Deep learning in jet reconstruction at CMS

Markus Stoye[1] on behalf of the CMS collaboration
CERN[1]

E-mail: markus.stoye@cern.ch

Abstract. Deep learning has led to several breakthroughs outside the field of high energy physics, yet in jet reconstruction for the CMS experiment at the CERN LHC it has not been used so far. This report shows results of applying deep learning strategies to jet reconstruction at the stage of identifying the original parton association of the jet (jet tagging), which is crucial for physics analyses at the LHC experiments. We introduce a custom deep neural network architecture for jet tagging. We compare the performance of this novel method with the other established approaches at CMS and show that the proposed strategy provides a significant improvement. The strategy provides the first multi-class classifier, instead of the few binary classifiers that previously were used, and thus yields more information and in a more convenient way. The performance results obtained with simulation imply a significant improvement for a large number of important physics analysis at the CMS experiment.

1. Introduction

The reconstruction of jets, sprays of particles that originate from the hadronization process of a quark or gluon, is a central element in high energy physics collider experiments. Jets can originate from the following particles: b quarks, c quarks, light quarks (u, d, s quark), and gluons. The b and c quarks are heavier that the other aforementioned quarks. If the focus is on identifying b and c quarks we speak of (heavy) flavor tagging. If we want to classify gluons, we cannot speak of flavor tagging as they do not have a physical flavor. We speak either of generic tagging if the gluon class is added or of quark vs. gluon tagging, if it is only a binary classifier. Recently several studies, using simplified simulations, were made on applying deep neural networks in jet tagging. It was proposed to use the analogy of the calorimeter cells to pixels in images to apply convolutional or dense networks that are often used for computer vision [1, 2, 3, 4]. The results ranged from no significant improvement to slight improvements with respect to established methods and did not include jets originating from heavy flavor quarks. Also, the use of recurrent neural networks was proposed [5, 6] in context of heavy flavor and other jet tagging.

In section 2, we present the jet tagging inputs in more detail in terms of the intrinsic data hierarchy and complexity. We present in section 3 the currently recommended flavor tagger of the CMS experiment [7] as well as the new proposal that is built on deep learning in the sense that layers for feature engineering are added that use the intrinsic hierarchy of the data. Finally, in section 5, we present results of the former standard CMS flavor tagger (CSVv2), the currently recommended flavor tagger (DeepCSV) and the generic new proposal (DeepJet). We also compare the generic tagging capabilities to other deep neural network architectures for quark vs. gluon classification.

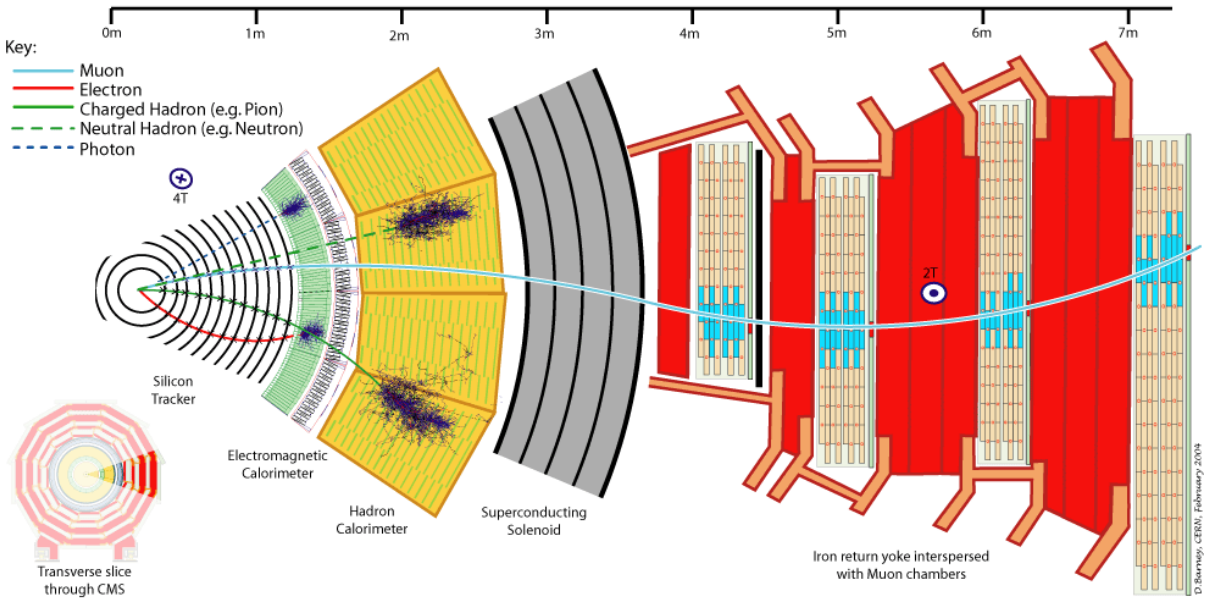

Figure 1. Slice of the CMS detector. It illustrates the different detector types and their different response to different particle types, such as muons or hadrons.

2. Particle flow jets

Particle flow jets consist of a list of particles reconstructed by the particle flow algorithm [12]. A single particle leaves various individual signals in the CMS subsystems. The subsystems are composed of different detector technologies as can be seen in Figure 1, which all have different signal responses to different particles. So, the particle data is very heterogeneous. As different particles carry different information, we split the particles, that are reconstructed by the particle flow algorithm, into two categories; charged particles which we measure in the tracker, and neutral that are only measured in the calorimeter. The object lists are sorted in descending order of a measure of their displacement significance, which for heavy flavor tagging is known to be a key observable.

There is a natural data hierarchy as jets are composed of particles, all of which have features. To build these lists we use the clustering algorithm anti-k_T [13] that identifies particles belonging to one particle spray initiated by a quark or gluon. Typically, fewer than 50 particles are found in such lists. The clustering algorithm has a parameter that is related to the radius of the cone of the particles spray which is set to 0.4. Tracks, i.e. the trajectories of charged particles are used to find vertices from where several tracks originated. Light quarks and gluons hadronize dominantly to short lived hadrons and their tracks originate directly from the proton-proton collision point, called the primary vertex. Secondary vertices are displaced from the primary vertex and are produced by heavy flavor hadrons that do not decay immediately and stem from b or c qaurks. If secondary vertices are found inside the cone of the jet, we add them as additional information to the jet.

3. Neural network designs

For the DeepCSV tagger we first apply quality criteria to tracks. Then we use the first six most displaced tracks and build seven features by hand that are traditionally used for b tagging [14]. Also, information from the most displaced vertex is used and again the features are similar to

those in [14] as well as the seven global features that we use, such as the transverse momentum of the jet or number of charged particles in the jet. All these features (around 60) are the input to a dense neural network with six layers and 100 nodes per layer.
For the deep learned DeepJet tagger the input is significantly extended. No quality criteria are applied for the charged particles of the jet. We use the features that are used for the heavy flavor tagging of DeepCSV and, in addition we also use features that are likely useful for jet classification, e.g. features that are related to the quality of tracks. All together this makes 18 features per charged particle. For neutral particles, we only use six features; some of which are calculated with respect to the secondary vertex, e.g. the opening angle between the particle and the first secondary vertex, if present. Finally, we use up to four secondary vertices that were reconstructed in the jet cone and use again the same features as in DeepCSV and a few additional ones, adding up to 12 features per vertex. These lists are then input to the neural network architecture. The first step is to engineer features per particle or vertex. For this purpose, several 1D 1x1 convolutional layers are applied to the lists of objects. For charged particles and secondary vertices, four layers with 64, 32, 32, and eight nodes are used. For the neutral particles with considerably less information content only 3 layers are used with 32, 16 and 8 nodes. This strategy allows us to use many input features per particle, which is important as it allows to use rather complete information for each particle or vertex. Altogether the number of input features is around 650. The output of the convolutional layer is then given to a LSTM [15] recurrent neural network. The output of the three recurrent networks that correspond to the charged particles are 150, 50, and 50 intermediate features that correspond to the charged particles, the neutral particles and the secondary vertices, respectively. These intermediate features are concatenated with eight global features, that are used traditionally in flavor tagging, and given to a dense neural network with six layers with 200 nodes for the first and 100 nodes for the remaining layers. For both DeepCSV and DeepJet the ReLU [16] activation function is used, and the softmax function for the last layer. We add DropOut [17] (rate 0.1) layers and batch normalization [18]. For optimization, the Adam optimizer [19] is used with a learning rate of $\epsilon = 10^{-8}$ without automatic learning rate decay. Instead, the learning rate is decreased manually during training time if the loss remains constant. For the technical implementation, we use Keras [20] and Tensorflow [21].

4. Training strategy

Both DeepCSV and DeepJet differ from previous taggers also in the training strategy. In both cases for the first time in CMS a multi-class classification approach is used for the classes. For DeepCSV the light quark class and the gluon class are merged, as usually done in flavor tagging. For DeepJet though these are individual classes to provide a generic tagger.
Another change with respect to previous taggers is that a larger and more diverse sample is used for the training. We use simulated events from two processes, namely QCD and top quark pair production. The more diverse samples reduce the danger of obtain a classifier too specific to a process. The larger sample sizes allow to increase the complexity of the tagger without danger of overfitting. Depending on the simulation version we use 40 to 100 million jet samples. Typically, we reserve 10% of the samples for both, validation and development. The training needs about 30 minutes on a NVIDIA GTX 1080 GPU for DeepCSV and 24 hours for DeepJet.

5. Performance comparison in real data and simulations

Figure 2 compares the DeepCSV to the old tagger (CSVv2) [14] of the CMS experiment and a significantly better performance is observed. At the same true positive rate (b jet efficiency) of about 65%, the false positive rate (misidentification probability) for uds quarks and gluon

is reduced from 1% to 0.6%. The 1% false positive rate is a typical working point used for classification. The improvement originate from the changes in feature selection, new training samples, and a new machine learning algorithm. The efficiencies (true positive rate) of the tagger above a certain threshold on the discriminator are evaluated in real collision data as well as in simulation, details of the procedure are described in Ref. [14]. Figure 3 illustrates the so-called scaling factors that measure the difference in tagging efficiencies in simulation and real collision data. DeepCSV and CSVv2 show the similar agreement in simulation and data, which means that DeepCSV is also more performant in real data collisions. Also for other classes DeepCSV outperformed the old CMS taggers, specifically the binary c quark vs. light quark and c quark vs. b quark taggers.

Figure 4 shows the comparison of DeepCSV to DeepJet (labeled DeepFlavour as it is used for

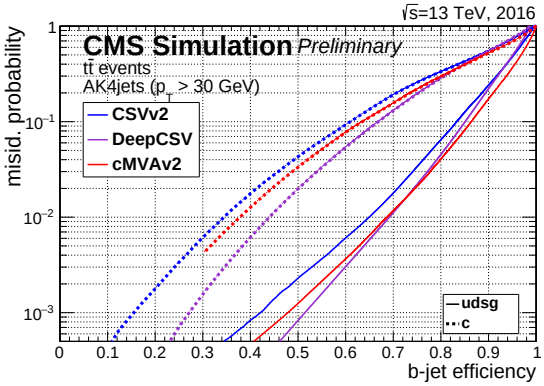

Figure 2. [8] True positive (b jet efficiency) rate vs. false positive rate of light jets and c quarks in simulation with 2016 conditions for jets with more than 30 GeV transverse momentum from top quark pair production. DeepCSV and CSVv2 are explained in the text. cMVA uses muon information, which for validation purposes is not used by other taggers.

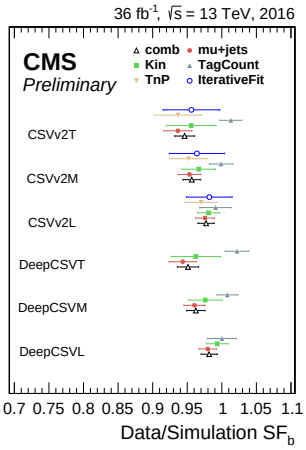

Figure 3. Difference in b jet efficiency between real data and simulation for different working points (loose (L), medium (M), and tight (T) with 10%, 1%, 0.1% false positive rate, respectively) for DeepCSV and CSVv2. Several complementary methods are used as described in [14, 8].

flavor tagging in this case). A further significant gain is observed; the false positive rate is reduced from 1% to 0.7% for 78% true positive rate. We also tested using the extended input of DeepJet as the input to an extended dense neural network. The resulting performance is labeled noConv in Figure 4 and shows that the more sophisticated custom architecture is needed. We also tested using the input of DeepCSV as input for the DeepJet architecture and effectively reproduced the DeepCSV result. We tested in several steps increasing the input from DeepCSV to DeepJet and found that a significant improvement was achieved by not using the quality criteria for particle selection of DeepCSV and CSVv2. The second biggest impact were the additional features per charged particles. The additional secondary vertices and neutral particles gave only modest gain, but are also important for other classes of the generic DeepJet tagger. In Figures 6 and 5 DeepCSV and DeepJet are compared at very high transverse momentum of the b jets. The gain is very significant; the false positive rate is reduced from 1% to only 0.12% at the usual working point. Also here a main limitation was the charged particle quality selection and as well the upper limit of only six charged particles (tracks) that were used in DeepCSV. We thus

learned that in order to reduce the input feature dimension, useful information was lost in the past flavor taggers of CMS.
For quark vs. gluon separation we compare DeepJet to custom deep neural networks. In [3] it was shown that these slightly outperform the classical approaches, namely using only a few handcrafted features that are input machine learning tools. As reference we used a neural network as described in [3], i.e. a 2D convolution layers working on a jet image. To produce images the continuous particle positions are made discrete, i.e. they are pixelized. We also built a slimmed down version of DeepJet, as for light quark vs. gluon separation it is evident from first principle arguments that just a fraction of the input to DeepJet is relevant. We used only 4 features per particles and removed the 1x1 convolutional layers. The secondary vertices were dropped as well. The slimmed down version of DeepJet is labeled recurrent in the Figure. Figure 7 compares the three networks, that all were trained with the same samples. DeepJet and the slimmed down version of DeepJet performed similarly well and only marginally better than the 2D convolution. The 2D convolution works well because quark vs. gluon separation mostly dependent on particle densities and energies densities, which can well be represented in images. Heavy flavor tagging however is much more complex.

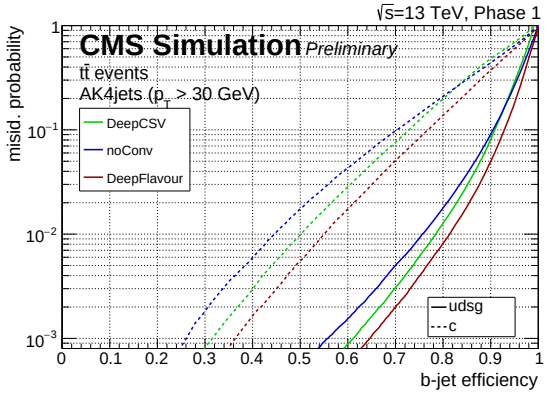

Figure 4. [9] True positive rate (b jet efficiency) vs. false positive rate of light jets and c quarks in simulation with 2017 conditions for jets with more than 30 GeV transverse momentum from top pair production. The jet-taggers DeepJet, DeepCSV, and noConv are described in the text.

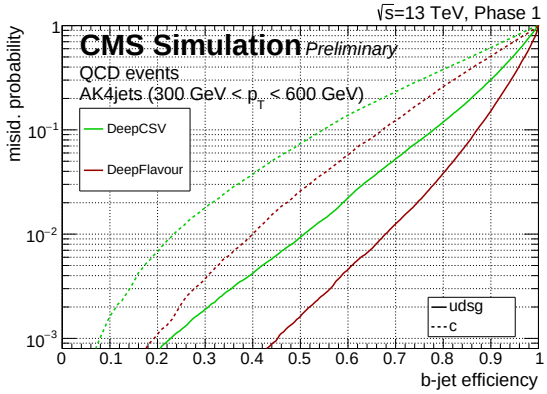

Figure 5. [9] True positive rate (b jet efficiency) vs. false positive rate of light jets and c quarks in simulation with 2017 conditions for jets in the range of 300-600 GeV of transverse momentum from QCD processes. The DeepJet and DeepCSV taggers are described in the text.

6. Conclusion

We present a new network architecture for jet tagging for the LHC experiments. DeepCSV, the currently recommended tagger of CMS based on a deep neural network, led to significant gain in tagging performance and was validated in real collision data. An even more recent development is DeepJet, a generic tagger for all hadron classes, which is based on deep learning. It outperforms the other taggers and especially at large transverse momenta of the jets the gain is very significant, nearly on order of magnitude less false positive rate for the standard threshold definitions. If the expectations from simulations are confirmed in real data that would yield to a significant improvement of the scientific output of the CMS experiment.

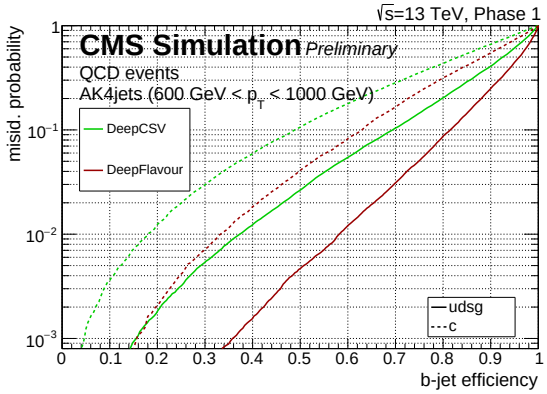

Figure 6. [9] True positive rate (b jet efficiency) vs. false positive rate of light jets and c quarks in simulation with 2017 conditions for jets in the range of 600-1000 GeV of transverse momentum from QCD processes. The DeepJet and DeepCSV taggers are described in the text.

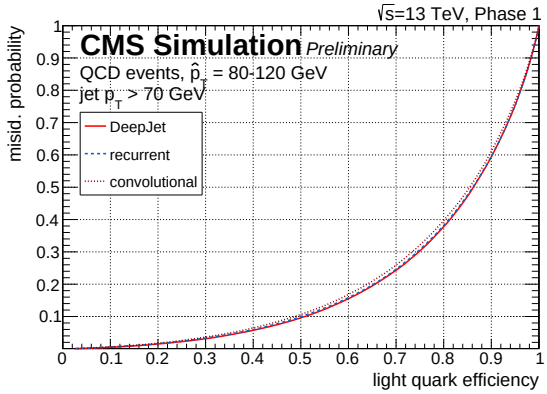

Figure 7. [10] True positive rate (light jet efficiency) vs. false positive rate of gluon jets in simulation with 2017 conditions for jets in the range of 80-120 GeV of transverse momentum from QCD processes. Three different (DeepJet, recurrent, convolutional) neural network based taggers are tested. Only DeepJet is a multi-class classifier that can also be used for flavor tagging.

References

[1] J. Cogan, M. Kagan, E. Strauss and A. Schwartzman, "Jet-Images: Computer Vision Inspired Techniques for Jet Tagging," JHEP **1502**, 118 (2015)

[2] G. Kasieczka, T. Plehn, M. Russell and T. Schell, "Deep-learning Top Taggers or The End of QCD?," JHEP **1705**, 006 (2017)

[3] P. T. Komiske, E. M. Metodiev and M. D. Schwartz, "Deep learning in color: towards automated quark/gluon jet discrimination," JHEP **1701**, 110 (2017)

[4] P. Baldi, K. Bauer, C. Eng, P. Sadowski and D. Whiteson, "Jet Substructure Classification in High-Energy Physics with Deep Neural Networks," Phys. Rev. D **93**, no. 9, 094034 (2016)

[5] D. Guest, J. Collado, P. Baldi, S. C. Hsu, G. Urban and D. Whiteson, "Jet Flavor Classification in High-Energy Physics with Deep Neural Networks," Phys. Rev. D **94**, no. 11, 112002 (2016)

[6] G. Louppe, K. Cho, C. Becot and K. Cranmer, "QCD-Aware Recursive Neural Networks for Jet Physics," arXiv:1702.00748 [hep-ph] (2017)

[7] CMS Collaboration, "The CMS Experiment at the CERN LHC," JINST **3**, S08004 (2008)

[8] CMS Collaboration, "Heavy flavor identification at CMS with deep neural networks," Detector Performance Summary: CMS-DPS-17-005, http://cds.cern.ch/record/2255736, (2017)

[9] CMS Collaboration, "CMS Phase 1 heavy flavour identification performance and developments," Detector Performance Summary: CMS-DPS-17-013, http://cds.cern.ch/record/2263802, (2017)

[10] CMS Collaboration, "New developments for jet substructure reconstruction in CMS," Detector Performance Summary: CMS-DPS-17-027, https://cds.cern.ch/record/2275226, (2017)

[11] ATLAS Collaboration, "Identification of Jets Containing b-Hadrons with Recurrent Neural Networks at the ATLAS Experiment," ATLAS note: ATL-PHYS-PUB-2017-003, http://cds.cern.ch/record/2255226, (2017)

[12] CMS Collaboration, "Particle-flow reconstruction and global event description with the CMS detector," JINST **12**, no. 10, P10003 (2017)

[13] M. Cacciari, G. P. Salam and G. Soyez, "The Anti-k(t) jet clustering algorithm," JHEP **0804**, 063 (2008)

[14] CMS Collaboration, Identification of b-quark Jets at the CMS Experiment in the LHC Run2 Startup," CMS public analysis summary BTV-15-001, (2015)

[15] S. Hochreiter and J. Schmidhuber, "Flat Minima," Neural Computation 9(1):1-42, http://people.idsia.ch/ juergen/fm/ (1997)

[16] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," Proceedings of the 27th international conference on machine learning (ICML-10), pp. 807–814. (2010)

[17] N. Srivastava *et al.*, "Dropout: a simple way to prevent neural networks from overfitting," Journal of machine

learning research **15**, no. 1, 1929–1958. (2014)
[18] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," International Conference on Machine Learning, pp. 448–456. (2015)
[19] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," arXiv: 1412.6980 [cs.LG] (2014)
[20] Chollet, François et al., "Keras," https://github.com/fchollet/keras, (2015)
[21] Martín Abadi et al., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," https://www.tensorflow.org (2015)