Bundles, Lenses & Machine Learning

Jules Hedges\(^1\)
joint work with
Brendan Fong\(^2\)  Eliana Lorch\(^3\)  David Spivak\(^2\)

\(^1\)Max Planck Institute for Mathematics in the Sciences

\(^2\)MIT

\(^3\)University of Oxford

SYCO 5, Birmingham
Bundles, Lenses & Machine Learning

Jules Hedges\textsuperscript{1}
joint work with
Brendan Fong\textsuperscript{2} Eliana Lorch\textsuperscript{3} David Spivak\textsuperscript{2}

\textsuperscript{1}Max Planck Institute for Mathematics in the Sciences
\textsuperscript{2}MIT
\textsuperscript{3}University of Oxford

SYCO 5, Birmingham

Featuring zero string diagrams :(
Motivation

Machine learning is categorical in 2 different ways:

**Backprop As Functor**
(compositional description of ML with monoidal categories)

+ ML as differential geometry
Machine learning is categorical in 2 different ways:

**Backprop As Functor**
(compositional description of ML with monoidal categories)

ML as differential geometry

In this talk: *smoosh them together*

(why? Why not)
Machine learning is categorical in 2 different ways:

### Backprop As Functor
(compositional description of ML with monoidal categories)

+ ML as differential geometry

In this talk: smoosh them together

(why? Why not)

It clarifies Backprop as Functor more than anything else
Open learners

**Definition** (Fong, Spivak & Tuyéras) : An open learner $X \rightarrow Y$ consists of:

- A set $P$ of parameters
Open learners

Definition (Fong, Spivak & Tuyéras): An open learner \( X \rightarrow Y \) consists of:

- A set \( P \) of parameters
- A function \( I : P \times X \rightarrow Y \) (the implementation)

Composition of open learners is fiddly
They form a symmetric monoidal category called Learn
Who cares about monoidal bicategories
Open learners

**Definition** (Fong, Spivak & Tuyéras): An open learner $X \to Y$ consists of:

- A set $P$ of parameters
- A function $I : P \times X \to Y$ (the implementation)
- A function $u : P \times X \times Y \to P$ (the update)
Open learners

Definition (Fong, Spivak & Tuyéras): An open learner $X \to Y$ consists of:

- A set $P$ of parameters
- A function $I : P \times X \to Y$ (the implementation)
- A function $u : P \times X \times Y \to P$ (the update)
- A function $r : P \times X \times Y \to X$ (the request)
**Open learners**

**Definition** (Fong, Spivak & Tuyéras) : An open learner $X \rightarrow Y$ consists of:

- A set $P$ of parameters
- A function $I : P \times X \rightarrow Y$ (the implementation)
- A function $u : P \times X \times Y \rightarrow P$ (the update)
- A function $r : P \times X \times Y \rightarrow X$ (the request)

Composition of open learners is fiddly
Open learners

**Definition** (Fong, Spivak & Tuyéras): An open learner $X \to Y$ consists of:

- A set $P$ of parameters
- A function $I : P \times X \to Y$ (the implementation)
- A function $u : P \times X \times Y \to P$ (the update)
- A function $r : P \times X \times Y \to X$ (the request)

Composition of open learners is fiddly

They form a symmetric monoidal category called **Learn**

who cares about monoidal bicategories
A lens $X \to Y$ is a function $X \to Y$ and a function $X \times Y \to X$. Composition of lenses is also fiddly!

Theorem (Fong & Johnson): Open learners compose by pullback of lenses:

$P \times Q \times X \to P \times X \\downarrow \\uparrow \\pi_2 \\ell_1 \\pi_2 \\ell_2 \hspace{1cm} Q \times Y \to X \hspace{1cm}$
A lens $X \rightarrow Y$ is a function $X \rightarrow Y$ and a function $X \times Y \rightarrow X$.

Composition of lenses is also fiddly!
A lens $X \to Y$ is a function $X \to Y$ and a function $X \times Y \to X$.

Composition of lenses is also fiddly!

**Theorem (Fong & Johnson):** Open learners compose by pullback of lenses:

$$
egin{array}{c}
  P \times Q \times X \\
  \downarrow \\
  P \times X \\
  \downarrow \ell_1 \\
  X \\
  \downarrow \pi_2 \\
 \\
  \downarrow \\
  Y \\
  \downarrow \pi_2 \\
  Q \times Y \\
  \downarrow \ell_2 \\
  Z
\end{array}
$$

The Para construction

Let $C$ be a monoidal category

Define a category\(^1\) $\text{Para}(C)$ by:

\(^1\)who cares about monoidal bicategories
Let $\mathcal{C}$ be a monoidal category

Define a category\(^1\) **Para**($\mathcal{C}$) by:

- **Objects**: objects of $\mathcal{C}$
- **Morphisms** $X \to Y$: pair $(A, f)$, $A$ object of $\mathcal{C}$, $f : X \otimes A \to Y$

\(^1\) who cares about monoidal bicategories
The Para construction

Let $C$ be a monoidal category

Define a category $\text{Para}(C)$ by:

- **Objects**: objects of $C$
- **Morphisms** $X \to Y$: pair $(A, f)$, $A$ object of $C$, $f : X \otimes A \to Y$
- **Identity on** $X$: $(I, X \otimes I \overset{\text{ir}}{\to} X)$

$^1$ who cares about monoidal bicategories
The **Para** construction

Let $\mathcal{C}$ be a monoidal category

Define a category

\[ \text{Para}(\mathcal{C}) \]

by:

- **Objects**: objects of $\mathcal{C}$
- **Morphisms** $X \to Y$: pair $(A, f)$, $A$ object of $\mathcal{C}$, $f : X \otimes A \to Y$
- **Identity on $X$**: $(I, X \otimes I \xrightarrow{\cong} X)$
- **Composition of $(B, g) \circ (A, f)$**:

\[
(A \otimes B, X \otimes A \otimes B \xrightarrow{f \otimes B} Y \otimes B \xrightarrow{g} Z)
\]

\[1\text{who cares about monoidal bicategories}\]
Let $\mathcal{C}$ be a monoidal category

Define a category\(^1\) $\text{Para}(\mathcal{C})$ by:

- **Objects**: objects of $\mathcal{C}$
- **Morphisms** $X \to Y$: pair $(A, f)$, $A$ object of $\mathcal{C}$, $f : X \otimes A \to Y$
- **Identity on** $X$: $(I, X \otimes I \xrightarrow{\text{id}} X)$
- **Composition of** $(B, g) \circ (A, f)$:

\[
(A \otimes B, X \otimes A \otimes B \xrightarrow{f \otimes B} Y \otimes B \xrightarrow{g} Z)
\]

$\otimes$ lifts to a monoidal product on $\text{Para}(\mathcal{C})$

\(^1\)who cares about monoidal bicategories
The structure of \( \text{Para}(-) \)

A lax symmetric monoidal functor \( F : \mathcal{C} \to \mathcal{D} \) lifts to

\[
\text{Para}(F) : \text{Para}(\mathcal{D}) \to \text{Para}(\mathcal{D})
\]

by

\[
F(A, f) : F(X) \otimes F(A) \xrightarrow{\varphi} F(X \otimes A) \xrightarrow{F(f)} F(Y)
\]
The structure of \( \text{Para}(\_\_\_) \)

A lax symmetric monoidal functor \( F : \mathcal{C} \to \mathcal{D} \) lifts to

\[
\text{Para}(F) : \text{Para}(\mathcal{D}) \to \text{Para}(\mathcal{D})
\]

by

\[
F(A, f) : F(X) \otimes F(A) \xrightarrow{\varphi} F(X \otimes A) \xrightarrow{F(f)} F(Y)
\]

\[\textbf{Proposition}\ (\text{probably}):\ \text{Para}(\_\_)\ \text{defines a monad on}\]

\[\text{[symmetric monoidal categories, lax symmetric monoidal functors]}\]
Backprop as Functor

**Theorem** (Fong, Spivak & Tuyéras): Fix a **learning rate** $\varepsilon > 0$ and a differentiable **cost function** $C : \mathbb{R}^2 \to \mathbb{R}$.

---

$^2$such that every $\frac{\partial}{\partial y} C(x, y)$ is invertible
Theorem (Fong, Spivak & Tuyéras): Fix a learning rate $\varepsilon > 0$ and a differentiable cost function $^2 C : \mathbb{R}^2 \to \mathbb{R}$.

Then there is a symmetric monoidal functor $F_{\varepsilon, C} : \text{Para(Euc)} \to \text{Learn}$ defined by

- On objects $X \mapsto$ underlying set of $X$

---

$^2$such that every $\frac{\partial}{\partial y} C(x, y)$ is invertible
Backprop as Functor

Theorem (Fong, Spivak & Tuyéras): Fix a learning rate $\varepsilon > 0$ and a differentiable cost function $^2 C : \mathbb{R}^2 \to \mathbb{R}$.

Then there is a symmetric monoidal functor $F_{\varepsilon, C} : \text{Para}(\text{Euc}) \to \text{Learn}$ defined by

- On objects $X \mapsto$ underlying set of $X$
- On morphisms $f : P \times X \to Y$:
  - Parameters $P$
  - Implementation $I = f$

\[\text{such that every } \frac{\partial}{\partial y} C(x, y) \text{ is invertible}\]
Backprop as Functor

**Theorem** (Fong, Spivak & Tuyéras): Fix a learning rate $\varepsilon > 0$ and a differentiable cost function $^2 C : \mathbb{R}^2 \to \mathbb{R}$.

Then there is a symmetric monoidal functor $F_{\varepsilon, C} : \text{Para(Euc)} \to \text{Learn}$ defined by

- On objects $X \mapsto$ underlying set of $X$
- On morphisms $f : P \times X \to Y$:
  - Parameters $P$
  - Implementation $I = f$
  - Update $U(a, x, y) = a - \varepsilon \nabla_a E(a, x, y)$
  - Request $r(a, x, y) = \text{(too awkward to write down)}$

where $E(a, x, y) = \sum_{i=1}^{\text{dim}(Y)} C(f(p, x)_i, y_i)$ is total error

---

$^2$such that every $\frac{\partial}{\partial y} C(x, y)$ is invertible
Backprop as Functor

**Theorem** (Fong, Spivak & Tuyéras): Fix a learning rate $\varepsilon > 0$ and a differentiable cost function $^2 C : \mathbb{R}^2 \to \mathbb{R}$.

Then there is a symmetric monoidal functor $F_{\varepsilon, C} : \text{Para}(\text{Euc}) \to \text{Learn}$ defined by

- **On objects** $X \mapsto$ underlying set of $X$
- **On morphisms** $f : P \times X \to Y$:
  - Parameters $P$
  - Implementation $I = f$
  - Update $U(a, x, y) = a - \varepsilon \nabla_a E(a, x, y)$
  - Request $r(a, x, y) = (\text{too awkward to write down})$

where $E(a, x, y) = \sum_{i=1}^{\text{dim}(Y)} C(f(p, x)_i, y_i)$ is total error

Update is gradient descent, and request is backpropagation

---

$^2$such that every $\frac{\partial}{\partial y} C(x, y)$ is invertible
ML doesn’t work like that

Actual backpropagation backpropagates gradients
ML doesn’t work like that

Actual backpropagation backpropagates gradients

Request backpropagates a finite step in the gradient direction
ML doesn’t work like that

Actual backpropagation backpropagates **gradients**

Request backpropagates a **finite step** in the gradient direction

This is a hack because objects of **Learn** doesn’t have differentiable structure
ML doesn’t work like that

Actual backpropagation backpropagates gradients

Request backpropagates a finite step in the gradient direction

This is a hack because objects of Learn doesn’t have differentiable structure

(The benefit is Learn is more general than just ML)
Work in a category with finite limits

A **bundle** over $X$ is a morphism $\xymatrix{E \ar[r]_p & X}$
Bundles, Lenses & Machine Learning

Motivation
Backprop as Functor
Bundles
Putting it together

Bundles

Work in a category with finite limits

A bundle over $X$ is a morphism $E \xrightarrow{p} X$

Examples:

1. Trivial bundle $\xrightarrow{\pi_1} X$

2. Tangent bundle over a differentiable manifold $T_M \pi$

3. Cotangent bundle $T^* M \pi^*$
Work in a category with finite limits

A bundle over $X$ is a morphism $E \xrightarrow{p} X$

Examples:

1. **Trivial bundle** $X \times Y \xrightarrow{\pi_1} X$

2. **Tangent bundle** over a differentiable manifold $TM \xrightarrow{\pi} M$
Bundles

Work in a category with finite limits

A bundle over $X$ is a morphism $E \xrightarrow{p} X$

Examples:

1. **Trivial bundle**
   
   $X \times Y \xrightarrow{\pi_1} X$

2. **Tangent bundle** over a differentiable manifold
   
   $T\mathcal{M} \xrightarrow{\pi} \mathcal{M}$

3. **Cotangent bundle**
   
   $T^*\mathcal{M} \xrightarrow{\pi^*} \mathcal{M}$
Bundles over Euclidean spaces

If $X = \mathbb{R}^n$ is a Euclidean space then

- every $T_x(X) \cong X$
Bundles over Euclidean spaces

If $X = \mathbb{R}^n$ is a Euclidean space then

- every $T_x(X) \cong X$
- so, $T(X) \cong X \times X$
Bundles over Euclidean spaces

If $X = \mathbb{R}^n$ is a Euclidean space then

- every $T_x(X) \cong X$
- so, $T(X) \cong X \times X$

- so, the tangent bundle is trivial:

$$
\begin{array}{ccc}
T(X) & \downarrow \pi_2 \\
\hookrightarrow & & \\
X & & \\
\end{array}
$$
Bundles over Euclidean spaces

If $X = \mathbb{R}^n$ is a Euclidean space then

- every $T_x(X) \cong X$
- so, $T(X) \cong X \times X$

Moreover:

- every $T^*_x(X) \cong X \text{ unnaturally} \ (\text{since } X^* \cong X)$
Bundles over Euclidean spaces

If $X = \mathbb{R}^n$ is a Euclidean space then

- every $T_x(X) \cong X$
- so, $T(X) \cong X \times X$

Moreover:

- every $T^*_x(X) \cong X$ unnaturally (since $X^* \cong X$)
- so, $T^*(X) \cong X \times X$ unnaturally
Bundles over Euclidean spaces

If $X = \mathbb{R}^n$ is a Euclidean space then

- every $T_x(X) \cong X$
- so, $T(X) \cong X \times X$

Moreover:

- every $T^*_x(X) \cong X$ unnaturally (since $X^* \cong X$)
- so, $T^*(X) \cong X \times X$ unnaturally
- elements of $X \times X$ are called dual numbers
Bundles over Euclidean spaces

If $X = \mathbb{R}^n$ is a Euclidean space then

- every $T_x(X) \cong X$
- so, $T(X) \cong X \times X$

Moreover:

- every $T_x^*(X) \cong X$ unnaturally (since $X^* \cong X$)
- so, $T^*(X) \cong X \times X$ unnaturally
- elements of $X \times X$ are called dual numbers
- the cotangent bundle is unnaturally equivalent to a trivial bundle
Bundles over Euclidean spaces

If $X = \mathbb{R}^n$ is a Euclidean space then

- every $T_x(X) \cong X$
- so, $T(X) \cong X \times X$

Moreover:

- every $T_x^*(X) \cong X$ unnaturally (since $X^* \cong X$)
- so, $T^*(X) \cong X \times X$ unnaturally
- elements of $X \times X$ are called dual numbers
- the cotangent bundle is unnaturally equivalent to a trivial bundle

Nb. Euc doesn’t have finite limits, so we work in Top
Morphisms of bundles

A bundle morphism $f : E \xrightarrow{p} F \xrightarrow{q} X \times_Y F$ is:

- Morphisms $f : X \rightarrow Y$ and $f^\# : X \times Y \rightarrow E$ such that $X \times Y F \rightarrow E$ is a pullback.
- Equivalently: $f$ such that $X \times Y F \rightarrow X$ factors through $p$.

"Every algebraic geometer knows this definition" – David Spivak
Morphisms of bundles

A bundle morphism $f : X \to Y$ and $f^\# : X \times_Y F \to E$
Morphisms of bundles

A bundle morphism $f : E \rightarrow F$ is:

- Morphisms $f : X \rightarrow Y$ and $f^\# : X \times_Y F \rightarrow E$
- such that

$$
\begin{array}{ccc}
X \times_Y F & \rightarrow & F \\
\downarrow f^\# & & \downarrow q \\
E & \rightarrow & Y
\end{array}
$$

is a pullback
Morphisms of bundles

A bundle morphism $f : E \rightarrow F$ is:

- Morphisms $f : X \rightarrow Y$ and $f^\#: X \times_Y F \rightarrow E$
- such that

$$
\begin{array}{ccc}
X \times_Y F & \longrightarrow & F \\
\downarrow^{f^\#} & & \downarrow \\
E & \longrightarrow & F \\
\downarrow^{p} & & \downarrow^{q} \\
X & \longrightarrow & Y \\
\end{array}
$$

is a pullback
- Equivalently: $f$ such that $X \times_Y F \rightarrow X$ factors through $p$
Morphisms of bundles

A bundle morphism \( f : E \rightarrow F \) is:

- Morphisms \( f : X \rightarrow Y \) and \( f^\#: X \times_Y F \rightarrow E \)
- such that

\[
\begin{array}{ccc}
X \times_Y F & \rightarrow & F \\
\downarrow f^\# & & \downarrow \\
E & \rightarrow & q \\
\downarrow p & & \downarrow \\
X & \leftarrow f & \rightarrow Y
\end{array}
\]

is a pullback

- Equivalently: \( f \) such that \( X \times_Y F \rightarrow X \) factors through \( p \)

“Every algebraic geometer knows this definition” – David Spivak
The category of bundles

\[ X \times_X E \cong E \]

Identity morphism:

\[ \begin{array}{c}
E \\
\downarrow p \\
X
\end{array} \cong \begin{array}{c}
E \\
p \\
X
\end{array} \]
The category of bundles

Identity morphism:

\[ X \times_X E \cong E \]

Composition of morphisms:

\[ X \times_Z G \rightarrow X \times_Y F \rightarrow Y \times_Z G \rightarrow G \]

\[ f^*(g^#) \rightarrow f^* (g^#) \rightarrow \]

\[ E \rightarrow E \]

\[ p \rightarrow p \]

\[ X \rightarrow X \]

\[ f \rightarrow f \]

| Motivation | Backprop as Functor | Bundles | Putting it together |
Where does this come from?

From the Grothendieck construction:

$$\text{Bund}(\mathcal{C}) = \int_{X \in \mathcal{C}} (\mathcal{C}/X)^{\text{op}}$$
Where does this come from?

From the Grothendieck construction:

$$\text{Bund}(\mathcal{C}) = \int_{X \in \mathcal{C}} (\mathcal{C}/X)^{\text{op}}$$

This buys us (conjecture) a monoidal structure:

$$\begin{array}{ccc}
E & F & E \times F \\
\downarrow p \otimes & \downarrow q = & \downarrow p \times q \\
X & Y & X \times Y
\end{array}$$

(this might not be the right one!)
A (bimorphic) lens $\lambda : (S, T) \to (A, B)$ consists of:

- a morphism $\lambda_v : S \to A$ called **view**
- a morphism $\lambda_u : S \times B \to T$ called **update**
A (bimorphic) lens $\lambda : (S, T) \to (A, B)$ consists of:

- a morphism $\lambda_v : S \to A$ called view
- a morphism $\lambda_u : S \times B \to T$ called update

Composition of lenses is fiddly
A (bimorphic) lens $\lambda : (S, T) \to (A, B)$ consists of:

- a morphism $\lambda_v : S \to A$ called view
- a morphism $\lambda_u : S \times B \to T$ called update

Composition of lenses is fiddly

Where does this come from? The Grothendieck construction:

$$\text{Lens}(C) = \int_{X \in C} \text{coKl}(X \times -)^{op}$$
A (bimorphic) lens $\lambda : (S, T) \rightarrow (A, B)$ consists of:

- a morphism $\lambda_v : S \rightarrow A$ called view
- a morphism $\lambda_u : S \times B \rightarrow T$ called update

Composition of lenses is fiddly

Where does this come from? The Grothendieck construction:

$$\text{Lens}(C) = \int_{X \in C} \text{coKl}(X \times -)^{op}$$

**Theorem** (Lambek): $\text{coKl}(X \times -) \cong C[x]$, where $C[x]$ is the polynomial category formed by freely adjoining $x : 1 \rightarrow X$ and closing under finite products.
Lenses are bundle morphisms

Another theorem (Lambek): \( \text{coEM}(X \times -) \cong C/X \)
Lenses are bundle morphisms

Another theorem (Lambek): $\text{coEM}(X \times -) \cong \mathcal{C}/X$

So there is a canonical embedding $\mathcal{C}[x] \hookrightarrow \mathcal{C}/X$
Lenses are bundle morphisms

Another theorem (Lambek): $\text{coEM}(X \times -) \cong C/X$

So there is a canonical embedding $C[x] \hookrightarrow C/X$

Grothendieck them all together: $\text{Lens}(C) \to \text{Bund}(C)$
It takes a lens $\lambda : (S, T) \to (A, B)$ to the bundle morphism
Lenses are bundle morphisms

Another theorem (Lambek): \( \text{coEM}(X \times -) \cong \mathcal{C}/X \)

So there is a canonical embedding \( \mathcal{C}[x] \hookrightarrow \mathcal{C}/X \)

Grothendieck them all together: \( \text{Lens}(\mathcal{C}) \rightarrow \text{Bund}(\mathcal{C}) \)

It takes a lens \( \lambda : (S, T) \rightarrow (A, B) \) to the bundle morphism

\[
\begin{array}{ccc}
S \times B & \longrightarrow & A \times B \\
\downarrow_{\langle \pi_1, \lambda \nu \rangle} & & \downarrow \\
S \times T & \longrightarrow & \pi_2 \\
\downarrow_{\pi_2} & & \downarrow \pi_2 \\
S & \longrightarrow & A \\
\lambda \nu & & \\
\end{array}
\]
Morphisms of cotangent bundles

There is a functor $\text{Cot}(-) : \text{DiffMfd} \rightarrow \text{Bund}(\text{Top})$
Morphisms of contangent bundles

There is a functor \( \text{Cot}(\_): \text{DiffMfd} \rightarrow \text{Bund}(\text{Top}) \)

It takes \( f : X \rightarrow Y \) to

\[
\begin{array}{ccc}
X \times_Y T^*(Y) & \longrightarrow & T^*(Y) \\
\downarrow_{f'} & & \downarrow \\
T^*(X) & \downarrow_{\pi^*} & \\
\downarrow_{\pi^*} & & \downarrow \\
X & \longrightarrow & Y
\end{array}
\]

where \( f' : (x, c) \mapsto (x, c \circ J_x(f)) \)
Morphisms of contangent bundles

There is a functor \( \text{Cot}(\cdot) : \text{DiffMfd} \to \text{Bund}(\text{Top}) \)

It takes \( f : X \to Y \) to

\[
\begin{array}{ccc}
X \times_Y T^*(Y) & \longrightarrow & T^*(Y) \\
\downarrow f' & & \downarrow \\
T^*(X) & \longrightarrow & T^*(Y) \\
\downarrow \pi^* & & \downarrow \\
X & \longrightarrow & Y
\end{array}
\]

where \( f' : (x, c) \mapsto (x, c \circ J_x(f)) \)

\( J_x(f) \) is the Jacobian (matrix of partial derivatives) of \( f \) at \( x \)
The chain rule

Functorality of $\text{Cot}(-)$:

\[
\begin{align*}
X \times_Z T^*(Z) & \xrightarrow{f^*(g')} Y \times_Z T^*(Z) & \xrightarrow{g'} T^*(Z) \\
X \times_Y T^*(Y) & \xrightarrow{f'} T^*(Y) & \xrightarrow{\pi^*} T^*(Y) \\
T^*(X) & \xrightarrow{\pi^*} & \\
X & \xrightarrow{f} & Y & \xrightarrow{g} & Z
\end{align*}
\]

$\pi^*(f^*(g')) \circ \pi^* = f^* \circ g'$ is the chain rule in differential geometry.
Functorality of $\text{Cot}(-)$:

\[
\begin{align*}
X \times Z & \xrightarrow{T^*(Z)} Y \times Z \xrightarrow{T^*(Z)} T^*(Z) \\
X \times Y & \xrightarrow{T^*(Y)} T^*(Y) \\
X & \xrightarrow{T^*(X)} Y \xrightarrow{T^*(Y)} Z
\end{align*}
\]

\[(g \circ f)' = f' \circ f^*(g')\] is the chain rule in differential geometry.
Consider a morphism of \( \text{Para}(\text{Bund}(\text{Top})) \) in the image of

\[
\text{Para}(	ext{Cot}) : \text{Para}(	ext{Euc}) \to \text{Para}(\text{Bund}(\text{Top}))
\]

It looks like

\[
(X \times A) \times_Y T^*(Y) \xrightarrow{f'} \rightarrow Y
\]

We're going to turn it into an open learner, given \( \varepsilon > 0 \) and differentiable \( C : \mathbb{R}^2 \to \mathbb{R} \).
From \( \text{Para}(\text{Bund}(\text{Top})) \) to \textbf{Learn}

Consider a morphism of \( \text{Para}(\text{Bund}(\text{Top})) \) in the image of

\[ \text{Para}(\text{Cot}) : \text{Para}(\text{Euc}) \rightarrow \text{Para}(\text{Bund}(\text{Top})) \]

It looks like

\[
\begin{array}{ccc}
(X \times A) \times_Y T^*(Y) & \longrightarrow & Y \\
\downarrow^{f'} & & \downarrow \\
T^*(X \times A) & \underset{\pi^*}{\longrightarrow} & Y
\end{array}
\]

We’re going to turn it into an open learner, given \( \varepsilon > 0 \) and differentiable \( C : \mathbb{R}^2 \rightarrow \mathbb{R} \)
Obviously, **parameters** are $A$ and **implementation** is $f$
The setup

Obviously, parameters are $A$ and implementation is $f$

We need to define $\langle U, r \rangle : A \times X \times Y \rightarrow A \times X$

so, fix $a \in A$, $x \in X$ and $y \in Y$

and fix the total error $C_y(y') = \sum_{i=1}^{\dim(Y)} C(y_i, y'_i)$
The setup

Obviously, parameters are $A$ and implementation is $f$

We need to define $\langle U, r \rangle : A \times X \times Y \to A \times X$

so, fix $a \in A$, $x \in X$ and $y \in Y$

and fix the total error $C_y(y') = \sum_{i=1}^{\dim(Y)} C(y_i, y'_i)$

Consider the diagram...
The brain exploding part

\[ \mathbb{R} \cong T_{C_y(f(x,a))}(\mathbb{R}) \rightarrow (X \times A) \times_\mathbb{R} T^*(\mathbb{R}) \rightarrow Y \times_\mathbb{R} T^*(\mathbb{R}) \rightarrow T^*(\mathbb{R}) \]

\[ f^*(C'_y) \]

\[ T^*_f(x,a)(Y) \rightarrow (X \times A) \times_Y T^*(Y) \rightarrow T^*(Y) \]

\[ f' \]

\[ T^*_f(x,a)(X \times A) \leftarrow T^*(X \times A) \]

\[ 1 \cong T^*(1) \]

\[ (x,a) \rightarrow X \times A \rightarrow Y \rightarrow C_y \rightarrow \mathbb{R} \]
The part we don’t understand

Now: Chase $1 \in \mathbb{R}$ to $T^*(X \times A)$ and then apply

$$\mu_\varepsilon : T^*(X \times A) \rightarrow X \times A$$

The result is $\langle r, U \rangle (a, x, y)$
The part we don’t understand

Now: Chase $1 \in \mathbb{R}$ to $T^*(X \times A)$ and then apply

$$\mu_\varepsilon : T^*(X \times A) \rightarrow X \times A$$

The result is $\langle r, U \rangle (a, x, y)$

$\mu_\varepsilon$ takes a finite step in the gradient direction:

$$\mu_\varepsilon((x, a), (v, w)) = (x + v, a + \varepsilon w)$$
The part we don’t understand

Now: Chase $1 \in \mathbb{R}$ to $T^*(X \times A)$ and then apply

$$\mu_\varepsilon : T^*(X \times A) \to X \times A$$

The result is $\langle r, U \rangle (a, x, y)$

$\mu_\varepsilon$ takes a finite step in the gradient direction:

$$\mu_\varepsilon((x, a), (v, w)) = (x + v, a + \varepsilon w)$$

What is $\mu_\varepsilon$? We couldn’t find any nice properties
The part we don’t understand

Now: Chase $1 \in \mathbb{R}$ to $T^*(X \times A)$ and then apply

$$\mu_\varepsilon : T^*(X \times A) \rightarrow X \times A$$

The result is $\langle r, U \rangle (a, x, y)$

$\mu_\varepsilon$ takes a finite step in the gradient direction:

$$\mu_\varepsilon((x, a), (v, w)) = (x + v, a + \varepsilon w)$$

What is $\mu_\varepsilon$? We couldn’t find any nice properties

It looks a bit like a thing called an exponential map
The catch

Conjecture: This defines a symmetric monoidal functor

\[ \text{Para}(\text{Bund}(\text{Top})) \supseteq \text{Im}(\text{Para}(\text{Cot})) \to \text{Learn} \]
The catch

**Conjecture:** This defines a symmetric monoidal functor

\[ \text{Para}(\text{Bund}(\text{Top})) \supseteq \text{Im}(\text{Para}(\text{Cot})) \to \text{Learn} \]

**Another conjecture:** This commutes:

\[ \begin{array}{ccc}
\text{Para}(\text{Euc}) & \xrightarrow{\text{Para}(\text{Cot})} & \text{Im}(\text{Para}(\text{Cot})) \\
\downarrow & & \downarrow \\
\text{Learn} & & \text{Learn}
\end{array} \]
The catch

**Conjecture:** This defines a symmetric monoidal functor

\[ \text{Para}(\text{Bund}(\text{Top})) \supseteq \text{Im}(\text{Para}(\text{Cot})) \rightarrow \text{Learn} \]

**Another conjecture:** This commutes:

\[ \text{Para}(\text{Euc}) \xrightarrow{\text{Para}(\text{Cot})} \text{Im}(\text{Para}(\text{Cot})) \rightarrow \text{Learn} \]

**The catch:** We think \( \text{Para}(\text{Cot}) \) is an equivalence of categories onto its image
The catch

**Conjecture:** This defines a symmetric monoidal functor

\[ \text{Para}(\text{Bund}(\text{Top})) \supseteq \text{Im}(\text{Para}(\text{Cot})) \rightarrow \text{Learn} \]

**Another conjecture:** This commutes:

\[ \text{Para}(\text{Euc}) \xrightarrow{\text{Para}(\text{Cot})} \text{Im}(\text{Para}(\text{Cot})) \]

\[ \downarrow \]

\[ \text{Learn} \]

**The catch:** We think \( \text{Para}(\text{Cot}) \) is an equivalence of categories onto its image

So, we’ve just rewritten Backprop as Functor in a different way!
Even more hard questions

What happens if we extend the functor to the whole of $\text{Para}(\text{Bund}(\text{Top}))$? We have no idea!

Optimistic hope: This allows defining general “ML-like” systems, not necessarily involving gradients (eg. “discrete ML” on Bayesian networks)