} -\mathbf{R}[f \circ \pi_0 + g \circ \pi_1]_1 &= -\mathbf{R}[f \circ \pi_0 + g \circ \pi_1] \circ \langle \mathrm{id}_{X \times Y}, 1_{X \times Y,R} \rangle \\ &= -(\mathbf{R}[f \circ \pi_0] + \mathbf{R}[g \circ \pi_1]) \circ \langle \mathrm{id}_{X \times Y}, 1_{X \times Y,R} \rangle \qquad \mathbf{RD.1} \\ &= -\mathbf{R}[f \circ \pi_0] \circ \langle \mathrm{id}_{X \times Y}, 1_{X \times Y,R} \rangle - \mathbf{R}[g \circ \pi_1] \circ \langle \mathrm{id}_{X \times Y}, 1_{X \times Y,R} \rangle \ \mathbf{LA.1} \\ &= -\mathbf{R}[f \circ \pi_0]_1 - \mathbf{R}[g \circ \pi_1]_1. \end{aligned} \tag{21}$$

These can be shown equivalent following the same reasoning as for naturality, namely by applying the chain rule. The unit comparison diagram commutes trivially as $D(*)$ is terminal. □

Corollary 2.5.1 (Functoriality of Generalized Gradient Flow)**.** *Given an optimization domain $(\mathcal{C}, R)$ such that $\mathsf{Lin}(\mathcal{C})$ has finite limits, there is an identity on objects hypergraph functor $\mathsf{GD}_\mathcal{C} \colon \mathbf{Opt}_\mathcal{C}^R \to \mathbf{Dynam}_\mathcal{C}$ which takes an open objective $(X \xleftarrow{l} N \xrightarrow{r} Y, f \colon N \to R)$ to the open optimizer $(X \xleftarrow{l} N \xrightarrow{r} Y, -\mathbf{R}[f]_1 \colon N \to N)$.*

Proof. This follows by applying Theorem 4.1 of [9] to the monoidal natural transformation gd. □

Taking stock, there are a many implications of the categories and functor just defined. For one, corollary 2.5.1 says we have the following equality given any composable pair $F := (X \leftarrow N \to Y, f \colon N \to R), G := (Y \leftarrow M \to Z, g \colon M \to R)$ of open objectives:

$$\mathsf{GD}_\mathcal{C}(G \circ F) = \mathsf{GD}_\mathcal{C}(G) \circ \mathsf{GD}_\mathcal{C}(F). \tag{22}$$

In particular, this yields the following equivalence of open optimizers:

$$-\mathbf{R}[f \circ \pi_0 \circ \phi + g \circ \pi_1 \circ \phi]_1 = -\phi^\dagger \circ (\iota_0 \circ \mathbf{R}[f]_1 \circ \pi_0 + \iota_1 \circ \mathbf{R}[g]_1 \circ \pi_1) \circ \phi, \tag{23}$$

where we are using ϕ to denote the pairing $\langle b_0, b_1 \rangle$ from (5). Although these two optimizers are *extensionally* equivalent (i.e., they compute the same output for a given input), they are not *computationally* equivalent. In particular, the optimizer on the RHS of (23) is far better suited to implementation in a distributed computing environment as it can be interpreted with message passing semantics. Specifically, the distribute step is given by applying the linear map ϕ to the input followed by the application of the projections to send the desired components to the subsystems. The parallel computation step is given by computing the gradients of each objective with respect to their current local values, and the collect step is given by injecting the results into a single vector and applying the linear map $\phi^\dagger$.

Another benefit of open objectives and optimizers is the ability to easily specify composite objective functions using the graphical syntax of string diagrams. In the following section we will model hard parameter sharing for mulit-task learning in our category $\mathbf{Opt}^R_{\mathcal{C}}$. We do so to show that, rather than existing as an exercise in abstraction, our framework is applicable to a prevalent machine learning paradigm.

3 Multitask Learning

Multitask learning (MTL) [4] is a machine learning paradigm in which learning signals from different tasks are aggregated in order to train a single model. Precise definitions of what a task is vary, but for our purposes we will consider a task $\mathcal{T}_i \coloneqq (\mathcal{D}_i, \mathcal{L}_i)$ to be a pair of data set $\mathcal{D}_i$ and loss function $\mathcal{L}_i$. Usually (see [21]) the tasks $\mathcal{T}_i$ are supervised, so $\mathcal{D}_i \coloneqq \mathcal{X}_i \times \mathcal{Y}_i$ for input data $\mathcal{X}_i$ and labels $\mathcal{Y}_i$. The primary (or at least original) motivation of MTL is to exploit additional learning signals for improved model performance on a given "main" task, beyond what might be possible if the model were trained on any single task [4].

A common approach (see [21]) to multitask learning is to leverage parameterised models where at least some parameters are trained on all tasks. Such an approach is called *hard parameter sharing* because the shared parameters are updated *directly* using gradient information from multiple tasks. This is in contrast to *soft parameter sharing* where the parameters are only *penalized* for differing from counterparts trained on other tasks. Theoretically, hard parameter sharing helps to avoid overfitting of shared parameters [2]. When the underlying model is a neural network, hard parameter sharing is often achieved by sharing the parameters of the first several layers while training a unique set of predictive final layers per task. However, recent work suggests that more flexible schemes may lead to improved performance [20]. With this motivation, we will now frame hard parameter sharing as a composite optimization problem that allows arbitrary parameters to be shared amongst arbitrary tasks.

For each task $\mathcal{T}_i$ in a collection of N tasks, consider a neural network f_i that has task-specific parameters P_i and shared parameters P_0. We will call an assignment of parameters to real numbers $W_i \in \mathbb{R}^{P_i}$ the *weights* of a neural network, though note that the machine learning literature often uses the terms "weights" and "parameters" interchangeably. The composite optimization problem given by such a multitask learning setup (cf. [16], (1)) is

$$\min_{W_0,\ldots,W_N} \quad \sum_{i=1}^{N} \quad \mathcal{L}_i^D(f_i; W_0, W_i) \tag{24}$$

where

$$\mathcal{L}_i^D(f_i; W_0, W_i) \quad \coloneqq \tfrac{1}{|\mathcal{D}_i|} \sum_{d=1}^{|\mathcal{D}^i|} \mathcal{L}_i(f_i(\mathcal{X}_i^d; W_0, W_i), \mathcal{Y}_i^d) \tag{25}$$

with $f_i(\mathcal{X}_i^d; W_0, W_i)$ denoting the prediction of the network f_i weighted by (W_0, W_i) on input datum $\mathcal{X}_i^d$. We will now describe how problem 24 is a composite of open objectives, and therefore how to mechanically recover a distributed gradient descent algorithm for it.

3.1 Compositional Formulation

We will take our CRDC $\mathcal{C}$ to be **Euc** with objects euclidean spaces $\mathbb{R}^N$ and morphisms $f : \mathbb{R}^N \to \mathbb{R}^M$ the smooth maps between spaces. The monoidal product is given by $\times$, the standard product of Euclidean spaces. Each task's loss is a smooth map $\mathcal{L}_i^D \colon \mathbb{R}^{P_i} \times \mathbb{R}^{P_0} \to \mathbb{R}$, meaning our optimization domain is $(\mathbf{Euc}, \mathbb{R})$. Note that we have avoided some difficulty by treating each loss as an atomic map, as opposed to the composition of a neural network applied to data followed by a loss function. In the latter case, we would need to consider the set of parameters for the network in addition to the set of data points, and handle the subtleties of updating parameters while ignoring gradients w.r.t the input data. See [8] for a thorough discussion of such concerns. Because the loss $\mathcal{L}_i^D$ is computed with respect to the entire data set $\mathcal{D}_i$ for a task, we have restricted ourselves to optimization methods that perform true gradient descent as opposed to the stochastic or mini-batch variants which are more common in the machine learning literature. We leave an extension of our work to the SGD setting for future efforts.

With the goal of specifying problem 24 as a morphism of $\mathbf{Opt}_{\mathbf{Euc}}^{\mathbb{R}}$, we first consider the following span that relates shared and task specific parameters for some i.

$$\mathbb{R}^{P_0} \xleftarrow{\pi_0} \mathbb{R}^{P_0} \times \mathbb{R}^{P_i} \xrightarrow{\pi_1} \mathbb{R}^{P_i} \tag{26}$$

To promote 26 to a decorated span, we need only to specify an objective function $\mathbb{R}^{P_0} \times \mathbb{R}^{P_i} \to \mathbb{R}$. With $\mathcal{L}_i^D$ being the obvious choice, we construct a decorated span $F^i := \left(\mathbb{R}^{P_0} \xleftarrow{\pi_0} \mathbb{R}^{P_0} \times \mathbb{R}^{P_i} \xrightarrow{\pi_1} \mathbb{R}^{P_i}, \mathcal{L}_i^D\right)$. Each F^i is a morphism in the hypergraph category $\mathbf{Opt}_{\mathbf{Euc}}^{\mathbb{R}}$, meaning that we may take the monoidal product $\bigotimes_{i=1}^N F^i$. By Cor. 2.2.1, specifically Eq. 7, we know that the monoidal product of our task specific decorated spans is as follows.

$$\bigotimes_{i=1}^N F^i = \left(\prod_{i=1}^N \mathbb{R}^{P_0} \xleftarrow{\prod_{i=1}^N \pi_0^i} \prod_{i=1}^N (\mathbb{R}^{P_0} \times \mathbb{R}^{P_i}) \xrightarrow{\prod_{i=1}^N \pi_1^i} \prod_{i=1}^N \mathbb{R}^{P_i}, \varphi(\mathcal{L}_1^D, \ldots, \mathcal{L}_N^D)\right) \tag{27}$$

Note that by Thm. 2.2 the decorating objective function is given by the comparison map:

$$\varphi(\mathcal{L}_1^D, \ldots, \mathcal{L}_N^D) \colon \prod_{i=1}^N O(\mathbb{R}^{P_0} \times \mathbb{R}^{P_i}) \to O(\prod_{i=1}^N \mathbb{R}^{P_0} \times \mathbb{R}^{P_i}), \tag{28}$$

where $\varphi(\mathcal{L}_1^D, \ldots, \mathcal{L}_N^D)$ maps the tuple of task specific losses to their sum:

$$(\mathcal{L}_1^D, \ldots, \mathcal{L}_N^D) \mapsto \sum_{i=1}^N \mathcal{L}_i^D \circ \pi^i, \tag{29}$$

where π^i denotes the i'th projection of $\prod_{i=1}^N (\mathbb{R}^{P_0} \times \mathbb{R}^{P_i})$. We now wish to specify that the parameter space $\mathbb{R}^{P_0}$ is shared among all N loss functions. This is accomplished as follows.

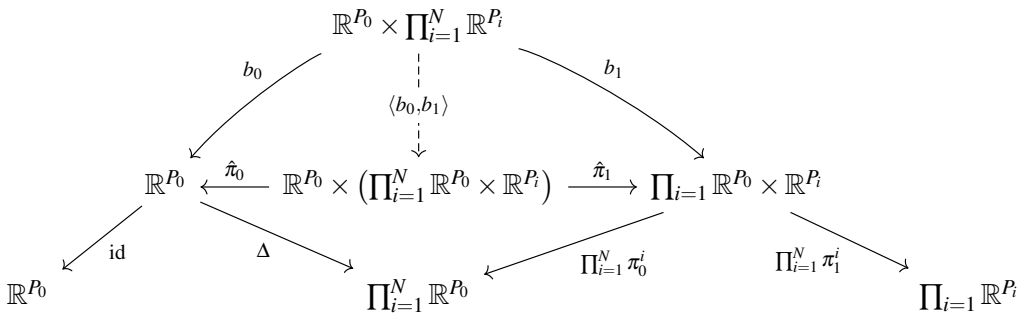
Recall that in a hypergraph category like $\mathbf{Opt}_{\mathbf{Euc}}^{\mathbb{R}}$, we have comultiplication maps

$$\delta_p := \left(\mathbb{R}^p \xleftarrow{\mathrm{id}_p} \mathbb{R}^p \xrightarrow{\Delta} \mathbb{R}^p \times \mathbb{R}^p, 0 \colon \mathbb{R}^p \to \mathbb{R}\right), \tag{30}$$

where $0 \colon \mathbb{R}^p \to \mathbb{R}$ is the "empty" decoration, i.e., the constant 0 objective function $0(v) = 0$. The map $\Delta \colon \mathbb{R}^p \to \mathbb{R}^p \times \mathbb{R}^p$ is the duplication map $v \mapsto (v, v)$ built-in to any Cartesian category. Thus, to copy the single parameter space $\mathbb{R}^{P_0}$ to all tasks, we simply apply δ_{P_0} N times. We denote this N-fold application $\delta_{P_0}^N$. To recover problem 24, all we must do is precompose our monoidal product of task specific objectives with the N-fold copy, yielding the morphism in $\mathbf{Opt}_{\mathbf{Euc}}^{\mathbb{R}}$

$$\begin{pmatrix} \left(\bigotimes_{i=1}^{N} F^i\right) \circ \delta_{P_0}^N, O(\langle b_0, b_1 \rangle) \circ \left(0 + \sum_{i=1}^{N} \mathcal{L}_i^D \circ \pi^i\right) \end{pmatrix} \\ \left(\left(\bigotimes_{i=1}^{N} F^i\right) \circ \delta_{P_0}^N, \mathcal{L}^D \colon (W_0, W_1, \ldots, W_N) \mapsto \sum_{i=1}^{N} \mathcal{L}_i^D(W_0, W_i)\right) \end{pmatrix} = \tag{31}$$

which can be pictured diagramatically as:



In Eq. 31, the computation of the pullback has effectively filtered the multiple copies of $\mathbb{R}^{P_0}$ and left us with the loss we want. Note that the objective function 0 of the comultiplication morphism does not contribute to the overall objective.

3.2 Distributed Optimization via Functoriality of Gradient Descent

Algorithm 1: Distributed MTL

Input: Initial shared weights $W_0^0 \in \mathbb{R}^{P_0}$, and task-specific weights $W_i^0 \in \mathbb{R}^{P_i}$ for $i \in \{1, \ldots, N\}$
Input: A positive real learning rate γ
$W_i \leftarrow W_i^0$ for $i \in \{0, \ldots, N\}$;
while *a stopping criterion is not reached* **do**
  Compute each iteration of the following for-loop in parallel;
  for $i \leftarrow 1$ **to** N **do**
    $\texttt{grad\_}W_{0,i} \leftarrow \nabla_{W_0} \mathcal{L}(W_0, W_i)$;
    $\texttt{grad\_}W_i \leftarrow \nabla_{W_i} \mathcal{L}(W_0, W_i)$;
    $W_i \leftarrow W_i - \gamma * \texttt{grad\_}W_i$; `/* Update non-shared weights */`
  end
  $\texttt{grad\_}W_0 \leftarrow \sum_{i=1}^{N} \texttt{grad\_}W_{0,i}$; `/* Gradient of shared weights is sum of gradients from each task-specific learner */`
  $W_0 \leftarrow W_0 - \gamma * \texttt{grad\_}W_0$;
end
return $W_0, W_1, \ldots, W_N$

Having represented problem 24 as a morphism in $\mathbf{Opt}_{\mathbf{Euc}}^{\mathbb{R}}$, we can create a gradient descent based optimizer by applying Cor. 2.5.1. In particular we aim for a distributed gradient descent scheme. Exploiting the functoriality of $\mathsf{GD}_{\mathbf{Euc}}$, we prefer to compute the gradient for each morphism separately and then compose systems. The generalized gradient functor is illustrated in Section 1, Figure 1.

Algorithm 1 describes the distributed scheme for multitask learning derived by applying $\mathsf{GD}_{\mathcal{C}}$ to the individual subproblems and composing the resulting dynamical systems. Note that there is value in distributing computation this way as computing the gradients of weights will typically involve non-trivial backpropogation over potentially large networks. With this, we have successfully recovered hard parameter sharing for multitask learning and shown how to make a gradient descent optimizer that can be implemented in a distributed fashion.

4 Discussion and Future Work

Taking stock, it is fair to wonder what has been gained by specifying parameter sharing via decorated spans. One classic argument for the utility of applied category theory is the tight link between graphical problem depiction and its realization in a mathematical or computational data structure. Figure 1 contains two string diagrams which visually capture the structure of multitask learning objectives and optimizers respectively, enabling better communication of models between machine learning practitioners. Moreover, the relative ease with which one can describe compositional objectives supports more complex problem structures. Objectives with traditional regularization terms can be trivially described as compositional objectives, and soft parameter sharing should be fairly easily accommodated via the addition of penalization terms. More exotic hierarchical parameter sharing (see e.g. [18]) should be readily modeled using our approach to MTL. There is a computational benefit too; distributed optimization schemes become easier to implement with a rigorous definition of problem structure. Finally the generality of the techniques discussed allows the parameter sharing pattern of the given morphism to be applied to settings beyond smooth real-valued functions and traditional gradient descent.

As for the general framework defined in Sec. 2, we believe there are interesting connections with previous work on CRDCs. Perhaps the most obvious direction for future work would be to understand the connections with the Para and Lense constructions of [8]. In particular, we hope that a unified treatment of both the compositional structure of models and the compositional structure of objectives can be achieved, perhaps using tools from double category theory. Naturally another direction of future work is to find more examples of CRDCs and study not only learning problems in those settings, but other optimization problems such as those arising in operations research.

5 Conclusion

In summary, we have presented hypergraph categories $\mathbf{Opt}_{\mathcal{C}}^{R}$ and $\mathbf{Dynam}_{\mathcal{C}}$ of open objectives and dynamical systems respectively, and shown that there exists a hypergraph functor between them that maps a generalized objective to its corresponding gradient descent optimizer. Both $\mathbf{Opt}_{\mathcal{C}}^{R}$ and $\mathbf{Dynam}_{\mathcal{C}}$ are constructed with respect to an underlying Cartesian reverse derivative category, allowing for composition of optimization problems that are defined on a general optimization domain. Such compositionality induces a template for distributed optimization, and provides an accompanying graphical syntax that facilitates communication and sharing. Lastly, to provide evidence that existing machine learning paradigms can be described by our compositional framework, we described parameter sharing models for multitask learning using the hypergraph structure of $\mathbf{Opt}_{\mathcal{C}}^{R}$, and highlighted the distributed gradient descent algorithm induced by this structure.

References

[1] John C Baez & Blake S Pollard (2017): *A compositional framework for reaction networks*. *Reviews in Mathematical Physics* 29(09), p. 1750028.

[2] Jonathan Baxter (1997): *A Bayesian/information theoretic model of learning to learn via multiple task sampling*. *Machine learning* 28, pp. 7–39.

[3] Christopher Bishop (2006): *Pattern Recognition and Machine Learning*. Springer. Available at `https://www.microsoft.com/en-us/research/publication/pattern-recognition-machine-learning/`.

[4] Rich Caruana (1997): *Multitask learning*. *Machine learning* 28, pp. 41–75.

[5] Robin Cockett, Geoffrey Cruttwell, Jonathan Gallagher, Jean-Simon Pacaud Lemay, Benjamin MacAdam, Gordon Plotkin & Dorette Pronk (2019): *Reverse derivative categories*. arXiv:1910.07065.

[6] Bob Coecke & Ross Duncan (2007): *A graphical calculus for quantum observables*. *Preprint*.

[7] Bob Coecke & Ross Duncan (2008): *Interacting quantum observables*. In: *International Colloquium on Automata, Languages, and Programming*, Springer, pp. 298–310.

[8] Geoffrey SH Cruttwell, Bruno Gavranović, Neil Ghani, Paul Wilson & Fabio Zanasi (2022): *Categorical foundations of gradient-based learning*. In: *European Symposium on Programming*, Springer International Publishing Cham, pp. 1–28.

[9] Brendan Fong (2015): *Decorated Cospans*. arXiv:1502.00872.

[10] Brendan Fong (2016): *The algebra of open and interconnected systems*. *arXiv preprint arXiv:1609.05382*.

[11] Tyler Hanks, Matthew Klawonn, Evan Patterson, Matthew Hale & James Fairbanks (2024): *A Compositional Framework for First-Order Optimization*. *arXiv preprint arXiv:2403.05711*.

[12] Sophie Libkind, Andrew Baas, Evan Patterson & James Fairbanks (2022): *Operadic Modeling of Dynamical Systems: Mathematics and Computation*. *Electronic Proceedings in Theoretical Computer Science* 372, pp. 192–206, doi:10.4204/EPTCS.372.14. Available at `http://arxiv.org/abs/2105.12282`. ArXiv:2105.12282 [math].

[13] Seppo Linnainmaa (1970): *The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors*. Ph.D. thesis, Master's Thesis (in Finnish), Univ. Helsinki.

[14] Makoto Matsumoto & Takuji Nishimura (1998): *Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator*. *ACM Transactions on Modeling and Computer Simulation (TOMACS)* 8(1), pp. 3–30.

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

[16] Ozan Sener & Vladlen Koltun (2018): *Multi-task learning as multi-objective optimization*. *Advances in neural information processing systems* 31.

[17] Dan Shiebler (2022): *Generalized Optimization: A First Step Towards Category Theoretic Learning Theory*. In Pandian Vasant, Ivan Zelinka & Gerhard-Wilhelm Weber, editors: *Intelligent Computing & Optimization*, Springer International Publishing, Cham, pp. 525–535.

[18] Anders Søgaard & Yoav Goldberg (2016): *Deep multi-task learning with low level tasks supervised at lower layers*. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 231–235.

[19] Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun & Rob Fergus (2013): *Regularization of neural networks using dropconnect*. In: *International conference on machine learning*, PMLR, pp. 1058–1066.

[20] Lijun Zhang, Qizheng Yang, Xiao Liu & Hui Guan (2022): *Rethinking hard-parameter sharing in multi-domain learning*. In: *2022 IEEE International Conference on Multimedia and Expo (ICME)*, IEEE, pp. 01–06.

[21] Yu Zhang & Qiang Yang (2021): *A survey on multi-task learning*. *IEEE Transactions on Knowledge and Data Engineering* 34(12), pp. 5586–5609.

A Generalized Optimization

In this section, we recall the necessary definitions and concepts from Cartesian reverse derivative categories [5] and generalized optimization [17].

Definition A.1 (Definition 3.1 in [17])**.** A **Cartesian left-additive category** is a Cartesian category $\mathcal{C}$ with terminal object $*$ in which for any pair of objects X, Y, the hom-set $\mathcal{C}(X,Y)$ is a commutative monoid with addition operation $+$ and additive identities $0_{X,Y}\colon X \to Y$. In addition, the following axioms must be satisfied.

- **LA.1** For any morphisms $h\colon X \to Y$ and $f,g\colon Y \to Z$, we have

$$(f+g)\circ h = (f\circ h)+(g\circ h)\colon X \to Z \text{ and } 0_{Y,Z}\circ h = 0_{X,Z}\colon X \to Z. \tag{32}$$

- **LA.2** For any projection map $\pi_i\colon Y \to Z$ and morphisms $f,g\colon X \to Y$ we have

$$\pi_i\circ(f+g) = (\pi_i\circ f)+(\pi_i\circ g)\colon X \to Z \text{ and } \pi_i\circ 0_{X,Y} = 0_{X,Z}\colon X \to Z. \tag{33}$$

The additive identity of $\mathcal{C}(*,X)$ is denoted 0_X.

Rough Definition A.2 (Definition 3.3 in [17])**.** A **Cartesian differential category** is a Cartesian left-additive category $\mathcal{C}$ equipped with a **Cartesian derivative combinator D** taking morphisms $f\colon X \to Y$ to morphisms $\mathbf{D}[f]\colon X \times X \to Y$. This combinator must satisfy several axioms requiring that it behave like a standard forward derivative.

Rough Definition A.3 (Definition 3.2 in [17])**.** A **Cartesian reverse derivative category** is a Cartesian left-additive category $\mathcal{C}$ equipped with a **Cartesian reverse derivative combinator R** taking morphisms $f\colon X \to Y$ to morphisms $\mathbf{R}[f]\colon X \times Y \to X$. This combinator must satisfy several axioms requiring that it behave like a standard reverse derivative. In particular, we will make use of the following axioms in this paper.

- **RD.1** $\mathbf{R}[f+g] = \mathbf{R}[f]+\mathbf{R}[g]$ and $\mathbf{R}[0] = 0$.
- **RD.5** $\mathbf{R}[g\circ f] = \mathbf{R}[f]\circ(\mathrm{id}\times\mathbf{R}[g])\circ\langle\pi_0,\langle f\circ\pi_0,\pi_1\rangle\rangle$.

Intuitively, **RD.1** says that the reverse derivative combinator must be additive while **RD.5** is the chain rule cast in the abstract setting of CRDCs. Every Cartesian reverse derivative category (CRDC) is also a Cartesian differential category (Theorem 16 in [5]) by defining the derivative combinator of a morphism $f\colon X \to Y$ as

$$\mathbf{D}[f] = \pi_1\circ\mathbf{R}[\mathbf{R}[f]]\circ(\langle \mathrm{id}_X, 0_{X,Y}\rangle\times \mathrm{id}_X)\colon X\times X \to Y. \tag{34}$$

This allows us to refer to the Cartesian derivative combinator of a CRDC. Of crucial importance in the sequel is the notion of a linear map in CRDC.

Definition A.4. A **linear map** in a CRDC $\mathcal{C}$ is a morphism f such that $\mathbf{D}[f] = f\circ\pi_1$. The linear maps in $\mathcal{C}$ form a subcategory of $\mathcal{C}$ which we denote $\mathsf{Lin}(\mathcal{C})$. By Proposition 24 in [5], $\mathsf{Lin}(\mathcal{C})$ is a †-category with finite †-biproducts. Moreover, $\mathbf{R}[f] = f^\dagger\circ\pi_1$ and $(f+g)^\dagger = f^\dagger + g^\dagger$.

There are two canonical examples of CRDCs in the literature.

1. The CRDC **Euc** has Euclidean spaces as objects and smooth functions as morphisms. The terminal object is $\mathbb{R}^0$ and the reverse derivative of a smooth function $f\colon \mathbb{R}^m \to \mathbb{R}^n$ is

$$\mathbf{R}[f](x,x') \coloneqq J_f(x)^T x',$$

where $J_f(x)$ is the Jacobian of f evaluated at x. The linear maps in **Euc** are the usual linear maps between vector spaces, with the dagger structure corresponding to dual maps.

2. Given a commutative ring r, the CRDC $\mathbf{Poly}_r$ has natural numbers as objects. A morphism from n to m is an m-tuple of polynomials over r, each in n variables. The terminal object is 0 and the reverse derivative of a polynomial $P(x) = (p_1(x), \ldots, p_m(x))$ is

$$\mathbf{R}[P](x,x') \coloneqq \Big(\sum_{i=1}^{m} \frac{\partial p_i}{\partial x_1}(x)x_i', \ldots, \sum_{i=1}^{m} \frac{\partial p_i}{\partial x_n}(x)x_i'\Big),$$

where $\frac{\partial p_i}{\partial x_j}(x)$ denotes the formal derivative of p_i in x_j at x. The linear maps are polynomials $P(x) = (p_1(x), \ldots, p_m(x))$ where each $p_i(x)$ is of the form $p_i(x) = \sum_{i=1}^{n} r_i x_i$ for $r_i \in r$ [5].

Definition A.5 (Definition 3.4 in [17])**.** An **optimization domain** is a pair $(\mathcal{C}, R)$ where $\mathcal{C}$ is a CRDC and R is an object of $\mathcal{C}$ equipped with the following additional structures.

- Each morphism $f\colon X \to Y$ in $\mathcal{C}$ has an additive inverse $-f$.
- Each hom-set $\mathcal{C}(*, X)$ out of the terminal object is equipped with a multiplication operation fg and a multiplicative identity $1_X\colon * \to X$ to form a commutative ring with the left additive structure.
- The hom-set $\mathcal{C}(*, R)$ is totally ordered to form an ordered commutative ring.

Given a unique map $!_X\colon X \to *$ into the terminal object, the map $1_Y \circ !_X\colon X \to Y$ is denoted $1_{X,Y}$. An **objective** in $(\mathcal{C}, R)$ is a morphism $\ell\colon X \to R$ in $\mathcal{C}$.

Definition A.6 (Definition 3.6 in [17])**.** Given an optimization domain $(\mathcal{C}, R)$, the **generalized gradient** of an objective $\ell\colon X \to R$ is $\mathbf{R}[\ell]_1\colon X \to X$ defined by

$$\mathbf{R}[\ell]_1 \coloneqq \mathbf{R}[\ell] \circ \langle \mathrm{id}_X, 1_{X,R} \rangle. \tag{35}$$

The following are examples given in [17] of optimization domains and generalized gradients.

1. The **standard domain** is $(\mathbf{Euc}, \mathbb{R})$. Objectives are smooth functions $\ell\colon \mathbb{R}^n \to \mathbb{R}$ and the gradient of $\ell\colon \mathbb{R}^n \to \mathbb{R}$ is $\nabla\ell\colon \mathbb{R}^n \to \mathbb{R}^n$.
2. The **r-polynomial domain** is $(\mathbf{Poly}_r, 1)$. Objectives are polynomials $\ell\colon n \to 1$ and the gradient of $\ell\colon n \to 1$ is $\Big(\frac{\partial \ell}{\partial x_1}(x), \ldots, \frac{\partial \ell}{\partial x_n}(x)\Big)$, where these are again formal derivatives.

Lemma A.7. *Let $\phi\colon X \to Y$ and $\psi\colon X' \to Y'$ be arbitrary morphisms in* $\mathrm{Lin}(\mathcal{C})$*. Then the following equations hold:*

1. $(\phi \times \psi)^\dagger = \phi^\dagger \times \psi^\dagger$,
2. $\pi_0 \circ (\phi \times \psi) = \phi \circ \pi_0$,
3. $\pi_1 \circ (\phi \times \psi) = \psi \circ \pi_1$.

Proof. For (1), we have

$$\begin{aligned}
(\phi \times \psi)^\dagger &= \langle \phi \circ \pi_0, \psi \circ \pi_1 \rangle^\dagger \\
&= (\iota_0 \circ \phi \circ \pi_0 + \iota_1 \circ \psi \circ \pi_1)^\dagger && \text{Lemma 3 in [5]} \\
&= (\iota_0 \circ \phi \circ \pi_0)^\dagger + (\iota_1 \circ \psi \circ \pi_1)^\dagger && \text{Property of } \dagger\text{-biproducts} \\
&= \pi_0^\dagger \circ \phi^\dagger \circ \iota_0^\dagger + \pi_1^\dagger \circ \psi^\dagger \circ \iota_1^\dagger && \text{Contravariant functoriality} \\
&= \iota_0 \circ \phi^\dagger \circ \pi_0 + \iota_1 \circ \psi^\dagger \circ \pi_1 && \text{Lemma 21 in [5]} \\
&= \langle \phi^\dagger \circ \pi_0, \psi^\dagger \circ \pi_1 \rangle = \phi^\dagger \times \psi^\dagger.
\end{aligned} \tag{36}$$

For (2), let $(x, x') \in X \times X'$ be generalized elements. Then

$$\pi_0 \circ (\phi \times \psi)(x, x') = \phi(x) = \phi(\pi_0(x, x')). \tag{37}$$

A symmetric argument holds for (3). □