Approximating likelihood ratios with calibrated classifiers

Gilles Louppe

June 22, 2016
MLHEP, Lund

NYU

Joint work with

Kyle Cranmer

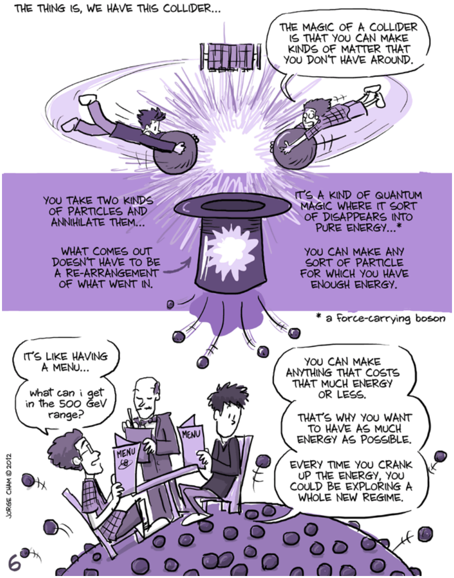
New York University

Juan Pavez

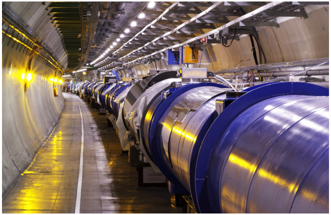
Federico Santa María University

See paper (Cranmer et al., 2015) for full details.

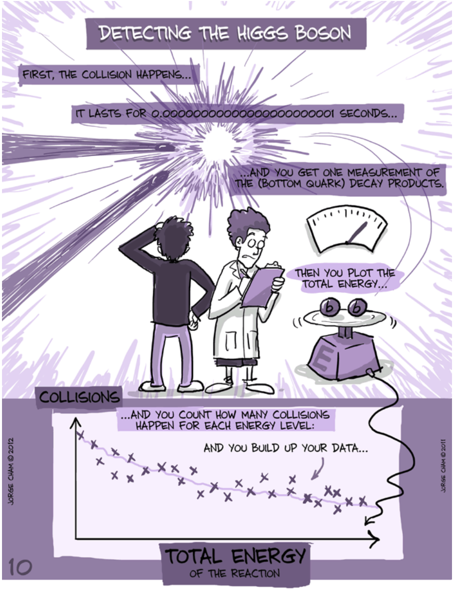
Studying the constituents of the universe



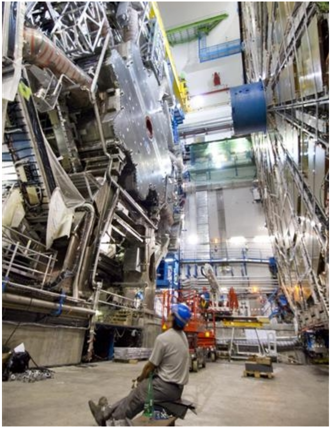
(c) Jorge Cham

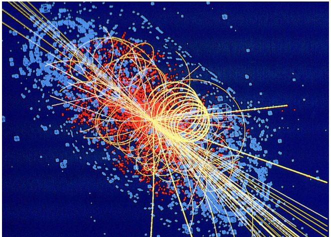


Collecting data

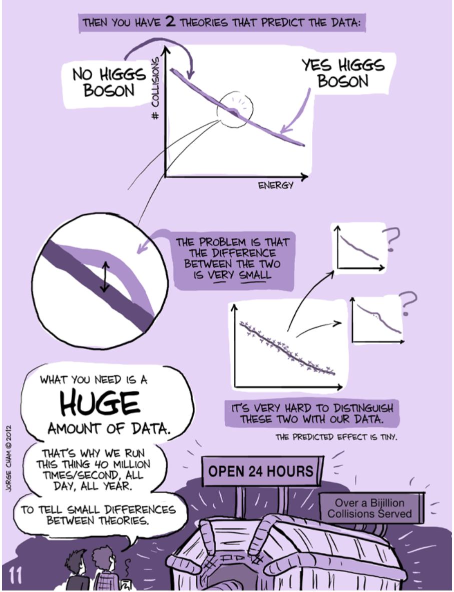


(c) Jorge Cham

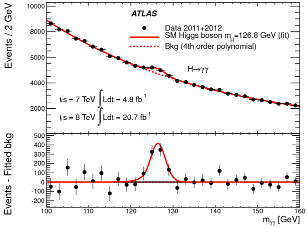




Testing for new physics



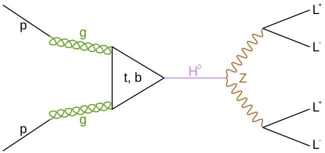
(c) Jorge Cham

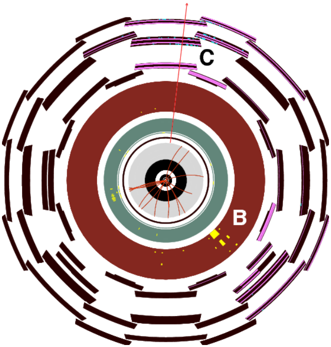


$$\frac{p(\text{data}|\text{theory}+X)}{p(\text{data}|\text{theory})}$$

Likelihood-free setup

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

$p =$ 

$\otimes$ 

Simple hypothesis testing

- Assume some observed data $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$;
- Test a null $\theta = \theta_0$ against an alternative $\theta = \theta_1$;
- The Neyman-Pearson lemma states that the most powerful test statistic is

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

- ... but neither $p_{\mathbf{X}}(\mathbf{x}|\theta_0)$ nor $p_{\mathbf{X}}(\mathbf{x}|\theta_1)$ can be evaluated!

Straight approximation

1. Approximate $p_{\mathbf{X}}(\mathbf{x}|\theta_0)$ and $p_{\mathbf{X}}(\mathbf{x}|\theta_1)$ individually, using density estimation algorithms;
2. Evaluate their ratio $r(\mathbf{x}; \theta_0, \theta_1)$.

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!

$$\frac{p_{\mathbf{X}}(\mathbf{x}|\theta_0)}{p_{\mathbf{X}}(\mathbf{x}|\theta_1)} = r(\mathbf{x}; \theta_0, \theta_1)$$

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 $\mathbf{U} = s(\mathbf{X})$, provided $s(\mathbf{x})$ is monotonic with $r(\mathbf{x})$.

$$r(\mathbf{x}) = \frac{p_{\mathbf{X}}(\mathbf{x}|\theta_0)}{p_{\mathbf{X}}(\mathbf{x}|\theta_1)} = \frac{p_{\mathbf{U}}(s(\mathbf{x})|\theta_0)}{p_{\mathbf{U}}(s(\mathbf{x})|\theta_1)}$$

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

$$\begin{aligned}\hat{\theta} &= \arg\max_{\theta} p(\mathcal{D}|\theta) \\ &= \arg\max_{\theta} \prod_{\mathbf{x}\in\mathcal{D}} \frac{p(\mathbf{x}|\theta)}{p(\mathbf{x}|\theta_1)} \\ &= \arg\max_{\theta} \prod_{\mathbf{x}\in\mathcal{D}} \frac{p(s(\mathbf{x};\theta,\theta_1)|\theta)}{p(s(\mathbf{x};\theta,\theta_1)|\theta_1)} \end{aligned} \tag{1}$$

where θ_1 is fixed and $s(\mathbf{x};\theta,\theta_1)$ is a family of classifiers parameterized by (θ,θ_1).

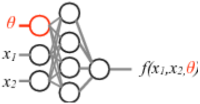
Accordingly, generalized (or profile) likelihood ratio tests can be evaluated in the same way.

Parameterized learning

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

- One could build a classifier s independently for all θ, θ_1. But this is computationally expensive and would not guarantee a smooth evolution of $s(\mathbf{x}; \theta, \theta_1)$ as θ varies.
- Solution: build a single parameterized classifier instead, where parameters are additional input features (Cranmer et al., 2015; Baldi et al., 2016).

$\mathcal{T} := \{\}$;
while size($\mathcal{T}$) $< N$ **do**
  Draw $\theta_0 \sim \pi_{\Theta_0}$;
  Draw $\mathbf{x} \sim p(\mathbf{x}|\theta_0)$;
  $\mathcal{T} := \mathcal{T} \cup \{((\mathbf{x}, \theta_0, \theta_1), y = 0)\}$;
  Draw $\theta_1 \sim \pi_{\Theta_1}$;
  Draw $\mathbf{x} \sim p(\mathbf{x}|\theta_1)$;
  $\mathcal{T} := \mathcal{T} \cup \{((\mathbf{x}, \theta_0, \theta_1), y = 1)\}$;
end while
Learn a single classifier $s(\mathbf{x}; \theta_0, \theta_1)$ from $\mathcal{T}$.

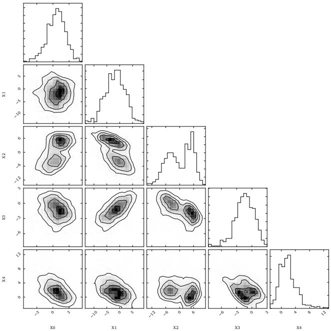


Example: Inference from multidimensional data

Let assume 5D data $\mathbf{x}$ generated from the following process p_0:

1. $\mathbf{z} := (z_0, z_1, z_2, z_3, z_4)$, 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. $\mathbf{x} := R\mathbf{z}$, where R is a fixed semi-positive definite 5×5 matrix defining a fixed projection of $\mathbf{z}$ into the observed space.

Our goal is to infer the values α and β based on $\mathcal{D}$.



Observed data $\mathcal{D}$

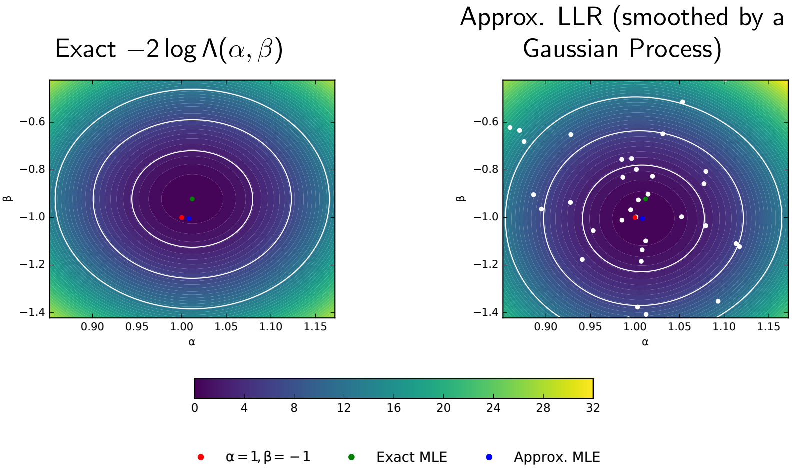
Check out (Louppe et al., 2016) to reproduce this example.

Example: Inference from multidimensional data

Recipe:

1. Build a single parameterized classifier $s(\mathbf{x};\theta_0,\theta_1)$, 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, $p(s(\mathbf{x};\theta_0,\theta_1)|\theta)$ can be calibrated on-the-fly, for every candidate (α,β), e.g. using histograms.
3. Construct the log-likelihood ratio (LLR) statistic

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

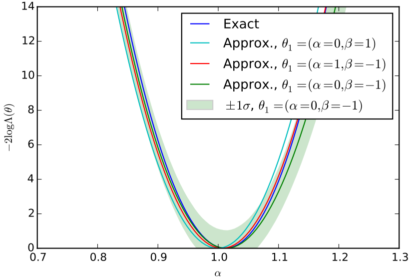


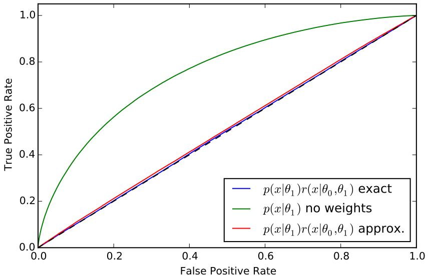
Exact $-2\log\Lambda(\alpha,\beta)$
Approx. LLR (smoothed by a Gaussian Process)
β
α
−0.6
−0.8
−1.0
−1.2
−1.4
0.90
0.95
1.00
1.05
1.10
1.15
0
4
8
12
16
20
24
28
32
α = 1, β = −1
Exact MLE
Approx. MLE

Diagnostics

In practice $\hat{r}(\hat{s}(\mathbf{x};\theta_0,\theta_1))$ 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 θ_1 used in the denominator of the ratio.
2. Train a classifier to distinguish between unweighted samples from $p(\mathbf{x}|\theta_0)$ and samples from $p(\mathbf{x}|\theta_1)$ weighted by $\hat{r}(\hat{s}(\mathbf{x};\theta_0,\theta_1))$.





Density ratio estimation

The density ratio $r(\mathbf{x}; \theta_0, \theta_1) = \frac{p(\mathbf{x}|\theta_0)}{p(\mathbf{x}|\theta_1)}$ appears in many other fundamental statistical inference problems, including

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

For all of them, the proposed approximation can be used as a drop-in replacement!

Transfer learning

Under the assumption that train and test data are drawn iid from a same distribution p,

$$\frac{1}{N}\sum_{\mathbf{x}_i} L(\varphi(\mathbf{x}_i)) \rightarrow \int L(\varphi(\mathbf{x}))p(\mathbf{x})d\mathbf{x},$$

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

Minimizing L over training data is therefore a good strategy.

Transfer learning

~~Under the assumption that train and test data are drawn iid from a same distribution p,~~

$$\frac{1}{N}\sum_{\mathbf{x}_i} L(\varphi(\mathbf{x}_i)) \to \int L(\varphi(\mathbf{x}))p_{\text{train}}(\mathbf{x})d\mathbf{x},$$

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

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

$$\int L(\varphi(\mathbf{x}))p_{\text{test}}(\mathbf{x})d\mathbf{x}.$$

Minimizing L over training data is therefore a bad strategy!

Importance weighting

Reweight samples by $\frac{p_{\text{test}}(\mathbf{x}_i)}{p_{\text{train}}(\mathbf{x}_i)}$, such that

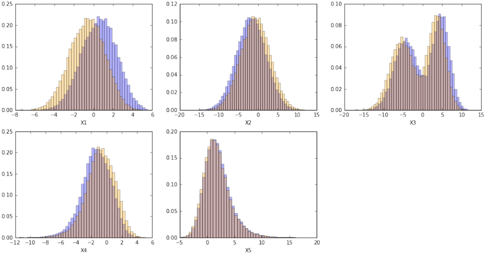
$$\frac{1}{N}\sum_{\mathbf{x}_i} \frac{p_{\text{test}}(\mathbf{x}_i)}{p_{\text{train}}(\mathbf{x}_i)} L(\varphi(\mathbf{x}_i)) \to \int \frac{p_{\text{test}}(\mathbf{x})}{p_{\text{train}}(\mathbf{x})} L(\varphi(\mathbf{x})) p_{\text{train}}(\mathbf{x}) d\mathbf{x},$$

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

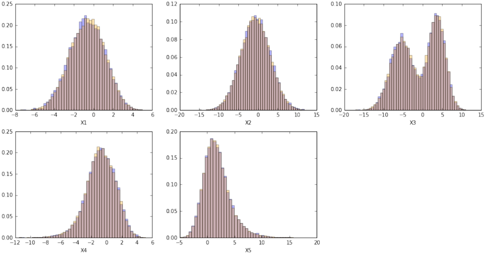
Again, $\frac{p_{\text{test}}(\mathbf{x}_i)}{p_{\text{train}}(\mathbf{x}_i)}$ cannot be evaluated directly, but approximated likelihood ratios can be used as a drop-in replacement.

Example

$p_0 : \alpha = -2, \beta = 2$ *versus* $p_1 : \alpha = 0, \beta = 0$

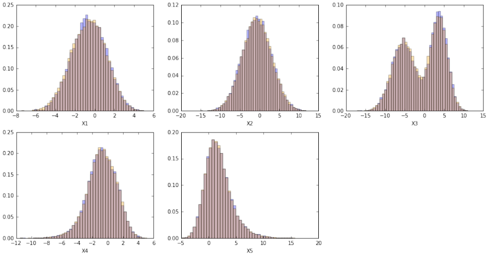


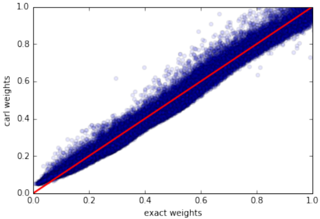
p_0 *versus* $\frac{p_0}{p_1} p_1$



Example

p_0 *versus* $\hat{r}p_1$





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.

References

Baldi, P., Cranmer, K., Faucett, T., Sadowski, P., and Whiteson, D. (2016). Parameterized Machine Learning for High-Energy Physics. *arXiv preprint arXiv:1601.07913*.

Cranmer, K., Pavez, J., and Louppe, G. (2015). Approximating likelihood ratios with calibrated discriminative classifiers. *arXiv preprint arXiv:1506.02169*.

Louppe, G., Cranmer, K., and Pavez, J. (2016). carl: a likelihood-free inference toolbox. `http://dx.doi.org/10.5281/zenodo.47798`, `https://github.com/diana-hep/carl`.