Computing the Bayes-optimal classifier and exact maximum likelihood estimator with a semi-realistic generative model for jet physics

Matthew Drnevich
New York University
mdd424@nyu.edu

Lauren Greenspan
New York University

Sebastian Macaluso
New York University
seb.macaluso@nyu.edu

Kyle Cranmer
University of Wisconsin-Madison
New York University
kyle.cranmer@wisc.edu

Duccio Pappadopulo

Abstract

Deep learning techniques have proven to be extremely effective in studying complicated, collimated sprays of particles found in high energy particle collisions known as *jets*. As with most realistic classification tasks, the Bayes-optimal classifier is unknown or intractable, even when trained with simulated data. Here we consider `Ginkgo`, a semi-realistic simulator for jets that captures the essential physics and produces data with similar features and format. By using a recently-developed hierarchical trellis data structure and dynamic programming algorithm, we are able to exactly marginalize over the combinatorically large space of latent variables associated to this generative model. This allows us to compute the Bayes-optimal classifier and the exact maximum likelihood estimator for this model, which can serve as a powerful benchmarking tool for studying the performance of machine learning approaches to these problems.

1 Introduction

High energy particle collisions copiously produce collimated sprays of particles called *jets*. Jets are complicated objects, and deep learning techniques have proven to be extremely effective for many tasks encountered in jet physics (see Larkoski et al. [2020] for a recent review). For example, a variety of deep learning architectures including convolutional networks, recurrent networks over sequences and trees, deep sets, and graph neural networks of various types have been employed for a particular binary classification task Larkoski et al. [2020], Butter et al. [2019]. In addition, there is significant activity in physics-inspired approaches that incorporate domain knowledge in various ways. Progress has been steady, but it is unclear how close the field is to saturating the performance of the Bayes-optimal classifier. In this work, we make progress to address this question.

Many tasks in jet physics can be framed in probabilistic terms Cranmer et al. [2021b]. Here we consider two of them. The first is the binary classification problem mentioned above. Here the Bayes-optimal classifier is equivalent to the likelihood ratio as it yields the most powerful hypothesis test according to the Neyman-Pearson lemma. The second task we consider is "tuning" the parameters of the simulators, which can be formalized as a parameter estimation problem with the maximum likelihood estimate as the desired target. While these formulations are helpful conceptually, they are not practically of much value as the likelihood for our most high-fidelity simulations for jets is intractable.

Machine Learning and the Physical Sciences workshop, NeurIPS 2022.

Contributions of this paper We consider `Ginkgo` [Cranmer et al., 2019]: a simplified simulator of jets that provides a tractable joint likelihood. We pair this with a classical data structure and dynamic programming algorithm (the *cluster trellis* [Greenberg et al., 2020]), that exactly and efficiently marginalizes over the space of configurations, that grows super-exponentially. Having access to the marginal likelihood allows us to directly characterize the discrimination power of the optimal classifier (without any training), defined by the likelihood ratio [1]. It also allows us to compute the exact maximum likelihood estimate for the parameters of the simulator, which provides an asymptotically unbiased and minimum-variance estimator. These references serve as valuable benchmarks for studying the effectiveness of proposed methods based on deep learning in a highly controlled setting.

2 Reframing jet physics in probabilistic terms

In high energy particle collisions, quarks and gluons are produced, which subsequently go through a *showering process*, where they radiate many other quarks and gluons in successive binary splittings. This ultimately leads to a collimated spray of stable particles whose energy and momenta can be measured in a detector. The group of particles is collectively called a *jet*. This showering process can be represented as a binary tree, where the leaves correspond to the stable particles observed in the detector. Any given jet could have originated from a multitude of unobserved showering histories, corresponding to a latent space of binary trees that grows super-exponentially – specifically as $(2N-3)!!$, where N is the number of leaves (jet constituents).

There are sophisticated simulators for jets such as `Pythia` [Sjostrand et al., 2006], `Herwig` [Bellm et al., 2016], and `Sherpa` [Gleisberg et al., 2009]. These simulators are grounded in quantum chromodynamics, but make a number of approximations, which introduce parametric modelling choices and corresponding parameters θ. These parameters are not predicted from underlying theory, so it is common that they are tuned to match the data. These simulators can also be used to describe different types of particle interactions, each of which can be thought of as a discrete class label. For notational convenience, we will absorb the settings of the simulator used to control the type of particle interaction into the parameter vector θ. Following the notation of Brehmer et al. [2018a], Cranmer et al. [2020], we denote the observable output of the simulator x and latent variables (aka Monte Carlo truth record or showering history) z. Here we focus on the following quantities

- Joint likelihood for latent shower and observed constituents: $p(x, z|\theta)$
- Marginal likelihood for observed constituents: $p(x|\theta) = \int dz\ p(x, z|\theta)$
- Maximum likelihood estimate: $\hat{\theta} = \text{argmax}_\theta\, p(x|\theta) = \text{argmax}_\theta \int dz\ p(x, z|\theta)$
- Bayes-optimal classifier for θ_0 vs. θ_1: $r(x) = p(x|\theta_1)/p(x|\theta_0)$

Most of these quantities are intractable to compute or inconvenient to access. Quantities such as the marginal likelihood $p(x|\theta)$ and the maximum likelihood parameter $\hat{\theta}$ involve integration (sums) over all possible showering histories. This super-exponential growth in the number of showering histories is at the heart of many computational bottlenecks in jet physics.

These simulators are typically autoregressive models, that follow a Markov process that encodes the physics of each splitting. We consider simulators based on successive $1 \to 2$ splittings, where the joint likelihood is given by

$$p(x, z|\theta) = p(x|z_{\text{leaves}}, \theta) \prod_{s \in \text{splittings}} p(z_{s,L}, z_{s,R}|z_{s,P}, \theta)\ , \tag{1}$$

with $z_{s,P}$, $z_{s,L}$, and $z_{s,R}$, being respectively the data needed to encode the state of the parent and left and right children for the s^{th} splitting, and z_{leaves} are the terminal leaves of the showering process.

Another barrier in working with simulators such as `Pythia`, `Herwig`, and `Sherpa` is that neither the joint likelihood $p(x, z|\theta)$ nor the likelihood of individual splittings $p(z_{s,L}, z_{s,R}|z_{s,P})$ is exposed in a way that is convenient to access. The joint likelihood is encoded within the models but often in terms of accept-reject sampling and procedural code that does not explicitly expose the probabilities

[1] The optimal classifier is based on the Neyman–Pearson lemma and defined by the likelihood ratio as the most powerful variable or test statistic (for a proof and a particle physics application see [J. Stuart and Arnold, 1994, Cranmer and Plehn, 2007]).

themselves. This motivates `Ginkgo`, which provides convenient access to these quantities in a simplified Monte Carlo parton shower [Cranmer et al., 2019].

3 Data generation and the computation of the marginal likelihood

We demonstrate the effectiveness of this approach using the `Ginkgo` simulator, which has two parameters $\theta = (\lambda, t_{\text{cut}})$ that define the showering (generative) process. The first parameter, λ, determines the decay rate of the parton showering such that larger values encourage a shorter latent path. The second parameter, t_{cut}, is the stopping criterion for the generative process. Whenever a particle invariant mass squared falls below t_{cut} the showering process stops for that particle, obtaining a leaf in the binary tree. For each dataset, we begin the showering process with a particle of mass 30 GeV and momentum $|\mathbf{p}| = 400$ GeV.

The data output by `Ginkgo` includes the energy-momentum vectors of every final state particle in the jets (leaves), as well as the joint likelihood of $p(x, z|\lambda, t_{\text{cut}})$, where z labels the latent path and x_i the energy-momentum vectors of the final state particles. In order to compute the Bayes optimal classifier and the exact maximum likelihood estimator for these datasets, we need to compute the marginal likelihood $p(x|\lambda, t_{\text{cut}})$. This requires marginalizing over the whole space of latent path configurations. With the *cluster trellis* [Greenberg et al., 2020]) algorithm, we are able to efficiently evaluate the marginal likelihood over spaces of configurations that make a brute force sum intractable.

4 Exact optimal classifier based on the marginal likelihood

In a typical binary classification problem, one denotes the class labels as y and considers a dataset of labeled training data $(x_i, y_i) \sim p(x, y)$. The Bayes optimal classifier is simply $p(y|x)$. Here we assign class label $y = 0$ for data generated with parameters θ_0 and $y = 1$ to data generated with parameters θ_1. It is well known that in the limit of enough data and a sufficiently expressive classifier, that the binary cross entropy loss function is minimized by the Bayes optimal classifier. However, in the real world with finite training data, computation, and model capacity as well as a lack of guarantees in the context of non-convex optimization, it is difficult to know how close a learned classifier is to this theoretical optimum.

Let us consider the posterior probability for the $y = 1$ label in the case of balanced training data, *e.g.* $p(y = 0) = p(y = 1) = 1/2$. Define $s(x) \equiv p(y = 1|x) = p(x|y = 1)/(p(x|y = 0) + p(x|y = 1))$, which is a monotonic transformation of the likelihood ratio $r(x) = p(x|y = 1)/p(x|y = 0)$. Since the rates of false positives and negatives are invariant to such a monotonic transformation, the Bayes optimal classifier $s(x)$ is equivalent to the likelihood ratio $r(x)$, which is also well known as the most powerful test statistic in classical hypothesis testing J. Stuart and Arnold [1994].

Since the trellis algorithm allows us to compute the exact marginal likelihood $p(x|\theta)$ for any jet, we can also compute the likelihood-ratio $r(x) = p(x|\theta_1)/p(x|\theta_0)$ for any choices of model parameters, θ_0 and θ_1. We use this Bayes-optimal classifier – which is not learned, but simply computed – as a benchmark to evaluate the performance of other classifiers based on machine learning.

We generated datasets for two values of the parameters θ_0 and θ_1. For the first class we used $\lambda = 4$ and $t_{\text{cut}} = 36$ GeV^2, and for the second class we used $\lambda = 1.5$ and $t_{\text{cut}} = 36$ GeV^2. We refer to these classes as "QCD-like" and "W-like", respectively. For each dataset, 5,000 jets were produced with `Ginkgo`.

Figure 1 (left) shows the receiver operating characteristic (ROC) curve for the Bayes-optimal classifier, which provides an upper-bound on the performance for this task. Ordinarily, it wouldn't be possible to compute the Bayes-optimal classifier and characterize its performance. However, this is precisely what makes this work interesting. Ginkgo generates data of similar complexity to a real-world problem of interest to the physics community, but with an exposed joint likelihood that can then be marginalized with a special technique (*i.e.* the trellis) to establish an upper bound on performance.

This provides a well-controlled benchmark where those exploring deep learning and other approaches to problems like this can better understand how well their methods are working relative to the theoretical optimum.

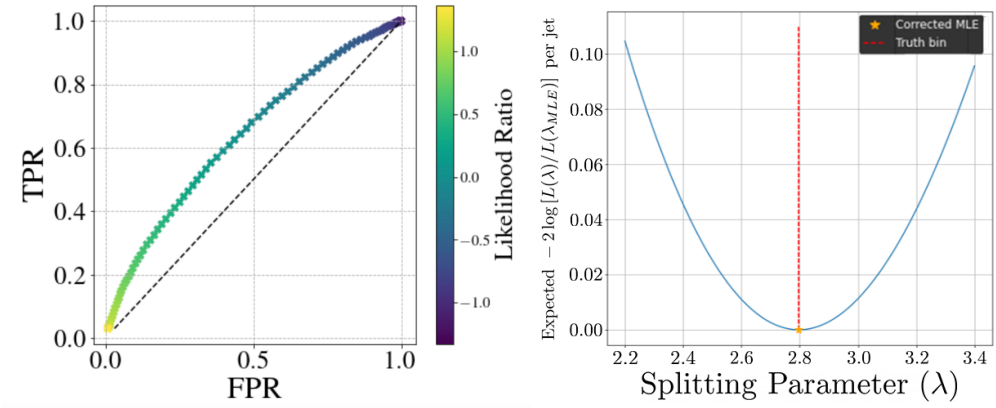

Figure 1: Left: Receiver-operating curve for the optimal classifier, defined by the likelihood-ratio of data classes $\lambda = 4$ (QCD-like jets) and $\lambda = 1.5$ (W-like jets) generated by `Ginkgo`. Right: The negative log-likelihood as a function of `Ginkgo`'s splitting parameter, λ. The maximum likelihood estimate corresponds to the minimum point on the graph and corresponds to the true value of the parameter used to generate the data.

5 Parameter tuning via maximum likelihood

Next we consider the task of estimating the parameters θ of a simulator like `Pythia`. In particle physics, this process is often referred to as "Monte Carlo tuning". Estimating these parameters improves the fidelity of the simulator and all the downstream data analysis that depend upon it. Furthermore, some model parameters may provide us with meaningful physics information or deeper insight into physical phenomena.

Ordinarily the likelihood $p(x|\theta)$ is intractable, so maximum likelihood estimation is not applicable. Traditional approaches for tuning the parameters of simulator in particle physics, e.g. `Professor` [Buckley et al., 2010], compare one-dimensional projections (marginal distributions) of physics observables (summary statistics), which is blind to various forms of mismodelling in the high-dimensional distribution. An emerging technique for parameters tuning is *likelihood-free inference* or *simulation-based inference* [Brehmer et al., 2018a,b,c]. These methods approximate the intractable marginal likelihood using machine learning.

Just as it was in the classification setting, it is powerful to have a well controlled benchmark with optimal performance guarantees in the parameter estimation setting. Here we compute the exact maximum likelihood estimator, which enjoys the properties of being asymptotically unbiased and minimum variance (*e.g.* the Cramér–Rao bound).

With the parameters set to $\lambda = 2.8$ and $t_{\text{cut}} = 30$ GeV^2, we generated 100,000 jets with `Ginkgo`. We chose these internal parameters of the model to yield jets with a relatively small number of leaves for computational efficiency in this proof-of-concept work.

For this demonstration, we fix t_{cut} and treat λ as the unknown parameter to be estimated. Then we generate a one dimensional grid of possible values $\{\lambda_i\}$ and compute $p(x|\lambda_i, t_{\text{cut}})$ for each λ_i using the cluster trellis algorithm. Finally, we estimate the value of λ using maximum likelihood, i.e. $\lambda_{MLE} = \text{argmax}_{\lambda_i}\, p(x|\lambda_i, t_{\text{cut}})$, to obtain $\lambda_{MLE} = 2.796$ [2]. Figure 1 (right) shows the likelihood-ratio test statistic as a function of the splitting parameter λ and that the maximum likelihood estimate has good agreement with the true value of $\lambda = 2.8$.

6 Conclusion

We showed how the combination of a semi-realistic generative model for jet physics (`Ginkgo`) and a recently-developed data structure / dynamic programming algorithm (the cluster trellis) can be used to provide a compelling benchmark dataset where the Bayes-optimal classifier and exact maximum

[2] We had to make a small correction to the likelihood computation to account for a normalization discrepancy that we were able to detect through reweighting. We were unable to identify the source of this small normalization discrepancy, and suspect a subtle numerical issue or bug in one of the libraries we rely upon. For details see [Cranmer et al., 2021a]

likelihood estimate can be computed in practice. Thus, this benchmark provides not only data, but also theoretically optimal solutions that can be used to better understand the performance of current and emerging machine learning techniques for two problems of interest to the particle physics community.

7 Broader Impact

While this work grew out of research in particle physics, the higher-level point that is being made is that we can leverage physical systems to provide a scalable class of problems that can be used for well-controlled experiments that test our theoretical understanding of machine learning. These systems can often scale in such a way that the problem goes from being easy to effectively impossible. Nevertheless, physical insight and centuries of effort have led to impressive results where some of these physical systems are effectively solved or there exist very strong baselines. Together, these properties make for a good test bed for machine learning research. Most of the physical systems where this has been achieved come from statistical physics. This is one of the only examples from particle physics that the authors are aware of where these two ingredients are both manifest. In this case, the problem complexity grows as the number of final state particles (leaves of the tree) grows or as the parameter values θ_0 and θ_1 become closer.

A diverse range of deep learning architectures have been used in the context of jet physics. It is an area where incorporating symmetries and other forms of inductive bias have been effective. It is also an area where graph-based algorithms and geometric deep learning have been very effective. Therefore, this benchmark is relevant for better understanding modern architectures that often provide state of the art results in a wide range of tasks and datasets.

We do not see any direct ethical concerns associated to this research. The impact on society is primarily through the over-arching context of research in machine learning in general.

The computational requirements (and associated environmental) of this study were modest. The computing was conducted with the computing resources available to all researchers at a specific R1 US University without any exceptional access or funding. Generating the datasets required on the order of hundreds of CPU hours, while running the cluster trellis algorithm required only tens of CPU hours for this special case. No GPUs were used.

References

J. Bellm et al. Herwig 7.0/Herwig++ 3.0 release note. *Eur. Phys. J. C*, 76(4):196, 2016. doi: 10.1140/epjc/s10052-016-4018-8.

J. Brehmer, K. Cranmer, G. Louppe, and J. Pavez. A Guide to Constraining Effective Field Theories with Machine Learning. *Phys. Rev. D*, 98(5):052004, 2018a. doi: 10.1103/PhysRevD.98.052004.

J. Brehmer, K. Cranmer, G. Louppe, and J. Pavez. Constraining Effective Field Theories with Machine Learning. *Phys. Rev. Lett.*, 121(11):111801, 2018b. doi: 10.1103/PhysRevLett.121.111801.

J. Brehmer, G. Louppe, J. Pavez, and K. Cranmer. Mining gold from implicit models to improve likelihood-free inference. 2018c.

A. Buckley, H. Hoeth, H. Lacker, H. Schulz, and J. E. von Seggern. Systematic event generator tuning for the LHC. *Eur. Phys. J. C*, 65:331–357, 2010. doi: 10.1140/epjc/s10052-009-1196-7.

A. Butter et al. The Machine Learning landscape of top taggers. *SciPost Phys.*, 7:014, 2019. doi: 10.21468/SciPostPhys.7.1.014.

K. Cranmer and T. Plehn. Maximum significance at the lhc and higgs decays to muons. *The European Physical Journal C*, 51(2):415–420, Jun 2007. ISSN 1434-6052. doi: 10.1140/epjc/s10052-007-0309-4. URL `http://dx.doi.org/10.1140/epjc/s10052-007-0309-4`.

K. Cranmer, S. Macaluso, and D. Pappadopulo. Toy Generative Model for Jets Package, 2019. https://github.com/SebastianMacaluso/ToyJetsShower.

K. Cranmer, J. Brehmer, and G. Louppe. The frontier of simulation-based inference. National Academy of Sciences, 2020. doi: 10.1073/pnas.1912789117.

K. Cranmer, M. Drnevich, and S. Macaluso. Tuning the parton shower parameters with the marginal likelihood. In *ML4Jets Workshop*, 2021a. URL https://indico.cern.ch/event/980214/contributions/4413534/.

K. Cranmer, M. Drnevich, S. Macaluso, and D. Pappadopulo. Reframing jet physics with new computational methods. *EPJ Web of Conferences*, 251:03059, 2021b. doi: 10.1051/epjconf/202125103059. URL https://doi.org/10.1051%2Fepjconf%2F202125103059.

T. Gleisberg, S. Hoeche, F. Krauss, M. Schonherr, S. Schumann, F. Siegert, and J. Winter. Event generation with SHERPA 1.1. *JHEP*, 02:007, 2009. doi: 10.1088/1126-6708/2009/02/007.

C. S. Greenberg, S. Macaluso, N. Monath, J. A. Lee, P. Flaherty, K. Cranmer, A. McGregor, and A. McCallum. Compact representation of uncertainty in hierarchical clustering. *CoRR*, abs/2002.11661, 2020. URL https://arxiv.org/abs/2002.11661.

A. O. J. Stuart and S. Arnold. Kendall's advanced theory of statistics. In *Vol 2A (6th Ed.) (Oxford University Press, New York*, 1994.

A. J. Larkoski, I. Moult, and B. Nachman. Jet Substructure at the Large Hadron Collider: A Review of Recent Advances in Theory and Machine Learning. *Phys. Rept.*, 841:1–63, 2020. doi: 10.1016/j.physrep.2019.11.001.

T. Sjostrand, S. Mrenna, and P. Z. Skands. PYTHIA 6.4 Physics and Manual. *JHEP*, 05:026, 2006. doi: 10.1088/1126-6708/2006/05/026.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] **We emphasize that we are working with a simplified simulator as a special case study.**
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] **See Broader Impacts**
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] **Not at this time, but we have this available and will link to it if the paper is accepted**
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] **These are specified in sections 3, 4, and 5**
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A] **There's no learning in this method so no error bars**
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] **See end of Broader Impacts**
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] **Ginkgo and Cluster Trellis Algorithm**
   (b) Did you mention the license of the assets? [No] **Licenses are mentioned in the links associated with individual code packages.**
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]