Journal of the Korean Physical Society, Vol. 74, No. 3, February 2019, pp. 219∼223

Quark-Gluon Jet Discrimination Using Convolutional Neural Networks

Jason Sang Hun Lee,* Inkyu Park, Ian James Watson† and Seungjin Yang‡
Department of Physics, University of Seoul, Seoul 02504, Korea

(Received 20 June 2018, in final form 5 September 2018)

Currently, newly developed artificial intelligence techniques, in particular convolutional neural networks, are being investigated for use in data-processing and classification of particle physics collider data. One such challenging task is to distinguish quark-initiated jets from gluon-initiated jets. Following previous work, we treat the jet as an image by pixelizing track information and calorimeter deposits as reconstructed by the detector. We test the deep learning paradigm by training several recently developed, state-of-the-art convolutional neural networks on the quark-gluon discrimination task. We compare the results obtained using various network architectures trained for quark-gluon discrimination and also a boosted decision tree (BDT) trained on summary variables.

PACS numbers: 13.90.+i, 13.87.−a, 12.38.Qk, 13.87.Fh
Keywords: Machine learning, Jet tagging, QCD, Jet, Fragmentation
DOI: 10.3938/jkps.74.219

I. INTRODUCTION

In recent years, new techniques have been developed for image analysis and classification. This arose from the finding that neural networks composed of successive layers of convolutional filters operating on the previous layer can be successfully and efficiently trained with networks dozens of layers deep. This is to be compared to traditional densely connected neural networks, where adding multiple layers makes training unstable and slow and generally does not improve performance. This new paradigm of a neural network goes under the moniker of Deep Learning and was most famously adapted for use in the creation of AlphaGo, a Go AI which achieved the unprecedented feat of defeating a Go world champion [1].

Recently, these techniques have been applied to jet physics by interpreting the energy depositions forming a two-dimensional image in (η, ϕ) space [2,3]. The space is pixelized, and the pixel luminosity is proportional to the amount of energy carried by particles of the jet travelling in the direction of the pixel. Convolutional neural network techniques were applied to these jet-images [2]. This has been extended by treating the different types of particles as being different color channels producing a color image representation of the jets [4]. In this paper, we train various recently-developed, start-of-the-art convolutional neural network types to discriminate quark-initiated jets and gluon-initiated jets and compare the results from the different networks. In particular, we discuss the expected performance from 13 TeV LHC data with the CMS detector [5]. Other approaches being investigated for jet physics include geometric deep learning for processing non-Euclidean data, such as graphs and manifolds [6]. The authors of these studies argue that this approach can reduce the loss of information that occurs with pixelization so that classification performance can be improved [7,8].

*E-mail: jason.lee@uos.ac.kr
†E-mail: ian.james.watson@cern.ch
‡E-mail: seungjin.yang@cern.ch

II. MONTE CARLO MODELS

We use MadGraph5_aMCatNLO v2.6.0 to generate the hard process for dijet and Z+jet events at leading order [9]. We separately generate events for quarks and gluons and label the jets "quark" or "gluon" based on the hard process being generated. We do this to avoid ambiguities in the matching process. As outlined in the event selection below, we also require dijets to be balanced to reduce further ambiguities due to hard radiation producing further gluon-like jets. We interface the generated hard-process events to PYTHIA 8.2 with the default PYTHIA tune for parton showering and underlying event generation [10]. We use the fast detector simulator DELPHES to approximate CMS reconstruction particle-flow algorithms [11]. DELPHES uses the FASTJET package for anti-k_T algorithm with a jet radius of $R = 0.5$ [12]. The default settings of the packages have been used to generate events.

