Blueprints for Training Information Bottlenecks for Collider Analyses

FERMILAB-SLIDES-22-216-QIS

Prasanth Shyamsundar, Fermilab Quantum Institute

ML4Jets 2022, Rutgers University

1-4 November 2022

Collaborators/Related projects

Has strong connections to

- “Deep-Learned Event Variables for Collider Phenomenology”, arXiv:2105.10126 [hep-ph]
  Doojin Kim, Kyoungchul Kong, Konstantin T. Matchev, Myeonghun Park, PS
- “An upcoming pheno paper”, **Kevin Pedro, PS**

Has loose connections to

- “New Machine Learning Techniques for Simulation-Based Inference: InferoStatic Nets, Kernel Score Estimation, and Kernel Likelihood Ratio Estimation”, arXiv:2210.01680 [stat.ML]
  Kyoungchul Kong, Konstantin T. Matchev, Stephen Mrenna, PS
- “Optimal event selection and categorization in high energy physics, Part 1: Signal discovery”, arXiv:1911.12299 [physics.data-an], **Konstantin T. Matchev, PS**

And most of the representation learning work in the literature

The goal

Learn analysis variables (selection cuts, etc.) with ML

WHY:

1. Most collider analyses use analysis variables.
2. See (1)

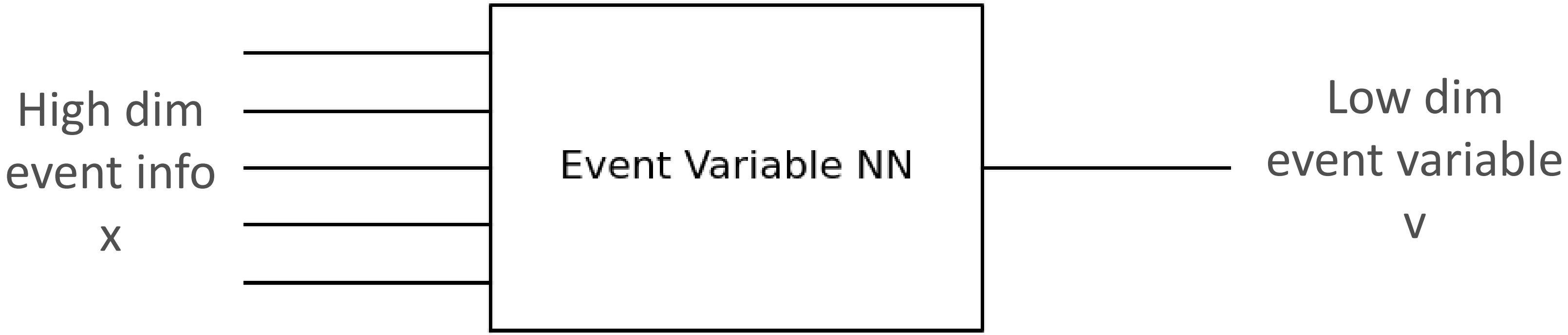
How they are typically used:

Histograms, function fits, etc. to compare data to production level simulations.
We know how to do this part very well.

How they are typically picked:

Human intelligence, simulation studies ("theorist's" simulators are okay)

Event variables are information bottlenecks



- Information bottlenecks
- Representation learning
- Feature engineering

Many ways to model such networks (Interpretable and/or physics informed architectures).

How can we train them?

Route 1: General purpose representation learning

- Examples: Self-supervision, contrastive learning, VAEs,
  other smart latent space constructions.
- Some techniques only perform well when learning high-dim representations.
- Great for industry applications (sometimes a priori unexpected applications).
- Great for anomaly detection strategies.
- Great as intermediate representations to be processed further.
- Not always great as analysis variables.

  a) Analysis variables often need to be ultra-low dimensional.
  b) Their typical use-case is very specific: signal discovery, parameter estimation.

Route 2: Fully differentiable analysis pipeline

- The idea here is to make the entire (or most of the) analysis pipeline including selection cuts, histogram bin edges, neural networks, etc. differentiable (read: tunable via backprop).
- Now one can do a full analysis with simulated data, compute the sensitivity, use it as the gain function and train the analysis pipeline via gradient ascent.

Pros:

The possibilities opened by a fully tunable analysis pipeline.

Cons (once available):

Training steps will be computationally expensive (one full analysis per step)
Small batch sizes may not be reliable for sensitivity estimation
Risk of over-tuning the analysis if not careful

The common feature in ALL these approaches

Every (I think?) representation learning technique

1) Designs a task to perform with the representation

2) Trains the representation on performance on that task

What tasks do we perform in particle physics?

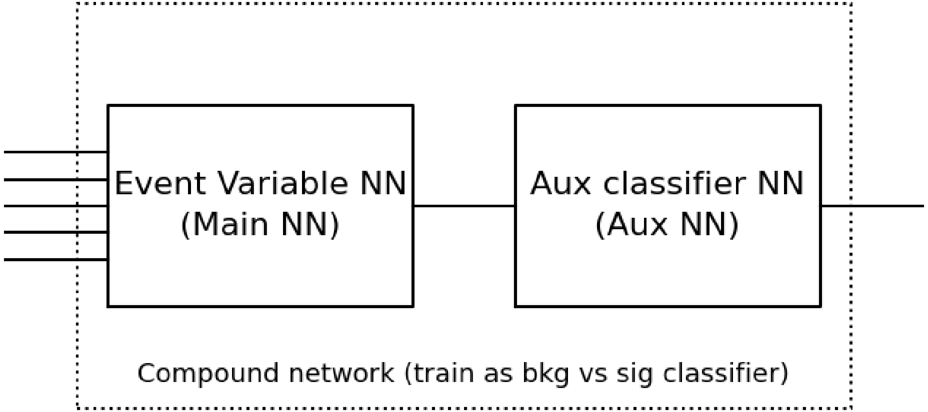
Signal discovery and parameter estimation are both classification tasks

- Background vs Background+Signal
- mW = 80.434 GeV vs mW = 80.357 GeV

Idea: Use classification as the task to perform with the NN variable

Blueprint 0

Point background and signal hypotheses (i.e., no unknown parameters)

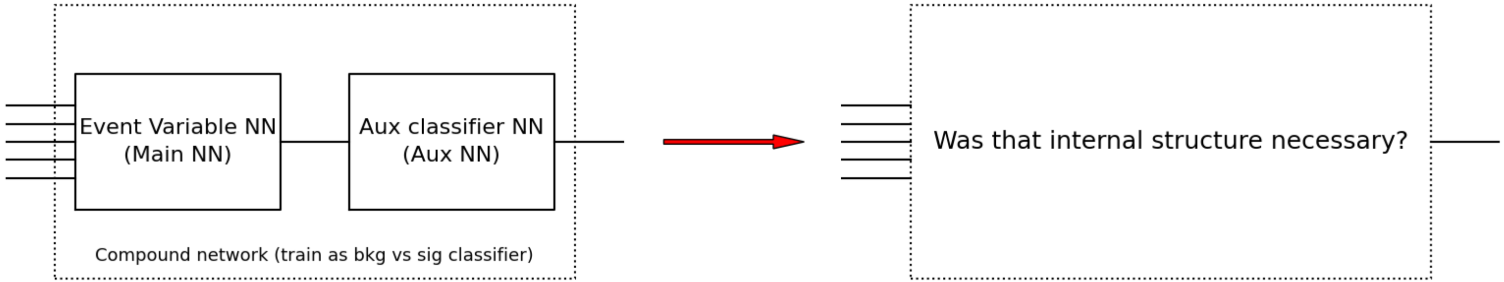


- Aux classifies bkg vs sig using main's output.
- Main does its best to help aux **(not adversarial)**
- Main is being evaluated on David's metric defined as: (more distinguishable is better)

$$D(\text{main network}) = -\text{Trained classifier cost} + \text{constant shift}$$

Well, that was silly…

Q: Why not use one network to do the whole thing?



A: Unknown signal parameters…

(Prasanth's conceptual test for comparing approaches:
Can it learn the invariant mass?)

Handling unknown parameters

How to handle unknown signal parameters with regular classifiers?

1. Parameterized NNs: NN(event | parameter)

   This maps each event to a function of the parameter, not a number.

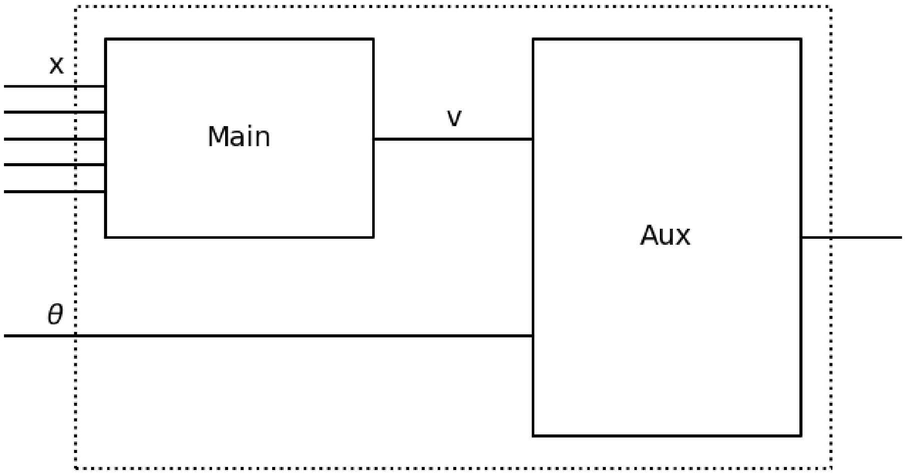
   Space age stuff. I want stone age.

2. Mix signals events from different parameter values together in one global dataset.

   Cannot learn the invariant mass (mass peaks get washed out).

   Will only work if there's a feature that doesn't move around much as the parameter changes.

Blueprint 1: Variable for signal discovery with unknown params



$P_0(x,\theta) = \pi(\theta)P_b(x)$ $\quad$ $P_1(x,\theta) = \pi(\theta)P_s(x|\theta)$

Assorted thoughts:

- Events in real data come from one signal param value.
  So, aux should get to condition on θ.
- What's maximized is
  Avg-over-theta$[D(P_b||P_{s,\theta})]$
- π is NOT a prior.
  It specifies our priority.
- Aux isn't interesting to me, post-training. But v(x) from main is.

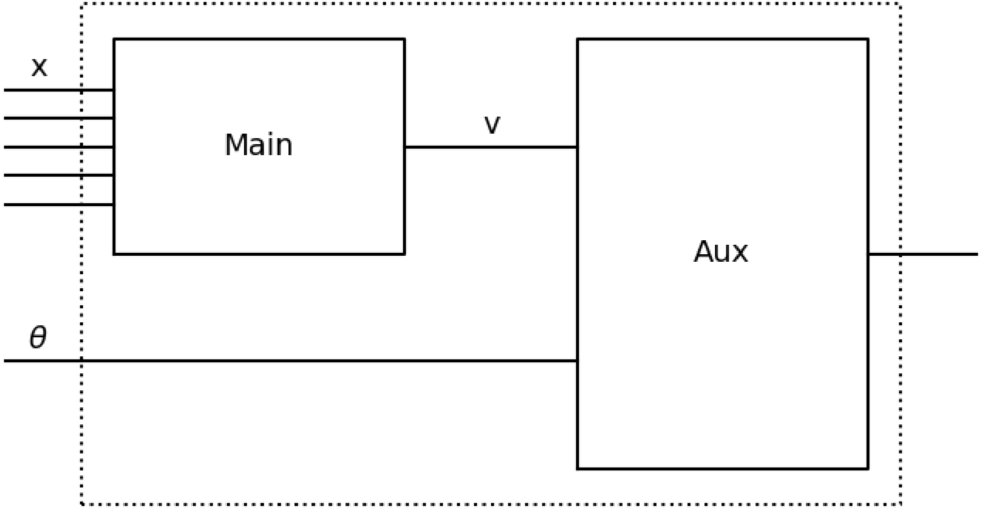
- What will this learn for sharp resonant signals?

  Some linear combination of likelihoods at different param values?
  No. It'll learn the invariant mass!

Fermilab

Blueprint 2: Param estimation

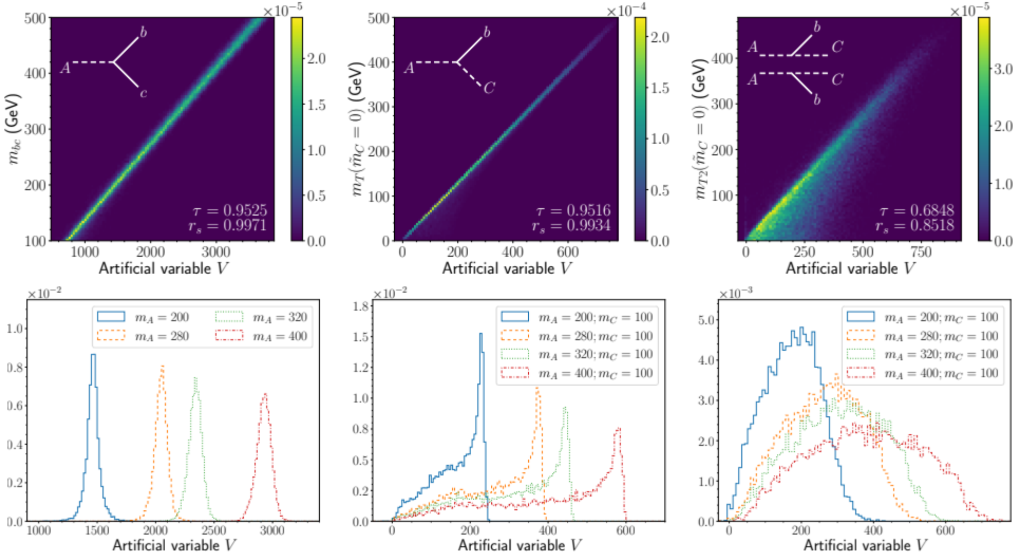
"Deep-Learned Event Variables for Collider Phenomenology", arXiv:2105.10126 [hep-ph]
Doojin Kim, Kyoungchul Kong, Konstantin T. Matchev, Myeonghun Park, Prasanth Shyamsundar



$P_0(x,\theta) = \pi(\theta)\int d\theta' \pi(\theta')P(x|\theta')$ $P_1(x,\theta) = \pi(\theta)P(x|\theta)$

This maximizes the information content in a single event about theta.

Blueprint 2: Results

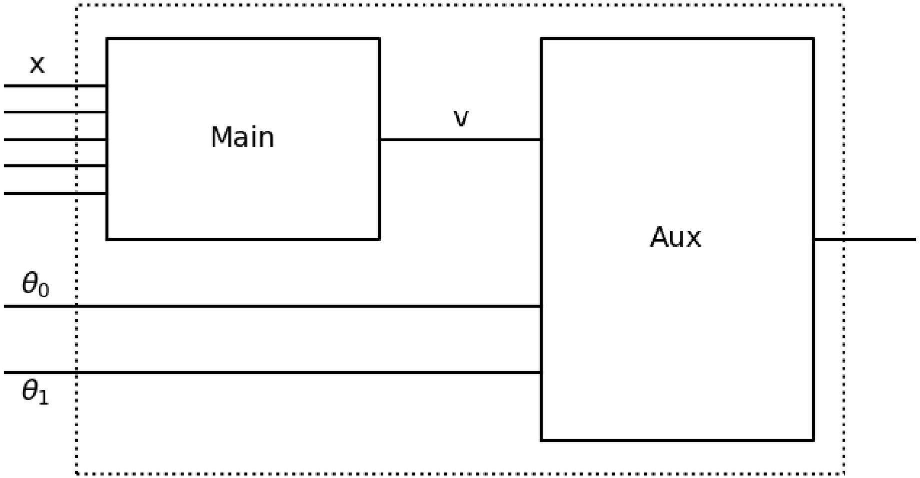


More results for harder analyses in Kevin Pedro's talk.

The architecture was able to learn **invariant mass**, **mT**, and a variable correlated with **mT2** for the appropriate topologies. This was a successful first demonstration of the principle.

This approach can be refined (classifier loss doesn't correlate perfectly with sensitivity in this approach).

Blueprint 3: Refined blueprint for param estimation



$P_y(x, \theta_0, \theta_1) = \pi(\theta_0, \theta_1) P(x|\theta_y)$

This maximizes Average-over-thetas[$D(P_{\theta_0}||P_{\theta_1})$]

Some results in Kevin Pedro's talk.

Some notes:
θ_0 and θ_1 need not be iid.
π can be a correlated distribution.

Aux can be modeled as
$Aux(v, \theta_0, \theta_1) = \varphi(v, \theta_1) - \varphi(v, \theta_0)$
where φ is a backend nn sitting inside Aux.

See: arXiv:2210.01680 [stat.ML],
Kyoungchul Kong, Konstantin T. Matchev, Stephen Mrenna, Prasanth Shyamsundar

More blueprints exist…

- Can be extended from continuous NN-variables to NN-event-selectors and categorizers
- Make analysis variables/event selectors sensitive
  - Over a range of unknown theory parameters
  - Over a range of analysis choices like pre-selection cuts (avoid fine tuning)
  - Haven't figured out the best way to handle nuisance params yet.
- **Train variables complementary to a known variable**
- **Decorrelate event selectors from a known variable**
- Train variables for parameter estimation in the presence of background
- By choosing the right loss functions, we can maximize statistical distances between (Bkg) and (Bkg + α Sig).

Summary

- One can train information bottlenecks with blueprints like these to **improve the sensitivity** of modeled searches and parameter measurements **without compromising on robustness**.
- A variety of useful properties can be incorporated into the bottlenecks.
- Trivially deployment-ready technique, with a lot of use cases in analyses.

- Happy to discuss more if anyone wants to try these out.
- Undecided about the publication pipeline (there might be a minimal preprint some day).

This talk focused on the roses.
Thorns, buds (and interesting physics) in Kevin Pedro's talk later today.

Funding Information

This research is supported by the U.S. Department of Energy, Office of Science, Office of High Energy Physics QuantISED program under the following grants:

- “HEP Machine Learning and Optimization Go Quantum”, Award Number 0000240323
- “DOE QuantiSED Consortium QCCFP-QMLQCF”, Award Number DE-SC0019219