LundNet: Robust jet identification using graph neural networks

Frédéric A. Dreyer
Rudolf Peierls Centre for Theoretical Physics, University of Oxford
Clarendon Laboratory, Parks Road, Oxford OX1 3PU, UK
`frederic.dreyer@physics.ox.ac.uk`

Huilin Qu
CERN
`huilin.qu@cern.ch`

Abstract

The identification of heavy particles such as top quarks or vector bosons is one of the key issues at the Large Hadron Collider. An efficient way to describe the radiation patterns within a jet is the Lund plane. In this article, we introduce a novel jet tagging method which relies on graph neural networks and the Lund plane to optimally disentangle signatures of boosted objects from background QCD jets. We apply this framework to several different benchmarks, showing improved performance for top tagging compared to existing algorithms. Finally, we study the sensitivity to non-perturbative effects and show how kinematic cuts in the Lund plane can improve the robustness of the neural network.

1 Introduction

As the Large Hadron Collider continues to explore proton collisions at the energy frontier, an important task to search for signals of new physics beyond the Standard Model (SM) is the identification of heavy particles at the electroweak scale, which might emerge from decays of yet unknown heavy particles.

An entity of particular interest at the LHC is the jet, which is essentially defined as a collimated bunch of particles with a certain energy and direction, typically determined using a sequential recombination algorithm [1]. One important problem is that electroweak scale particles such as vector bosons or top quarks produced from yet heavier new states can become sufficiently boosted such that their hadronic decays are reconstructed as single jets. It is therefore crucial to have efficient tools to probe the radiation patterns within jets and determine their physical origin. This topic has been the focus of much attention over the past decade, with a range of approaches being developed to extract information from a jet's substructure. In recent years, a new generation of tools based on machine learning models have emerged, which can achieve very high performance on specific benchmarks [2]. One important limitation of such methods is the difficulty to estimate their uncertainties, as well as their proneness to rely on unphysical features present in the training data to achieve their high performance, as this data is generally derived from Monte Carlo (MC) simulations of proton collisions.

In this article, we introduce a novel method to identify jets using graph networks. To this end, we represent jets through their so-called Lund plane, associating each Lund declustering with a node on the graph. Compared with other state-of-the-art tools, our new method shows improved performance, notably for processes with complicated topologies such as top decays. Finally, we will investigate the robustness of our new tagger, and show how through kinematic cuts on the Lund variables one can mitigate overfitting to the MC simulations, reducing the reliance of the neural network on non-perturbative contributions.

2 LundNet: Tagging jets in the Lund plane

The Lund plane contains a rich set of information about jet substructure and has shown powerful potential for jet tagging [3]. The Lund plane provides a description of the clustering sequence of a jet in terms of the transverse momentum and angle of each splitting. This allows for a useful visual representation of the clustering history of a jet, as well as an efficient encoding of a jets radiation patterns that can be measured experimentally [4]. However, so far, only the primary Lund plane has been used for jet tagging, which inevitably leads to some loss of information due to the omission of the secondary splittings. In these proceedings, we propose LundNet, a new approach capable of using the full Lund plane as input, relying on graph neural networks to better exploit the complex structure associated with this representation.

Two variants of LundNet models are presented in this paper: the `LundNet` aims at maximizing the performance of jet tagging with no particular constraints on computational cost, while the `FastLundNet` strives on computation efficiency and robustness. This leads to different choices of input features and network architectures.

2.1 LundNet

The `LundNet` adopts the dynamic graph convolutional neural network architecture used in the ParticleNet [5]. The inputs, however, are changed from jet constituent particles to the emissions in the full Lund plane. The graph network starts by constructing a graph for a jet, with each emission being one node of the graph. For `LundNet`, we use the 5-dimensional Lund coordinates,

$$(\ln z, \ln \Delta, \psi, \ln m, \ln k_t),$$

as the node feature, which are defined through the two p_t ordered subjets i and j of a given declustering: $z = p_{t,j}/(p_{t,i} + p_{t,j})$ is the momentum fraction of the softer subjet j; $\Delta^2 = (y_i - y_j)^2 + (\phi_i - \phi_j)^2$, where y_i is the rapidity, a measure of relativistic velocity along the beam axis, and ϕ_i is the azimuthal angle of particle i around the same axis; $\psi = \tan^{-1}\left(\frac{y_j - y_i}{\phi_j - \phi_i}\right)$ is the azimuthal angle around the subjet i axis; m is the invariant mass of the two subjets; $k_t = p_{t,j}\Delta$ is the transverse momentum of j relative to i. Then, each node is connected to its k nearest neighbor nodes in the pseudorapidity–azimuth plane, forming edges of the graph.

The graph convolution is based on EdgeConv [6], which operates in two steps. First, a shared multi-layer perceptron (MLP) is applied to each edge of the graph, using the features of the two nodes connected by the edge as inputs, and produces an "edge feature". Then, an aggregation step is performed for each node, averaging edge features of all the connecting edges and assigning it as the new node feature. Such EdgeConv operation can be stacked, and between two consecutive EdgeConv's, the graph structure can be dynamically updated by redefining the k nearest neighbor–based edges in the latent space learned from the previous EdgeConv layer, thus building a dynamic graph convolutional neural network. A global average pooling is applied to the outputs of the last EdgeConv layer, followed by a dense layer with 256 units and a dropout layer with a drop probability of 0.1, before the final classification output.

The `LundNet` shares the same network architecture as ParticleNet, consisting of three EdgeConv layers with the number of nearest neighbors $k = 16$. Each EdgeConv layer is built with a three-layer MLP. The number of channels of the MLPs are $(64, 64, 64)$, $(128, 128, 128)$ and $(256, 256, 256)$ for the three EdgeConv layers, respectively.

2.2 FastLundNet

The `FastLundNet` is derived from the `LundNet` with a couple of modifications to reduce the complexity and improve the robustness. While both networks use a graph of the emissions to represent a jet, the `FastLundNet` uses the Cambridge/Aachen declustering tree [7, 8] as the graph structure directly, and the graph structure remains unchanged throughout the graph convolutions. This modification significantly reduces the computational cost as it gets rid of the expensive k nearest neighbor search, and also substantially reduces the number of edges of the graph (as each node is connected to no more than three other nodes now). In addition, a simplified, 3-dimensional Lund coordinates,

$$(\ln z, \ln \Delta, \ln k_t),$$

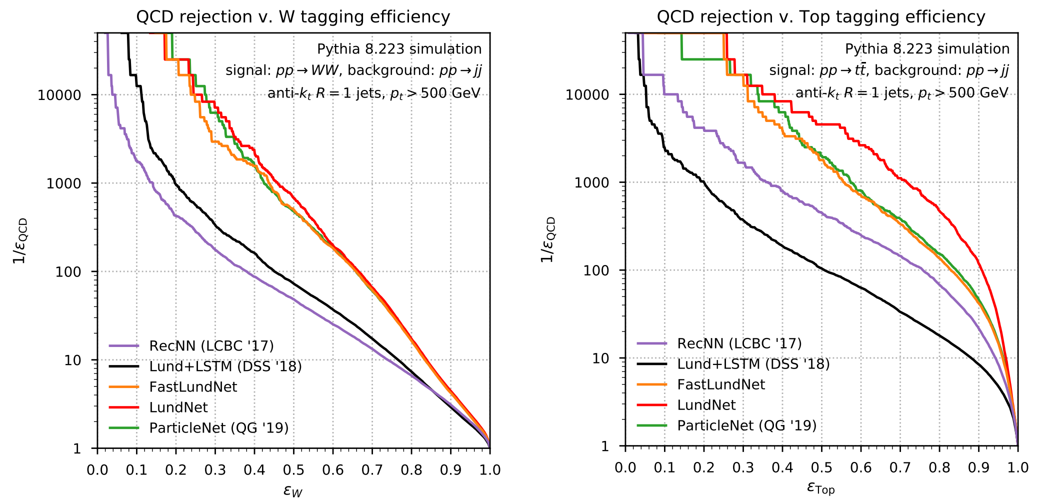

Figure 1: Background rejection for W (left) and top (right) tagging with $p_t > 500$ GeV.

is used as the input features for each emission.

The `FastLundNet` consists of six EdgeConv layers, each built with a two-layer MLP. The number of channels of the MLPs are $(64, 64)$, $(64, 64)$, $(128, 128)$, $(128, 128)$, $(256, 256)$, $(256, 256)$ for the six EdgeConv layers, respectively.

2.3 Implementation

The LundNet is implemented with the Deep Graph Library [9] using the PyTorch [10] backend. The training is performed on a Nvidia GTX 1080 Ti graphics card with a minibatch size of 256. The Adam optimizer [11] is used to minimize the cross entropy loss. The training is performed for 30 epochs, with an initial learning rate of 0.001, and subsequently lowered by a factor of 10 after the 10th and the 20th epochs. A snapshot of the model is saved at the end of each epoch, and the model snapshot showing the best accuracy on the validation dataset is selected for the final evaluation.

3 Identifying jets

As benchmarks, we consider the potential of our models for identifying two hallmark signals at the LHC, which are hadronically decaying boosted electroweak W bosons and top quarks. The data sample consists of 500 000 signal and background jets simulated with `Pythia` 8.223 [12] using fast detector simulation with Delphes v3.4.1 particle flow [13], and is taken from [14]. Jets are clustered using the anti-k_t algorithm [15] with a radius $R = 1.0$ using `FastJet` 3.3.2 [16], and are required to pass a selection cut, with transverse momentum $p_t > 500$ GeV and rapidity $|y| < 2.5$. In fig. 1 (left) we show for each model the background rejection $1/\epsilon_{\mathrm{QCD}}$ against the signal efficiency ϵ_W for W bosons. We compare the `FastLundNet` and `LundNet` models with three recent benchmarks: the ParticleNet model introduced in [5], the RecNN model from [17] and the Lund+LSTM model from the original Lund plane paper [3], which uses an LSTM network on the primary Lund sequence. Both the RecNN and the Lund+LSTM models, while superior to heuristic substructure algorithms, are vastly outperformed by all of the graph based methods considered. The `LundNet` model provides slightly higher performance, but is substantially slower to train than the `FastLundNet` model, and, as we will see below, less robust to non-perturbative effects.

In fig. 1 (right), we show similar results for the identification of top quarks. In this case, the Lund+LSTM model performs poorly, as one would expect, because it uses only the primary Lund declustering sequence, which can contain information about the structure of only either the bottom quark or the vector boson that it initially decays into. It is interesting to see that in this more complex case, the `LundNet` model provides a substantial performance gain over existing state-of-the-art methods such as ParticleNet.

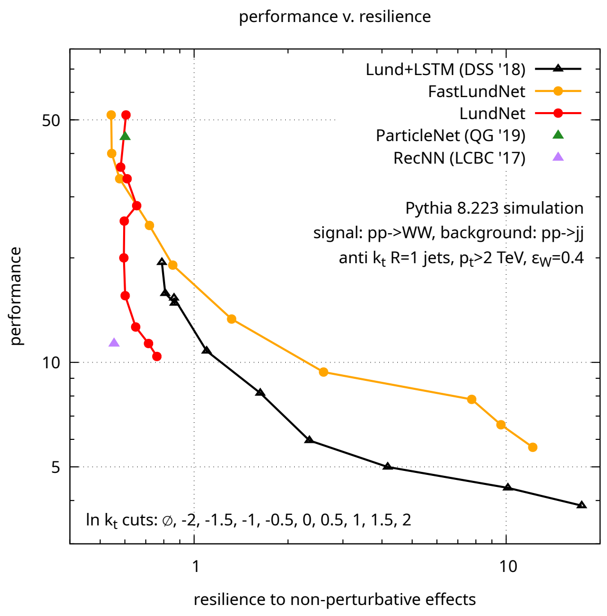

Figure 2: Performance $\frac{\epsilon_W}{\sqrt{\epsilon_{\rm QCD}}}$ versus resilience for non-perturbative effects.

4 Robustness study

Beyond its raw performance, it is important for practical applications that a tagger be relatively robust to model-dependent non-perturbative effects. To carry out studies of sensitivity to non-perturbative effects, we compare performance between a data sample produced at parton level, and a sample after hadronisation (with underlying event models turned on in the event generator). The same model, trained on a hadron-level sample, is used for the comparison.

Figure 2 shows the robustness of the tagger in conjunction with its performance. This robustness is measured through the resilience ζ [18], calculated using both the efficiency on the hadron-level sample, ϵ, and that on the parton-level sample, ϵ':

$$\zeta = \left(\frac{\Delta\epsilon_W^2}{\langle\epsilon\rangle_W^2} + \frac{\Delta\epsilon_{\rm QCD}^2}{\langle\epsilon\rangle_{\rm QCD}^2} \right)^{-1/2}, \qquad (1)$$

where $\Delta\epsilon = \epsilon - \epsilon'$ and $\langle\epsilon\rangle = 1/2\,(\epsilon + \epsilon')$. The efficiencies are obtained with a fixed cut corresponding to a signal efficiency $\epsilon_W = 40\%$ on the hadron-level sample. The curves are obtained by increasing a transverse momentum cut on the k_t variable of the Lund plane, progressively removing declustering nodes that fall below the cut. We can observe that despite their good performance, ParticleNet and RecNN models have very little resilience to non-perturbative effects. Somewhat surprisingly, the `LundNet` also offers relatively poor robustness to non-perturbative effects. This is because of its higher dimensional input state, which allows it to extrapolate some information on emissions in the non-perturbative regime despite the presence of the transverse momentum cut. In contrast, the `FastLundNet` model becomes very resilient to non-perturbative effects as the transverse momentum cut is increased, outperforming the Lund+LSTM model by a factor two for the same resilience value.

5 Conclusions

In these proceedings, we have introduced a novel algorithm to detect signals at the LHC, showing substantial improvements on key benchmarks. We have also provided a concrete study of its resilience to model-dependent non-perturbative effects, establishing an algorithm that combines both performance and robustness. These results offer a promising avenue to construct an effective machine-learning based tagger that can be robust to model-dependent effects present in the training data, a key feature for real-life applications of artificial intelligence at the LHC.

Broader impact

The work presented in these proceedings is focused on applications to collider physics, which presents no ethical or societal concerns, but could potentially find useful applications in related fields with pattern recognition problems in hierarchical data.

References

[1] Gavin P. Salam. Towards Jetography. *Eur. Phys. J. C*, 67:637–686, 2010.

[2] Andrew J. Larkoski, Ian Moult, and Benjamin Nachman. Jet Substructure at the Large Hadron Collider: A Review of Recent Advances in Theory and Machine Learning. 2017.

[3] Frédéric A. Dreyer, Gavin P. Salam, and Grégory Soyez. The Lund Jet Plane. *JHEP*, 12:064, 2018.

[4] Georges Aad et al. Measurement of the Lund Jet Plane Using Charged Particles in 13 TeV Proton-Proton Collisions with the ATLAS Detector. *Phys. Rev. Lett.*, 124(22):222002, 2020.

[5] Huilin Qu and Loukas Gouskos. ParticleNet: Jet Tagging via Particle Clouds. *Phys. Rev. D*, 101(5):056019, 2020.

[6] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds. *ACM Trans. Graph.*, 38(5):146, October 2019.

[7] Yuri L. Dokshitzer, G. D. Leder, S. Moretti, and B. R. Webber. Better jet clustering algorithms. *JHEP*, 08:001, 1997.

[8] M. Wobisch and T. Wengler. Hadronization corrections to jet cross-sections in deep inelastic scattering. In *Monte Carlo generators for HERA physics. Proceedings, Workshop, Hamburg, Germany, 1998-1999*, pages 270–279, 1998.

[9] Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*, 2019.

[10] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[11] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[12] Torbjörn Sjöstrand, Stefan Ask, Jesper R. Christiansen, Richard Corke, Nishita Desai, Philip Ilten, Stephen Mrenna, Stefan Prestel, Christine O. Rasmussen, and Peter Z. Skands. An Introduction to PYTHIA 8.2. *Comput. Phys. Commun.*, 191:159–177, 2015.

[13] J. de Favereau, C. Delaere, P. Demin, A. Giammanco, V. Lemaître, A. Mertens, and M. Selvaggi. DELPHES 3, A modular framework for fast simulation of a generic collider experiment. *JHEP*, 02:057, 2014.

[14] Stefano Carrazza and Frédéric A. Dreyer. JetsGame/data v1.0.0, March 2019. This repository is git-lfs.

[15] Matteo Cacciari, Gavin P. Salam, and Gregory Soyez. The Anti-k(t) jet clustering algorithm. *JHEP*, 04:063, 2008.

[16] Matteo Cacciari, Gavin P. Salam, and Gregory Soyez. FastJet User Manual. *Eur. Phys. J.*, C72:1896, 2012.

[17] Gilles Louppe, Kyunghyun Cho, Cyril Becot, and Kyle Cranmer. QCD-Aware Recursive Neural Networks for Jet Physics. *JHEP*, 01:057, 2019.

[18] J. R. Andersen et al. Les Houches 2017: Physics at TeV Colliders Standard Model Working Group Report. 2018.