Loop Amplitudes from Precision Networks

Tilman Plehn

Universität Heidelberg

ML4Jets 11/2022

Surrogate loop amplitudes

Problem in precision simulation [Ramon's talk; Danziger etal]

- NLO and NNLO with one-loop and two-loop amplitudes
- computed in terms of poly-logarithms, etc, not fast
- per-mille precision the key requirement
- control of prediction critical

$\rightarrow$ Typical for LHC, atypical for ML

Surrogate amplitudes

- benchmark dataset $pp \to \gamma\gamma g(g)$ [Aylett-Bullock, Badger, Moodie]
- simple regression task [Bishara & Montull, Badger & Bullock]
- exact, nonstochastic training data
- wide range of amplitude values [soft-collinear, on-shell]
- physics-informed architectures succcessful [Maitre & Truong]
- naive MSE-trained regression pointless

$\rightarrow$ ML-solution for precision training?

Likelihood loss from Bayesian networks

Bayesian networks [LHC regression, classification, generation since 2019]

- variational approximation [think $q(\omega)$ as Gaussian with mean and width]

$$p(A) = \int d\omega \; p(A|\omega) \; p(\omega|T) \approx \int d\omega \; p(A|\omega) \; q(\omega)$$

- similarity through KL-divergence [well-defined through ELBO]

$$\text{KL}[q(\omega), p(\omega|T)] = \text{KL}[q(\omega), p(\omega)] - \int d\omega \; q(\omega) \; \log p(T|\omega) + \log p(T)$$

- loss combining likelihood $p(T|\omega)$ and prior $p(\omega)$

$$L = - \int d\omega \; q(\omega) \; \log p(T|\omega) + \text{KL}[q(\omega), p(\omega)]$$

Likelihood loss from Bayesian networks

Bayesian networks [LHC regression, classification, generation since 2019]

- variational approximation [think $q(\omega)$ as Gaussian with mean and width]

$$p(A) = \int d\omega\ p(A|\omega)\ p(\omega|T) \approx \int d\omega\ p(A|\omega)\ q(\omega)$$

- loss combining likelihood $p(T|\omega)$ and prior $p(\omega)$

$$L = -\int d\omega\ q(\omega)\ \log p(T|\omega) + \mathrm{KL}[q(\omega), p(\omega)]$$

Uncertainties

- expectation value using trained network $q(\omega)$

$$\langle A\rangle = \int d\omega\ q(\omega)\overline{A}(\omega) \qquad \text{with} \qquad \overline{A}(\omega) = \int dA\ A\ p(A|\omega)$$

- output variance

$$\sigma^2_{\text{tot}} = \int dA d\omega\ \ (A - \langle A\rangle)^2\ \ p(A|\omega)\ q(\omega) \equiv \sigma^2_{\text{syst}} + \sigma^2_{\text{stat}}$$

- 'statistical' contribution vanishing for $q(\omega) \to \delta(\omega - \omega_0)$ [predictive unc, from ω-sampling]

$$\sigma^2_{\text{stat}} = \int d\omega\ q(\omega) \left[\overline{A}(\omega) - \langle A\rangle\right]^2$$

- 'systematic' contribution in ω-space [stochasticity or model]

$$\sigma^2_{\text{syst}} = \int d\omega\ q(\omega) \left[\overline{A^2}(\omega) - \overline{A}(\omega)^2\right] = \int d\omega\ q(\omega)\ \sigma_{\text{syst}}(\omega)^2$$

Likelihood loss from Bayesian networks

Bayesian networks [LHC regression, classification, generation since 2019]

- variational approximation [think $q(\omega)$ as Gaussian with mean and width]

$$p(A) = \int d\omega \; p(A|\omega) \; p(\omega|T) \approx \int d\omega \; p(A|\omega) \; q(\omega)$$

- loss combining likelihood $p(T|\omega)$ and prior $p(\omega)$

$$L = -\int d\omega \; q(\omega) \; \log p(T|\omega) + \text{KL}[q(\omega), p(\omega)]$$

Likelihood loss

- focus here: network output in weight and phase space

$$\text{BNN} : x, \omega \to \begin{pmatrix} \overline{A}(\omega) \\ \sigma_\text{syst}(\omega) \end{pmatrix}$$

- Gaussian likelihood [heterostedastic loss]

$$L = \int d\omega \; q(\omega) \sum_{\text{points } j} \left[\frac{\left| \overline{A}_j(\omega) - A_j^\text{truth} \right|^2}{2\sigma_{\text{syst},j}(\omega)^2} + \log \sigma_{\text{syst},j}(\omega) \right] + \text{KL}[q(\omega), p(\omega)]$$

$\to$ Likelihood loss with extracted uncertainty

Boosted network training

Self-consistency improvement [Badger, Butter, Luchmann, Pitz, TP]

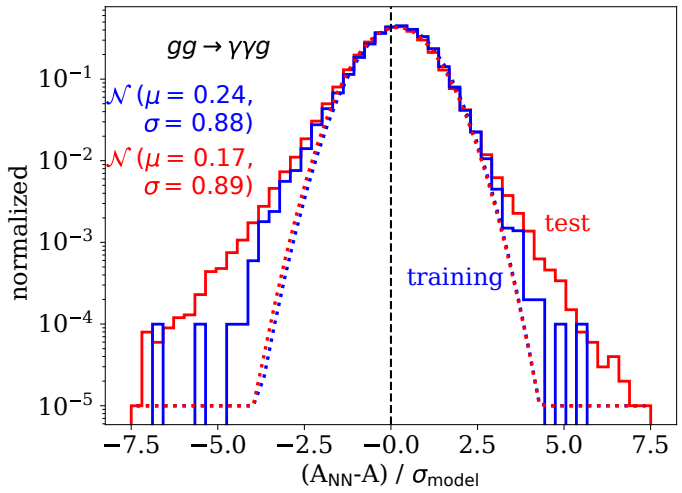
- check weight-dependent pull

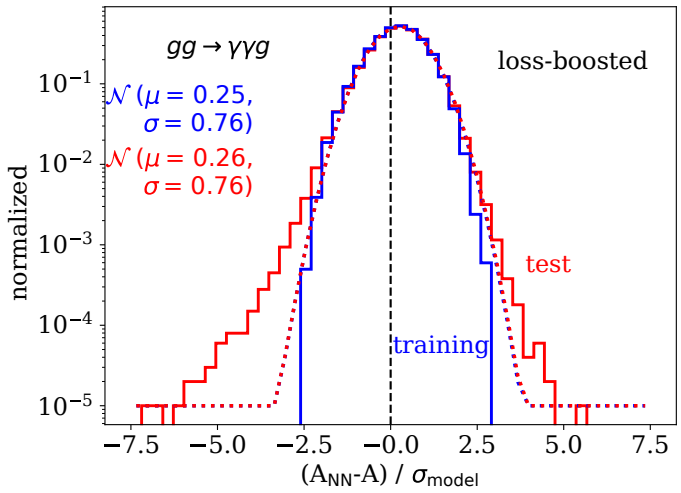
$$\frac{\overline{A}_j(\omega) - A_j^{\text{truth}}}{\sigma_{\text{model},j}(\omega)}$$

- after boosting amplitudes with large pull

$$L_{\text{boost}} = \int d\omega\; q(s\omega) \sum_{\text{points } j} n_j \times \left[\frac{\left|\overline{A}_j(\omega) - A_j^{\text{truth}}\right|^2}{2\sigma_{\text{model},j}(\omega)^2} + \log \sigma_{\text{model},j}(\omega) \right] \cdots$$

- more improvement on training data than test data $\rightarrow$ controlled overtraining





Boosted network training

Self-consistency improvement [Badger, Butter, Luchmann, Pitz, TP]

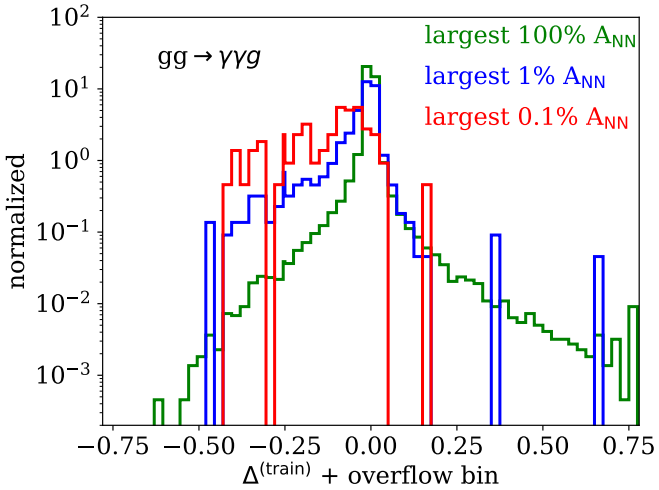
- check weight-dependent pull

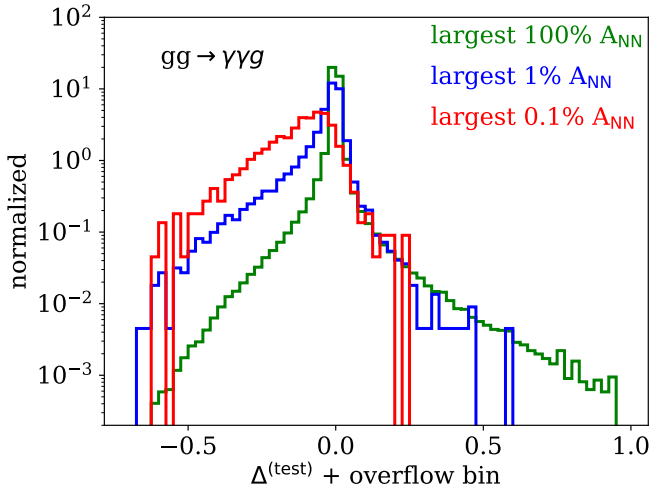
$$\frac{\overline{A}_j(\omega) - A_j^{\text{truth}}}{\sigma_{\text{model},j}(\omega)}$$

- after boosting amplitudes with large pull

$$L_{\text{boost}} = \int d\omega\; q(s\omega) \sum_{\text{points } j} n_j \times \left[\frac{\left|\overline{A}_j(\omega) - A_j^{\text{truth}}\right|^2}{2\sigma_{\text{model},j}(\omega)^2} + \log \sigma_{\text{model},j}(\omega) \right] \cdots$$

- more improvement on training data than test data $\rightarrow$ controlled overtraining
- but not precise enough for largest amplitudes... $[\Delta = (A - A)/A]$





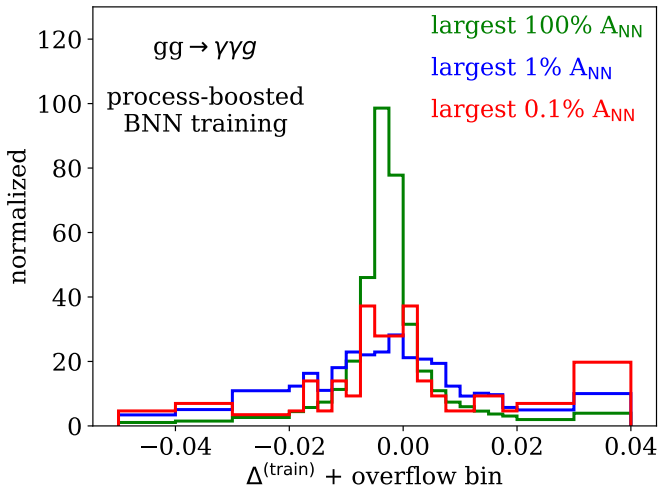
More boosting

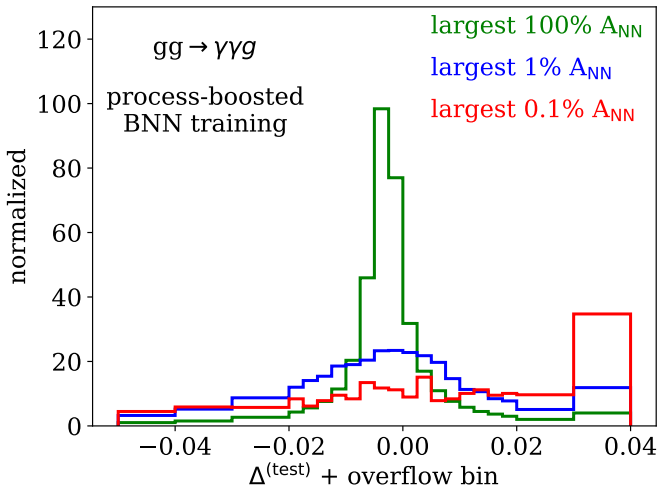
Process-dependent improvement

- boosting amplitudes by σ

$$L_\text{boost} = \int d\omega\; q(s\omega) \sum_{\text{points } j} n_j \times \left[\frac{\left| \bar{A}_j(\omega) - A_j^\text{truth} \right|^2}{2\sigma_{\text{model},j}(\omega)^2} + \log \sigma_{\text{model},j}(\omega) \right] \cdots$$

- even more controlled overtraining
- stable performance for 90k $\to$ 9k training amplitudes





More boosting

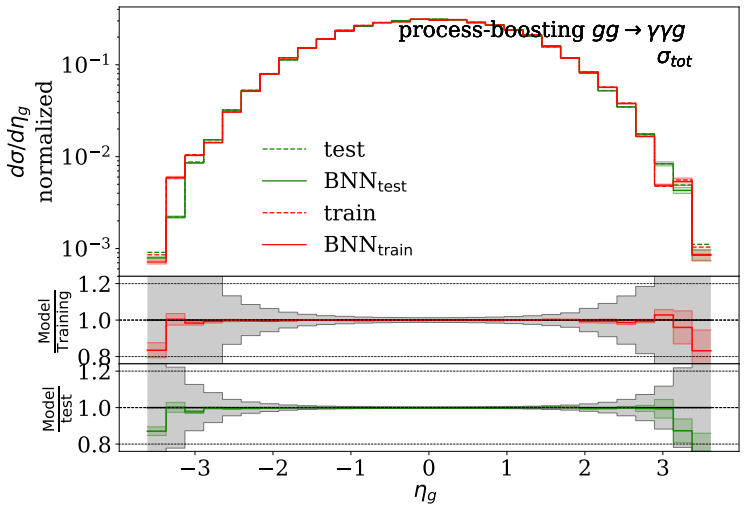
Process-dependent improvement

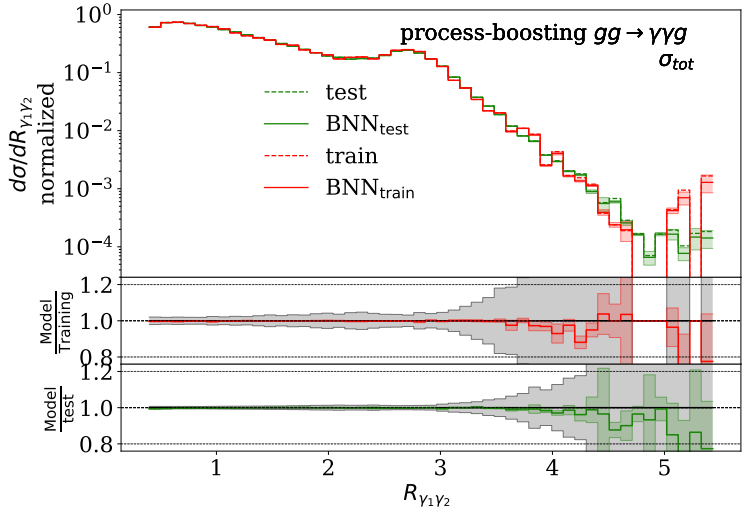
- boosting amplitudes by σ

$$L_{\text{boost}} = \int d\omega\; q(s\omega) \sum_{\text{points } j} n_j \times \left[\frac{\left| \bar{A}_j(\omega) - A_j^{\text{truth}} \right|^2}{2\sigma_{\text{model},j}(\omega)^2} + \log \sigma_{\text{model},j}(\omega) \right] \cdots$$

- even more controlled overtraining
- stable performance for 90k $\to$ 9k training amplitudes
- looking at kinematic distributions

$\to$ From fit to interpolation

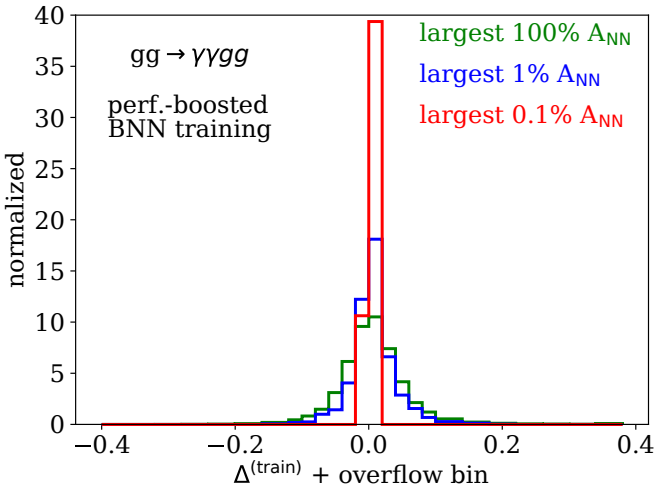


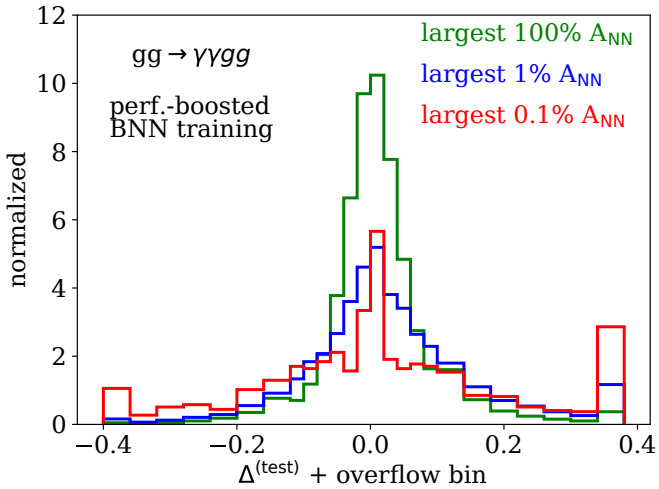


One more jet

From $\gamma\gamma g$ to $\gamma\gamma gg$ [Luchmann, Victor Breso]

- 6k $\to$ 600k parameters, 90k training amplitudes...
- depressing results, problem with scaling?

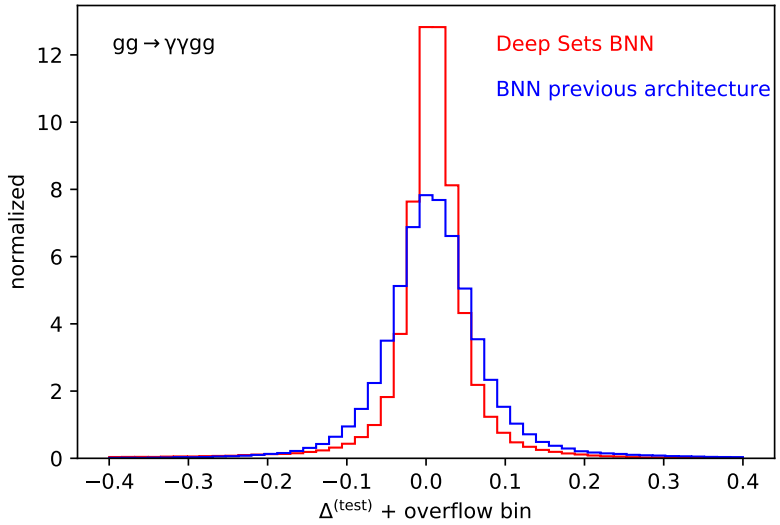




One more jet

From $\gamma\gamma g$ to $\gamma\gamma gg$ [Luchmann, Victor Breso]

- 6k $\rightarrow$ 600k parameters, 90k training amplitudes...
- depressing results, problem with scaling?
- check DeepSets preprocessing as a start...
  ...for more ideas, let's talk over coffee...



Bayesian networks

Initially developed for inference they work for...

...regression with error bars

...classification with error bars

...generation with error bars

...turning fit-networks into interpolation-networks

Modern Machine Learning for LHC Physicists

Tilman Plehn[a,*], Anja Butter[a,b], Barry Dillon[a], and Claudius Krause[a,c]

[a] Institut für Theoretische Physik, Universität Heidelberg, Germany
[b] LPNHE, Sorbonne Université, Université Paris Cité, CNRS/IN2P3, Paris, France
[c] NHETC, Dept. of Physics and Astronomy, Rutgers University, Piscataway, USA

November 2, 2022

Abstract

Modern machine learning is transforming particle physics, faster than we can follow, and bullying its way into our numerical tool box. For young researchers it is crucial to stay on top of this development, which means applying cutting-edge methods and tools to the full range of LHC physics problems. These lecture notes are meant to lead students with basic knowledge of particle physics and significant enthusiasm for machine learning to relevant applications as fast as possible. They start with an LHC-specific motivation and a non-standard introduction to neural networks and then cover classification, unsupervised classification, generative networks, and inverse problems. Two themes defining much of the discussion are well-defined loss functions reflecting the problem at hand and uncertainty-aware networks. As part of the applications, the notes include some aspects of theoretical LHC physics. All examples are chosen from particle physics publications of the last few years. Given that these notes will be outdated already at the time of submission, the week of ML4Jets 2022, they will be updated frequently.