# The Poly-shaped ingredients of predictive coding

AUTHOR
Toby St Clere Smithe

PUBLISHED
March 1, 2024

ABSTRACT
Intelligent systems in the world seem to make predictions, with some uncertainty. How do they do this? Some of the ingredients of this story are Poly-shaped, and we sketch them here.

## 1 Polynomials with stochastic feedback

The form of **Poly** with which we may be most familiar can be obtained by constructing *Grothendieck lenses* from the self-indexing $\mathbf{Set}/-$ of **Set**.

**Definition 1** $\mathbf{Set}/-$ is an indexed category $\mathbf{Set}^{\text{op}} \to \mathbf{Set}$. It maps a set $B$ to the slice category $\mathbf{Set}/B$ whose objects are pairs $(E, p)$ of a set $E$ and a function (or 'bundle') $p : E \to B$. The morphisms $(E, p) \to (E', p')$ of $\mathbf{Set}/B$ are functions $\varphi : E \to E'$ such that $p' \circ \varphi = p$. Given a function $f : A \to B$, $\mathbf{Set}/f$ is the functor $\mathbf{Set}/B \to \mathbf{Set}/A$ which acts by pullback, sometimes written $f^*$. □

**Proposition 1** The category of *Grothendieck lenses* $\mathbf{Lens}(\mathbf{Set}/-)$ in $\mathbf{Set}/-$ is **Poly**. (This is why **Poly** is sometimes known as a category of dependent lenses.) □

We don't have to construct dependent lenses in **Set**, however. If we choose another fibration (another model of dependent types), we will obtain another category of dependent lenses, and this may behave much like **Poly**. (For example, it will often have a tensor product $\otimes$ — and more structures, too!)

In particular, we can define an indexed category whose Grothendieck lenses will behave like "**Poly** with stochastic feedback". We can do this in great generality, using the notion of *kernel* between measurable spaces.

**Definition 2** If $X, Y$ are measurable spaces, a *kernel* $k : X \rightsquigarrow Y$ is a function $k : X \times \Sigma_Y \to [0, \infty]$ which is measurable in the first argument $X$ and which is an 's-finite' measure in the second argument $\Sigma_Y$, the $\sigma$-algebra associated to $Y$. □

**Proposition 2** Measurable spaces and kernels between them form the objects and morphisms of a category **Krn**. Composition $X \overset{k}{\rightsquigarrow} Y \overset{h}{\rightsquigarrow} Z$ is given by the Chapman-Kolmogorov equation:

$$h \circ k : (x, C) \mapsto \int_{y:Y} g(x, \mathrm{d}y)\, h(y, C)\ .$$

Identity kernels map points to Dirac delta ('indicator') measures. □

**Proposition 3** There is an embedding $\delta : \mathbf{Meas} \to \mathbf{Krn}$ of measurable functions into kernels. A measurable function $f : X \to Y$ is mapped to the kernel $\delta_f : X \rightsquigarrow Y$ defined by

$$(x, U) \mapsto [f(x) \in U]$$

where the indicator $[f(x) \in U]$ equals 1 if $f(x) \in U$ and 0 otherwise. □

**Proposition 4** There is an indexed category $\mathcal{K} : \mathbf{Meas}^{\mathrm{op}} \to \mathbf{Cat}$ defined by mapping a measurable space $B$ to the category $\mathcal{K}_B$ whose objects are pairs $(E, p)$ of a measurable space $E$ and a measurable function $p : E \to B$. The morphisms $(E, p) \rightsquigarrow (E', p')$ of $\mathcal{K}_B$ are kernels $k : E \rightsquigarrow E'$ such that $\delta'_p \circ k = \delta_p$. This means that $k$ is a kernel *fibrewise*: for any generalized element $b : I \to B$ in $\mathbf{Meas}$, $k$ yields a kernels $k[b] : p[b] \rightsquigarrow p'[b]$, where $p[b]$ is the pullback object of $p$ along $b$. Given a function $f : A \to B$, there is a functor $\mathcal{K}_f : \mathcal{K}_B \to \mathcal{K}_A$ which maps objects to their pullback along $f$ and which acts on kernels $k$ to yield kernels $k[f]$ that are defined fibrewise by $k[f(a)]$. □

The category of Grothendieck lenses in $\mathcal{K}$ behaves something like **Poly** where the backward components of morphisms may be stochastic.

**Proposition 5** The objects of $\mathbf{Lens}(\mathcal{K})$ are pairs $(B, p)$ of a measurable space $B$ and an object $p$ of $\mathcal{K}_B$. We can write these simply as $p$, by defining $p(1)$ to denote the base object $B$. We can likewise define formal notation $\sum_{i:p(1)} p[i]$ to denote the total space $E$ of $p$, thinking of $i$ as a generalized element $i : I \to p(1)$ of the base of $p$. Finally, we can follow this formal intuition further, to write $p$ itself as $\sum_{i:p(1)} y^{p[i]}$.

The morphisms $\varphi : p \to q$ of $\mathbf{Lens}(\mathcal{K})$ are then pairs $(\varphi_1, \varphi^\sharp)$ of a measurable function $\varphi_1 : p(1) \to q(1)$ and a fibrewise kernel $\varphi^\sharp[i] : q[\varphi_1(i)] \rightsquigarrow p[i]$ — *i.e.*, a kernel $q[\varphi_1] \rightsquigarrow p$ in $\mathcal{K}_{p(1)}$. Composition of morphisms in $\mathbf{Lens}(\mathcal{K})$ is as in **Poly**. □

*Remark*. The fibration $\mathcal{K}$ is actually a bifibration, meaning that each functor $\mathcal{K}_f$ has a left adjoint, $\Sigma_f$. These left adjoints formally justify the foregoing $\Sigma$ notation. □

**Example 1** A $p$-coalgebra in $\mathbf{Lens}(\mathcal{K})$ is a pair $(S, \theta)$ of a measurable space $S$ and a morphism $\theta : Sy^S \to p$ in $\mathbf{Lens}(\mathcal{K})$. This is a stochastic dependent Moore machine: a pair of an 'output' function $\theta_1 : S \to p(1)$ and a family of update kernels $\theta^\sharp_s : p[\theta_1(s)] \rightsquigarrow S$ for each $s \in S$. A morphism of $p$-coalgebras is a measurable function that commutes with the dynamics. □

**Proposition 6** There is an opindexed category $\mathsf{Coalg} : \mathbf{Lens}(\mathcal{K}) \to \mathbf{Cat}$ which maps each polynomial $p$ to the category $\mathsf{Coalg}(p)$ of stochastic dependent Moore machines. Given a morphism $\varphi : p \to p'$ in $\mathbf{Lens}(\mathcal{K})$, $\mathsf{Coalg}(\varphi)$ acts by post-composition. □

## 1.1 Random variables and 'quasi-Borel' kernels

During the workshop, David Spivak pointed out to me that we can still work with measurable spaces[1] and stay within the usual **Poly** defined within **Set**, by adopting an analogue of the monad $\mathsf{lott} := \sum_{n:\mathbb{N}} \sum_{d:\Delta(n)} y^n$ (here $\Delta(n)$ is the set of distributions on the finite set with size $n$). We could call this polynomial $\mathsf{rand}$, as it captures a very broad notion of *random variable*:

$$\mathsf{rand} := \sum_{X:\mathbf{Meas}} \sum_{\mu:MX} y^X$$

where $MX$ denotes the set of (s-finite) measures on $X$.

We did not prove that $\mathsf{rand}$ is a monad in **Poly**, although it seems likely to be, but we can still gain an intuition for its Kleisli morphisms. These behave a little like a possibly more familiar notion: quasi-Borel kernels.

A quasi-Borel space is a set $X$ along with a subset of $X^{\mathbb{R}}$ that we could think of as random variables on $X$ (satisfying some 'sheaf' axioms). Quasi-Borel spaces form a category **QBS**, with which we can define a notion of kernel.

**Definition 3** A *quasi-Borel kernel* $k : X \rightsquigarrow Y$ between quasi-Borel spaces $X$ and $Y$ is a function $k : X \to \mathbf{QBS}(\mathbb{R}, Y)$. Two quasi-Borel kernels $k, k' : X \rightsquigarrow Y$ are considered equal if pushing forward the uniform measure on the unit interval yields the same measure on $Y$ for all $x \in X$. (To obtain a notion of s-finite quasi-Borel kernel, we can work with 'partial' functions $X \to \mathbf{QBS}(\mathbb{R}, Y + 1)$ instead, and push forward the uniform measure on $\mathbb{R}$. But we will not concern ourselves with such details here.) □

Quasi-Borel spaces and kernels between them form a category **qbKrn**. Composition ends up being much as in **Krn** — but again we will not spell out the details.

Now consider a morphism $Ay \to \mathsf{rand} \triangleleft By$ in **Poly**. This consists of a function $A \to \sum_X \sum_\mu B^X$ — that is, a function that returns a combination of a "sample space" $X$ with a "noise source" $\mu$ and a "random variable" in $B^X$ from the sample space $X$ to $B$. Notice that this generalizes the quasi-Borel picture: there, a kernel yields a random variable from a privileged sample space $\mathbb{R}$ equipped with the uniform noise source; now, there is freedom to choose.

## 2 Animated and dynamic categories

Using some of the ingredients sketched so far, we can render categories 'dynamical'. For our purposes, a *systems theory* will be an opindexed category defined on a *category of interfaces*, which will usually be **Poly** or another lens category, such as **Lens**($\mathcal{K}$). Thus, the opindexed category Coalg above is a systems theory.

If we have a category enriched in a category of interfaces, we can *change its base of enrichment* along a corresponding systems theory, which will yield a bicategory whose objects are the objects of the starting category, whose 1-cells are dynamical systems on the 'hom' interfaces, and whose 2-cells are the corresponding morphisms of those systems. This process, of turning a category into a bicategory of dynamical systems, is what I call *animation*.

**Example 2 (Animating monoidal categories)** Given any monoidal category $(\mathscr{C}, I, \otimes)$, we can write down a corresponding **Poly**-enriched category $\mathscr{C}_P$ which has the same objects as $\mathscr{C}$. Between every pair $(A, B)$ of objects in $\mathscr{C}$, we have a hom polynomial $\mathscr{C}(A, B)\, y^{\mathscr{C}(I,A)}$. For every triple $(A, B, C)$ of objects, there is a composition morphism $\mathscr{C}(B, C)\, y^{\mathscr{C}(I,B)} \otimes \mathscr{C}(A, B)\, y^{\mathscr{C}(I,A)}$ which acts in the forward direction by composition and in the backward direction, given $(g, f) \in \mathscr{C}(B, C) \times \mathscr{C}(A, B)$ by mapping $s : \mathscr{C}(I, A)$ to $(f \circ s, s)$.

If we choose $(\mathbf{Set}, 1, \times)$ as our monoidal category, and animate $\mathbf{Set}_P$ along Coalg, then we obtain a bicategory whose 1-cells $A \to B$ are Mealy machines $Sy^S \to B^A\, y^A$. □

**Example 3 (Org)** Note that **Poly** itself is **Poly**-enriched, via its internal hom. Thus, we can change *its* base along Coalg, to obtain a bicategory of dynamical systems on hom polynomials. This bicategory is sometimes known as **Org**. □

A *dynamic category* is then a category enriched in the bicategory **Org**. We can generalize this using animation.

**Definition 4** A *generalized dynamic category* is a category enriched in a bicategory resulting from some animation.

*Remark*. In discussion with Matteo Capucci and Sophie Libkind, it was observed that every animation yields a monad in **Prof** (the bicategory of categories and profunctors). Such monads can be thought of as *bidirectional systems theories* (and may be accordingly generalized). Hopefully, other artifacts from the workshop discuss this!

discuss this.

## 3 Predictive coding, via animation

It is known that deep learning systems yield dynamic categories (in **Org**), and predictive coding feels similar, except that instead of passing forward vectors, one passes forward 'beliefs'; and instead of passing backward tangent vectors, one passes back prediction errors.

It turns out that we can think of predictive coding as a generalized form of deep learning: but instead of working in Euclidean spaces, we must work with *information geometry*. In this way, we can define a *predictive coding* dynamic category (in **Org**, or perhaps a stochastic variant of it, obtained from $\mathbf{Lens}(\mathcal{K})$).

A dynamic category (really, dynamic 'operad') in **Org** is defined first by a base polynomial $t$. In the case of deep learning, this is $\mathbb{R}\,y^{\mathbb{R}}$, where we think of the exponent as the tangent space to each point of $\mathbb{R}$.

In the case of predictive coding, we fix a type of belief that the system predicts with: usually, this is Gaussians (with fixed variance $\sigma$). Thus, we can define a base polynomial $\sum_{\gamma:\mathbf{Gauss}(\mathbb{R})} y^{\mathsf{Tan}_\gamma \mathbf{Gauss}(\mathbb{R})}$. The set of tangent vectors $\mathsf{Tan}_\gamma \mathbf{Gauss}(\mathbb{R})$ at a given Gaussian $\gamma$ with mean $\mu_\gamma$ is the set of functions $x \mapsto \sigma^{-1}(\mu_\gamma - x)$ — *i.e.*, the set of "precision-weighted" prediction errors, given the prediction $\mu_\gamma$.

With this base polynomial, the dynamic category story for deep learning may be re-told — but now one obtains *predictive coding*! There are some details to fix, however, which we have no space (or time!) to detail here! For example, we need to define a weakening of the dynamic category structure, in order to incorporate interesting correlations.

A sketch of this story — and another variant, making use of animation primarily (rather than dynamic categories) — may be found in the bonus slides of my DPhil viva presentation.

---

### Footnotes

1. I will ignore size issues here. ↩

### Reuse