Application of deep learning in top pair and single top quark production at the LHC

Ijaz Ahmed\textsuperscript{1,a}, Anwar Zada\textsuperscript{2,b}, Muhammad Waqas\textsuperscript{3,c}, M. U. Ashraf\textsuperscript{4,d}

\textsuperscript{1} Federal Urdu University of Arts, Science and Technology, Islamabad, Pakistan
\textsuperscript{2} Institute of High Energy Physics, University of Chinese Academy of Sciences, Beijing 100049, China
\textsuperscript{3} Riphah International University, Islamabad, Pakistan
\textsuperscript{4} Centre for Cosmology, Particle Physics and Phenomenology, Université catholique de Louvain, Ottignies-Louvain-la-Neuve, Belgium

Received: 24 January 2022 / Accepted: 23 August 2023
© The Author(s), under exclusive licence to Società Italiana di Fisica and Springer-Verlag GmbH Germany, part of Springer Nature 2023

Abstract We demonstrate the performance of a very efficient tagger applies on hadronically decaying boosted top quark pairs as signal based on deep neural network algorithms working with Lorentz Layer and the Minkowski metric. Due to limited computing resources, we could show only the receiver ordering characteristic curve, accuracy and loss which illustrates the trade-off between signal acceptance against huge QCD multi-jet background acceptance. Alternatively, we also report the modern machine learning approaches and apply multivariate technique on single top quark production through weak interaction at $\sqrt{s} = 14$ TeV proton-proton Collider to demonstrate its observability against the most relevant Standard Model backgrounds through the techniques of boosted decision tree (BDT), likelihood and multilayer perceptron (MLP). The analysis is trained to observe the performance of classifiers in comparison with the conventional cut based and counting approach.

1 Introduction

The top quark is the heaviest elementary particle discovered so far which is very uncommon quark in many ways. One of the possible fact is, the electroweak decay of the top quark is faster than the timescale of hadronization indicating its existence as the only free quark. The comparison of precise measurements from the Standard Model (SM) predictions would be helpful to better understand the possible effects from new physics. More than twenty years ago, $(t\bar{t})$ pair-production process was discovered and was entered in the domain of precision physics" later after many years [1, 2]. Tevatron experiment was the first to observe the production of single top-quark just few years before its shut-down [3, 4]. Later on the CMS [5] and ATLAS [6] experiments recorded data at $\sqrt{s} = 7$ TeV in 2010 [8] and in early 2011 [9] respectively and re-discovered” the production of single top-quark from the Feynman diagrams given in Fig. 1. In the s-channel diagram The W-boson which is involved in this process is time-like ($q^2_W \geq 0$) process involves two incoming quark weak isospin partners. A top quark and a hard $\bar{b}$ quark are produced in the final state. While in t-channel the involved W-boson is space-like ($q^2_W \leq 0$) is the largest source of single top at the LHC with a cross section comparable with the $t\bar{t}$ cross section. This channel is described by two diagrams, a $2 \rightarrow 2$ and a $2 \rightarrow 3$. A high momentum and forward light quark and a top quark are expected in the $2 \rightarrow 2$ and $2 \rightarrow 3$ processes. An additional b-quark is produced in the $2 \rightarrow 3$ process. The involved W-boson in this process is a real W-boson (This process involves a b-quark from one proton sea and a gluon from another proton sea. The products are a real W-boson and a real top quark. At various hadron colliders, the production of single top-quark is useful tool to study and hence understand the electroweak $Wtb$ vertex and Cabibbo-Kobayashi-Maskawa (CKM) matrix $V_{tb}$. The cross sections of single top processes are proportional to $|V_{tb}|^2$, $\sigma \propto |M|^2 \propto |V_{tb}|^2$. Therefore, single top processes provide the only way to measure $V_{tb}$ directly. Top quark decays before hadronization scale. Therefore its spin information is transferred to its final decay products. Single top quark provides a possibility to study the polarization of a single quark (Fig. 2).

The top pair quark is the most common process by strong interaction in both Tevatron and LHC. First top was observed in pair form at Tevatron in 1995. The annihilation of quark anti-quark and gluon fusion produces top pairs. In compared to gluon fusion, top-pair at Tevatron using annihilation of quark anti-quark was dominating, while gluon fusion produces the top pair at LHC as shown in Fig. 16.

\textsuperscript{a} e-mail: ijaz.ahmed@fuuast.edu.pk (corresponding author)
\textsuperscript{b} e-mail: azekhan72@yahoo.com
\textsuperscript{c} e-mail: wiki_agral@yahoo.com
\textsuperscript{d} e-mail: muhammad.ashraf@uclouvain.be

Published online: 09 September 2023
The ATLAS experiment is one of the most important LHC experiments for detecting the decay products of top quark. The top quark decays in $\approx 100\%$ to a $b$ quark and a $W$ boson, according to the measured CKM matrix elements and the SM. The $b$ quark then hadronizes to form a narrow cone of hadrons (jet). In ATLAS detector, exact signature of the top quark depends on $W$ boson decay mode from the top quark. The $W$ boson can decay leptonically ($\approx 32\%$) or hadronically ($\approx 68\%$), with signature of one lepton or two jets respectively and missing energy from a neutrino. In case of hadronic decay of the top quark’s $W$, the top quark is known as hadronic and in case of leptonic decay it is referred as leptonic.

In this work, we studied the applications of deep learning in top pair and single top production at the LHC. In Sect. 2, the brief discussion of deep learning and artificial intelligence is discussed. Dataset is described in Sect. 5. In Sect. 4, Multivariate analysis for Jet substructure study is discussed. Results and discussion is presented in Sect. 4.3.1. A preprint of this study has previously been published as arXiv of Cornell University [7].

2 Deep learning and artificial intelligence (AI)

Intelligence that has been designed by the humans to accomplish the human tasks is known as Artificial Intelligence (AI) and is integrated with computer systems to develop AI tools that eventually works as “thinking machines” units. It also employs computer science and data to allow machines to solve problems. Some important types of AI is discussed briefly. Machine learning (ML) is kind of artificial intelligence which allows an AI system to automatically adapt from its surroundings and apply that knowledge to make good judgments. It employs a range of algorithms to continuously learn, explain, and enhance data in order to anticipate better results. Computer scientists can use machine learning to ‘train’ a machine by providing it vast volumes of data. To evaluate and make inferences from data, the machine employs a series of rules known as an algorithm. The more data a machine processes, the better it can perform a task or make a conclusion. ML is further divided into three categories, one is supervised ML, unsupervised ML and semi-supervised ML. ML uses algorithms (processes) to output models which can be likened to computer programs. By comparison, deep learning (DL) uses a layered structure of algorithms collectively called an artificial neural network to produce DL models which are highly accurate. Both ML and DL “learn” from training data which is data that is relative to the task and representative of the data the models use after training. Specifically, both find and note patterns within huge quantities of data. These processes are also incredibly fast, reducing time for such tasks from months, in some cases to mere minutes, or a few days. This creates incredible efficiencies over using human workers to complete the same tasks. A convolutional neural network (CNN) are distinguished from other neural networks by their superior performance with image, speech, or audio signal inputs.

2.1 Machine learning in experimental high energy physics

The difficulties of identifying small signals from the massive LHC backgrounds have aided the introduction of machine learning (ML) approaches for classification. These methods are a sort of supervised learning in which the outputs are constrained to a small number of variables or classes, such as signals or backgrounds. The data analysis of Higgs boson discovery is a very good example of application of ML in high energy physics. The CMS [5] and ATLAS [6] collaborations discovered the Higgs boson in 2012, marking
the first time that Boosted Decision Trees (BDT) were used in such a high-stakes investigation for the isolation of small signals (irreducible mass peaks) across huge smoothly dropping backgrounds [10]. Another sort of supervised ML is regression algorithms, which produce continuous outputs with any numerical value inside a range. They can be used for reconstruction in high energy physics, for example, where exact measurements of continuous parameters such as track momenta, hit positions or jet energies are required [10]. Machine-based categorization methods are quickly replacing conventional HEP procedures. The LHCb experiment at Large Hadron Collider (LHC) focuses on the physics of ‘beauty’ quarks. The ring-imaging Cherenkov detectors, hadron calorimeters, tracker, electromagnetic calorimeters and muon chambers all require identifying the sorts of long-lived charged particles. Machine learning approaches are being used to develop global particle identification (PID) [11]. Machine learning is used to detect jets made up of heavy quarks e.g., (c, b, t) or light quarks and m (u, d, s) quarks, gluons, and W, Z, and H bosons in a variety of jet classification challenges. Typically, these categorization issues have been divided into three categories. 1) Flavour tagging is the process of identifying jets containing b-hadrons (b-jets), dis-criminating them from jets containing c-hadrons and no b-hadrons (c-jets) and from those containing neither c nor b-hadrons (light-flavour jets). 2) jet substructure tagging method is very useful in such events where boosted partons arise from TeV scale resonances which make use of the substructure of large jets e.g., W, Z, t and H are necessary to discriminate from QCD jets. 3) quark-gluon tagging technique being very useful in distinguishing quarks from gluons based on low-level detector output is one of the most challenging applications of multivariate and machine learning techniques [12]. The details can be found in two most important classification publications references [13, 14]. Quality monitoring is an important step in ensuring that only accurate data, in which all systems work as designed, enter into the physics analysis chain, whereas poor data is identified. At the same time, eliminating useful data is not a wise idea. This takes time at the LHC and includes detectors and online expertise. This is performed in the CMS research by comparing multiple histograms having detector variables to conventional working point references [11]. The CMS collaboration proposes a new method for anomaly identification based on auto-encoders [15]. A luminosity section (LS) last for ≈ 23 s at the LHC and is determined by a specific number of protons orbits in the accelerator. Inside the run, each LS gets its own label, which starts at one and goes up from there. The idea is to use machine learning to give data assurance at the LS level [11]. Top quark partons play a significant role in a variety of scenarios that are beyond the SM. The top quark have a very high transverse momentum when created by a massive new particle, and well known as boosted top quark. In this phase, studying the large radius jet’s substructure, which completely comprises the hadronic decay products, is more effective than resolved reconstruction. Top taggers are the algorithms that detect these decays [16]. The decay products of the top quarks are collimated within one big jet at high transverse momenta (p_T), identifying boosted hadronically decaying top quarks is difficult. Substructure method are used by top tagging algorithms to differentiate these jets from background jets. In many circumstances, the algorithms use various subject finding approaches to recreate the jet’s decay products. The top quark’s mass is another significant variable that can be accessed by different methods. Several algorithms are currently available, with different properties and methods [16]. One of the major purposes of modern machine learning is to extract as much information as possible from a dataset. The data structure is used to develop models in successful implementations. Particle collisions in experiments are recreated in high-energy physics (HEP) by mixing the energy deposits left by particles after passing through different segments of a detector. Sub-detector’s data can be merged to provide a comprehensive description of each particle generated. Jets are ubiquitous items created in pp collisions at the LHC. Jets are results of the hadronization of gluons and quarks, which produce a collimated spray of particles. The task of determining the origin of a jet generated by the hadronization of a quark or a gluon is known as tagging. Top tagging is a recent approach to identifying boosted hadronic top quarks. It avoids reconstructing individual top decay products and instead uses a jet algorithm to reconstruct the entire top decay. The general concept for top tagging is to combine all three jets from a boosted top into single largeRJet, named as top jet by using the anti-kt jet algorithm with large radius e.g. 1.0. Then the largeRJets can be tagged to distinguish between background jets and top jets resulting from other processes, by using the internal structure of jets. For example, in pp collisions, QCD production of two jets. The observable, is one jet shape variable having potential for detecting the top jets, is called N-subjettiness. In a largeRJet, it “counts” the number of subjects.

Top and anti-top, which exist in pairs, is the most well-known top-quark production process, and it is based on strong contact. In strong interaction, two strategies for creating top quarks are possible: gg fusion and qg annihilation. Higher order adjustments are done to these processes at NLO, considering both real and virtual gluon emission. Other mixed networks, like 𝑞𝑞 or 𝑔𝑔, become available as well.

3 Datasets

The public datasets [17–19] are used for current measurements which consists of hadronic top anti-top pair production as signal process, produced at √s =14 TeV with qcd di-jets as background being the main process, while for detector simulation, delphes ATLAS detector card with Pythia8 is used. We cluster particle-flow entries (produced by Delphes E-flow) into anti-kT 0.8 jets in the P_T range [550, 650] GeV. All top jets are matched to a parton-level top within ΔR=0.8, and to all top decay partons within 0.8. We also require |η|_{jet}<2. The mixed quark gluon jets (dijet production) are produced by using pythia 8.2.15 [20] by ignoring multiple interactions and pile-up at √s = 14 TeV. In order to include the detector simulation by default ATLAS detector card, Delphes 3.3.2 [21] has been used. The curvature radius of a particle trajectory in the magnetic field of 2 Tesla is taken as 1.15 m. This helps to evaluate the tracking efficiency and momentum smearing changes with η. FastJet 3.1.3 [22, 23] is used to define substructure
Fig. 3 The details of the dataset containing the test, train and validation samples of the signal events.

Fig. 4 a Transverse momentum, b pseudo-rapidity ($\eta$), c mass and d $\phi$ of Jets of the signal and background events.

container largeRjet through anti $k_t$ algorithm with the $R = 0.8$. In each event only the leading jets are stored having the transverse momentum range: $p_T > 550 – 650$ GeV. For signal of jets, the partonic level top quark and its decay product are considered to be in the largeRjet ($\Delta R = 0.8$ from the jet axis). No matching is done for QCD jets and also the $|\eta_j| < 2$. Particle flow objects (constituents) are extracted through the Delphes energy-flow algorithm, and the 4-momenta of the leading 200 jet constituents are stored. For jets with less than 200 constituents, simply zero-vectors are added, and their four momentum are represented as $(E_0, P_x, P_y, P_z)$ in data as 4-vectors with the energy and the momentum of the jet constituents.

Particle collisions at high energies produce many particles which are often short-lived. These short-lived particles decay into particles of the final state, sometimes through a cascade of multiple decays. Intermediate particles can be reconstructed by using energy-momentum conservation when a parent particle decays into its daughter particles. By assigning to each particle a four-vector defined by the particle energy and its momentum vector, the sum of the four-vectors of the daughter particles gives the four-vector $(E, P_x, P_y, P_z)$ of the parent particle. The four-vectors of the final-state particles uses as input. They are sorted on the basis of ascending order of $p_T$. They are splitted into three categories: First, training contains total 1.2 M events having 600 k signals and background jets each. Second, validation contains total 400 k events having 200 k signals and background jets each. Finally, the testing samples contains 400 k events having 200 k signals and 200k background jets. All the algorithms are optimised for better comparison using training and validation of data samples, and the results are produced using the test sample.

Figure 3 shows the transverse momentum distributions of jets corresponding to train, test and validated events.

In Fig. 4, the signal and backgrounds events are plotted together before selection cuts.
3.1 Model

A new machine learning approach is used to jet substructure TAGGER instead of depending on analogies to natural language recognition or image, investigation of the components of the largeRjets has been done directly, separating signal from background using only special relativity elements such as the Minkowski metric and Lorentz group. Lorentz layer (LoLa) with a Combination layer (CoLa) is used, and two fully connected layers to generate a deep neural network (DNN) for our DeepTopLoLa tagger. The input 4-moments relate to the calorimeter towers in the standard setup [19].

3.1.1 Combination layer (CoLa)

A set consisting of \( N \) measured 4-vectors are the basic components of any subjet analysis, sorted by \( p_T \), for example organized as the matrix

\[
k_{i,j} = \begin{pmatrix}
  k_{0,1} & k_{1,1} & \cdots & k_{0,n} \\
  k_{1,1} & k_{1,2} & \cdots & k_{1,n} \\
  \vdots & \vdots & \ddots & \vdots \\
  k_{3,1} & k_{3,2} & \cdots & k_{3,n}
\end{pmatrix}
\]

Two physics-inspired parts comprises our tagger. We initiated a process by multiplying \( C_{i,m} \) with the four vectors above. This defines our Combination layer, which is motivated by the treatment of jet clustering in the non-deterministic Q-jets technique.

\[
k_{r,l} \xrightarrow{\text{CoLa}} k_{r,l} C_{i,m}
\]

It generates \( M \) coupled four vectors \( \tilde{k}_m \) from the original \( N \) input four-vectors, with and \( m = 1 \ldots M \) and \( l = 1 \ldots N \). Our neural network’s CoLa matrix has the following trainable form in general.

\[
C = \begin{pmatrix}
  1 & 0 & \cdots & 0 & C_{1,N+1} & \cdots & C_{1,M} \\
  0 & 1 & \cdots & 0 & C_{2,N+1} & \cdots & C_{2,M} \\
  \vdots & \vdots & \ddots & \vdots & \vdots & \cdots & \vdots \\
  0 & 0 & \cdots & 1 & C_{N,N+1} & \cdots & C_{N,M}
\end{pmatrix}
\]

It ensures that the set of \( M \) 4-moments \( \tilde{k}_m \) contains both a trainable set consisting of linear combinations \( M - N \) and the original momentum \( k_i \). A deep neural network (DNN) will be used to investigate these \( \tilde{k}_m \).

3.1.2 Lorentz layer (LoLa)

The relevant distance measured in both two substructure items, according to fundamental theory, is the Minkowski metric. We use it to build a weight function that makes learning the underlying features relatively easy for the DNN. We can choose a conversion that maps the constituent 4-vectors to values more closely related to physical observables because each constituent momentum is described individually by four degrees of freedom. To achieve so, we develop a Lorentz layer as the second portion of the DNN, that maps the constituent 4-vectors to values more closely related to physical observables because each constituent momentum is described individually by four degrees of freedom. We can choose a conversion that converts the \( M \) 4-moments \( \tilde{k}_m \) into the equal number of measurement-motivated items \( \tilde{k}_m \).

\[
\tilde{k}_m \xrightarrow{\text{LoLa}} \tilde{k}_m = \begin{pmatrix}
  m^2(\tilde{k}_m) \\
  p_T(\tilde{k}_m) \\
  u_{(E)mn}(\tilde{k}_m)
\end{pmatrix}
\]

where \( \xi_{mn} \) represent the Minkowski distance between \( \tilde{k}_m \) and \( \tilde{k}_m \), 4-momenta.

\[
\xi_{mn}^2 = (\tilde{k}_m - \tilde{k}_n)_\mu \gamma^\mu (\tilde{k}_m - \tilde{k}_n)_\mu
\]

and \( u_{mn} \) is the weights matrices updated all through the network’s training. These four entries show various structures that can be used in Lorentz layer. The first Individual \( \tilde{k}_m \) are mapped on invariant mass, and second one is mapped on transverse momentum by the \( \tilde{k}_m \). With a trainable vector of weights \( u_{(E)mn} \), where \( n = 1, 2 \ldots M \), the third entry creates a linear combination of all energies. We can get a set of \( M \) copies of this linear combination by changing the value of \( m \). The fourth entry manages to combine all \( \tilde{k}_m \) with a fixed \( \tilde{k}_m \), along with a trainable weights vector \( u_{(E)mn} \). We enhance the performance of the network by inserting 4 copies with trainable weights that are independent for the last entry with the weights \( u_{(E)mn} \). From these 4-copies, two minimize over internal index and remaining two sum over it. The network model i.e. LoLa [24] is trained using KERAS [25] with THEANO [26] and ADAM [27] optimizer. The learning rate is 0.001. The learning rate determines the step size in the numerical minimization of loss-function.
Fig. 5 a Epoch versus Accuracy, b Epoch versus Loss, c The ROC curve summarizing the performance for the tagger DeepTopLoLa

Fig. 5 a Epoch versus Accuracy, b Epoch versus Loss, c The ROC curve summarizing the performance for the tagger DeepTopLoLa

training of the model terminates, when the performance of test sample does not improve after several epochs. After training of the model with initial seed weights, the performance is then tested by the validation sample.

Figure 5a shows the epoch as a function of accuracy curve, which approaches 90% and shows the efficiency of our model. Its classification accuracy trained to recognize the signal (top quark) from the backgrounds. Figure 5b shows the training and validation loss versus number of epochs, that shows how after each epoch loss is reducing. Training terminates after 17 epoch and shows how after each epoch loss is reducing. Figure 5c is Receiver Operating Characteristic curve (ROC curve), False positive rate (FPR) vs True Positive Rate (TPR). As area under the curve (AUC) is 0.97 so we conclude from ROC curve that used model 3.1.2 method have high classification accuracy of separating signal from backgrounds. These plots are drawn for LoLa algorithm, so similarly we can obtain plots for other algorithms by making change in our main file. LoLa is for Learning with Opponent-Learning Awareness, a system where each entity modifies the expected learning of other entities in the environment.

4 Multivariate analysis for jet substructure study

The TMVA [28] stands for Toolkit for Multivariate Data Analysis with the ROOT framework. In HEP, to explore for the smaller signals from huge datasets, maximum information has to be extracted for analysis. Based on machine learning, the Multivariate classification technique becomes basic for many of the data analysis methods. Incorporated with the ROOT framework, the TMVA toolkit has a huge diversity of classification algorithms. Training, testing, progress assessment and use of all accessible classifiers are executed at the same time and are simple to use. Supervised machine learning is used in all TMVA methods. They utilize the
Fig. 6 Separation of signal and background of initial tree of BDT with adaptive booster by applying selection cuts of $\text{Jet}_{\text{PT}} > 20$ GeV

Fig. 7 Input variables of Deep-Top for multivariate analysis

Fig. 8 Network Architecture for the multivariate analysis

training events which determines the required outputs. TMVA is mainly planned for the requirements of HEP applications, however ought not be limited to these.

The TMVA classification and regression analysis are mainly classified into two independent stages: Firstly, the training stage where the testing, training and evaluation has been done for the multivariate methods governed by TMVA factory and second is the application stage where the picked methods have been applied by using TMVA Reader to specific regression or classification problems that were trained before.
4.1 Data pre-processing, training and testing

The datasets need to be provided to the factories that are required for training and testing. The classifiers are trained and tested using datasets in the form of ROOT that contain known event classification. In this scenario, the factory divides the tree into two sections: one for training and another for testing. When provided in the data collection, individual event weights can be assigned. The connection between user and the TMVA analysis processes is organized by the factory class, which includes pre-analysis and pre-processing of the data needs to be trained to determine basic features of the variables used as an input to the classifiers. Once the training has completed, each classifier stores its transformation in its weight file. The weight file includes all of the information needed for the afterwards application of the trained classifier. The transformation is taken from its weight file for testing and application of the classifier. The classifiers are tested and evaluated once they have been trained in order to determine their performance.

In machine learning, a classifier is an algorithm which automatically sorts or classifies data within one or more ‘classes’. Email classifier is the most common example, which examines emails and filters them according to whether they are spam or not. A classifier is an algorithm - the principles that machines use to categorise data. In contrast, the classification model is the final outcome of a classifier’s machine learning. The classifier is used to train the model, and the model is then used to classify your data. The signal is overlaid by background processes with the same signature in the high energy. Widely used techniques of classification into signal and background events meet their limits, whenever the signal is very weak, or when a piece of information about whether an event is signal or background is hidden and not very well established correlations between the observables. The Multivariate Analysis Toolkit TMVA allows you to efficiently extract the information from the observables.

In TMVA, classification is performed by generating a classifier output from the input variables (observables) in which signal events have values equivalent to 1 and background values approaches to 0. The mapping to the background class and signal is carried out by labelling all events as signal with a classifier output \( y > y_{\text{cut}} \) and remaining are classified as background. Purity of the signal efficiency \( \epsilon_{\text{sig,eff}} \) and background rejection \((1 - \epsilon_{\text{bkg,eff}})\) are measured for each cut value \( y_{\text{cut}} \) [29]. The light-weight Reader object is used to apply the trained classifiers to the chosen events from a data sample with signal and background composition. It could be involved in any C++ executable, ROOT files, or python analysis program and reads and interprets the weight files of the specified classifier [30].

| MVA classifier | AUC (with cut) | AUC (without cut) |
|---------------|---------------|-------------------|
| MLP           | 0.877         | 0.865             |
| BDT           | 0.875         | 0.864             |
| Likelihood    | 0.829         | 0.793             |
Fig. 10 a and b represents BDT Signal Significance with and without applying cuts respectively.

Fig. 11 a and b represents MLP Signal Significance with and without applying cuts respectively.

Fig. 12 a and b represents likelihood Signal Significance with and without applying cuts respectively.

4.2 Methodology

The boosting strategy has proven to be a very successful method of enhancing performance not only for decision trees, but also for any type of classifier. In high-energy physics, boosting, and particularly boosted decision trees, has grown increasingly popular [31]. The D0 experiment uses boosted decision trees for the first time to conduct a search, and it resulted in the first indication (and then observation) of single top quark production [32]. A decision tree takes a collection of input features and divides input data depending on those features in a recursive manner. They only evaluate signal and background trees because they only consider trees...
Table 2 Signal Significance with and without cut value

| MVA Classifier | Signal Sig. (with cut) | Signal Sig. (without cut) |
|----------------|------------------------|---------------------------|
| MLP            | 26.07                  | 25.99                     |
| BDT            | 26.05                  | 25.93                     |
| Likelihood     | 25.31                  | 25.16                     |

Fig. 13 a and b represents BDT response with and without applying cuts respectively

Table 3 Comparison of classifier’s response with and without cut value

| MVA          | Signal (background) (with cut) | Signal (background) (without cut) |
|--------------|--------------------------------|----------------------------------|
| Classifier   |                               |                                  |
| MLP          | 0.73 (0.223)                  | 0.086 (0.175)                   |
| BDT          | 0.059 (0.053)                 | 0 (0.001)                       |
| Likelihood   | 0.015 (0.018)                 | 0.213 (0.902)                   |

having two classifications. A decision tree’s initial node is the root node. Until a stopping condition is reached, each node can be split into two daughters or we can say branches [31]. The phase space is divided into numerous areas that are finally designated as signal or background based on the number of training events that wind up in the final leaf node. In a regression tree, each output node represents a different value of the target variable. The trees are constructed from the same training ensemble by reweighting events, and then combined into a single classifier (regressor) that is average of a individual decisions (regression) trees. Boosting stabilizes the behavior of decision trees regarding variations in the training sample and can significantly improve performance when compared to a single tree [30]. Consider the problem of distinguishing between signal and background activities using a set of particle identification (PID) variables. The decision tree is a set of binary data divisions. The tree is trained using a collection of well-known training events. The findings are measured using a distinct set of known testing events. Assume that all data is stored on a single node. The best PID variable to classify the data into signal and background is discovered. There are two nodes after that. The operation is continued on these new nodes until a particular number of end nodes (referred to as “leaves”) is produced, or until all of the leaves are pure, or until a node has fewer events [33]. There are many widely used criteria for determining the appropriate PID variable and the best location for splitting a node. We used gini criterion for boosted decision trees. Assume that $W_j$ is the weight of event $j$. The term purity $P$ of node can be defined as, the ratio of the weight of signal events $W_s$ on the node and total weight ($W_s + W_b$) of events on that node. i.e

$$P = \frac{W_s}{W_s + W_b}$$  

(3)

gini is defined for a specific node as:

$$gini = P \left(1 - P \sum_j W_j \right)$$  

(4)
where the value of \( gini \) is 0 for both \( P = 0 \) and 1. The split that minimises \( gini_{left} + gini_{right} \) is preferred as the best. The next node to split is determined by identifying which node’s splitting maximises \( gini \) change. A decision tree is created in this manner. Signal leaves have value of \( P \) greater than or equal to 0.5, while background leaves have value \( P \) less than 0.5. Decision trees are significant, but they are also unstable. A minor change in the training data can lead to a large shift in the tree. Boosting is used to solve this problem. The AdaBoost is a widely used boosting classifier. An exponential loss function is used in adaptive boost. The weights of the training events that were misclassified are raised (boosted) for boosting, and a new tree is built. Then for the new tree, the procedure is repeated. Many trees are formed in this manner.

If the event occurs on a signal leaf, then the score \( T_n \) from the \( n \)th individual tree is +1, and if the event occurs on a background leaf, then score is -1. The final score \( T \) is calculated as a weighted sum of the individual leaf scores \( T_n \).

\[
T = \sum_{n=1}^{N_{tree}} \alpha_n T_n
\]

(5)

where,

\[
\alpha_n = \beta \log \frac{1 - err_n}{err_n}
\]

(6)
and here \( err_m \) represents the error function (misclassification rate) for the \( n \)th tree, given by

\[
err_m = \frac{\text{weight misclassified}}{\text{total weight for tree } m}
\]

and \( \beta \) is constant its typically used values are 1 or 1/2. In a decision tree, the dividing criteria for every node are defined by the process of training, building, or growing the tree. The training process begins with root node, which determines an initial dividing criterion for the entire training sample. When the training events are separated, they are divided into two subsets, each of which is subjected to the same algorithm to decide the next dividing iteration. We need to repeat this process till the entire tree is constructed. At every node, the splitting is determined by calculating the variable and related cut value that offers the best signal-to-background separation. When the least number of events set in the BDT configuration is achieved, the node splitting stops (option nEventsMin). On the basis of how many events are present, the leaf nodes are categorised as signal and background nodes. In theory, the splitting might go on till every leaf node only includes signal or background events, indicating that complete classification is possible. Such a decision tree, on the other hand, would be severely over trained. A decision tree must have been pruned to prevent over training. Although a neural network with \( n \) number of neurons can theoretically have \( n^2 \) directional links, the complexity can be minimised by layering the neurons and permitting only direct links from one layer to the next. The multi-layer perceptron is the name given to this type of neural network; in TMVA all neural network implementation are of this type. The input layer of a multilayer perceptron is the first layer, the output layer is the last, and the rest are hidden layers. In a classification problem with \( n_{\text{var}} \) input variables, the input layer contains \( n_{\text{var}} \) neurons that keep the input values, while the output layer contains one neuron that keeps the output variable.

In Multilayer Perceptron (MLP), the data is transferred from input layer to output layer in a forward direction, similar to a feed forward neural network. The neurons of the MLP are trained using the back propagation learning algorithm (a technique for optimising the weights of an MLP that use the outputs as inputs). We used default TMVA configuration of MLP having 600 number of epoch (NCyles), Hidden Layers = \( N + 5 \) with \( n_{\text{var}} - 1 \) neurons and \( \text{tanh} \) activation function.

The maximum likelihood method involves in creating a model out of probability density functions (PDFs) that regenerates the signal and background input variables. The likelihood of an event that is of signal type is calculated by multiplying the signal probability densities of all input variables, that are considered to be independent (Naive Bayes), and normalising by the total of
Fig. 17 Input variables distribution for signals and backgrounds after application of the Gaussian Transformation

Fig. 18 Receiver Operating Characteristic (ROC) curves from TMVA for several multivariate analysis. This represents signals efficiency ($\epsilon_{\text{sig, eff}}$) and background rejection ($1 - \epsilon_{\text{sig, background}}$)

the signal and background likelihoods for a given event. This PDE method is also known as the “naive Bayes estimator” because correlations between variables are neglected. The Naive Bayes is a probability-based algorithm used in machine learning technique that may be applied to a wide range of classification tasks. For event $j$ the likelihood ratio $y_L(j)$ is defined as follows:

$$y_L(j) = \frac{L_s(j)}{L_s(j) + L_B(j)}$$

(8)

where $[L_{S,B}(j) = \prod_{i}^{n_{\text{var}}} p_{S,B,i}(z_i(j))]$ the background and signal PDFs for the ith input variable $z_i$ are $p_{B,i}$ and $p_{S,i}$ respectively. The PDFs have been normalised as,
Fig. 19 Signal efficiency, background efficiency, signal purity, signal efficiency multiplied with purity and significance \( S \sqrt{S+B} \) as function of the BDT cut value.

Fig. 20 The MLP and Likelihood cut efficiencies as a function of applies cut values respectively.

\[
\int_{-\infty}^{+\infty} p_{S(B),i}(z_i)dz_i = 1 \quad \forall i
\]

In absence of model inaccuracies (that is correlations among input variables that were not eliminated by the de-correlation method, or a probability density model that was inaccurate), it can be demonstrated that, For the given set of input variables, the ratio (8) gives the perfect signal obtained from background splitting. As an outcome of TMVA, the likelihood output can be transformed using an inverse sigmoid function that zooms in with the peaks.

4.3 DeepTop

Jets from the LHC can be studied using machine learning based on neural networks. LargeRjet top tagging provides a clear structure for establishing a DeepTop methodology. Based on a large neural network using the Minkowski metric and Lorentz vectors, this new tagger for hadronically decaying top quarks is quite effective, known as Deep Top.

For the training purpose, two types of trees are involved: the signal tree and the background tree. The signal tree consists of all the features describing the signal events. The background tree consists of exactly the same features passed through the same preselection cuts as the signal. The distributions for these input variables for DeepTop datasets Sect. 3 are shown in Fig. 6 where signal significance is calculated, Fig. 7 shows the input distributions of kinematical variables and the network architecture used for analysis is shown in Fig. 8.
4.3.1 Results

The curve of background rejection against signal efficiency provides a reasonable estimate of a classifier’s performance. A high background rejection and a high signal efficiency are required to achieve the best separation of signal and background candidates. A classifier’s performance is measured by the area under the background rejection vs signal efficiency curve, which is also used to rank the classifier. The bigger the area, the better a classifier’s predicted separation power. The order of the classifiers in Figs. 8 and 9a corresponds to their rank with applying cut and without applying cut respectively, with the best performing classifier appearing first (Table 1).

Each classifier is trained for 50000 number of signal and background events. To check the visibility of the signal process, signal significance is calculated for each classifier. Signal significance is calculated by using a cut on the jet $P_T$ with large cone radius $\Delta R = 0.8$

as follows:

$P_T^{jet} > 20$ GeV

Figure 10a and b shows the comparison of signal significance for BDT classifier with and without applying cut value respectively. It is clear that the signal significance increases by applying cut on jet $P_T$.

Similarly signal significance is calculated for MLP and Likelihood classifiers with and without applying cut value on jet $P_T$, as shown in Figs. 11a to 12b.

The comparison of signal significance for each classifier is shown in Table 2:

![Normalized BDT response without cuts](image1)

(a) Normalized BDT response without cuts

![MLP response](image2)

(b) SMLP Response.

![Signal and Background for Likelihood](image3)

(c) Signal and Background for Likelihood.

Fig. 21 a–c Represents BDT, MLP and likelihood classifier responses for over-training check.

![Image](image4)
Classifier’s response gives signal to background probability by using Kolmogorov-Smirnov (KS-Test). For the BDT classifier signal (background) probability is almost zero without applying cut, but it increases if a cut is applied on jet $P_T$ as shown in Fig. 13a and b respectively. The over-training decreases and the performance of the classifier improve by applying cut (Table 3).

Similar behavior can be observe in the MLP classifier as shown in Fig. 14a and b. But for the likelihood the classifier performance decreases and over-training increases by applying cuts which can be seen in Fig. 15a and b.

For the same sample of data, three classifiers i.e. BDT, MLP and Likelihood are trained. Background rejection, signal efficiency, signal significance and classifier response is computed for each classifier with and without $P_T^{jet}$ cut. The comparison of computed results shows that among all three classifiers Multi-layer Perception (MLP) has the best output for all the calculated parameters.

4.4 Multivariate analysis for single top quark

Our second kind of datasets belong to single top quark production and the relevant background processes for TMVA has been created through Calchep 3.1.1 by using ‘Standard Model’ with the 7 TeV energy of each colliding particle and resulted output in the ‘.lhe’ format which is then converted into root by Delphes 3.1.2 using the ATLAS detector card. The data consists of 100 k events of T-channel as signal tree and background Trees comprising of S-channel excluding W’s, s-channel excluding H’s (using 2HDM-II-BSM model) and $t\bar{t}$. Each tree consists of 100 k events. The input variables we used for training are transverse momentum ($P_T$) of jet(Jet,$P_T$), angle $\phi$ and $\eta$ (pseudorapidity) of jet and last one is mass of jet (Fig. 16).

The input variables we used for training are jet (Jet,$P_T$), angle $\phi$ and $\eta$ and last one is mass of jet as shown in Fig. 11b. We observe the peak in $P_T$ at almost 50 GeV for signals and 200 GeV for backgrounds with selection cut of $P_T$ greater than 20 GeV. The backgrounds tends to give larger $P_T$ than the signals. 177,396 signal events passed and 577,556 number of events passed of backgrounds, out of which 5000 events from each are used for training and remaining for testing. After applying the gaussian distribution, the input variables distribution for the signal and background can be seen in Fig. 17.

5 Results and discussion

There are several multivariate analysis methods but currently we choose three of them. For each technique, we use the default TMVA setting. We choose 5000 events for each to train the signals and backgrounds. To enhance the performance of the trained BDT, we used 850 trees, with node splitting permitted only when the number of events in a node exceeds 2.5% of the total number of events in the training sample. The maximum tree depth has been set at three. Adaptive Boost is used for training, with a learning rate of $\beta = 0.5$. At the end of each boosting iteration, half of the training sample is chosen at random. The separation index of the parent node and the sum of the indices of the two daughter nodes are compared to optimise the cut value on the variable in a node. For the separation index, we use the Gini Index. Finally, the variable’s entire range is evenly graded into 20 cells.

Figure 18 shows the receiver operating characteristic (ROC) curves for the selected methods. ROC curves indicates the quality of the training for the booked multivariate methods. The ROC curve plots the rate of false positive (background rejection) against the rate of true positive (signal efficiency). The true positive rate (sensitivity) refers to the number of true signal events that have been identified correctly. The false positive rate (1-specificity), on the other hand, is a measure of how many actual background events have been correctly identified. The larger the area under the curve, better the performance. It has been observed that the multilayer perceptrons (MLP) shows better efficiency and background rejection as compared to other methods. BDT performance is in between MLP and likelihood methods. Area Under the Curve (AUC) of ROC is 0.761, 0.773 and 0.734 for BDT, MLP and Likelihood methods respectively.

For a given number of signal and background events, Fig. 19 depict the significance $\sqrt{S + B}$ as a function of the BDT cut value. This is the parameter that most properly indicates the BDT’s training performance. The significance output is used to evaluate a method’s performance, with a higher significance indicating a better classifier. However, statistics must be considered, which means that the signal efficiency that corresponds to the ideal cut value must be considered. A lot of signal gets filtered away if this number is really low. If the real data has low statistics, the quality of the cut cannot be guaranteed because the output distribution differs from that of the training sample.

Figure 20a and b shows the SMLP and Likelihood cut efficiencies as a function of applies cut values respectively. The significance is, 23.16 for BDT, 23.18 for MLP and 22.92 for Likelihood. It has been observed that MLP have higher significance as compared to the BDT and Likelihood method. Kolmogorov-Smirnov test statistics is used in order to determine whether the overtraining occurred. The rule of thumb is used in which we want $d_{KS} \geq 0.01$. We have considered two different cases for determining the overtraining, first one is without cuts and second is with cuts.

Figure 21a shows the normalized BDT response for the samples of test (histograms) and train (dots along with error bars). BDT responses for both signals (blue) and background (red) samples, spreads over the positive and negative BDT response zones respectively. The signals and backgrounds are partially separated from each other and are superimposed. Figure 21b shows the response of SMLP of signal and background combined together. The fact that the KS test statistics are lesser than 0.01 shows that
there is overtraining for backgrounds. It can be seen from Fig. 21c that the signal and background histograms almost overlap each other. Table 4 lists the \( K - S \) probability for signal and background from various methods used for current study.

### 6 Conclusion

The goal of this study is to see how well different Multivariate methods can classify signals and backgrounds based on particles characterized by input variables (invariant masses, transverse momenta, pseudorapidities, and angular distances between them). Three different classifiers are trained to find which one is the most effective in distinguishing signal from backgrounds. The Multivariate analysis performed on datasets of single top quark we conclude that MLP outperforms the other two classifiers (BDT and LH) for the classification of signal and backgrounds.

We also train a deep neural network (DeepTopLoLa tagger) to jet substructure which consists of two fully connected layers; Combination layer and Lorentz layer that translates the 4-vectors into quantities more directly related to physical observables. The Lorentz group and Minkowski metric can be used to distinguish the signal from the backgrounds. With our limited computing resources we could figure out the performance of the DeepTop model at the level of ROC curves. A study will be carried out in near future where we can apply the Top tagging in fat jets offers a well-defined framework to establish our DeepTop approach and compare its performance to QCD-based top taggers e.g., HepTopTaggers. This approach will be effective in those processes in which heavy resonance production occurs and it may decays into multi top quark events preferably.

The value of significance of MLP (23.18) is higher than the BDT (23.16) and likelihood (22.92) methods. It concludes that MLP performance is slightly better than BDT and also their ROC curves are also close to each other. Moreover, the signal purity \( P \) of MLP is splitted earlier and distinguishes the signals \( (P \geq 0.5) \) from the backgrounds \( (P < 0.5) \) efficiently. The signal purity response of MLP is much efficient as compared to other two classifiers.

From the K-S test probability we see that MLP classifier is not overtrained and also after the application of cuts, the overtraining of the signals in MLP classifier decreases, so the MLP outperforms the other two classifiers (BDT and LH) for the classification of signal and backgrounds. It can be concluded that MLP method is to be preferred in our scenario. Three different classifiers are trained to find which one is the most effective in distinguishing signal from backgrounds.

### Acknowledgements
We gratefully acknowledge support from the Simons Foundation and member institutions of Cornell University. The current submitted version of manuscript is available on arXiv pre-prints home page https://arxiv.org/abs/2203.12871 with arXiv ID:2203.12871. The corresponding author specially would like to thanks Prof. Jamil Muhammad from Sang-Ho College, and Department of Physics, Konkuk University, Seoul 05029, South Korea. He indeed contributed substantially in the couple of revision in replying the reviewer comments and helps in improving the format of the manuscript.

### Data Availability Statement
The data samples that support the findings of this study are available from the corresponding author, upon reasonable request.

### References

1. F. Abe et al., Phys. Rev. Lett. **74**, 2626–2631 (1995). https://doi.org/10.1103/PhysRevLett.74.2626. arXiv:hep-ex/9503002 [hep-ex]
2. S. Abachi et al., Phys. Rev. Lett. **74**, 2632–2637 (1995). https://doi.org/10.1103/PhysRevLett.74.2632. arXiv:hep-ex/9503003 [hep-ex]
3. V.M. Abazov et al., Phys. Rev. Lett. **103**, 092001 (2009). https://doi.org/10.1103/PhysRevLett.103.092001. arXiv:0903.0850 [hep-ex]
4. T. Aaltonen et al., Phys. Rev. Lett. **103**, 092002 (2009). https://doi.org/10.1103/PhysRevLett.103.092002. arXiv:0903.0885 [hep-ex]
5. S. Chatrchyan et al., JINST **3**, S08004 (2008). https://doi.org/10.1088/1748-0221/3/08/S08004
6. G. Aad et al., JINST **3**, S08003 (2008). https://doi.org/10.1088/1748-0221/3/08/S08003
7. I. Ahmed, A. Zada, M. Waqas, M.U. Ashraf, Application of deep learning in top pair and single top quark production at the LHC, arXiv:2203.12871 [hep-ph]
8. S. Chatrchyan et al., Phys. Rev. Lett. **107**, 091802 (2011). https://doi.org/10.1103/PhysRevLett.107.091802. arXiv:1106.3025 [hep-ex]
9. G. Aad et al., Phys. Lett. B **717**, 330–350 (2012). https://doi.org/10.1016/j.physletb.2012.09.031. arXiv:1205.3130 [hep-ex]
10. D. Bourilkov, Int. J. Mod. Phys. A **34**(35), 1930019 (2020). https://doi.org/10.1142/S0217751X19300199. arXiv:1912.08245 [physics.data-an]
11. D. Dermant et al., EPJ Web Conf. **214**, 06011 (2019). https://doi.org/10.1051/epjconf/201921406011
12. D. Guest, K. Cranmer, D. Whiteson, Ann. Rev. Nucl. Part. Sci. **68**, 161–181 (2018). https://doi.org/10.1146/annurev-nucl-101917-021019. arXiv:1806.11484 [hep-ex]
13. D. Guest, J. Collado, P. Baldi, S.C. Hsu, G. Urban, D. Whiteson, Phys. Rev. D **94**(11), 112002 (2016). https://doi.org/10.1103/PhysRevD.94.112002. arXiv:1607.08633 [hep-ex]
14. P. Baldi, K. Bauer, C. Eng, P. Sadowski, D. Whiteson, Phys. Rev. D 93(9), 094034 (2016). https://doi.org/10.1103/PhysRevD.93.094034. arXiv:1603.09349 [hep-ex]
15. A.A. Pol et al., EPJ Web Conf. 214, 06008 (2019). https://doi.org/10.1051/epjconf/201921406008
16. J. Caudron, https://cds.cern.ch/record/2642126
17. G. Kasieczka, T. Plehn, A. Butter, K. Cranmer, D. Debnath, B.M. Dillon, M. Fairbairn, D.A. Faroughy, W. Fedorko, C. Gay et al., SciPost Phys. 7, 014 (2019). https://doi.org/10.21468/SciPostPhys.7.1.014. arXiv:1902.09914 [hep-ph]
18. T. Heimel, G. Kasieczka, T. Plehn, J.M. Thompson, SciPost Phys. 6(3), 030 (2019). https://doi.org/10.21468/SciPostPhys.6.3.030. arXiv:1808.08979 [hep-ph]
19. A. Butter, G. Kasieczka, T. Plehn, M. Russell, SciPost Phys. 5(3), 028 (2018). https://doi.org/10.21468/SciPostPhys.5.3.028. arXiv:1707.08966 [hep-ph]
20. T. Sjöstrand, S. Ask, J.R. Christiansen, R. Corke, N. Desai, P. Ilten, S. Mrenna, S. Prestel, C.O. Rasmussen, P.Z. Skands, Comput. Phys. Commun. 191, 159–177 (2015). https://doi.org/10.1016/j.cpc.2015.01.024. arXiv:1410.3012 [hep-ph]
21. J. de Favereau et al., JHEP 02, 057 (2014). https://doi.org/10.1007/JHEP02(2014)057. arXiv:1307.6346 [hep-ex]
22. M. Cacciari, G.P. Salam, G. Soyez, Eur. Phys. J. C 72, 1896 (2012). https://doi.org/10.1140/epjc/s10052-012-1896-2. arXiv:1111.6097 [hep-ph]
23. M. Cacciari, G.P. Salam, Phys. Lett. B 641, 57–61 (2006). https://doi.org/10.1016/j.physletb.2006.08.037. arXiv:hep-ph/0512210 [hep-ph]
24. A. Butter, G. Kasieczka, T. Plehn, M. Russell, SciPost Phys. 5, 028 (2018).
25. F. Chollet, https://github.com/fchollet/keras (2015)
26. Theano Development Team, arXiv:1605.02688 [cs.SC]
27. D.P. Kingma, J. Ba, Adam: a method for stochastic optimization (2014) https://doi.org/10.48550/arXiv.1412.6980[cS.LG]
28. A. Hocker, P. Speckmayer, J. Stelzer, J. Therhaag, E. von Toerne, H. Voss, M. Backes, T. Carli, O. Cohen, A. Christov, et al. TMVA-toolkit for multivariate data analysis arXiv:physics/0703039 [physics.data-an]
29. P. Speckmayer, A. Hocker, J. Stelzer, H. Voss, J. Phys. Conf. Ser. 219, 032057 (2010). https://doi.org/10.1088/1742-6596/219/3/032057
30. H. Voss, A. Hocker, J. Stelzer, F. Tegenfeldt, PoS ACAT (2007). https://doi.org/10.22323/1.050.0040
31. Y. Coadou, EPJ Web Conf. 55, 02004 (2013). https://doi.org/10.1051/epjconf/20135502004
32. V.M. Abazov et al., Phys. Rev. Lett. 98, 181802 (2007). https://doi.org/10.1103/PhysRevLett.98.181802. arXiv:hep-ex/0612052 [hep-ex]
33. B.P. Roe, H. J. Yang, J. Zhu, Boosted decision trees, a powerful event classifier (2006)

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.