ETP-Bachelor-KA/2023-13

Karlsruhe Institute of Technology

# Generating Explanations for Graph Neural Networks for tt+X events at the CMS Experiment

Bachelor Thesis

Paul Oßwald

At the Department of Physics
Institute of Experimental Particle Physics

| | |
|---|---|
| Reviewer: | Prof. Dr. Ulrich Husemann |
| Second reviewer: | Dr. Michael Wassmer |
| Advisor: | Emanuel Pfeffer |

Karlsruhe, 08. November 2023

This thesis has been accepted by the first reviewer of the bachelor thesis.

**Karlsruhe, 08. November 2023**

.....

(**Prof. Dr. Ulrich Husemann**)

I declare that I have developed and written the enclosed thesis completely by myself, and have not used sources or means without declaration in the text.

**Paul Oßwald, 08. November 2023**

. . . . . .

(**Paul Oßwald**)

# Contents

# 1 Introduction

In 2012 the Higgs boson was discovered at the Large Hadron Collider (LHC) at CERN in Switzerland [1]. This was the last particle predicted by the Standard Model to be discovered. The Higgs boson is of special importance, because via the Higgs mechanism massive particles can be explained. The Standard Model classifies all known elementary particles and describes the electromagnetic, weak and strong interactions. While the Standard Model is the most successful theory in particle physics, there are some phenomena that the Standard Model is unable to explain. It does not include the theory of relativity and is thus unable to explain gravitational force. This leads to the search of physics beyond the Standard Model.

The top quark was first discovered at the Tevatron collider at Fermilab [2, 3], it was the last particle of the third generation of matter to be discovered. The top quark-antiquark pair production with an associated particle $X$ ($t\bar{t} + X$) is of special interest, especially for event classes that have similar final states ($t\bar{t} + X$, $X \to q\bar{q}$). Because of this, they are not distinguishable by their final state objects in data measured by a particle detector, for example, for the $ttH$ events with decays $H \to b\bar{b}$ which need to be distinguished from $ttZ$ ($Z \to b\bar{b}$) events and $ttB$ ($g \to b\bar{b}$) events. The correct identification of $ttH$ events is especially important because of the top-Higgs Yukawa coupling. An accurate event classification is important to test the understanding of the Standard Model and possibly uncover new physics.

In the last years, efforts have been made to use various machine learning models in the multiclass classification of the $t\bar{t} + X$ events [4–7]. Especially Graph Neural Networks (GNN) architectures have received attention, due to the data structure lending itself to the representation as a multi-relational graph. Naturally, when using a GNN to classify an event, it is of interest why it makes a particular prediction. GNNs however, due to the complex structure of graph data and black-box nature, are hard to interpret.

This thesis focuses on examining a GNN trained for this classification task using the GNNExplainer method to better understand the underlying decisions taken by the GNN when using different $t\bar{t} + X$ events as input.

This thesis is structured into 7 chapters. This introduction is chapter 1. Chapter 2 introduces the theoretical foundations of the physics processes examined in this thesis. Chapter 3 deals with the LHC and CMS Experiment, introducing the basic functionality and defining the kinematic variables. The GNN framework and the basics of Neural Networks

are introduced in chapter 4. The GNNExplainer method is introduced in chapter 5. Chapter 6 examines the GNN using the GNNExplainer introduced in the previous chapter. Finally, a conclusion is drawn in chapter 7.

# 2 High Energy Physics

This chapter attempts to provide a basic understanding of the underlying physics theories by taking a look at the Standard Model in section 2.1 and the $t\bar{t} + X$ events in section 2.2, which are the main focus of this thesis.

## 2.1 Standard Model

The Standard Model of particle physics forms the basis of the understanding of matter and interactions between matter, except for gravitation. The following summary is based on [8]. A systematic presentation of the individual particles can be seen in Fig. 2.1. Elementary particles can be divided in two categories, fermions with half-integer spin and bosons with integer spin value. Fermions form all forms of known matter, they can be further distinguished into leptons and quarks. Fermions can be sorted into three generations of matter. Only the first generation forms stable states, with the other generations eventually decaying into the former. There are six lepton-flavours, electron $e$ (I. generation), muon $\mu$ (II. generation) and tau $\tau$ (III. generation) which carry an electric charge of negative one in units of the elementary charge and distinguish them from the associated neutrinos. Electron-neutrino $\nu_e$ (I. generation), muon-neutrino $\nu_\mu$ (II. generation) and tau-neutrino $\nu_\tau$ (III. generation) are electrically neutral. Similarly, there are six quark-flavours, up $u$ (I. generation), charm $c$ (II. generation) and top $t$ (III. generation) with an electric charge of $+2/3$, as well as down $d$ (I. generation), strange $s$ (II. generation) and bottom $b$ (III. generation) with an electric charge of $-1/3$. Quarks in contrast to leptons possess an additional degree of freedom which is the colour charge red, blue and green. Due to colour confinement quarks cannot exist in isolation, only bound in hadrons where the colour charges neutralise each other. Furthermore, for every fermion there exists an antiparticle, which carries the same mass as their matter counterpart but carries an opposite electric charge and magnetic moment. Gauge bosons or vector bosons give rise to interactions between leptons and quarks, they are spin 1 particles. The photon $\gamma$ is the force carrier of the electromagnetic interaction. The $W^\pm$ bosons and $Z$ bosons govern the weak interaction. Furthermore, there exist eight gluons $g$, which are exchange particles for the strong interaction. There is only one known scalar boson with spin 0, the Higgs boson, which gives mass to other particles via the Higgs mechanism.

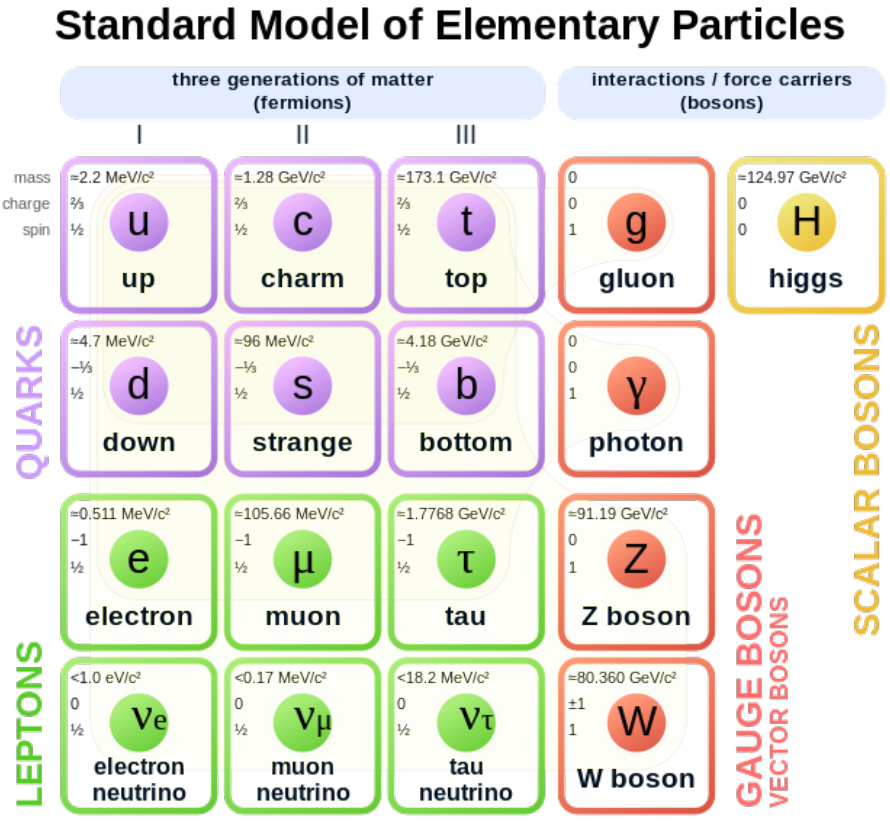



Figure 2.1: Elementary Particles of the Standard Model. Taken from [9].

The Standard Model can currently not explain gravitation, dark matter and dark energy. This is why there is a concerted effort to find physics beyond the Standard Model to explain these phenomena and reconcile the deficiencies of the Standard Model.

## 2.2 $t\bar{t} + X$ events

The top quark $t$ is the heaviest known elementary particle, it is also the only quark that does not form bound states because it is too short-lived. It can only decay through the weak interaction, for a large majority of more than 99% of the cases a $W$ boson and a bottom quark $b$ are the decay products. Less likely decays can result in a charm quark $c$ or a strange quark $s$. The resulting $W$ boson can decay via hadronic decay into a quark-antiquark pair or via leptonic decay into a lepton-neutrino pair [10].

At the LHC top quark-antiquark pairs are mostly generated via gluon-gluon fusion [11], there are three possible decay channels, the di-leptonic decay channel (10.5%) in which both $W$ bosons decay into a lepton and associated neutrino. In the semi-leptonic decay channel (43.8%) one of the $W$ bosons results in a leptonic decay and the other $W$ boson results in a hadronic decay. Lastly, in the hadronic decay (45.7%) both $W$ bosons decay into a quark-antiquark pair [10].

This thesis focuses on the di-leptonic decay channel illustrated in Fig. 2.2. For this channel events are expected to have two leptons, at least three jets and large missing transverse energy. This channel is particularly suitable for event classification because the products of the top decay are leptons and therefore easier to distinguish than for a hadronic or semi-leptonic decay of the top quarks.

When measuring such a top quark-antiquark decay in a particle detector only the final state objects are detected. The problem with measured data is that events with similar final states are not distinguishable from each other. This poses a challenge when trying to classify the final state products and events. Machine learning tasks require a true label in order to train a model. A solution for this problem is to simulate data using knowledge about the underlying physics processes. The data for the di-leptonic decay channel used in this thesis is generated using Monte Carlo event simulation.

To characterise the final state objects of the decays, they are sorted into seven categories. The $b$-jet stemming from the top decay into a $W^-$ boson and bottom quark is sorted into the category "*topb*". The $W^-$ boson decays into a lepton $l^-$ (electron $e^-$, muon $\mu^-$) and the corresponding antineutrino $\overline{\nu}_l$. Since taus $\tau$ can result in a hadronic decay, they are not considered. The lepton $l^-$ of the top quark decay is sorted into the "*toplep*" category. The antineutrino $\overline{\nu}_l$ cannot be directly detected in the CMS detector, only characterised by the missing transverse energy, which is denoted in the "*met*" category. Analogously the $b$-jet of the antitop $W^+$ decay can be sorted into the "*antitop*" category. While the antilepton $l^+$ of the $W^+$ boson decay is sorted into its own category "*antitoplep*" the same cannot be done for the corresponding neutrino $\nu_l$, which is characterised by its missing transverse energy. The missing transverse energy cannot be assigned to individual particles, a large number of neutrinos results in a large transverse energy which cannot be distinguished further.

The additional jets are produced through a coupling of the top quark-antiquark pair to a particle $X$ which can be a Higgs-boson $H$, $Z$ boson or a gluon $g$. This results in two hadronic quark $q$ antiquark $\overline{q}$ jets, the final state objects of the additional jet are sorted into the "*add*" category. Additionally, jets that are not identifiable are assigned the category *unknown*.

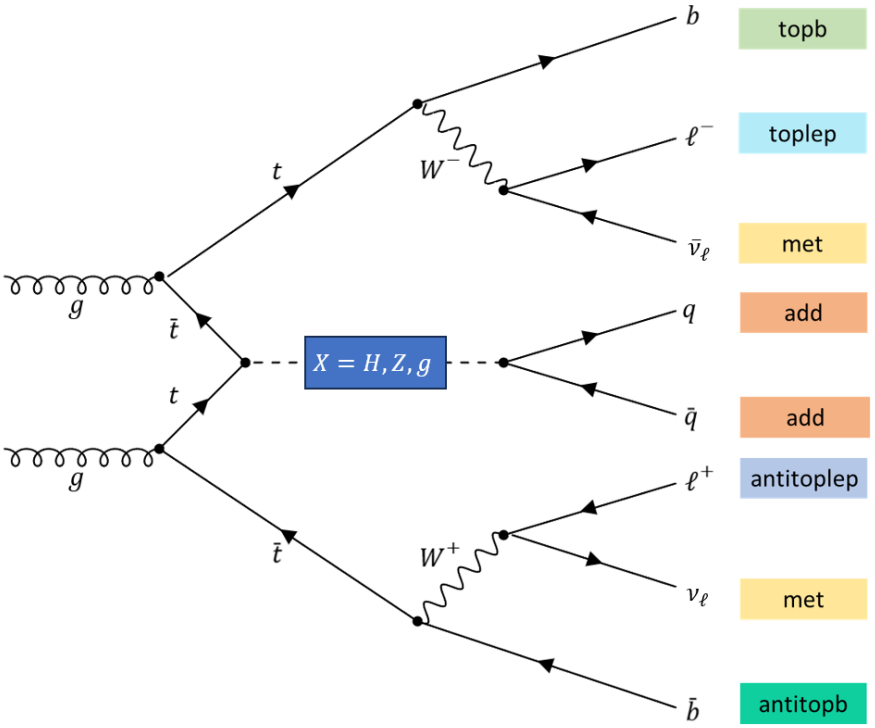



Figure 2.2: Illustration of an examplary di-letonic $t\bar{t} + X$ event in leading order.

The possible event classes are characterised by the coupling of the top quark-antiquark pair with the particle X, which can be a Higgs boson $H$, a $Z$ boson or a gluon $g$. The coupling between the top quark and the Higgs boson is called the top Higgs-Yukawa coupling. The Higgs boson decay can occur through many different processes [12]. The most common decay of the Higgs boson is the decay into a bottom quark-antiquark pair (57.2 %). In this thesis, only the $H \to b\bar{b}$ decay is considered.

Another possible coupling is the top quark-antiquark $Z$ boson coupling, resulting in a decay into a fermion-antifermion pair [13]. This decay can be leptonic $Z \to l\bar{l}$ or hadronic (69.9%) into quark-antiquark pairs $Z \to q\overline{q}$ or more exotic decays. In this thesis, only the hadronic decays are examined and special attention is paid to the $Z \to b\bar{b}$ decay.

Lastly, the additional gluon radiation can result in a light-flavour quark-antiquark pair $g \to q\overline{q}$, a charm quark-antiquark pair $g \to c\overline{c}$ and a bottom quark-antiquark pair $g \to b\bar{b}$ [14].

The flavour tagging of the jets is done using the multi-class flavour tagging algorithm "*DeepJet*" [15], utilising the heavy flavour tagging values "$CvL$" and "$CvB$" defined in equations 2.1 and 2.2. The heavy flavour taggers utilise that charm jets possess characteristics intermediate to $b$-jets and light-flavour jets, this makes it inefficient to define a singular charm tagger. The charm versus light-flavour tagging value "CvL" discriminates charm jets from light-flavour jets in this case $b$-jets act as extreme cases of $c$-jets. Low values of the "CvL" tagger indicate the presence of a light-flavour jet or gluon jet

$$CvL = \frac{P(c)}{P(c) + P(udsg)}. \tag{2.1}$$

Here $P(c)$ denotes the probability of the jet being a charm flavour jet and $P(udsg)$ is the probability of the jet being an up, down, strange or gluon flavour jet. Correspondingly the charm versus bottom tagging value "CvB" discriminates charm jets from $b$-jets in this case light-flavour jets act as extreme cases of $c$-jets

$$CvB = \frac{P(c)}{P(c) + P(b)}. \tag{2.2}$$

Low values of the "CvB" tagger indicate the presence of a $b$-jet. Here $P(b)$ denotes the probability of the jet being a bottom flavour jet. Only by leveraging both discriminators the jet flavour can be determined, a high value for both "$CvL$" and "$CvB$" result indicate a charm-tagged jet.

# 3 LHC and CMS Experiment

This chapter deals with the structure of the Large Hadron Collider (LHC) in section 3.1 based on [16]. A summary of the CMS experiment in section 3.2 is given based on [17]. It establishes the basic functionality and the basic kinematic variables in section 3.3, which are used in interpreting the measured data.

## 3.1 Large Hadron Collider (LHC)

The Large Hadron Collider (LHC) is the most powerful and with a circumference of 27km the largest particle accelerator in existence [16]. It is capable of proton-proton collisions at up to $\sqrt{s} = 14$ TeV center-of-mass energy. The LHC is located at the European Organization for Nuclear Research (CERN) sitting 100 meters underground. Particles are accelerated in pre-accelerators, each accelerator inserts the particle into the next accelerator of higher energy before they are inserted into the LHC rings. In these two rings the particles are accelerated in opposite directions, almost reaching the speed of light. At four interaction points the particle beams collide with each other and measurements are taken by detector systems. There are currently nine experiments at the LHC with the most famous being ALICE, LHCb (Large Hadron Collider beauty), ATLAS (A Toroidal LHC ApparatuS) and the CMS (Compact Muon Solenoid) experiments.

The Run 2 of the LHC, which is the basis of this thesis, observed proton-proton collisions at $\sqrt{s} = 13$ TeV center-of-mass energy.

## 3.2 CMS Detector

The Compact Muon Solenoid Experiment (CMS experiment) is designed to study the decay products of proton-proton beam collisions [17]. In order to achieve this, the CMS-Detector is designed as a multilayer detector, with each layer identifying a different particle or particle characteristic. The detector is installed cylindrically around the interaction point; the different layers of the CMS Experiment are shown in Fig. 3.1.

The inner tracking system is the innermost layer of the detector, in Fig. 3.1 it is referred to as Silicon Tracker because it consists of several layers of silicon pixel detectors and silicon strip trackers. The CMS tracker measures the trajectories and vertices of charged particles.

The electromagnetic calorimeter is a homogeneous crystal calorimeter made up of lead tungstate crystals, the purpose of the calorimeter is to measure the energy of electrons, positrons and photons using the scintillation properties of its material.

The hadron calorimeter is a sampling calorimeter to detect hadron jets of the final jet products, it is also important in determining the missing transverse energy caused by neutrinos and exotic particles. It consists of alternating brass absorbers and plastic scintillator layers.

The CMS tracker, electric and hadronic calorimeters are enclosed by the superconducting solenoid, the magnet is designed to generate a homogeneous 3.8 T field. This large magnetic flux allows the CMS tracker to determine the transverse momentum with the help of the bending radius of the trajectory for charged particles.

The CMS detector uses three gaseous particle detectors interspersed in an iron yoke to identify muons and measure their momentum. The momentum of the muons is measured in a similar fashion to the CMS tracker.

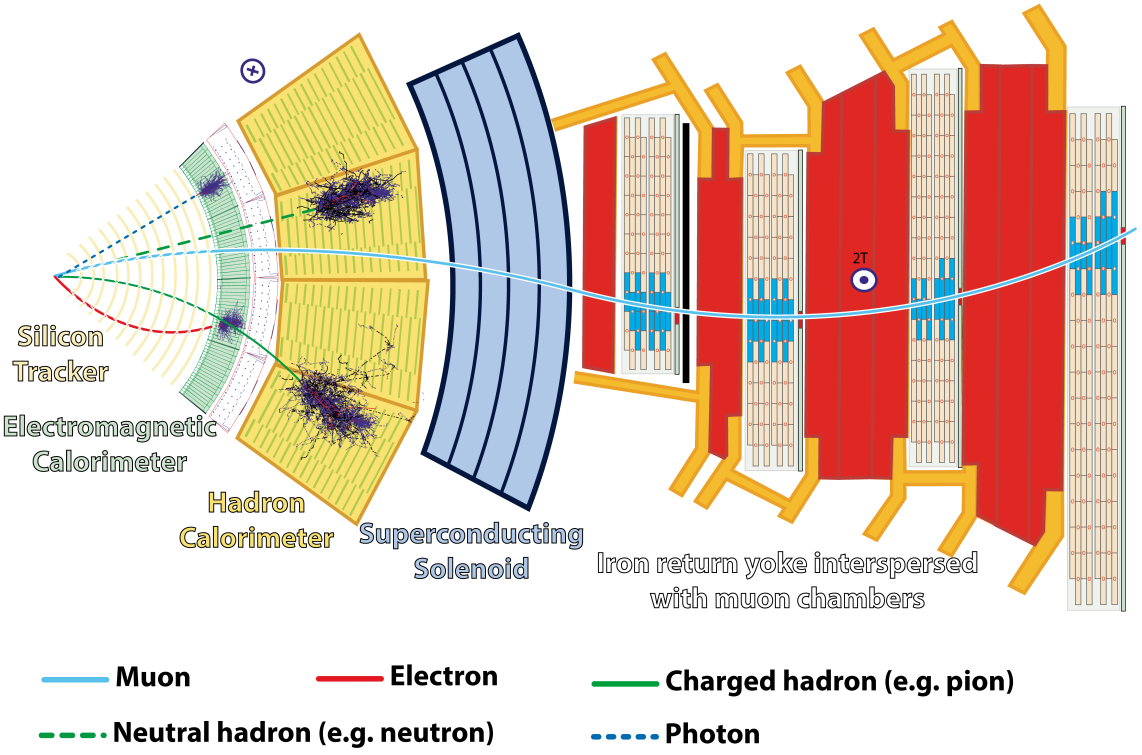



Figure 3.1: Illustration of the CMS Detectror. Taken from Ref. [18].

## 3.3 Basic kinematic quantities

The section of this chapter is based on [17, 19] if not stated otherwise.

The coordinate system used by the CMS detector is centred at the nominal collision point, the designed interaction point of the experiment. The $x$-axis is designed to point radially inwards, the $y$-axis positioned to point vertically upwards and the $z$-axis is oriented along the beam-axis. The CMS detector is cylindrical, so the cylindrical coordinates ($r$, $\phi$, $\theta$) are introduced, and the radial coordinate $r$ lies in the $x, y$-plane. The azimuthal angle is defined as the anticlockwise angle measured from the $x$-axis in the $x, y$-plane. The polar angle $\theta$ is defined as the angle between the $z$-axis and the radial coordinate $r$, measured from the $z$-axis.

The rapidity $y$ is a measure of the relativistic velocity of a particle along the beam axis. Using the previously introduced coordinate systems, the rapidity $y$ can be defined using the momentum $p_z$ along the beam axis and the energy $E$ of the particle

$$y = \frac{1}{2} \ln \frac{E + p_z}{E - p_z}. \tag{3.1}$$

For particles with masses significantly larger than their momentum, the pseudorapidity $\eta$ can be used to describe the direction of an object with respect to the polar angle $\theta$. The object travels at small angles to the beam-axis for large values of the pseudorapidity $\eta$ and perpendicular to the beam for values of zero

$$\eta = -\ln \tan \frac{\theta}{2}. \tag{3.2}$$

The spatial distance is the distance between two objects $i$ and $j$ is defined using the pseudorapidities $\eta_i$ and $\eta_j$ and the azimuthal angles $\phi_i$ and $\phi_j$

$$\Delta R_{ij} = \sqrt{(\eta_i - \eta_j)^2 + (\phi_i - \phi_j)^2}. \tag{3.3}$$

The transverse momentum is the momentum of a particle perpendicular to the beam axis. It can be defined as the components of the transverse momentum in the x-y-plane. The transverse momentum is used because protons are hadrons and not fundamental particles, so their partons and not the proton itself participate in the collisions

$$p_\mathrm{T} = \sqrt{p_x^2 + p_y^2}. \tag{3.4}$$

Alternatively, the transverse momentum of a charged particle can be determined using the electric charge $q$ of a particle, the magnetic flux $B$ of the superconducting solenoid of the CMS detector and the bending radius $r$ of the particle trajectory in the CMS tracker

$$p_\mathrm{T} = qBr. \tag{3.5}$$

Neutrinos and other exotic particles cannot be detected because of their lack of interaction. But the transverse momentum they carry can be reconstructed from the observed transverse momenta due to the conservation of energy and momentum. The missing transverse momentum is often substituted with the missing transverse energy (MET), which can be derived as the sum of the transverse momentum $p_\mathrm{T}$ of all visible particles $i$

$$E_\mathrm{T}^\mathrm{miss} = |-\sum_i p_{\mathrm{T},i}|. \tag{3.6}$$

When interpreting the MET, one has to keep in mind that one cannot distinguish the number of individual objects caused.

The invariant mass of a system of objects can be derived using equation 3.7 where $E$ denotes the energy, $p_\mathrm{T}$ the transverse momentum and $p_z$ momentum along the beam axis of the system

$$M = \sqrt{E^2 - p_\mathrm{T}^2 - p_z^2}. \tag{3.7}$$

# 4 Machine Learning and Graph Neural Networks

Machine learning methods have seen a rise in importance in a wide array of fields, especially in the sciences. This chapter introduces the basics of neural networks in section 4.1. Establishing a general understanding of the topic to introduce Graph Neural Networks in section 4.3, which are specialised in dealing with graph data, introduced in section 4.2.

## 4.1 Basics of Neural Networks

Neural networks are a machine learning method modelled after the information-passing capabilities of natural brain structures, using a so-called artificial neuron to process information. The following short summary of the basic functionality is based on [20]. An artificial neuron $j$ is illustrated in Fig. 4.1. The artificial neuron is a computation unit, which applies trainable weights $w_{1j}, w_{2j}, ..., w_{nj}$ to each input $x_1, x_2, ..., x_n$ the transfer function returns the network input $g_j$. The network input $g_j$ is a linear combination of the weighted inputs and possibly a bias. The activation function $f$ is applied to the network input and a threshold $\theta$ can be applied so that the activation function only passes on the output $y$ if the condition $g_j > \theta$ is fulfilled. The activation function $f$ is used to introduce nonlinearity, which increases the modelling power of the neural network because now the network can derive an output that varies non-linearly with the input. Without a non-linear activation function, a neural network with multiple hidden layers has no more expressive power than a neural network with a single hidden layer.

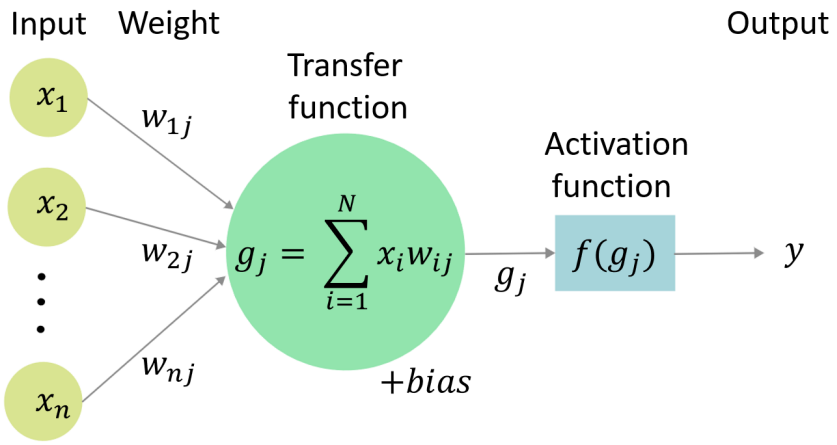



Figure 4.1: Illustration of an artificial neutron. Based on Ref. [20].

Neural networks are formed by artificial neurons, such a neural network is pictured in Fig. 4.2. Each input $x_1, x_2, ..., x_n$ is assigned to an input neuron, which is part of an input layer that transmits them to the hidden layer. The neurons of the hidden layer can take the form of the artificial neuron $j$ shown in Fig. 4.1. The output layer consists of the output neurons $y_1$ and $y_2$ returning the output values of the neural network.

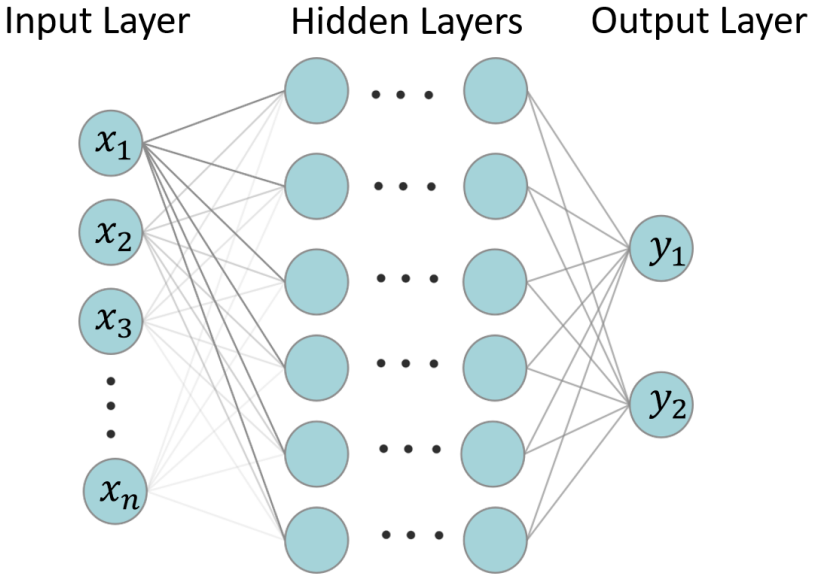



Figure 4.2: Illustration of a simple neural network consisting of an input layer, hidden layer and output layer consisting of artificial neurons. Based on Ref. [20].

Embedding layers or hidden embeddings are a special type of hidden layers that map information from a higher dimension to a lower dimension. In this thesis embeddings are used to allow the model to process graph data. Deep Neural Networks (DNNs) are neural networks that utilise multiple hidden layers. For training purposes, a set of input values and true labels are given; during training the input is propagated through all layers of the neural network, and the output is measured against the true label. The loss function $\mathcal{L}$ measures the loss between output and true label

$$\bar{w} \leftarrow w - \gamma \frac{\partial \mathcal{L}}{\partial w}. \tag{4.1}$$

The weights $w$ are trained via gradient descent using the loss function $\mathcal{L}$, where $\gamma$ denotes the learning rate. The learning rate determines how the weights are adjusted, if the learning rate is too large, the minimum of the loss function can be missed.

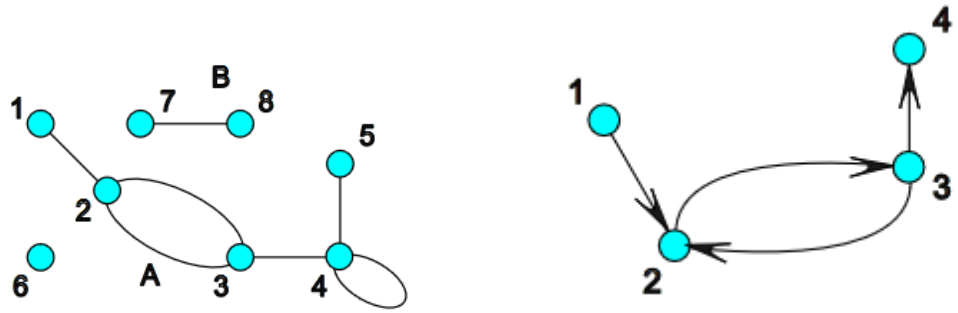
For a more detailed look at neural network methods, Ref. [20] can be consulted.

## 4.2 Graph Theory

If not stated otherwise, the following short summary of graph theory and graph data is based on [21–23].

A graph $G(V, E)$ consists of nodes $V$ and edges $E$ between nodes. An edge from node $u \in V$ to node $v \in V$ is typically denoted as $(u, v) \in E$. If the nodes $u$ and $v$ are connected by an edge $e$ they are called adjacent to each other. The nodes $u$ and $v$ are incident to such an edge $e$. An edge $e = (u, v)$ is called a loop if $u = v \in V$ joining the node with itself, as illustrated by node 4 in Fig. 4.3a.

A node is theoretically not limited in the number of connections it can form; the number of incident edges is denoted by the node degree. An end node has one edge connecting it to the graph and as a result a node degree of 1. For example node 1 or 5 in Fig. 4.3a. The nodes 2 and 3 have a node degree of 3, a loop as seen at node 4 adds two degrees to the node degree. An isolated node that is not connected to the rest of the graph has a node degree of 0, Fig. node 6. Nodes 7 and 8 connected by the edge (7,8) are isolated from the rest of the graph. The entire graph in Fig. 4.3a can be understood as the union of two disconnected graphs A and B (including the disconnected node 6).




(a) Demonstration graph (b) Directed graph

Figure 4.3: Two graphs for demonstration purposes (a) shows an undirected incomplete graph and (b) an directed incomplete graph.

Edges are called undirected if $(u, v) \in E$ and $(v, u \in E)$ are equivalent, in Fig. 4.3b a graph is pictured where that is not the case, a directed graph. In a directed graph, the edges are ordered, meaning that the edges $(u, v)$ and $(v, u)$ are not equivalent. Interpreting this as the flow of information means that information can travel from node 1 to node 2 but not in reverse.

There is no single representation of a graph, as illustrated in Fig. 4.4. Both figures are representations of the same graph, in both cases, their edges can be represented as $[(1, 2), (2, 3), (3, 4), (4, 5), (5, 1)]$.

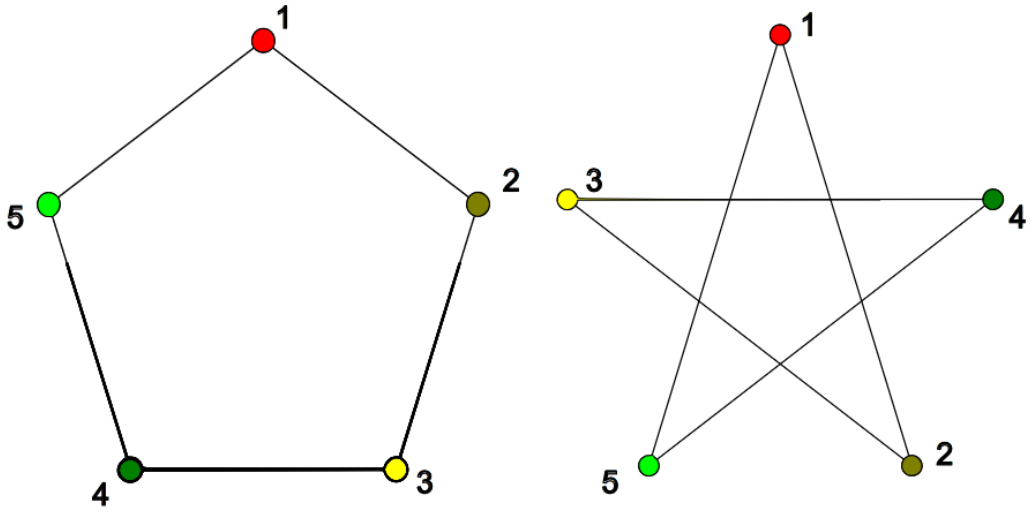



Figure 4.4: Two equally valid representations of the same graph. Figure adapted from Ref. [21].

The adjacency matrix $A$ is a mathematical representation of a graph. For a graph with n-nodes numbered from 1 to n, it is a $n \times n$ matrix which shows if nodes are adjacent. For graphs with multiple/parallel edges between two nodes $u$ and $v$ the entry $a_{uv}$ denotes the number of edges incident to the nodes $u$ and $v$, a graph with weighted edges will have entries between 0 and 1. In this thesis graphs do not have multiple edges, so like in the example eq. 4.2 for graph Fig. 4.4 an entry of 1 means that the two nodes are adjacent to each other. For an entry of zero, the opposite is true. For an undirected graph, the adjacency matrix is always symmetric, this can be seen in the adjacency matrix $A$ in eq. 4.2 for the undirected graph illustrated in Fig. 4.4.

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix} \tag{4.2}$$

A complete graph, as illustrated in Fig. 4.5, is a graph in which all nodes are adjacent to all other nodes, the graphs used in this thesis are complete graphs. All entries of an adjacency matrix $A$ for a complete (undirected) graph are one, except for the diagonal, which contains only zeros. The adjacency matrix of the complete graph Fig. 4.5 is shown in eq. 4.3.

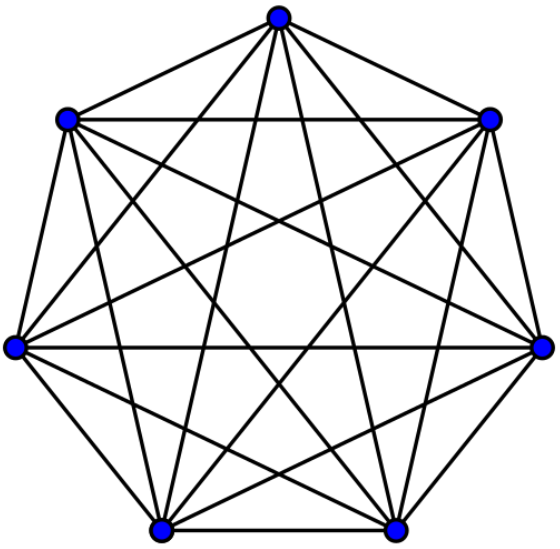


Figure 4.5: Illustration of an complete graph. Taken from Ref. [24]

$$A = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{pmatrix} \tag{4.3}$$

One of the reasons why graphs are so attractive as information tasks is feature information. Features or attributes can be assigned to nodes, edges and graphs themselves. Node-level attributes for $N$ nodes $V$ and $F$ node attributes are represented by the matrix $X$ with the dimensions $dim = |V| \times F = N \times F$

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1F} \\ x_{21} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ x_{N1} & \cdots & \cdots & x_{NF} \end{pmatrix} \tag{4.4}$$

Edge-level attributes can be assigned by modifying the adjacency matrix, simply replacing the entries of the adjacency matrix with the edge features. Lastly, there are graph-level or global attributes that are assigned to an entire graph or subgraph.

## 4.3 Graph Neural Networks (GNNs)

Graph Neural Networks are specialised in performing prediction tasks on graphs, the following summary is based on [23]. A Graph Neural Network (GNN) uses message passing (MSG) to exchange information between nodes and updates this information using neural networks. Information handled by a GNN can be node-level, edge-level or graph-level information, in order to explain the basic functionality in the following section only node-level information (node features) is referenced.

The Graph Neural Networks use a graph $G(V, E)$ as input and return a graph as output, updating information without changing the fundamental graph structure. The basis of any

GNN is the aggregation of information often called message passing. In Fig. 4.6 the message for a target node $A$ is generated, the GNN aggregates messages of the neighbourhood $\mathcal{N}(A)$ consisting of the nodes $B$, $C$ and $D$. The messages of the nodes adjacent to node $A$ are in turn the product of aggregation over their neighbourhoods. For example, the message aggregated of the neighbourhood $\mathcal{N}(B)$ consists of the messages of node $A$ and $C$. The tree structure of the hidden embeddings of nodes in order to compute the hidden embedding for node $A$ is called the computation graph $G_C$.

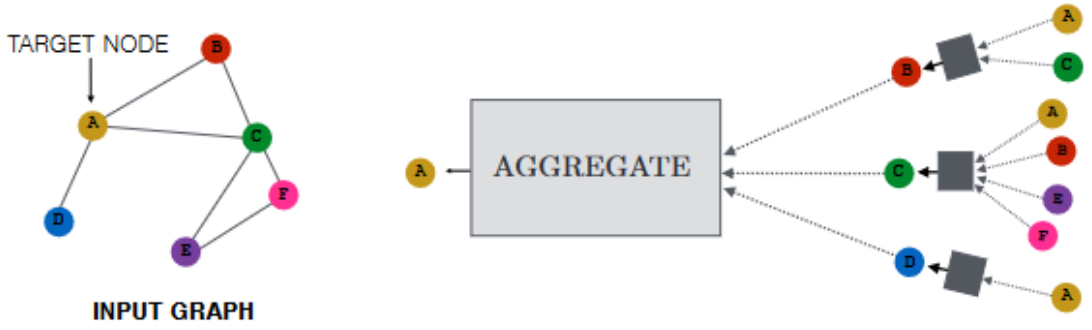



Figure 4.6: Message for target node $A$ aggregated over the neighbourhood of node $A$, consisting of the messages of the nodes $B$, $C$ and $D$. Which are in turn aggregated of the messages of their neighbourhoods. Taken from Ref. [23].

In computation terms, the message aggregated over the neighbourhood $\mathcal{N}(u)$ can be denoted as an aggregation operation $m^{(k)}_{\mathcal{N}(u)} = \text{AGGREGATE}(\{h^{(k-1)}_v, \forall v \in \mathcal{N}(u)\})$ that combines the previously computed hidden embedding $h^{(k-1)}_v$ of every node $v$ in the neighbourhood. The hidden embedding $h^{(k)}_u$ for node $u$ at iteration $k$ can now be expressed in order of update and message passing operations, a possible formulation can be seen in equation 4.5

$$h^{(k)}_u = \text{UPDATE}^{(k)}(h^{(k-1)}_u, m^{(k)}_{\mathcal{N}(u)}) \tag{4.5}$$

At each iteration $k$ the GNN updates the hidden embedding of the previous iteration $h^{(k-1)}_u$ with the aggregated neighbourhood information $m^{(k)}_{\mathcal{N}(u)}$ to the hidden embedding $k^{(k)}_u$. At iteration $k = 0$ the hidden embedding is simply the initial embedding, for example the node features $X$. Each additional layer of the GNN results in an additional message passing iteration, so the message in the figure 4.6 is being aggregated for a two-layer GNN. A one-layer GNN would only aggregate information of the nodes $B$, $C$ and $D$, for example the node features of those nodes. Iterating over $k$-Iterations can be understood as aggregation over the $k$-hop neighbourhood of a node, a one-hop neighbourhood is only aggregated over the immediate neighbours of a node.

In order to turn the eq. 4.5 into a workable GNN framework, trainable weights $W$ and a bias $b$ have to be introduced. Equation 4.6 shows the most basic GNN framework. The weights $W^{(k)}_{\text{self}}$ and $W^{(k)}_{\text{neighbour}}$ are matrices with the dimensions $dim = d^{(k)} \times d^{(k-1)}$ and the bias $b^{(k)}$ is a matrix with the dimension $dim = d^{(k)}$. Here, the dimension $d^{(k)}$ refers to the dimensionality of the node feature information at the current iteration and $d^{(k-1)}$ the dimensionality of the node feature information at the previous iteration $d^{(k)}$.

$$h_u^{(k)} = \sigma(W_{\text{self}}^{(k)} h_u^{(k-1)} + W_{\text{neighbour}}^{(k)} \sum_{v \forall \mathcal{N}(u)} h_v^{(k-1)} + b^{(k)}). \tag{4.6}$$

In this example, this basic GNN structure only aggregates node-level information (node features). However, in many cases edge-level and graph-level information is also of interest. So to aggregate this information, the message passing framework has to be updated. The new message needs to aggregate node-level, edge-level and graph-level information at each iteration step

$$h_{(u,v)}^{(k)} = \text{UPDATE}_{\text{edge}}(h_{(u,v)}^{(k-1)}, h_u^{(k-1)}, h_v^{(k-1)}, h_G^{(k-1)}). \tag{4.7}$$

The hidden embedding $h_{(u,v)}^{(k)}$ for the edge $(u, v)$ is derived by updating the embedding of the edge of the previous iteration $(k-1)$, the hidden embeddings of the nodes $u$ and $v$ incident to the edge of the previous iteration $(k-1)$ and the hidden embedding for graph-level information $h_G^{(k-1)}$ of the previous iteration. The message $m_{\mathcal{N}(u)}$ for the neighbourhood of the node $u$ is modified to aggregate the hidden embeddings of edges incident to node $u$ and a node $v \in \mathcal{N}(u)$ in the neighbourhood of node $u$

$$m_{\mathcal{N}(u)} = \text{AGGREGATE}_{\text{node}}(\{h_{(u,v)}^{(k)} \forall v \in \mathcal{N}(u)\}). \tag{4.8}$$

The hidden embedding $h_u^{(k)}$ of node $u$ can then be calculated as an UPDATE function taking the hidden embedding $h_u^{(k-1)}$ of the previous iteration, the new message $m_{\mathcal{N}(u)}$ aggregated over the neighbourhood of node $u$ and the graph-level information of the previous iteration as the input

$$h_u^{(k)} = \text{UPDATE}_{node}(h_u^{k-1}, m_{\mathcal{N}(u)}, h_G^{(k-1)}). \tag{4.9}$$

The embedding of the graph $h_G^{(k)}$ is derived by updating the hidden embedding $h_G^{(k-1)}$ of the previous iteration, the hidden embeddings of all nodes $u \in V$ of the input graph $G(V, E)$ and the hidden embeddings of all edges $(u, v) \in E$ of the input graph $G(V, E)$

$$h_G^{(k)} = \text{UPDATE}_{graph}(h_G^{(k-1)}, \{h_u^{(k)}, \forall u \in V\}, \{h_{(u,v)}^{(k)}, \forall (u,v) \in E\}). \tag{4.10}$$

The UPDATE functions for edge-, node- and graph-level information as well as the AGGREGATE functions are not identical but serve to update and aggregate hidden embeddings depending on the neighbourhood information.

The GNN framework shown in eq. 4.6 is the simplest way to implement AGGREGATE and UPDATE functions, the framework can be generalised and improved upon by modifying the AGGREGATE and UPDATE functions. An in depth look at these methods can be found in detail in Ref. [23].

# 5 GNNExplainer

This chapter introduces GNNExplainer an Explainable AI method for generating explanations for GNNs by assigning an importance to the input features of the GNN. The basic functionality of the GNNExplainer is introduced in section 5.1, while a closer look at the different mask configurations is taken in section 5.2. A short overview of the implementation is given in section 5.3. The summary of this method in the following chapter is based on [25].

## 5.1 Basic Functionality

### 5.1.1 Motivation

The GNNExplainer generates explanations for the behaviour of Graph Neural Networks (GNNs), but why would one use GNNs and not DNNs or other Neural Network structures instead?

Deep Neural Networks have a large drawback, they are only applicable to data in vector form, but not every machine learning task is best represented as a vector. A lot of information can be naturally represented as graphs. Examples include molecules, social networks or decay processes. Such graphs are challenging to work with because one has not only to consider feature information (node-, edge-, and graph-level) but also relational information between the individual nodes. Another difficulty when working with graphs is that the node ordering is unfixed, which leads to several possible but equally valid representations for a graph structure. GNNs solve this problem by aggregating information of a $k$-hop neighbourhood at each iteration step, as introduced in section 4.3.

Nevertheless, there is a serious disadvantage to Graph Neural Networks: Explainability. GNNs possess a complex structure due to them taking into account rich graph structural information as well as node, edge and graph feature information. Thus, a prediction made by a GNN cannot be easily explained. In contrast to a basic machine learning task, the steps a GNN takes to reach a prediction cannot be easily retraced. It is impossible to explain them through linear combinations; a computation tree cannot be drawn for a GNN model the decisions taken are simply too complex.

GNNs can be understood as a black box, producing a prediction after being given an input, but leaving the way how exactly this prediction is reached in the dark. The transparency of a model is important to eliminate mistakes and biases, for example to differentiate noise

from relevant results. Methods that strife to explain the predictions of such models are called Explainable AI, GNNExplainer is such a method that is specialised for GNNs.

There are generally two approaches to generate an explanation for a black-box, one is to approximate the model with another model and then probe this model for an explanation. Another method is to assign an importance to the input and probe the model for changes in the prediction when changing the input, this is the approach GNNExplainer takes.

### 5.1.2 Functionality

GNNExplainer is a post-hoc interpretability method, meaning this method requires an already trained model $\phi$. One advantage of GNNExplainer is that it is model agnostic, it does not require modification of the underlying GNN framework and no retraining of the GNN. The GNNExplainer can generate explanations for any Graph Neural Network (GNN) that can be formulated in terms of Message passing (MSG), Aggregation (AGG) and UPDATE computations as discussed in section 4.3. In order to generate an explanation the GNNExplainer uses that the prediction $\hat{y} = \phi(G_C, X_C)$ of a trained GNN model $\phi$ is only dependent on the computation graph $G_C$ and the associated node features $X_C$ of that graph.

GNNExplainer can generate explanations for Node-level prediction (NLP) and Graph-level prediction (GLP), the formulation for NLP explicitly refers to the node $v$. The computation graph $G_C$ for GLP and $G_C(v)$ for NLP are not necessarily the same graph depending on the aggregation of the hidden embedding for a node $v$ as introduced in section 4.3. The prediction for this node analogous to graph-level operations can be formulated as $\hat{y}(v) = \phi(G_C(v), X_C(v))$. The computation graph $G_C(v)$ and associated node features $X_C(v)$ can differ depending on the node.

In this thesis, the computation graphs $G_S$ and $G_S(v)$ are identical because the hidden embedding for a node $v$ is aggregated over all nodes. Furthermore, the input graph is complete, so for both NLP and GLP the computation graph corresponds with the input graph, the graph with which the underlying GNN is trained.

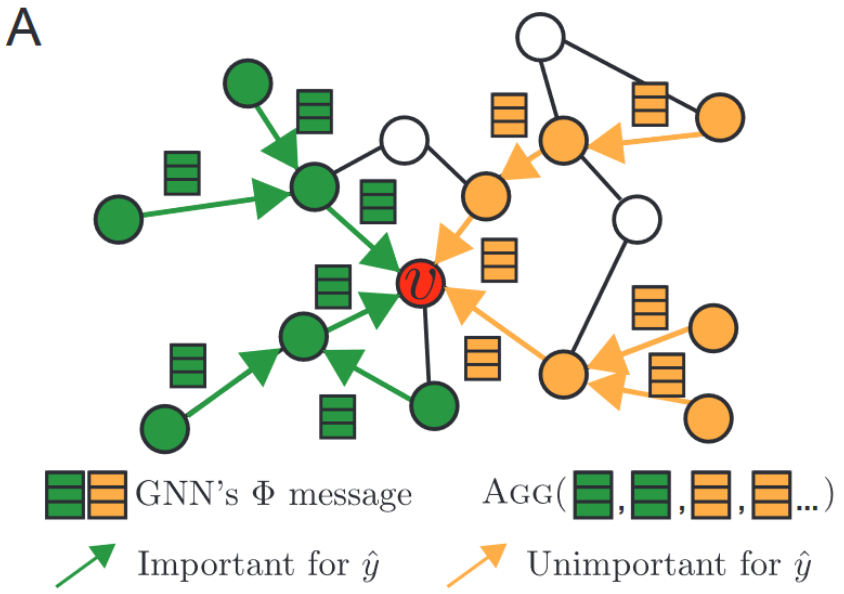



Figure 5.1: Illustration of a Computation Graph $G_C$ of a GNN, in order to determine the prediction $\hat{y}$ for node $v$, the GNN aggregates over the two subgraphs in green and yellow. The yellow subgraph is unimportant for the prediction $\hat{y}$, only the green subgraph is important. GNNExplainer extracts the green subgraph $G_S$, which only contains graph structural information and associated node features $X_S$, which are relevant for the prediction $\hat{y}$. Figure taken from Ref. [25].

The Basic idea behind the GNNExplainer is that only a small fraction of edges and node attributes have an impact on the prediction $\hat{y}$. The goal of the GNNExplainer is to find this compact subgraph $G_S \subseteq G_C$ of the computation graph and a small subset of associated node attributes $X_S$ of that subgraph. This can be illustrated by figure Fig. 5.1, the computation graph $G_C$ contains the two subgraphs in green and yellow, the uncoloured nodes and edges are not part of the computation graph because they do not contribute during message passing or aggregation, and are thus not part of the hidden embedding of node $v$. The subgraph $G_S$ is important for the prediction of the node $v$ marked in red is represented in green, the subgraph in yellow is not important in order to make a prediction for node $v$.

The subgraph $G_S$ contains all the relevant graph structural information and subset $X_S$ all the relevant node feature information for the prediction $\hat{X_S}$. Such a subgraph $G_S$ and subset of node features $X_S$ is illustrated in Fig. 5.2, the subgraph is again coloured in green and all unimportant nodes and edges are coloured out. The associated node features $X_S$ are displayed above the nodes, with the unimportant node features crossed out.

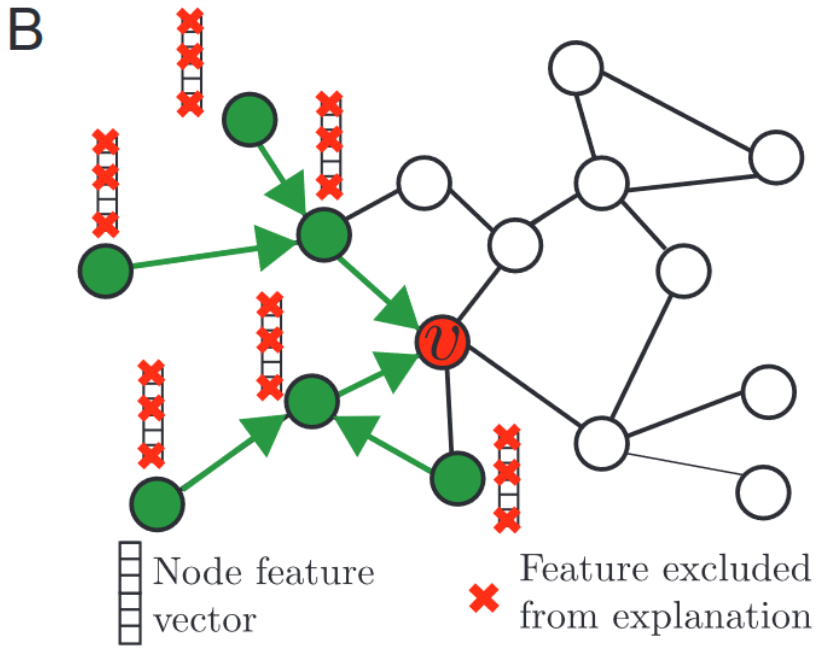



Figure 5.2: The explanation ($G_S$, $X_S$) provided by GNNExplainer only contains the compact subgraph $G_S$ (green), which contains the graph structural information relevant to the prediction and the subset of node features $X_S$. In order to denoise the computation graph, features and edges are excluded from the explanation, which do not contribute to the prediction $\hat{y}$ significantly. Figure taken from Ref. [25].

Because all irrelevant graph structural information (edges, nodes) and irrelevant node features are removed from the computation graph $G_C$ and subset $X_C$ resulting in a noise free subgraph $G_S$ and subset $X_S$ it can also be understood as a form of denoising. The collection ($G_S$, $X_S$) of subgraph and associated node features can be seen as an explanation of the prediction $\hat{y}$ because it contains all the relevant information in order to reach the prediction, e.g. used as an input graph $G_C$ and $G_S$ should result in approximately the same result.

GNNExplainer can now be formulated as an optimization problem in eq. 5.1, using mutual information (MI) between a predicted label distribution $Y$ and the explanation $(G_S, X_S)$.

$$\max_{(G_S, X_S)} (\mathrm{MI}(Y, (G_S, X_S))) \tag{5.1}$$

The mutual information measures how much information about the predicted label distribution $Y$ can be derived from a subgraph and subset of node features, a value of zero would indicate that the prediction is independent of the subgraph. Conversely, larger values signify a larger dependence of the prediction on the subgraph, by maximising the mutual information of the subgraph $G_S$ and a subset of node features $X_S$ most impactful for the prediction is derived.

$$MI(Y, (G_S, X_S)) = H(Y) - H(Y|G = G_S, X = X_S) \tag{5.2}$$

The marginal entropy $H(Y)$ of the predicted label distribution is derived from the computation graph $G_C$. The conditional entropy $H(Y|G, X)$ explicitly formulated as the entropy

of the subgraph $H(Y|G = G_S, X = X_S)$ quantifies the amount of information needed to derive the predicted label distribution $Y$ given the subgraph $G_S$ and associated node features $X_S$. Formulating the optimisation problem in this way shows why GNNEXplainer can be understood as a denoising task, seeing graph $G$ and d-dimensional node features $X$ as a noisy version of the predicted label distribution $Y$.

Being a post-hoc method the model $\phi$ being used is already trained, which is not changed during the optimisation steps. With $\phi$ being constant, the marginal entropy $H(Y)$ also remains constant during the optimisation process. The optimisation problem can be reduced to minimising the conditional entropy.

$$H(Y|G = G_S, X = X_S) = -\mathbb{E}_{(Y|G_S,X_S)}[logP_\phi(Y|G = G_S, X = X_S)] \tag{5.3}$$

Here $\mathbb{E}$ denotes the expected value operator and $P_\phi$ the sigmoid transformed logits, which represent the probability of the event in question being associated with a predicted label.

Direct optimisation of equation eq. 5.3 is impossible because not only one subgraph $G_S$ exists but potentially exponentially many subgraphs and thus exponentially many possible explanations.

In order to solve this problem, GNNEXplainer introduces two masks, an edge mask $M$ on the adjacency matrix of the computation graph and a feature mask $F$ on the node features. These masks control the influence of entries of the input. In essence, the influence the subgraph and associated node features have on the prediction $\hat{y}$. The masks can be understood as selectors masking out irrelevant entries. For example, if a feature is unimportant and masked out the prediction $\hat{y}$ will not change and vice versa if it is important $\hat{y}$ will change. Applying the sigmoid transformed feature selector $\sigma(F)$ on the associated node features $X_C$ of the computation graph $G_C$ via elementwise multiplication $\odot$ will return a subset of node features $X_S^F$ which are important for $\hat{y}$.

$$X_S^F = X_C \odot \sigma(F) \tag{5.4}$$

A subgraph $G_S^F$ can be generated in a similar fashion by applying the (edge mask) selector to the adjacency matrix $A_C$ of the computation graph $G_C$.

$$G_S^F = A_C \odot \sigma(M) \tag{5.5}$$

The subgraph $G_S^F$ and subset $X_S^F$ are not necessarily identical to the desired explanation $(G_S, X_S)$, only for a well-trained edge masks $M$ the subgraph $G_S^F$ is an accurate representation of the subgraph $G_S$. Entries of the node mask $F$ are randomly initialised and kept between zero and one by applying a sigmoid function $\sigma(F)$. The masks are derived by probing the model in this way, suppressing edges and node features during each optimisation step.

Often the goal of an analysis with GNNExplainer is not an exact prediction $\hat{y}$ but how or why a label $y$ of a prediction $\hat{y}$ is predicted, this is done via the cross entropy between an initial prediction and the sum of probabilities of a prediction $P_\phi$ of the model being a possible label $K$ at the specific subgraph $(G_S, X_S)$.

$$\mathcal{L} := -\sum_{k=1}^{K} \mathbb{I}_{y=k} logP_\phi(Y = y|G = A_C \odot \sigma(M), X = X_C \odot \sigma(F)) \tag{5.6}$$

The optimisation is done via gradient descent Ref. [25] using the loss function $\mathcal{L}$ in eq. 5.6. Subsequently regularisation terms are applied to the loss. Regularisation terms include sum over all elements in the edge mask, mean over all elements of the node mask, elementwise entropy over the masks and mean over the elementwise entropy. These terms ensure a compact form of the explanation, the elementwise entropy results in discrete masks and further constraints can be added through Lagrange multipliers.

## 5.2 Node Masks

The goal of GNNExplainer is to find an explanation $(G_S, X_S)$ containing the subgraph $G_S$ and subset of node features $X_S$, these are obtained by applying trainable masks as selectors

$$G_S = A_C \odot \sigma(M) \tag{5.7}$$

$$X_S = X_C \odot \sigma(F) \tag{5.8}$$

Information about the influence of edges and node features on the prediction $\hat{y}$ of the GNN are contained in the edge mask $M$ and node feature mask $F$. If an edge or feature is unimportant, the mask entries will be zero or close to zero. The remaining mask entries will be assigned values based on their impact on the prediction, with the most important features assuming values close to one. The masks are more compact than an actual subgraph or d-dimensional node features, they also apply a numeric importance to all mask entries. As a result, they are more practical than the actual subgraph and associated node features $(G_S, X_S)$, from this point onward the masks will be regarded as the explanation GNNEXplainer yields.

The edge mask $M$ contains graph structural information and is always of the dimension $\dim = N \times N$, with $N$ denoting the number of edges in the graph. The current implementation of GNNExplainer used in this thesis does not consider edge attributes when generating the node mask. This ignores crucial graph structural information, because of this and the difficulties in interpreting the edge mask for large input graphs in this thesis only node feature masks are used for the Analysis of GNNs.

The GNNExplainer implementation used in this thesis offers three mask configurations for the node mask: Attribute/Feature, Individual Feature and Scalar. In this thesis, the terms node feature and node attribute are used interchangeably. The masks differ in their dimensions $\mathbb{R}^{1xF}$, $\mathbb{R}^{NxF}$ and $\mathbb{R}^{Nx1}$, this determines which information particular masks hold and how they can be interpreted.

The Attribute/Feature configuration illustrated in Fig. 5.3, masks common node features of all nodes, taking the dimension $\dim = 1 \times F$, where $F$ denotes the number of node features. This configuration does not consider the importance of individual nodes, the same mask is trained for all nodes. This is the most computationally efficient mask configuration out of the three node mask types because the same node mask is trained for all nodes.

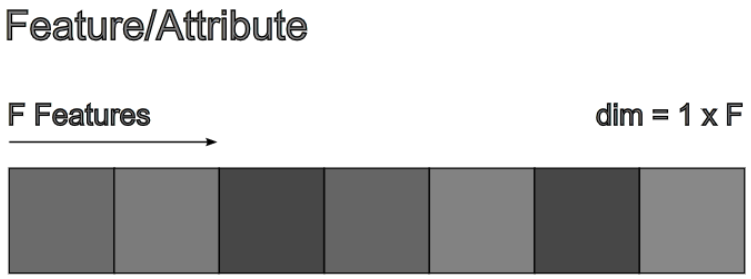



Figure 5.3: The Attribute/Feature configuration masks all common features of all nodes.

The Individual Feature configuration is shown in Fig. 5.4. This configuration masks node features for each node individually, i.e. it considers the importance of nodes as well as the features of these nodes. In terms of functionality, the mask can be understood as attribute mask applied to each node. Correspondingly, the mask takes the dimension dim $= N \times F$, $N$ denotes the number of nodes and $F$ the number of features. If a node is masked out in this configuration all node features of this node take values that are zero or close to zero.

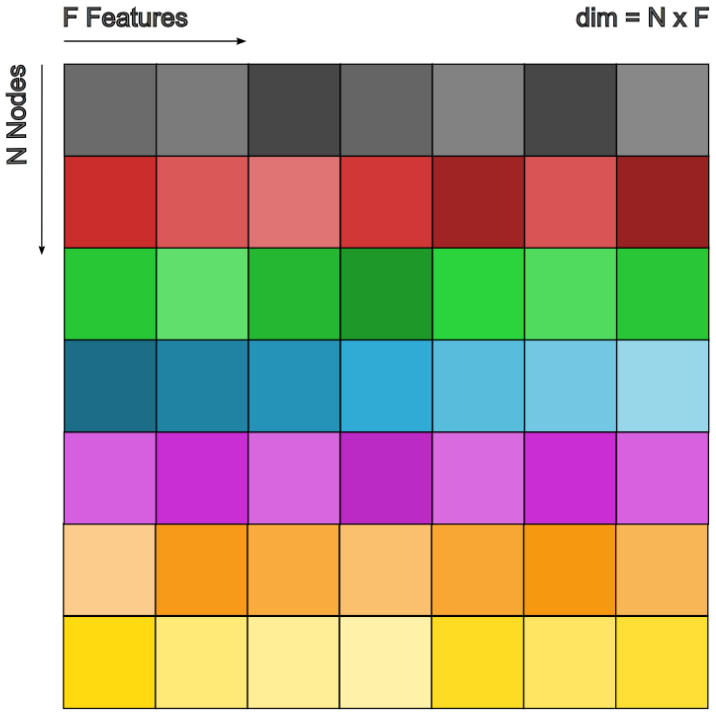



Figure 5.4: The Individual Feature configuration masks node features for each node individually.

The Scalar configuration masks all nodes, it does not consider the importance of node features. The dimension dim $= N \times 1$ of the mask is only influenced by the number of nodes $N$.

This mask configuration is more computationally efficient than the individual feature mask and less efficient than the feature configuration if there are more nodes than node features.

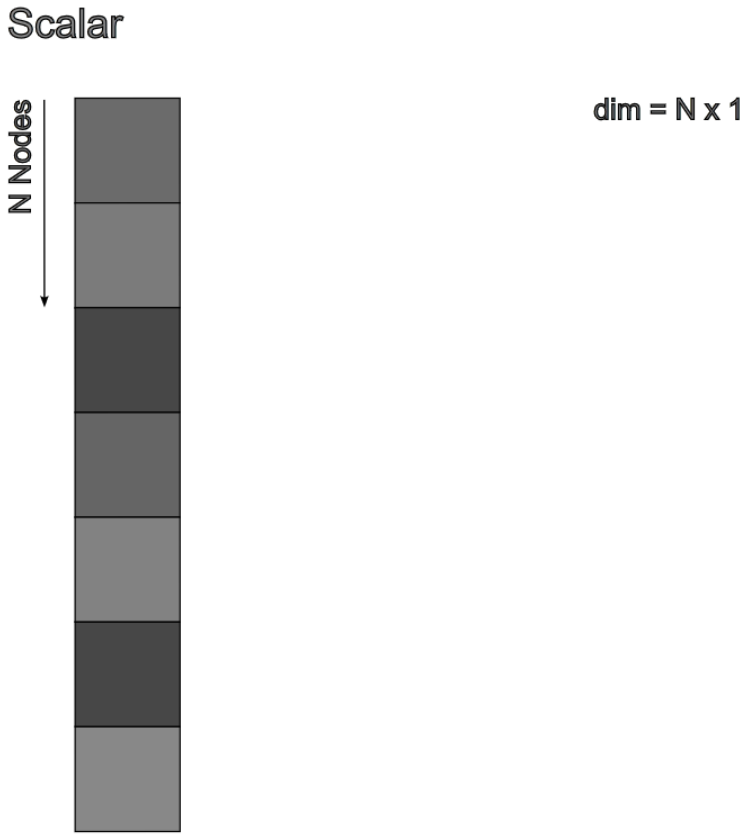



Figure 5.5: The Scalar configuration masks all nodes with no regard of feature information.

## 5.3 Implementation

This thesis uses the implementation of GNNExplainer found in the *torch_geometric.nn*-module [26], for the training of the masks Adam Optimizer found in the *torch.optim*-module [27] is used. The Parameters of the GNNExplainer are listed in tab. 5.1, as mentioned before the edge mask is not considered and will not be optimized, by setting the Parameter *allow_edge_mask* to $False$.

As learning rate, the default learning rate of $lr = 0.01$ is used. There are no values given for the *num_hops* variable because, as a complete graph, the hidden layer of a node is always aggregated over all nodes.

The *return_type* chosen in this paper is *prob*, which means that the mask entries correspond to percentages of the input. A mask entry of one corresponds to 100% of the input being used in the prediction, correspondingly a zero means that the feature is completely masked out. For example, an entry of 0.7 would mean that 70% of the input is being used in the prediction, with 30% of the feature being masked out.

Table 5.1: Most important parameters of the GNNExplainer implementation from *torch_geometric*-module Ref. [26]

| Parameter | description |
|---|---|
| *model* | Trained GNN-model $\phi$ passed to the GNNExplainer. |
| *epochs* | Number of optimization steps during training. |
| *lr* | Learning rate of the optimizer. |
| *num_hops* | How many k-hops during the neighbourhood aggregation are used, if not given the *num_hops* will be automatically calculated. |
| *return_type* | Output of the GNNEXplainer explanation. Offers three types of return *prob*, *log_prob*, *raw* as well as a return type for *regression*. |
| $feat\_mask\_type$ | Determines the configuration of the node mask $feature$, $individual\_feature$ and *scalar*. |
| *allow_edge_mask* | Determines if an edge mask will be trained. |

# 6 Analysis of GNNs with GNNExplainer

The goal of this chapter is to generate explanations utilising the GNNExplainer method introduced in chapter 5 and on the basis of these results to identify the relevant features for GNN prediction. These features are the basis of the decisions made by the underlying GNN. By examining the results and comparing them against the physics expectations for the event classes, an attempt is made to reach a better understanding of the decision-making process of the GNN. A summary of the results is made in section 6.3.

## 6.1 Interpretation

The explanations generated by GNNExplainer as introduced in chapter 5 are generated for 800 epochs. The choice of the optimising iteration is important because the masks are randomly initialised and then slowly optimised with respect to the initial prediction. So, if not enough epochs are used, this will result in an incorrect mask that does not reflect the input importance of the individual features. Using more epochs than necessary does not lead to a reduction in the expressive power of the mask, but it does increase computation time. Explanations are generated for each individual event in the dataset chosen for analysis and a node feature mask is trained for every event individually. For large datasets with a large number of individual events, this can be computationally demanding, so the configuration of 800 epochs achieves a high predictive power while being comparably efficient. All explanations are generated for the mask feature configuration as introduced in chapter 5.2.

When interpreting the node mask feature importance returned by the GNNExplainer one has to keep in mind how these are derived in the first place. The masks filter out features unimportant for the prediction by reducing the input and observing if the prediction of the GNN changes compared to the initial prediction. The initial prediction is treated as a kind of "true label". The importance given is a measure of how the input influences the prediction. Using this definition, a feature is important if the specific value is important for the prediction. In essence, this means changing it changes the prediction. An unimportant feature in contrast does not change the prediction if it is partly or completely filtered out by the feature mask. The more the prediction changes (compared to the initial prediction) when changing the input feature, the more important it is for the prediction. This behaviour is independent of the actual feature values. If a feature with a small feature value is important for the prediction, changing it will still result in a change in the prediction.

Another important point when interpreting the feature importance is that the explanation is in regard to the model, not in regard to the underlying physics processes. The node mask is explicitly trained in regard to an initial prediction made using the trained model, not the actual true label, which determines the class of the event. The GNNExplainer aims to generate explanations for the behaviour of GNNs, showing how the input values influence the prediction of the GNN.

## 6.2 Node Features

The features "$p_T$", "Phi" and "Eta" denote the transverse momentum, the azimuthal angle and the pseudorapidity respectively, as introduced in section 3.3. The feature "M" denotes the invariant mass of the jet or object in question and "E" denotes the energy of the object in question (this can be either a jet, a lepton or the missing transverse energy). The feature "charge" is a measure of the electric charge of the object, it assumes values of 1 for leptons, $-1$ for antileptons and zero for all other final state objects. The heavy flavour tagging values "CvL" and "CvB" are introduced in the chapter 2.2. The feature flags "is jet" and "is lep" determine if an object in question is a jet or a lepton. The "is jet" flag is one, if the object is a jet and zero if it is not. Similarly, the "is lep" flag is one if the object in question is a lepton and a value of zero if it is not. The missing transverse energy (MET) is characterised by having a value of zero in both feature flags. The "$p_{nominal}$" feature is the result of a previously made node-level prediction (NLP) using the same dataset. The goal of the node-level prediction is to predict if a node is an additional jet or not. The inclusion of this feature in the graph-level prediction (GLP) has led to an improvement in the predictive capabilities of the GNN model.

## 6.3 Graph-level prediction (GLP)

The goal of the graph-level prediction is to predict a class label for an event out of several event types. This predicted label can be measured against the true class of the event. So, it is of interest why a certain prediction is reached by the GNN and the GNNExplainer offers an explanation by assigning a feature importance to the input features. Only the di-leptonic decay channel is considered. The decay channel and the possible event classes are introduced in chapter 2.2. The importance of features for the prediction of an event class can vary for each event. In order to derive a valid explanation, the mean of masks over all similar events in a dataset, is taken. An example is the mean over all events identified as $ttLF$ events in figure 6.1.

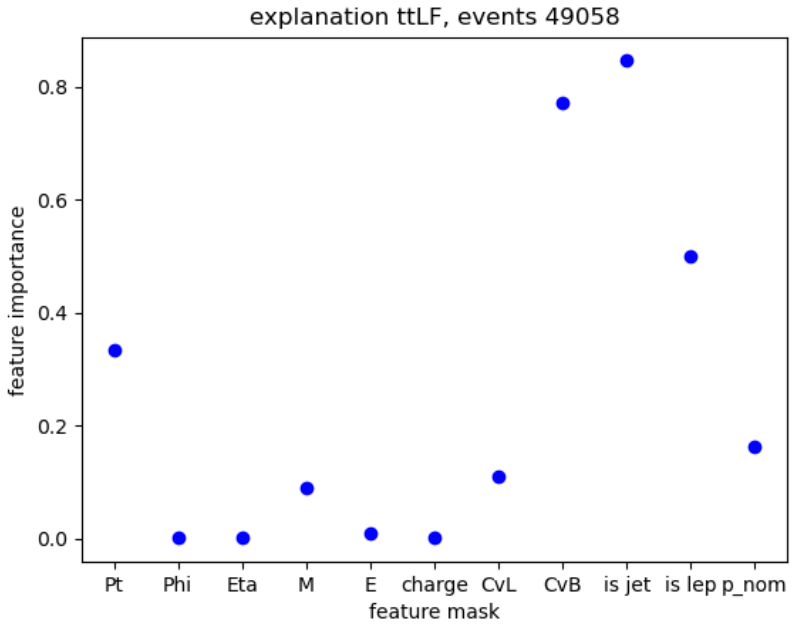
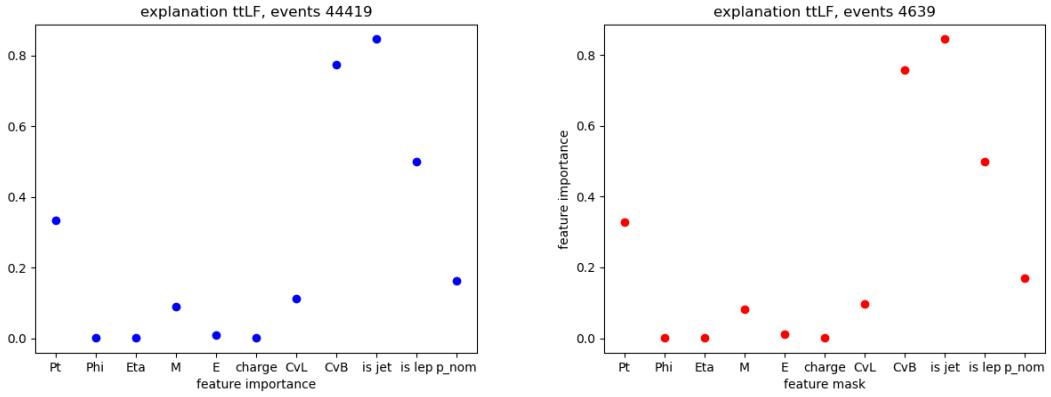



Figure 6.1: Mean over node mask features of 49058 events predicted to be *ttLF* event class.

For each event class some events are falsely predicted, so the explanations generated by the GNN can be separated into two categories: true positives in Fig. 6.2a, for which the class label is correctly predicted and false positives in Fig. 6.2b for which the class label is wrongly predicted. Plotting the node mask features of the events results in similar behaviour as shown in Fig. 6.1 for *ttLF* events. This is in line with the expectations put on the GNNExplainer. The node feature mask is a measure of how the input features influence the decisions made by the GNN. If the GNN predicts two events to be of a certain class, they are expected to have similar node feature masks. This is expected because the feature masks are trained explicitly in regard to the prediction of the GNN, not the true class of an event; this holds true for all event classes.




(a) True Positives (b) False Positives

Figure 6.2: Mean over node feature masks for a) all events correctly predicted as *ttLF* events and b) all events incorrectly predicted as *ttLF* events.

Examining the mean of the *ttLF* events in Fig. 6.1 shows that the most important features are by far the "is jet" and "CvB" features. The behaviour of the two-dimensional heavy flavour taggers "CvL" and "CvB" is not necessarily expected, for this event, the additional radiation in question is light-flavour one would expect "CvL" (input) feature values (not the importance generated by the GNNExplainer) to be small and "CvL" input values to be large. As explained in section 6.1 small values do not necessarily translate into small importance values, the guess that "CvL" has a high importance because for a *ttLF* event a low "CvL" input value is just as characteristic as "CvB" having a high input value is not far-fetched. Ultimately, the importance of a feature is determined by how the input influences the prediction of the GNN, in this case, a small importance is assigned to the "CvL" tagger. The underlying GNN seems to favour identifying the event class with the higher tagger (input) value, in this case assigning high importance values to the feature "CvL".

The feature flag "is lep" shows medium importance. Similarly, the transverse momentum "$p_T$" is not a top feature, but not an unimportant feature either. The invariant mass of the jet "M" and the energy "E" have a very low feature importance, but are not zero either. The behaviour of the feature "$p_{nominal}$" is similar. This feature is most likely used to distinguish additional jets from other jets, with the additional jets being light flavour jets they are probably easier to distinguish from the *b*-jets of the top quark decay, resulting in lower importance values for the feature "$p_{nominal}$". The features "Phi", "Eta" and "charge" are irrelevant for the prediction, this is in line with the physics expectations. An importance of the features "Phi" and "Eta" would mean the existence of a preferred spatial direction that contributes to predicting the event class. The feature "charge" is likely of no importance because the channel observed is the di-leptonic decay channel and the top quark decays for all events in a similar fashion.

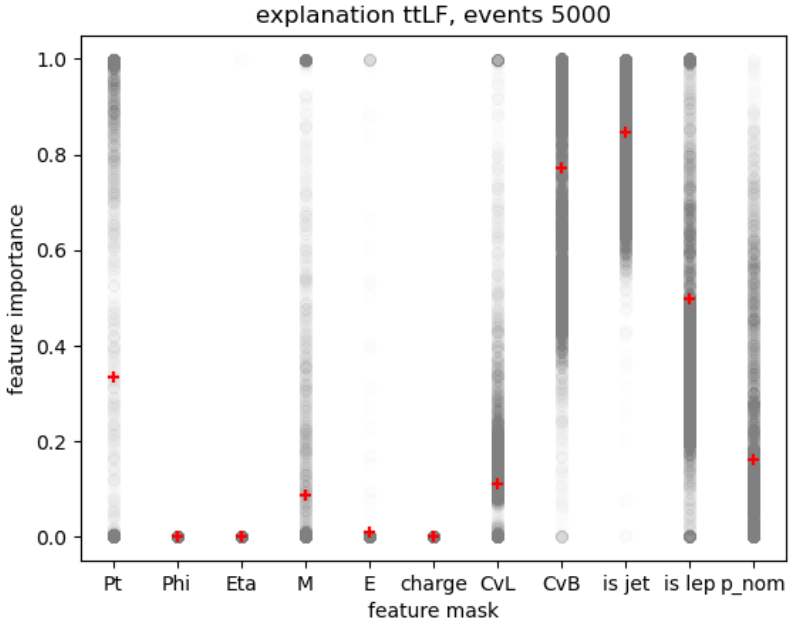



Figure 6.3: Scatter plot of the node feature masks of 5000 events with *ttLF* as predicted class in grey and the mean of each individual feature in red.

When interpreting the importance of features based on the feature mask in Fig. 6.1, one has to take into account that the result is derived as the average over a large number of

events. So, the results of the GNNExplainer can vary strongly depending on the event under scrutiny, so there is the possibility that an explanation derived from the mean is not helpful in trying to understand a certain event. So in order to better understand the predictions of the underlying GNN, the fluctuations of the feature importance of the masks have to be taken into account.

For the individual events, the feature importance fluctuates compared to the mean calculated over all events. These fluctuations are illustrated in Fig.6.3. To make these plots comparable, 5000 events are examined for each event class. Additionally, the mean is displayed as a red cross. When interpreting this plot, one has to consider that each point represents the feature importance of a single event, due to the transparency one event alone is not visible and 10 events are barely visible. A concentration of over 100 events results in a solid grey they cannot be further distinguished, so this diagram serves mainly as an illustration of the spread of the fluctuations.

**ttLF**

The observations made for the mean in Fig. 6.3 can be mostly reproduced for the distribution of the node mask importances. Similar to the mean, the "is jet" feature stands out mainly as being concentrated for high feature mask values above 0.5 in the upper part of the plot, reinforcing the importance of that feature. The heavy flavour tagger values "CvL" and "CvL" show an inverse behaviour, for "CvL" the concentration diminishes for higher importance values, while "CvB" diminishes for lower values of the distribution. The "CvL" feature value also shows some events with a feature importance of one, strongly deviating from the mean. So the GNN in some cases sees the smaller "CvL" input value as important for the prediction, but in much smaller quantities than the "CvL" feature.

The features "$p_T$" and "is lep" fluctuate a lot, in the mean this can be seen by them having average importance. Especially, the transverse momentum "$p_T$" is different from the mean by being mainly concentrated at the extremes, with a comparatively low concentration in between. In contrast, the "is lep" feature has a considerable concentration of events between the importance values 0 and 1. The "$p_{nominal}$" feature has a strong concentration in the bottom third of the graph, slowly thinning out towards the top. The invariant mass also shows fluctuations with a majority of events at importance value zero, some events with importance one and a smaller number of events in between. The energy "E" shows a small number of events having it as a very important feature but overall being concentrated at zero importance.

Similar to the examination of the mean, the features "Phi", "Eta" and "charge" are unimportant for the prediction.

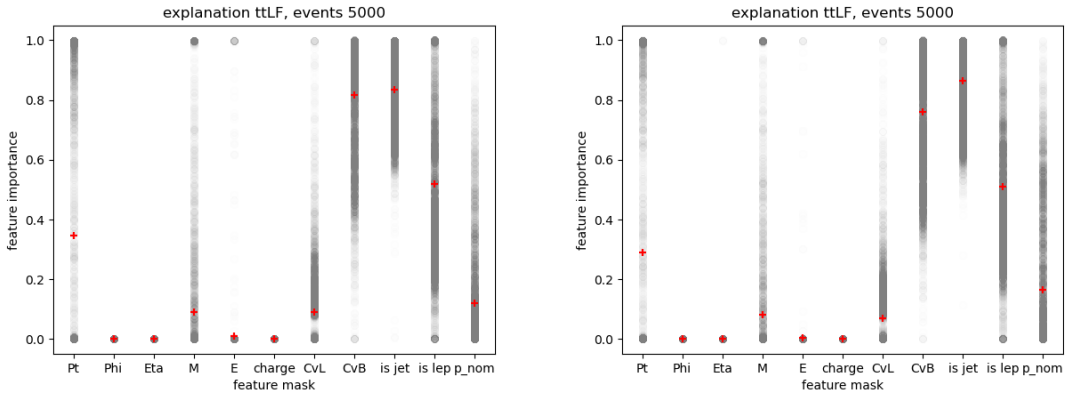



Figure 6.4: Scatter plot of node feature masks of 5000 events (falsely) classified as *ttLF* events for different datasets.

These results can be reproduced for events in different datasets (falsely) identified as *ttLF* events in Fig. 6.4, this is in line with the results for true and false positives in Fig. 6.2. This shows that falsely identified events have similar node feature masks as true positives independent of the dataset in use.

### ttC

The *ttC* event class, is characterised by the additional jets consisting of charm quarks because of the $g \to c\bar{c}$ decay in Fig. 6.5, shows much lower feature importance values for the top events in comparison to the *ttLF* events in Fig. 6.3. There are also more than one or two standout features, with the top features being the heavy flavour taggers "CvL" and "CvB", both with high importance. This is within expectations because high feature input values for "CvL" and "CvL" as charm tagging discriminants signify the presence of a *c*-flavour jet. So, it is to be expected that this would also result in high feature importance for the two heavy flavour taggers because determining a jet to be a charm jet requires both heavy flavour taggers to be present. The reason for the lower feature importance of the tagging features could lie in the identification of the *c*-flavour jets. Which require both "CvL" and "CvB" to assume significant values, making them harder to distinguish because they do not show the inverse behaviour they would for light-flavour or *b* jets. The distribution of events for the "CvL" feature is much more concentrated at the extremes than for the "CvB" feature, which shows significant concentrations for higher and lower feature importance different from one and zero. The "CvB" tagging value also shows a lower concentration for events with average feature importance values. Surprisingly the invariant mass "M" shows an importance nearly as high as the tagging features. This would be more expected for an additional *b*-jet with the bottom quark *b* having a mass several times larger than the charm quark *c*. But then again, the same is true for the charm quark *c* in contrast to the light flavour quarks up *u* down *d* and strange *s*. For the feature "M" most events are concentrated in the extremes, but there are also a significant number of events with importance values between zero and one. The "is jet" feature follows with slightly lower importance, showing fluctuations between one and zero importance values. The transverse momentum "$p_T$" mask shows a very strong fluctuation between an importance of zero and one. The distribution is a lot more concentrated in the extremes in contrast to the *ttLF* events. There is a small number of events that have a high feature importance for the energy "E" but for the majority of events the feature seems to be unimportant.

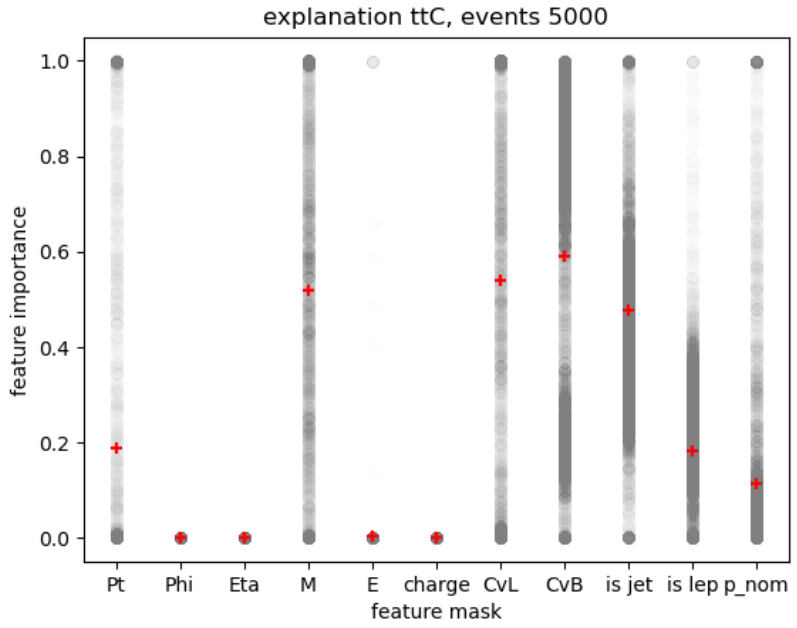



Figure 6.5: Scatter plot of the node feature masks of 5000 events with *ttC* as predicted class in grey and the mean of each individual feature in red.

For the "$p_{nominal}$" feature, events are mainly concentrated around low importance values, with a smaller number of events with slightly increased importance and some with importance one. Similar observations can be made for the "is lep" feature, mainly concentrated for lower values but lacking the concentration at importance value one. Similarly to the explanation for the *ttLF* events the features "Phi", "Eta" and "charge" are unimportant for the prediction, this is again in line with the physics expectations.

### ttB

For the *ttB* event class shown in Fig. 6.6 the feature "$p_{nominal}$" has the highest importance and is mainly concentrated around the mean at high mask values, but also possesses a concentration of events at the importance value of zero. A possible explanation for the high importance value is that it is used to distinguish between the additional *b*-jet and one originating from a top quark decay (*topb*, *antitopb*).

The heavy flavour tagger "CvL" is the feature with the second highest importance, looking at the fluctuations shows that the feature importance assumes a high mask value for a majority of the events but also assumes values over the whole range of possible values. Interestingly, the other tagging discriminator "CvL" is completely concentrated at value 0 unimportant for the prediction. The "CvL" tagging value assumes low importance values in the presence of *b*-flavour jets, so it is within expectations that "CvB" is of little importance for the prediction.

Similarly to the *ttC* events, the invariant mass "M" plays a role in the prediction but in contrast to the *ttC* events, the feature is much more important and has a stronger concentration for higher importance values. This can potentially be explained by the higher mass of the bottom quark, making it more distinct from the other additional radiation jets. But there is also a concentration of events with importance zero.

The mask of the transverse momentum "$p_T$" shows strong fluctuations with a significant number of events for all importance values, with high concentrations at the extremes and

also a concentration around the mean at 40% importance. The events for the feature flag "is lep" mainly assume lower importance values. In contrast to that, the events for the "is jet" flag are more spread out, having a high concentration for lower values but also some events for average and higher values. There are also some events with importance value one. The energy "E" shows a very distinct distribution, concentrated for some events at one and a majority at zero. In line with physics expectations, the features "Phi", "Eta" and "charge" show an importance of zero for the prediction.

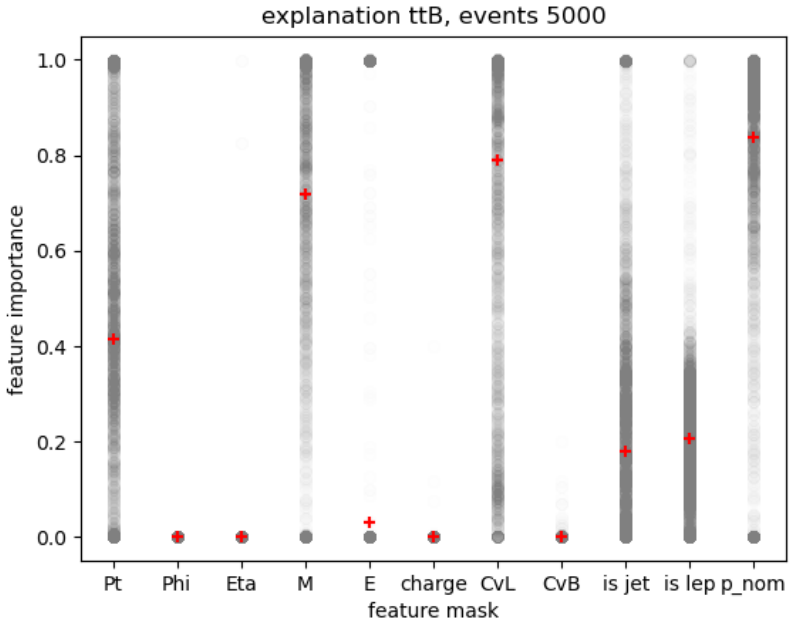



Figure 6.6: Scatter plot of the node feature masks of 5000 events with $ttB$ as predicted class in grey and the mean of each individual feature in red.

### ttH

In this thesis, only the events with an additional pair of $b$-jets originating from $H \to bb$ are considered.

The top feature for $ttH$ events Fig. 6.7 is the "is jet" flag, similar to $ttLF$ events concentrated at higher importance values. The feature "CvL" has the second highest importance; events are mainly concentrated at values above 0.3 and at 1. Events with very low importance are virtually absent. The heavy flavour tagging discriminators "CvL" and "CvB" show again an inverse behaviour with "CvL" being mainly concentrated at low values around zero. This is expected because of the additional jet being a $b$-jet similar to the $ttB$ event.

The mean of the "is lep" feature is of medium importance, with events either concentrated below the mean or around an importance value of one.

In contrast, the feature "$\mathrm{p}_{\mathrm{nominal}}$" which has a similar mean to "is lep" has a fluctuating importance value between zero and one with considerable event concentrations for most importance values. A possibility is that the feature is again used to distinguish the additional $b$-jets from the $b$-jets of the top decay. If this is the case, it is noteworthy that the importance value varies a lot more, a possible explanation could be that the NLP is much less effective in predicting additional jets for $ttH$ events when the additional radiation is a bottom quark pair.

In contrast to the previous event types, the transverse momentum $p_T$ is concentrated at very low importance values around zero, only a few events assume values other than zero. Similar behaviour can be observed for the invariant mass "M", which in contrast to the *ttB* events plays a minor role in the prediction. It is unclear why this is the case with the additional jet in both cases being a *b*-jet.

Another difference is that for some events, the "charge" feature plays a role in the prediction. Otherwise, the majority is concentrated at zero. The energy "E" and the pseudorapidity $\eta$ ("Eta") show an importance value of one for very few events. The azimuthal angle "$\phi$" plays no role in the prediction, as expected.

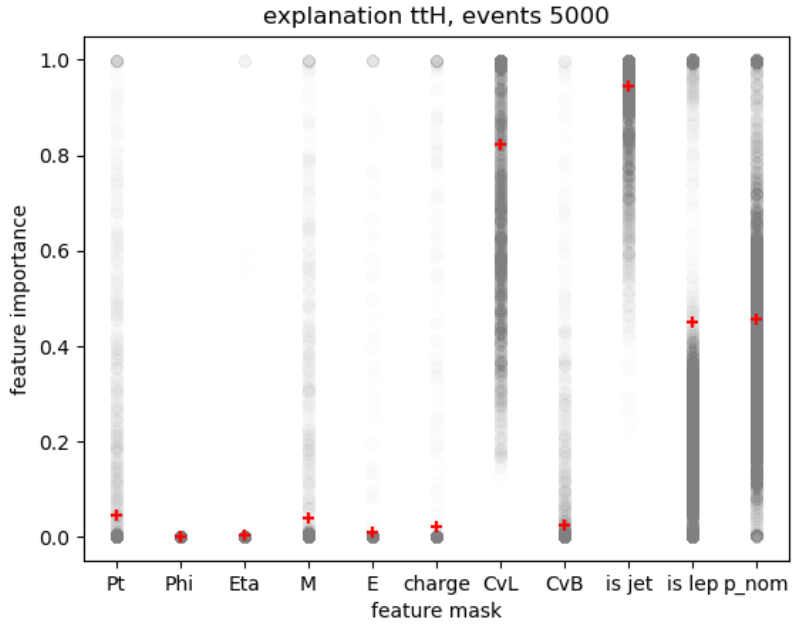



Figure 6.7: Scatter plot of the node feature masks of 5000 events with *ttH* as predicted class in grey and the mean of each individual feature in red.

### ttZB

The top quark-antiquark pair with an additional $Z$ boson to two bottom quarks decay events in Fig. 6.8 share similarities with the *ttH* events in Fig. 6.7. The most important feature is "$\mathrm{p_{nominal}}$" showing fluctuations mainly concentrated on higher values in the upper half of the distribution, this is in contrast to the *ttH* event which shows strong fluctuations. Similar to the *ttB* events, this feature could be relevant to distinguish additional *b*-jets from *b*-jets of the top decay.

The heavy flavour taggers "CvL" and "CvB" show an inverse behaviour for most events, but are less distinct than for *ttLF*, *ttB* and *ttH* events. The "CvL" tagger has a large number of events for high and middling importance values, thinning out at the bottom and virtually no events at zero. The "CvB" feature is mainly concentrated at lower importance values thinning out towards higher values, except that for some events the feature "CvB" has an importance of one, contrary to the inverse behaviour of the taggers.

The transverse momentum "$\mathrm{p_T}$" shows strong fluctuations, showing events with importance values between zero and one. The invariant mass "M" is mainly concentrated at importance values of zero but also some events with importance values different from zero. The feature

importance of "M" and fluctuations are similar to the ones of the $ttH$ events and in contrast to the $ttB$ events.

The invariant mass again is relevant for some events but is mainly concentrated at zero importance, the same is true for the "charge" feature. The behaviour of the "charge" feature is in contrast to the $ttLF$, $ttC$ or $ttB$ events, $ttH$ events show a similar behaviour but for fewer events. This is surprising because only leptons have a "charge" flag different from zero, 1 for leptons and $-1$ for antileptons. It is unclear why the leptonic top quark decay would influence the classification of the event when the leptonic decay is expected to be similar for all events.

The features "Phi" and "Eta" are of low importance for the prediction; the energy "E" shows a small fluctuation, but a large majority of events is concentrated at zero.

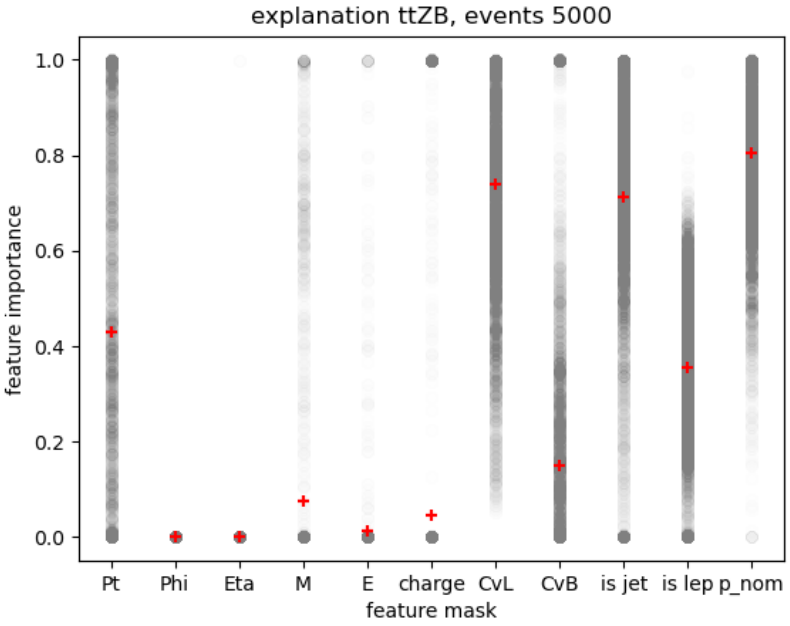



Figure 6.8: Scatter plot of the node feature masks of 5000 events with $ttZB$ as predicted class in grey and the mean of each individual feature in red.

### ttZnonB

The $ttZnonB$ event class is characterised by the additional radiation stemming from the $Z \to q\hat{q}$ decay, explicitly not being a pair of bottom quarks. The most important feature of the event shown in Fig. 6.9 is the "CvL" tagging mask, which is nearly completely concentrated at importance value one. This is in line with expectations: a high "CvL" value shows the absence of $b$-flavour jets, so for a decay that does not result in a bottom quark the importance should show a corresponding behaviour. The second highest mean importance value is the "CvL" feature, showing strong fluctuations between the values zero and one. A fluctuation for the feature importance of "CvL" is not unexpected because there is no distinction between the additional radiation being a light-flavour jet ($u$, $d$, $s$) or a charm-tagged jet, which could result in a different "CvL" importance depending on the jet flavour.

The feature flag "is lep" is centred around the importance value 0.4, having very low values at the top and virtually no values at the bottom of the distribution. This concentration

in the middle of the figure is different from the distributions of that feature in the other events, which are more spread out and also assume values at least one of the extremes.

The feature "$p_{nominal}$" shows strong fluctuations, showing fewer events for high importance values. This results in a similar mean as for the "is lep" feature.

For some events "charge" is a very important feature, but for the large majority, the feature is unimportant. The invariant mass "M" is mainly concentrated at a zero importance value, but shows small numbers of events with higher importance values. The transverse momentum "$p_T$" is concentrated around the importance value zero, thinning out for higher values. The features "Phi", "Eta" and "E" are of no importance for the prediction in line with expectations.

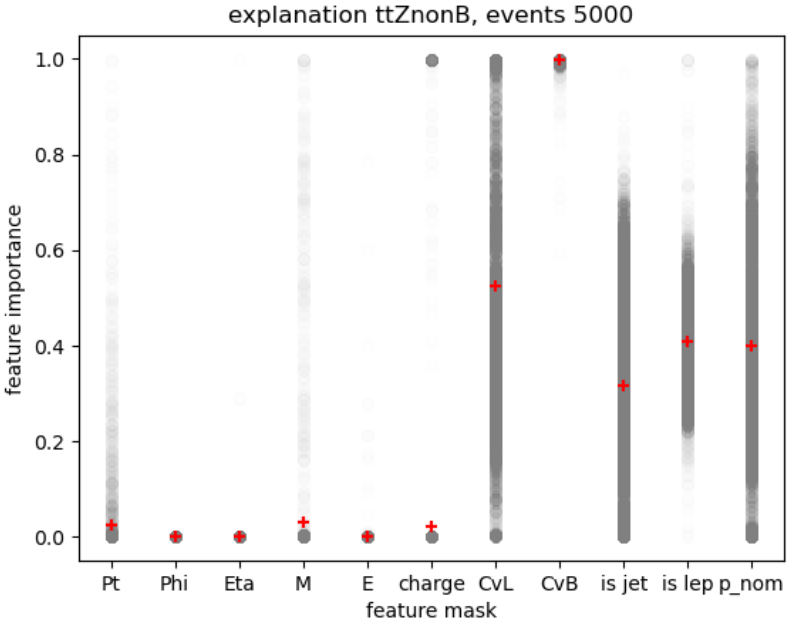



Figure 6.9: Scatter plot of the node feature masks of 5000 events with *ttZnonB* as predicted class in grey and the mean of each individual feature in red.

### other

Besides these event classes, there is also the possibility that the GNN identifies an event as *other*. This indicates this event is predicted to be part of a background process. There are no background processes simulated in the dataset used for the analysis, so they are all misidentified events of a different event class.

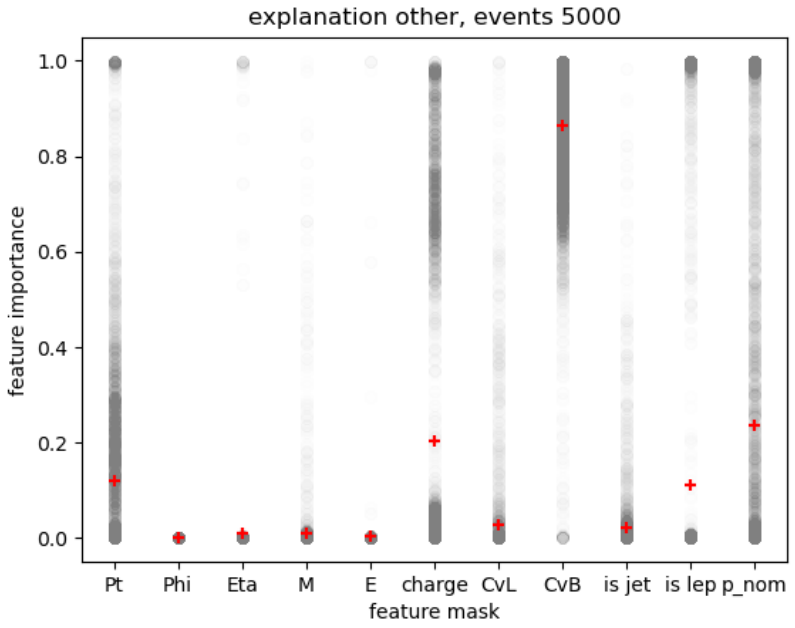



Figure 6.10: Scatter plot of the node feature masks of 5000 events with *other* as predicted class in grey and the mean of each individual feature in red.

This makes it hard to make meaningful observations about them. The top feature by far is the "CvL" tagger, mainly concentrated at events with high importance. A possible noteworthy observation is that in contrast to the other event classes, the feature "charge" plays a larger role in the prediction. The other events for which charge lightly fluctuated by having a few events with the importance around one. The "is lep" feature is concentrated at the extremes, also in contrast to the other event types.

### Summary

A surprising result of the analysis of the explanations is the fluctuations of the feature importance, a feature can have a high importance for one event while being unimportant for another, despite belonging to the same event class. These fluctuations can be very large for example, most of the $ttLF$ events in Fig. 6.3 have an importance value of zero for the invariant mass "M" but also some events with a feature importance of one. These results for "M" are literally the opposite of each other. At this stage of the analysis, the reason for this behaviour is unknown. A possible explanation could be that the GNNExplainer determines which features play a role in predicting a class label and not one of the other possible class labels for the underlying GNN, see section 5.1. For example, a $ttB$ event is characterised by a high "CvL" importance for most events and a low "CvB" importance. This is in stark contrast to the $ttLF$ event class, with most events having a low "CvL" and high "CvB" values, so the multitaggers are a good feature to distinguish the events from each other. Now considering a $ttB$ event in contrast to a $ttH$ event that is characterised by similar importance values for "CvL" and "CvB", the tagging features are now less helpful to distinguish the events from each other. In such a case, other features like the invariant mass "M", the transverse momentum "$p_T$" and others could play a role in distinguishing them. While these features are now assigned a high importance for the $ttB$ vs $ttH$ comparison it is unlikely that they would have the same importance for the comparison $ttB$ vs $ttLF$, causing the fluctuations in the feature importance depending on which event types are most similar to the event in question.

Another reason for the fluctuations could be that in the analysis, only the feature information in the form of the node feature mask is considered, graph structural information is not considered because of the focus on the feature information and the implementation of the GNNExplainer not considering edge attributes, see section 5.2.

Evaluating the results for all events, there is no single feature that stands out as the most important across all event classes. For each event, a combination of several events is impactful in determining the event class. Nonetheless, there are some features that seem to be more important than others, for most events at least one of the heavy flavour tagging values "CvL" and "CvB" maintains a high importance for the prediction. This is in line with expectations that the jet flavour plays an important role in order to find the event type. A $ttLF$ event in Fig. 6.3 is characterised by a light flavour jet resulting in a higher "CvB" importance value and a lower "CvL" importance value. Similarly, a $ttB$ event in Fig. 6.6, characterised by a $b$-flavour jet, possesses a high "CvL" importance for most events and a low "CvB" importance. With these features most likely being important for the prediction by playing a role in determining the flavour of the additional jets, an improvement in the accuracy of the heavy flavour taggers would probably play a greater role in improving the prediction than an improvement in the determination of the invariant mass "M" or the transverse momentum "$\mathrm{p_T}$".

The feature "$\mathrm{p_{nominal}}$" also stands out, generally assuming a high importance for $ttB$ and $ttZB$ events, as well as showing strong fluctuations for $ttH$ events. If this reflects the feature being used to distinguish the additional $b$-jets from the $b$-jets of the top decay, then improving the accuracy of the NLP classification could be crucial in improving the prediction for these events.

The importance of a feature cannot always be meaningfully interpreted in a physical sense. This is for example the case for the features "is jet" and "is lep". It is unclear why they would be influential in the prediction of the event class. The feature "is jet" simply determines if a node is a jet or not, the "is lep" feature does the same for leptons. While it makes sense that these features are used in distinguishing jets from leptons and neutrinos, it is unexpected that this would be more important for some events and less important for other events.

# 7 Conclusion

The goal of this thesis is to gain a better understanding of the underlying decision a Graph Neural Network (GNN) takes in order to reach a prediction for $t\bar{t} + X$ events in proto-proton collisions at the CERN Large Hadron Collider. A better understanding of the decision-making of the GNN could be helpful in improving the event classification. The interpretation of importance values generated by applying the GNNExplainer introduced in chapter 5 on the trained GNN serves as the main tool to archive this objective.

While the importance values of some features confirm physics expectations, such as the heavy flavour jet tagging values "CvL" and "CvB" being an important feature depending on the flavour of the final state object. This is expected because the events are characterised by the additional radiation of a particle $X$. The decay of that particle $X$ results in the additional jets in the dataset. The heavy flavour tagging value "CvL" discriminates charm from light flavour jets, "CvB" discriminates charm from $b$-jets. So, using both heavy flavour tagging values, one can determine the flavour of a jet. With this, a majority of event classes can be distinguished by the flavour of their additional jets. Another expected result is the "$\mathrm{p}_{\mathrm{nominal}}$" feature being important for events with additional $b$-jets. This is expected because the feature "$\mathrm{p}_{\mathrm{nominal}}$" is an NLP score, determining if a jet is an additional jet or not. So it is within expectations that this feature helps in differentiating an additional $b$-jet from a $b$-jet of the top quark decay. Some other features are important depending on the event class. Other feature importance values are hard to interpret in a physics sense, like the feature flags "is jet" and "is lep", which determine if the object in question is a jet or a lepton. Since some of the most important features are the heavy flavour tagging values and the product of an NLP prediction, the prediction of the GNN could be improved by increasing their accuracy. Especially the "$\mathrm{p}_{\mathrm{nominal}}$" feature is interesting because it could aid in the distinction of the $ttB$, $ttH$ and $ttZB$ events, which forms an irreducible background process when trying to investigate the $H \to b\bar{b}$ decay of the $ttH$ events.

One of the most important observations made in this thesis is the fluctuations of the feature importance between events of the same event class. The source of this behaviour is still unknown but it, being present in all event classes, hints at it being a fundamental behaviour of the underlying GNN. An in depth look into how the concrete input features influence the feature mask importance could be helpful in the understanding of the results.

A further look into these results is necessary, especially how the decisions taken by the GNN are influenced by graph structural information and how these interlock with the

feature information examined in this thesis. For this to be possible, the GNNExplainer method might have to be expanded to include edge feature information, which is utilised by the GNN as edge weights. Alternatively, another Explainable AI method could be utilised to complement these results. For example Taylor Coefficient Analysis [28]. This method was originally developed for DNNs, but can also be implemented for GNNs, as utilised in [7].

# Bibliography


[1] The ATLAS Collaboration. "Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC". In: *Physics Letters B* 716.1 (2012), pp. 1–29. ISSN: 0370-2693. DOI: https://doi.org/10.1016/j.physletb.2012.08.020.

[2] F. Abe et al. "Observation of Top Quark Production in $\overline{p}p$ Collisions with the Collider Detector at Fermilab". In: *Phys. Rev. Lett.* 74 (14 Apr. 1995), pp. 2626–2631. DOI: 10.1103/PhysRevLett.74.2626.

[3] S. Abachi et al. "Observation of the Top Quark". In: *Phys. Rev. Lett.* 74 (14 Apr. 1995), pp. 2632–2637. DOI: 10.1103/PhysRevLett.74.2632.

[4] E. Pfeffer. "Studies on tt+bb production at the CMS experiment". MA thesis. Karlsruhe Institute of Technology (KIT), 2021.

[5] T. Halenke. "Studien zu Graph Neural Networks in tt + bb- Prozessen am CMS-Experiment". BA Thesis. Karlsruhe Institute of Technology (KIT), 2021.

[6] C. Wolter. "Bayesian Optimization of Graph Neural Networks with Hypergraph Inputs in tt + bb Events at the CMS Experiment". BA thesis. Karlsruhe Institute of Technology (KIT), 2022.

[7] Y. C. Cung. "Feasibility and Reliability Studies of Graph Neural Networks for Multivariate tt+X Event Classification at the CMS Experiment at CERN". MA thesis. Karlsruhe Institute of Technology (KIT), 2022.

[8] D. J. Griffiths. *Introduction to Elementary Particles.* Second, Revised Edition. Physics textbook. Weinheim: Wiley-VCH, 2008. ISBN: 978-3-527-40601-2.

[9] *Standard-Modell der Elementarteilchen.* 2010. URL: https://commons.wikimedia.org/wiki/File:Standard_Model_of_Elementary_Particles-de.svg.

[10] P. Zyla et al. (Particle Data Group), Prog. Theor. Exp. Phys. 2020, 083C01 (2021) and 2021 update. URL: https://pdg.lbl.gov/2021/reviews/rpp2021-rev-top-quark.pdf.

[11] M. Worek. "Differential top pair cross section and top anti-top plus jets Physics". In: *Journal of Physics: Conference Series* 452.1 (July 2013), p. 012033. DOI: 10.1088/1742-6596/452/1/012033.

[12] R. Workman et al. (Particle Data Group), Prog. Theor. Exp. Phys. 2022, 083C01 (2022). URL: https://pdg.lbl.gov/2023/reviews/rpp2022-rev-higgs-boson.pdf.

[13] R. L. Workman et al. (Particle Data Group) Prog.Theor.Exp.Phys. 2022, 083C01 (2022) and 2023 update. URL: https://pdg.lbl.gov/2023/listings/rpp2023-list-z-boson.pdf.

[14] A. Sirunyan et al. “Measurements of $t\bar{t}$ cross sections in association with b jets and inclusive jets and their ratio using dilepton final states in pp collisions at s=13TeV”. In: *Physics Letters B* 776 (2018), pp. 355–378. ISSN: 0370-2693. DOI: https://doi.org/10.1016/j.physletb.2017.11.043. URL: https://www.sciencedirect.com/science/article/pii/S0370269317309358.

[15] E. Bols et al. “Jet flavour classification using DeepJet”. In: *Journal of Instrumentation* 15.12 (Dec. 2020), P12012. DOI: 10.1088/1748-0221/15/12/P12012.

[16] L. Evans and P. Bryant. “LHC Machine”. In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08001. DOI: 10.1088/1748-0221/3/08/S08001.

[17] The CMS Collaboration et al. “The CMS experiment at the CERN LHC”. In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08004. DOI: 10.1088/1748-0221/3/08/S08004.

[18] D. Barney. “CMS Detector Slice”. CMS Collection. 2016. URL: https://cds.cern.ch/record/2120661.

[19] R. Franceschini et al. *Kinematic Variables and Feature Engineering for Particle Phenomenology*. 2022. arXiv: 2206.13431 [hep-ph].

[20] C. C. Aggarwal. *Neural Networks and Deep Learning: A Textbook*. 2nd ed. 2023. Cham: Springer International Publishing, 2023. ISBN: 9783031296420. URL: https://doi.org/10.1007/978-3-031-29642-0.

[21] P. Hartmann. *Mathematik für Informatiker. Ein praxisbezogenes Lehrbuch. 6th ed.* Springer Vieweg Wiesbaden, 2015. ISBN: 978-3-658-03415-3. DOI: 0.1007/978-3-658-03416-0.

[22] R. J. Wilson. *Introduction to Graph Theory*. Fourth Edition. Addison Wesley Longman Limited, 1996. ISBN: 0-582-24993-7.

[23] W. L. Hamilton. “Graph Representation Learning”. In: *Synthesis Lectures on Artificial Intelligence and Machine Learning* 14.3 (2020), pp. 1–159. URL: https://www.cs.mcgill.ca/~wlh/grl_book/.

[24] *Complete graph K7*. wikimedia. URL: https://commons.wikimedia.org/wiki/File:Complete_graph_K7.svg.

[25] R. Ying et al. *GNNExplainer: Generating Explanations for Graph Neural Networks*. In: Advances in Neural Information Processing Systems (NeurIPS), Vol. 32. Ed. by H. Wallach et al., 2019. URL: https://arxiv.org/pdf/1903.03894.pdf.

[26] M. Fey and J. E. Lenssen. “Fast Graph Representation Learning with PyTorch Geometric”. In: *ICLR Workshop on Representation Learning on Graphs and Manifolds*. 2019.

[27] A. Paszke et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[28] S. Wunsch, R. Friese, R. Wolf, et al. “Identifying the Relevant Dependencies of the Neural Network Response on Characteristics of the Input Space.” In: *Comput Softw Big Sci 2* 5 (2018). DOI: https://doi.org/10.1007/s41781-018-0012-1.