Approximating likelihood ratios with calibrated classifiers

Gilles Louppe

April 13, 2016
Joint work with

Kyle Cranmer
New York University

Juan Pavez
Federico Santa María University

See paper (Cranmer et al., 2015) for full details.
Studying the constituents of the universe

The thing is, we have this collider...

The magic of a collider is that you can make kinds of matter that you don't have around.

You take two kinds of particles and annihilate them...

What comes out doesn't have to be a re-arrangement of what went in.

It's a kind of quantum magic where it sort of disappears into pure energy... *

You can make any sort of particle for which you have enough energy.

* a force-carrying boson

It's like having a menu...

What can I get in the 500 GeV range?

You can make anything that costs that much energy or less.

That's why you want to have as much energy as possible.

Every time you crank up the energy, you could be exploring a whole new regime.

(c) Jorge Cham
Collecting data

DETECTING THE HIGGS BOSON

FIRST, THE COLLISION HAPPENS...

IT LASTS FOR 0.000000000000000001 SECONDS...

...AND YOU GET ONE MEASUREMENT OF THE (BOTTOM QUARK) DECAY PRODUCTS.

THEN YOU PLOT THE TOTAL ENERGY...

...AND YOU COUNT HOW MANY COLLISIONS HAPPEN FOR EACH ENERGY LEVEL:

AND YOU BUILD UP YOUR DATA...

(c) Jorge Cham
Testing for new physics

THEN YOU HAVE 2 THEORIES THAT PREDICT THE DATA:

NO HIGGS BOSON

YES HIGGS BOSON

THE PROBLEM IS THAT THE DIFFERENCE BETWEEN THE TWO IS VERY SMALL

WHAT YOU NEED IS A HUGE AMOUNT OF DATA.

THAT’S WHY WE RUN THIS THING 40 MILLION TIMES/SECOND, ALL DAY, ALL YEAR.

TO TELL SMALL DIFFERENCES BETWEEN THEORIES.

OPEN 24 HOURS

Over a Biillion Collisions Served

\[
p(\text{data} | \text{theory} + X) \over p(\text{data} | \text{theory})
\]
Likelihood-free setup

- Complex simulator $p$ parameterized by $\theta$;
- Samples $x \sim p$ can be generated on-demand;
- ... but the likelihood $p(x|\theta)$ cannot be evaluated!

$p =$
Simple hypothesis testing

- Assume some observed data $\mathcal{D} = \{x_1, \ldots, x_n\}$;
- Test a null $\theta = \theta_0$ against an alternative $\theta = \theta_1$;
- The Neyman-Pearson lemma states that the most powerful test statistic is

$$\lambda(\mathcal{D}; \theta_0, \theta_1) = \prod_{x \in \mathcal{D}} \frac{p_X(x|\theta_0)}{p_X(x|\theta_1)}.$$

- ... but neither $p_X(x|\theta_0)$ nor $p_X(x|\theta_1)$ can be evaluated!
Straight approximation

1. Approximate $p_X(x|\theta_0)$ and $p_X(x|\theta_1)$ individually, using density estimation algorithms;

2. Evaluate their ratio $r(x; \theta_0, \theta_1)$.

Works fine for low-dimensional data, but because of the curse of dimensionality, this is in general a difficult problem! Moreover, it is not even necessary!

$$\frac{p_X(x|\theta_0)}{p_X(x|\theta_1)} = r(x; \theta_0, \theta_1)$$

When solving a problem of interest, do not solve a more general problem as an intermediate step. – Vladimir Vapnik
Likehood ratio invariance under change of variable

*Theorem.* The likelihood ratio is invariant under the change of variable $U = s(X)$, provided $s(x)$ is monotonic with $r(x)$.

$$r(x) = \frac{p_X(x|\theta_0)}{p_X(x|\theta_1)} = \frac{p_U(s(x)|\theta_0)}{p_U(s(x)|\theta_1)}$$
Approximating likelihood ratios with classifiers

- Well, a classifier trained to distinguish $\mathbf{x} \sim p_0$ from $\mathbf{x} \sim p_1$ approximates
  \[
s^*(\mathbf{x}) = \frac{p_{\mathbf{x}}(\mathbf{x}|\theta_1)}{p_{\mathbf{x}}(\mathbf{x}|\theta_0) + p_{\mathbf{x}}(\mathbf{x}|\theta_1)},
  \]
  which is monotonic with $r(\mathbf{x})$.

- Estimating $p(s(\mathbf{x})|\theta)$ is now easy, since the change of variable $s(\mathbf{x})$ projects $\mathbf{x}$ in a 1D space, where only the informative content of the ratio is preserved.
  - This can be carried out using density estimation or calibration algorithms (histograms, KDE, isotonic regression, etc).

- Disentangle training from calibration.
Inference and composite hypothesis testing

Approximated likelihood ratios can be used for inference, since

\[
\hat{\theta} = \arg\max_{\theta} p(D|\theta)
\]

\[
= \arg\max_{\theta} \prod_{x \in D} \frac{p(x|\theta)}{p(x|\theta_1)}
\]

\[
= \arg\max_{\theta} \prod_{x \in D} \frac{p(s(x; \theta, \theta_1)|\theta)}{p(s(x; \theta, \theta_1)|\theta_1)} \tag{1}
\]

where \(\theta_1\) is fixed and \(s(x; \theta, \theta_1)\) is a family of classifiers parameterized by \((\theta, \theta_1)\).

Accordingly, generalized (or profile) likelihood ratio tests can be evaluated in the same way.
Parameterized learning

For inference, we need to build a family $s(x; \theta, \theta_1)$ of classifiers.

- One could build a classifier $s$ independently for all $\theta, \theta_1$. But this is computationally expensive and would not guarantee a smooth evolution of $s(x; \theta, \theta_1)$ as $\theta$ varies.
- Solution: build a single parameterized classifier instead, where parameters are additional input features (Cranmer et al., 2015; Baldi et al., 2016).

\[
\mathcal{T} := \{\};
\]
\[
\textbf{while } \text{size}(\mathcal{T}) < N \textbf{ do}
\]
\[
\text{Draw } \theta_0 \sim \pi_{\Theta_0};
\]
\[
\text{Draw } x \sim p(x|\theta_0);
\]
\[
\mathcal{T} := \mathcal{T} \cup \{(x, \theta_0, \theta_1), y = 0\};
\]
\[
\text{Draw } \theta_1 \sim \pi_{\Theta_1};
\]
\[
\text{Draw } x \sim p(x|\theta_1);
\]
\[
\mathcal{T} := \mathcal{T} \cup \{(x, \theta_0, \theta_1), y = 1\};
\]
\[
\textbf{end while}
\]
Learn a single classifier $s(x; \theta_0, \theta_1)$ from $\mathcal{T}$.
Example: Inference from multidimensional data

Let assume 5D data $x$ generated from the following process $p_0$:

1. $z := (z_0, z_1, z_2, z_3, z_4)$, such that
   $z_0 \sim \mathcal{N}(\mu = \alpha, \sigma = 1)$,
   $z_1 \sim \mathcal{N}(\mu = \beta, \sigma = 3)$,
   $z_2 \sim \text{Mixture}(\frac{1}{2} \mathcal{N}(\mu = -2, \sigma = 1), \frac{1}{2} \mathcal{N}(\mu = 2, \sigma = 0.5))$,
   $z_3 \sim \text{Exponential}(\lambda = 3)$, and
   $z_4 \sim \text{Exponential}(\lambda = 0.5)$;

2. $x := Rz$, where $R$ is a fixed semi-positive definite $5 \times 5$ matrix defining a fixed projection of $z$ into the observed space.

Our goal is to infer the values $\alpha$ and $\beta$ based on $D$.

Check out (Louppe et al., 2016) to reproduce this example.
Example: Inference from multidimensional data

Recipe:

1. Build a single parameterized classifier \( s(x; \theta_0, \theta_1) \), in this case a 2-layer NN trained on 5+2 features, with the alternative fixed to \( \theta_1 = (\alpha = 0, \beta = 0) \).

2. Find the approximated MLE \( \hat{\alpha}, \hat{\beta} \) by solving Eqn. 1.
   - Solve Eqn. 1 using likelihood scans or through optimization.
   - Since the generator is inexpensive, \( p(s(x; \theta_0, \theta_1)|\theta) \) can be calibrated on-the-fly, for every candidate \( (\alpha, \beta) \), e.g. using histograms.

3. Construct the log-likelihood ratio (LLR) statistic

\[
-2 \log \Lambda(\alpha, \beta) = -2 \log \frac{p(D|\alpha, \beta)}{p(D|\hat{\alpha}, \hat{\beta})}
\]
Exact $-2 \log \Lambda(\alpha, \beta)$

Approx. LLR (smoothed by a Gaussian Process)

$\alpha = 1, \beta = -1$

Exact MLE

Approx. MLE
Diagnostics

In practice $\hat{r}(\hat{s}(x; \theta_0, \theta_1))$ will not be exact. Diagnostic procedures are needed to assess the quality of this approximation.

1. For inference, the value of the MLE $\hat{\theta}$ should be independent of the value of $\theta_1$ used in the denominator of the ratio.

2. Train a classifier to distinguish between unweighted samples from $p(x|\theta_0)$ and samples from $p(x|\theta_1)$ weighted by $\hat{r}(\hat{s}(x; \theta_0, \theta_1))$. 

![Graph showing comparison of exact and approximate calculations.]
Density ratio estimation

Approximating likelihood ratios relates to many other fundamental statistical inference problems, including

- transfer learning,
- outlier detection,
- divergence estimation,
- ...

Transfer learning: $p_{\text{train}} \neq p_{\text{test}}$

As training data increases, i.e. as $N \to \infty$,

$$\frac{1}{N} \sum_{x_i} L(\varphi(x_i)) \to \int L(\varphi(x)) p_{\text{train}}(x) \, dx.$$  

We want to be good on test data, i.e., minimize

$$\int L(\varphi(x)) p_{\text{test}}(x) \, dx.$$  

Solution: importance weighting.

$$\varphi^* = \arg \min_{\varphi} \frac{1}{N} \sum_{x_i} \frac{p_{\text{test}}(x_i)}{p_{\text{train}}(x_i)} L(\varphi(x_i))$$
Summary

• We proposed an approach for approximating LR in the likelihood-free setup.

• Evaluating likelihood ratios reduces to supervised learning. Both problems are deeply connected.

• Alternative to Approximate Bayesian Computation, without the need to define a prior over parameters.
