ETP-Bachelor-KA/2021-13

Karlsruher Institut für Technologie

Studien zu Graph Neural Networks in $t\bar{t} + b\bar{b}$-Prozessen am CMS-Experiment

Bachelor Thesis

Timo Halenke

An der Fakultät für Physik
Institut für Experimentelle Teilchenphysik

Erstgutachter:	Prof. Dr. U. Husemann
Zweitgutachter:	Dr. Michael Waßmer
Betreuender Mitarbeiter:	Emanuel Pfeffer

Karlsruhe, 18. Oktober 2021

Diese Arbeit wurde vom Erstgutachter der Bachelorarbeit akzeptiert.

Karlsruhe, den 18. Oktober 2021

.....

(**Prof. Dr. U. Husemann**)

Ich versichere wahrheitsgemäß, die Arbeit selbstständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde.

Karlsruhe, den 18. Oktober 2021

.....

(**Timo Halenke**)

Inhaltsverzeichnis

1 Einleitung

Im Verlauf des 20. Jahrhunderts haben verschiedene Experimente bestätigt, was schon in der Antike vor über tausend Jahren vermutetet wurde: Alle Materie im Universum besteht aus unteilbaren, elementaren Teilchen. In den 70er Jahren wurde hierfür eine Theorie entwickelt, die diese Elementarteilchen und die Funktionsweise von drei der vier fundamentalen Wechselwirkungen zwischen ihnen beschreibt: das Standardmodell der Teilchenphysik (SM). Heute gilt das SM als eine wohl getestete und anerkannte Theorie, die viele experimentelle Befunde einwandfrei beschreiben kann.

Dennoch ist eine genaue Untersuchung des SMs auch heute von großer Bedeutung: Zum einen beinhaltet das SM viele Parameter, die experimentell bestimmt werden müssen und zum anderen ist der Wissenschaft bewusst, dass das SM trotz seiner bisherigen Erfolge keine vollständige Beschreibung des Universums ermöglicht. Für diese Untersuchungen und der Jagd nach neuer Physik startete im Jahr 1984 die Planung für den Bau des größten Teilchenbeschleunigers der Welt: dem Large Hadron Collider (LHC), der im Jahre 2008 seine erste Protonenkollision erzeugte. Die Untersuchung solcher Teilchenkollisionen stellt eine wichtige Aufgabe in der Teilchenphysik dar, um weitere Einblicke in die Natur der Elementarteilchen zu erhalten. Hierfür werden häufig Methoden aus dem Bereich des maschinellen Lernens genutzt, die in der Vergangenheit sehr gute Ergebnisse erzielen konnten.

In den letzten Jahren hat sich das Feld der künstlichen Intelligenz stets weiterentwickelt und neue Methoden und Verfahren hervorgebracht. Ein noch recht junger Zweig im Bereich des maschinellen Lernens sind sogenannte Graph Neural Networks (GNNs). Dabei handelt es sich um einen speziellen Typ von neuronalen Netzen (NN), die auf Graphen operieren und bereits in vielen Bereichen signifikante Verbesserungen gegenüber herkömmlichen neuronalen Netzen erzielt haben.

Daher beschäftigt sich diese Arbeit mit Graph Neural Networks und der Analyse von Proton-Proton-Kollisionen am LHC. Genauer gesagt, untersucht diese Arbeit Methoden, mit denen die zusätzlichen Jets mit B-Hadronen im $t\bar{t} + b\bar{b}$-Prozess von anderen Jets im Ereignis getrennt werden können. Frühere Arbeiten haben für diese Aufgabe bereits Deep Neural Networks (DNNs) verwendet [1]. Da GNNs ein vielversprechender, neuer Zweig im Bereich des maschinellen Lernens sind, soll diese Arbeit erörtern, ob GNNs besser für dieses Klassifikationsproblem geeignet sind. Hierfür werden in Kapitel 2 zunächst die nötigen physikalischen Grundlagen besprochen und in Kapitel 3 die Grundlagen der Graphentheorie

sowie die Funktionsweise von GNNs diskutiert. In Kapitel 4 erfolgt schließlich die Anwendung und die Untersuchung verschiedener GNN-Modelle. Abschließend werden in Kapitel 5 weiterführende, physikalisch motivierte Untersuchungen vorgenommen. Ebenso erfolgt in diesem Kapitel dann eine finale Auswertung und der Vergleich mit den Ergebnissen der bisherigen Methode mittels DNNs.

2 Physikalischer Hintergrund

Kapitel 2 befasst sich mit den physikalischen Grundlagen, die für das Verständnis dieser Arbeit von Vorteil sind. Hierfür wird in Abschnitt 2.1 zunächst eine Einführung in das Standardmodell der Teilchenphysik gegeben und anschließend in 2.2 die experimentelle Umgebung vorgestellt. In Abschnitt 2.3 wird kurz die Rekonstruktion physikalischer Objekte aus den Messdaten des Detektors und die Identifikation von Jets mit B-Hadronen besprochen. Abschließend wird in Abschnitt 2.4 der für diese Arbeit relevante $\mathrm{t\bar{t}} + \mathrm{b\bar{b}}$-Prozess vorgestellt, sowie in Abschnitt 2.5 auf die Simulation solcher Prozesse und die in dieser Arbeit angewendeten Selektionen eingegangen.

2.1 Standardmodell der Teilchenphysik

Das Standardmodell der Teilchenphysik umfasst alle bisher bekannten Elementarteilchen und beschreibt die wichtigsten Wechselwirkungen zwischen ihnen: die elektromagnetische, die schwache und die starke Wechselwirkung. Der folgende Überblick zum SM basiert, wenn nicht anders angegeben, auf [2] und [3].

Das SM umfasst eine Vielzahl von Teilchen, die sich in verschiedene Unterkategorien einteilen lassen. Zu diesen Kategorien gehören zunächst die Fermionen (Leptonen und Quarks), die einen halbzahligen Spin haben und die Grundbausteine aller bisher bekannten Materie bilden. Die Bosonen besitzen einen ganzzahligen Spin und fungieren als Austausch-Teilchen und vermitteln die Kräfte zwischen den Fermionen. Eine Übersicht zum SM ist in Abb. 2.1 gegeben.

Zu den Fermionen gehören zunächst sechs Leptonen, die durch ihre elektrische Ladungsquantenzahl Q und ihre Flavour-Quantenzahl $L_{e,\mu,\tau}$ klassifiziert werden. Diese sind das Elektron e, Myon μ und das Tauon τ mit negativer Ladung $Q = -1\,e$, sowie ihre elektrisch neutralen Pendants: das Elektron-Neutrino ν_e, Myon-Neutrino ν_μ und das Tau-Neutrino ν_τ. Zu jedem Lepton existiert auch ein zugehöriges Antilepton mit entgegengesetzten Quantenzahlen, womit es tatsächlich 12 Leptonen gibt.

Ebenfalls zu den Fermionen gehören sechs Quarks, die sich durch ihre elektrische Ladung und fünf Quark-Flavour kategorisieren lassen: Isospin (für Up- und Down-Quarks), Charm, Strangeness, Topness und Bottomness. Das Up-, Charm- und Top-Quark (u, c, t) besitzen eine elektrische Ladung von $Q = +2/3\,e$ und für das Down-, Strange- und Bottom-Quark (d, s, b) gilt $Q = -1/3\,e$. Auch hier existiert zu jedem Quark ein Antiquark mit entgegengesetzten Quantenzahlen, womit es insgesamt auch 12 Quarks gibt.

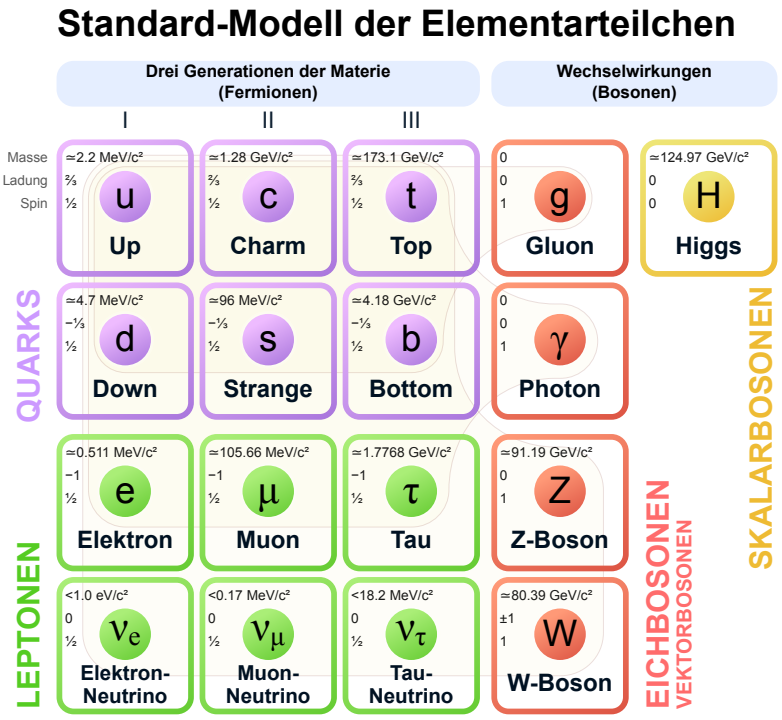


Abbildung 2.1: Standardmodell der Teilchenphysik, entnommen aus [4]: Das SM kategorisiert alle bisher bekannten Elementarteilchen und unterteilt diese in Fermionen und Bosonen. Zu den Fermionen gehören die Quarks und Leptonen, die sich nochmals in drei Generationen aufteilen lassen. Die Eichbosonen vermitteln die Wechselwirkungen zwischen den Elementarteilchen. Durch den Higgs-Mechanismus erhalten die Elementarteilchen ihre Masse.

Eine Besonderheit der Quarks ist, dass sie zusätzlich noch eine weitere Quantenzahl, die Farbladung rot, blau oder grün und Antiquarks eine Antifarbladung antirot, antiblau oder antigrün besitzen. Aufgrund des QCD-Confinments der starken Wechselwirkung, das nur farbneutrale Objekte erlaubt, können Quarks nicht als freie Teilchen auftreten (Verbot von isolierten Farbladungen). Dies führt zur sogenannten Hadronisierung, ein Prozess, bei dem sich einzelne Quarks und Gluonen anlagern und zusammen farbneutrale Hadronen (Mesonen und Baryonen) bilden. Bei Kollisionsprozessen in Teilchenbeschleunigern kann es vorkommen, dass bei hohen Energien die Quarks solcher Hadronen auseinandergerissen und voneinander getrennt werden. Dabei steigt die Wechselwirkungsenergie immer weiter an, bis sich ein neues Quark-Antiquark Paar bildet. Dieser Hadronisierungsprozess wiederholt sich mehrmals, wobei die dabei neu entstehenden Hadronen in etwa die gleiche Flugrichtung wie das ursprüngliche Teilchen aufweisen. Dieser gebündelte Strahl an Teilchen wird auch als Jet bezeichnet.

Die Bosonen vermitteln die Wechselwirkungen zwischen den Fermionen. Die Vektorbosonen besitzen einen Spin von 1 und vermitteln die starke, schwache und elektromagnetische Wechselwirkung. Das Gluon g dient als Vermittler der starken Wechselwirkung, welche auf alle Teilchen mit einer Farbladung wirkt. Das Gluon selbst besitzt ebenfalls eine Farbladung, bestehend aus einer Farbe und einer Antifarbe. Aufgrund der Symmetriegruppe SU(3) der starken Wechselwirkung gibt es insgesamt acht verschiedene Gluonen. Die schwache Wechselwirkung wirkt auf alle Leptonen und Quarks und wird durch das $W^{\pm}$-Boson und das Z^0-Boson vermittelt. Während das Z^0 elektrisch neutral ist, kann das $W^{\pm}$ sowohl positiv

als auch negativ geladen sein. Die elektromagnetische Wechselwirkung wird durch das Photon γ vermittelt und wirkt auf alle elektrisch geladenen Teilchen. Insgesamt klassifiziert das SM also 12 Vektorbosonen.

Als letztes Teilchen wurde das Higgs-Boson dem SM hinzugefügt. Es wurde bereits im Jahr 1964 unter anderem von Peter W. Higgs vorhergesagt [5] und konnte im Jahr 2012 am Large Hadron Collider nachgewiesen werden [6] [7]. Der Higgs-Mechanismus verleiht allen massebehafteten Elementarteilchen ihre Masse. Dies geschieht durch die Wechselwirkung mit dem Higgs-Feld, welches als Konsequenz die Existenz eines Higgs-Bosons H fordert. Das Higgs-Boson ist das einzige Skalarboson im SM und besitzt einen Spin von 0.

Das Standardmodell ist heute Basis aktueller Forschung. Es wurde in den vergangenen Jahren umfangreich getestet und kann viele Phänomene hervorragend erklären. Dennoch ist klar, dass das SM in seiner bisherigen Form nicht vollständig, und lediglich Teil einer noch umfassenderen Theorie ist. So findet zum Beispiel die vierte fundamentale Kraft, die Gravitation, im SM keine Berücksichtigung. Ebenso können die Phänomene Dunkle Materie und Dunkle Energie mit dem bisherigen SM nicht erklärt werden.

2.2 Das CMS-Experiment

2.2.1 Large Hadron Collider

Der Large Hadron Collider (LHC) ist mit einem Umfang von fast 27 km der größte Teilchenbeschleuniger der Welt und befindet sich am CERN, der europäischen Organisation für Kernforschung, in der Nähe von Genf in der Schweiz [8]. Er ist das letzte Element in einer Reihe von Beschleunigern und besitzt zwei Stahlrohre, in denen ein extrem starkes Vakuum herrscht. Die Protonen und Ionen durchlaufen die einzelnen Beschleuniger und werden dabei auf immer höhere Energien gebracht, bis sie anschließend in den LHC geleitet werden. Dort werden sie in den Hochfrequenz-Kavitäten weiter beschleunigt und erreichen ihre maximale Geschwindigkeit nahe der Lichtgeschwindigkeit. Mit Hilfe von supraleitenden Elektromagneten werden die Protonen und Ionen um den LHC-Ring geführt und anschließend zur Kollision gebracht. Der LHC ist ein wichtiger experimenteller Bestandteil aktueller Grundlagenforschung mit den Zielen das SM besser zu verstehen und Hinweise auf neue Physik jenseits des SMs zu entdecken. Hierfür werden hochenergetische Kollisionsprozesse mit Schwerpunktsenergien von bis zu $\sqrt{s} = 14\,\text{TeV}$ von vier verschiedenen Experimenten ausgewertet: ATLAS [9], ALICE [10], LHCb [11] und CMS [12]. Jedes dieser Experimente ist durch seine speziellen Detektoren charakterisiert. In Abb. 2.2 ist ein Überblick zum CERN-Beschleunigerkomplex gegeben.

2.2.2 Der CMS-Detektor

Das CMS-Experiment (Compact Muon Solenoid) am LHC untersucht unter anderem das Standardmodell und das Higgs-Boson und sucht nach möglichen Teilchen, die dunkle Materie ausmachen könnten. Es befindet sich ca. 100 m unter der Erde nahe dem französischen Dorf Cessy am Genfer See. Seinen Namen erhielt das Experiment aufgrund seiner kompakten Größe von $21\,\text{m} \times 15\,\text{m} \times 15\,\text{m}$, seiner herausragenden Auflösung von Myonenspuren und seinem starken, supraleitenden Solenoid-Magneten. Der CMS-Detektor besteht aus verschiedenen Schichten, dargestellt in Abb. 2.3, mit denen verschiedene Endprodukte der Kollisionsprozesse gemessen werden können. Diese Zusammenfassung basiert auf [12].

Der **Silizium-Spurdetektor** bildet die innerste Schicht des CMS-Detektors und befindet sich somit am nächsten zum Kollisionspunkt. Er besteht aus mehreren Schichten Silizium-Streifen-Detektoren und Silizium-Pixel-Detektoren. Seine Aufgabe ist es, die Trajektorie

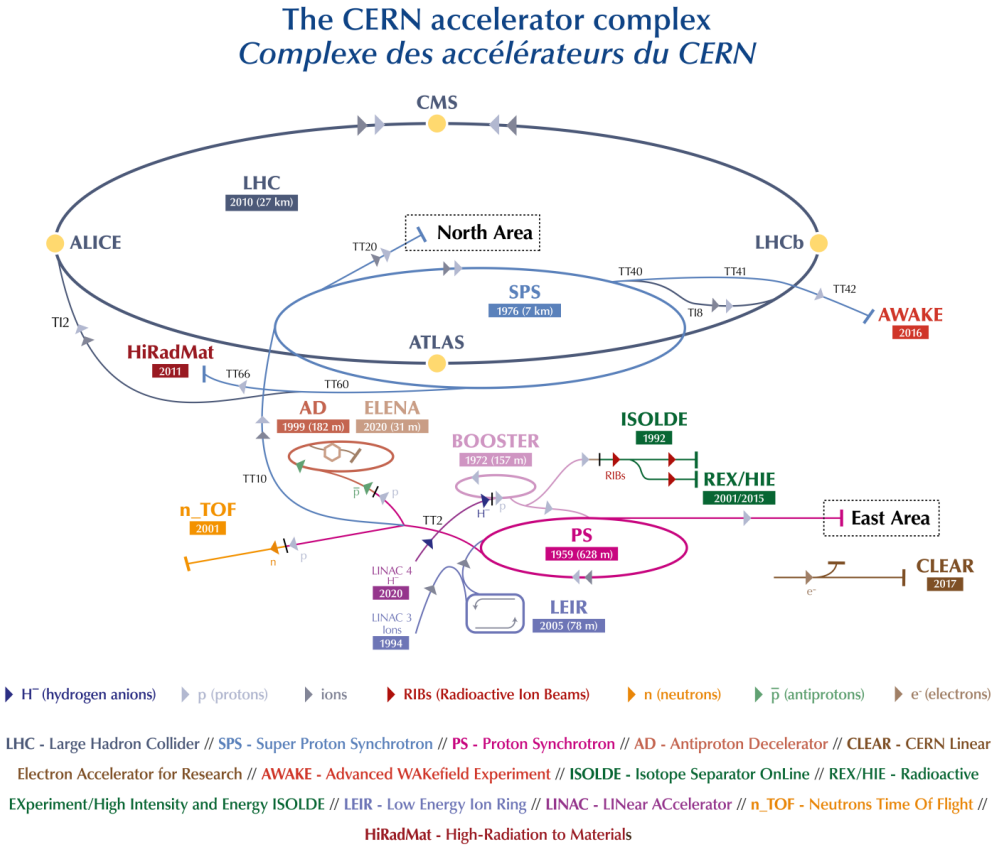


Abbildung 2.2: Der CERN-Beschleunigerkomplex, entnommen aus [13]: Der LHC (dunkelblau) ist das letzte Glied in einer Kette von mehreren Beschleunigern. Die gelben Punkte stellen die vier Experimente ATLAS, ALICE, LHCb und CMS dar, in denen die hochenergetischen Kollosionsereignisse detektiert werden.

geladener Hadronen und Leptonen zu bestimmen. Mit Hilfe eines starken Magnetfeldes werden die geladenen Teilchen von ihrer ursprünglichen Trajektorie abgelenkt, wodurch die Ladung und der Transversalimpuls der Teilchen bestimmt werden können.

Das **elektromagnetische Kalorimeter** (ECAL) wird dazu genutzt, die Energien von Elektronen/Positronen und Photonen zu rekonstruieren. Dies geschieht mittels elektromagnetischer Schauer, die bei hohen Energien hauptsächlich durch Bremsstrahlung und Positron-Elektron-Paarbildung entstehen.

Das **hadronische Kalorimeter** (HCAL) misst die Energien von Teilchen, die der starken Wechselwirkung unterliegen und spielt somit eine wichtige Rolle bei der Vermessung der Teilchenjets. Dies geschieht ebenfalls über Schauerentwicklung, womit die Energien von geladenen und neutralen Hadronen gemessen werden können.

Der **supraleitende Solenoid-Magnet** bildet das Herz des Detektors und umschließt die bisher genannten Schichten. Seine Aufgabe ist die Erzeugung der bereits erwähnten, starken magnetischen Felder mit magnetischen Flussdichten von bis zu 4 T.

Die **Myonenkammern** bilden das letzte Glied im CMS-Detektor und sind verantwortlich für die Detektion der Myonen. Ein Großteil der Elektronen und Photonen geben bereits im ECAL ihre Energie vollständig durch Paarbildung und Bremsstrahlung ab. Auch Hadronen

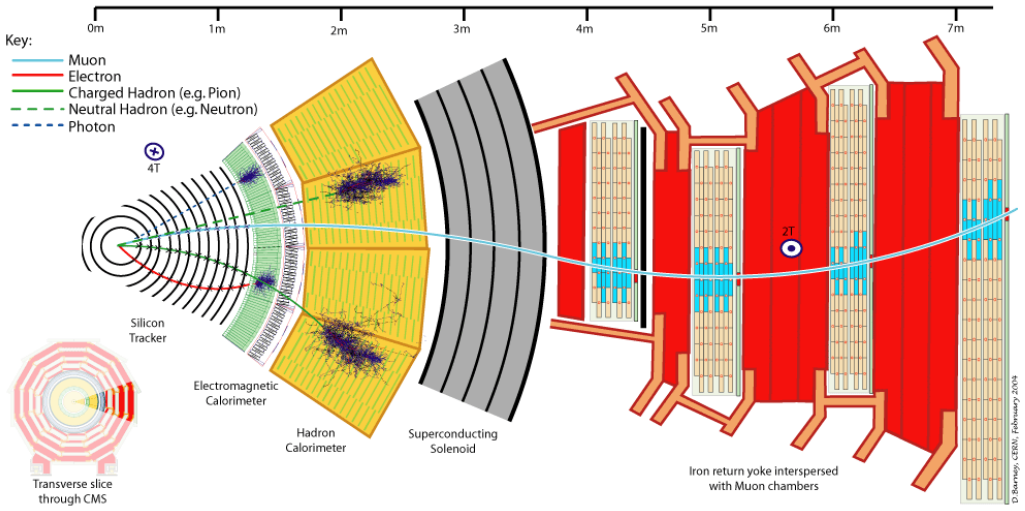


Abbildung 2.3: Querschnitt eines Ausschnitts des CMS-Detektors, entnommen aus [14]: Der CMS-Detektor besteht aus verschiedenen Schichten, die jeweils auf die Vermessung bestimmter Teilchen spezialisiert sind.

werden beinahe vollständig im HCAL absorbiert. Die Myonen hingegen durchlaufen die vorherigen Schichten beinahe interaktionslos und werden in den Gasionisationskammern detektiert.

Die verschiedenen Schichten des CMS-Detektors erlauben die Vermessung vieler physikalischer Kollisionsprozesse. Dabei ist die Auflösung des CMS-Detektors allerdings begrenzt, wodurch es zu einer gewissen Verschmierung der Messdaten kommt.

2.2.3 Kinematische Größen im CMS-Experiment

Das Koordinatensystem des CMS-Detektors hat seinen Ursprung in der Mitte des Detektors, dem Kollisionspunkt der Teilchen. Die x-Achse zeigt radial ins Zentrum des LHCs und die y-Achse steht senkrecht auf dieser und zeigt nach oben. Die x, y-Ebene wird auch als Transversalebene bezeichnet. Die z-Achse steht senkrecht auf der Transversalebene und zeigt entlang der Strahlrichtung entgegen dem Uhrzeigersinn. Aufgrund des zylindrischen Aufbaus des CMS-Detektors und der Rotationsinvarianz in der Transversalebene von Teilchen-Kollisionen, liegt die Beschreibung in zylindrischen Koordinaten nahe. Dabei ist ϕ der Azimutwinkel in der Transversalebene und wird von der x-Achse aus gemessen. Der Polarwinkel θ wird von der z-Achse aus gemessen. Der radiale Abstand zum Kollisionspunkt in der Transversalebene wird mit r bezeichnet.

Da der Impuls der beiden einlaufenden Teilchen entlang der z-Achse von null verschieden ist, bewegt sich das Ruhesystem der Kollision entlang dieser Achse. Daher ist es sinnvoll, transversale Größen in der x, y-Ebene, wie etwa den Transversalimpuls zu definieren:

$$p_{\mathrm{T}} = \sqrt{p_x^2 + p_y^2}. \tag{2.1}$$

Die Pseudorapidität ist definiert als

$$\eta = -\ln \tan\left(\frac{\theta}{2}\right) \tag{2.2}$$

und stellt ein Maß für die Flugrichtung eines Teilchens dar. Kleine Werte von η beschreiben eine Flugrichtung senkrecht zur z-Achse, wohingegen große Werte eine zur z-Achse geneigte Flugrichtung beschreiben. Der geometrische Abstand zweier Teilchen in der η, ϕ-Ebene ist definiert als

$$\Delta R = \sqrt{\Delta\eta^2 + \Delta\phi^2}. \tag{2.3}$$

Obwohl bestimmte Teilchen wie das Neutrino oder andere exotische Teilchen die Detektoren des CMS-Experiments interaktionslos durchlaufen, können dennoch Rückschlüsse auf deren Existenz gezogen werden. Dies geschieht mittels der fehlenden Transversalenergie, auch MET (engl.: Missing Transverse Energy) genannt. Es kann angenommen werden, dass der Transversalimpuls der Teilchen vor der Kollision vernachlässigbar klein ist. Daher müssen sich aufgrund der Impulserhaltung die Vektoren der Transversalimpulse nach der Kollision zu null aufaddieren. Die fehlende Transversalenergie ergibt sich also als Betrag der Summe der gemessenen Transversalimpulse $\boldsymbol{p}_{\mathrm{T},i}$:

$$\not{E}_{\mathrm{T}} = \left| \sum_i \boldsymbol{p}_{\mathrm{T},i} \right|. \tag{2.4}$$

2.3 Objektrekonstruktion

Bei der Vermessung verschiedener Kollisionsprozesse am LHC entstehen jährlich riesige Datenmengen im Petabyte-Bereich. Diese müssen zur weiteren Analyse zunächst in physikalische Objekte übersetzt werden. Dieser Prozess wird auch Objektrekonstruktion genannt. Dies geschieht am CMS-Experiment mittels verschiedener Algorithmen, die die Signale der einzelnen Detektorschichten (s. Abschnitt 2.2.2) verarbeiten.

Der Particle-Flow-Algorithmus [15] ist verantwortlich für die Rekonstruktion der Teilchen und nutzt hierfür die Informationen aus den verschiedenen Detektorschichten. Durch die Berechnung des Ursprungsorts der Teilchen können diese dem jeweiligen Kollisionsprozess zugeordnet werden. Mit Hilfe des anti-k_{T}-Algorithmus [16] werden dann Teilchen mit ähnlicher Flugrichtung zu Jets zusammengefasst.

Eine weitere wichtige Rolle spielt der Deep-Jet-Algorithmus [17], welcher für die Identifikation von Jets zuständig ist, die aus einem Bottom-Quark entstanden sind. Solche Jets lassen sich von anderen Jets unterscheiden, unter anderem da B-Hadronen eine längere Lebensspanne als andere Hadronen aufweisen und somit weiter entfernt vom Kollisionspunkt zerfallen. Durch verschiedene Algorithmen kann somit jedem Jet ein b-tag-Wert $\tilde{b}$ zwischen 0 und 1 zugewiesen werden, der seine b-Artigkeit beschreibt. Dies wird auch als „B-tagging“ bezeichnet. In dieser Arbeit werden Jets, für die $\tilde{b} \geq 0{,}277$ gilt, als „b-tagged“ Jets bezeichnet.

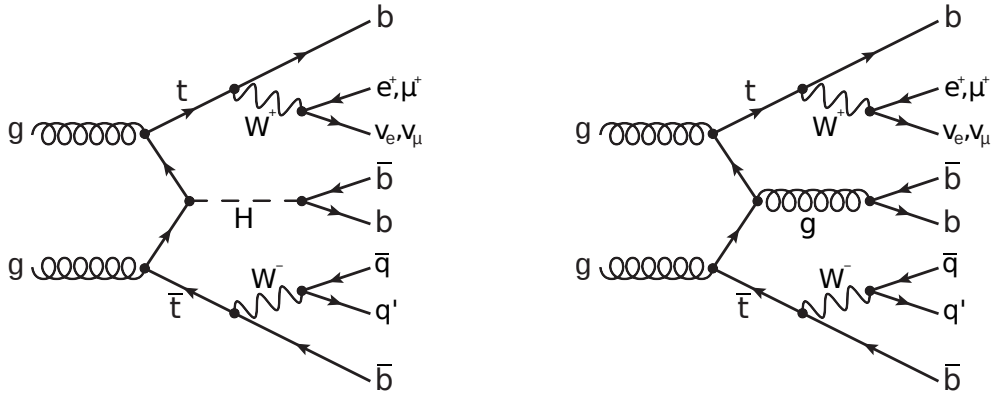
2.4 Der $\mathrm{t\bar{t}} + \mathrm{b\bar{b}}$-Prozess

Im Zentrum dieser Arbeit steht die Untersuchung des $\mathrm{t\bar{t}} + \mathrm{b\bar{b}}$-Prozess, welcher sich durch die Entstehung eines Gluons in assoziierter Produktion eines Paares aus Top-Quark und Top-Antiquark auszeichnet. Der Prozess entsteht unter anderem bei Proton-Proton-Kollisionen mit einer Schwerpunktsenergie von $\sqrt{s} = 13\,\mathrm{TeV}$ am CMS-Experiment [18]. Das Top-Quark zerfällt über die schwache Wechselwirkung in ein W^+-Boson und ein Bottom-Quark. Auf analoge Weise zerfällt auch das Top-Antiquark. Das W-Boson kann nun

hadronisch oder leptonisch zerfallen. Im Rahmen dieser Untersuchung liegt der Fokus auf dem semileptonischen Zerfallskanal des $t\bar{t} + b\bar{b}$-Prozess. Dabei zerfällt ein W-Boson leptonisch in ein Lepton und das zugehörige Neutrino. Das andere W-Boson zerfällt hadronisch in ein Quark-Antiquark Paar. Das Gluon selbst wandelt sich in zwei zusätzliche Bottom-Quarks um, welche in dieser Arbeit eine wichtige Rolle spielen werden. Hierbei handelt es sich lediglich um die Beschreibung des Prozesses in führender Ordnung. Durch Korrekturen höherer Ordnung können auch weitere Teilchen abgestrahlt werden. Die Jet-Multiplizität beschreibt die Anzahl der Jets in einem Kollisionsereignis.

Die Untersuchung des $t\bar{t} + b\bar{b}$-Prozess ist aus mehreren Gründen interessant. Zum einen finden der Prozess $pp \to t\bar{t}$ und die Produktion der beiden B-Jets auf unterschiedlichen Energieskalen statt, was zu großen Unsicherheiten bei der Berechnung und Simulation verschiedener Parameter führt [18]. Zum anderen stellt der $t\bar{t} + b\bar{b}$-Prozess einen der wichtigsten Untergrundprozesse des $t\bar{t}H(b\bar{b})$-Prozesses dar. Dieser ist dem $t\bar{t} + b\bar{b}$-Prozess sehr ähnlich und zerfällt in die gleichen Endprodukte. Der einzige Unterschied liegt in der Entstehung eines Higgs-Bosons mit assoziierter Produktion eines Paares aus Top-Quark und Top-Antiquark. Eine beispielhafte Darstellung der beiden Prozesse ist in Abb. 2.4 gegeben.

Eine genaue Untersuchung des $t\bar{t}H(b\bar{b})$-Prozesses und dem dabei entstehenden Higgs-Boson ist von zentraler Bedeutung für die Teilchenphysik. So ist dieser Prozess sehr gut für die Vermessung der Top-Higgs-Yukawa-Kopplung geeignet, was einen wichtigen Test des SMs darstellt. Für die Durchführung solcher Analysen ist es essenziell, den $t\bar{t}H(b\bar{b})$-Prozess von seinen Untergrundprozessen verlässlich trennen zu können. Da der einzige Unterschied der beiden Prozesse im Ursprung der zusätzlichen B-Jets liegt, ist die genauere Betrachtung dieser Jets der Schlüssel zum besseren Verständnis des $t\bar{t} + b\bar{b}$-Prozesses.



(a) $t\bar{t}H(b\bar{b})$-Prozess (b) $t\bar{t} + b\bar{b}$-Prozess

Abbildung 2.4: Beispielhafte Feynmandiagramme des $t\bar{t}H(b\bar{b})$- und des $t\bar{t} + b\bar{b}$-Prozesses in führender Ordnung und im semileptonischen Zerfallskanal, entnommen aus [1].

2.5 Ereignissimulation und Selektion

Zur Auswertung der Kollisionsprozesse im CMS-Detektor werden verschiedene Methoden aus dem Bereich des maschinellen Lernens angewandt. Die Entwicklung solcher Methoden erfolgt mit Hilfe von simulierten Daten aus Monte-Carlo-Ereignisgeneratoren. Die in dieser Arbeit verwendeten Simulationen entstammen den Generatoren POWHEG [19] und PYTHIA 8.2 [20]. Dabei werden simulierte Datensätze, die den Datennahmebedingungen des

Jahres 2018 entsprechen, verwendet. Bei der Simulation wird zwischen zwei verschiedenen Ansätzen unterschieden. Der wichtigste Unterschied dieser beiden Simulationsansätze liegt in der Herkunft der zusätzlichen Bottom-Quarks und der Berechnung der Matrixelemente, die entweder $t\bar{t}$- oder $t\bar{t} + b\bar{b}$-Produktion betrachten [21]. Je nach Ansatz können die Simulationsdaten also in ein $t\bar{t}$-Sample und ein $t\bar{t} + b\bar{b}$-Sample unterteilt werden, was in Abschnitt 5.3 für den finalen Vergleich mit DNNs wichtig sein wird. Die Simulation der Kollisionsprozesse ist in drei Level unterteilt. Das erste Level bilden die Monte-Carlo-generierten Partonen des Kollisionsprozesses, welche zunächst zu physikalischen Teilchen (Parton Shower) und dann zu Jets zusammengefasst werden. Diese Jets werden auch als Generator-Jets bezeichnet. Die wahre Information darüber, welche Jets welchen Teilchen zuzuordnen sind, ist auf diesem Level noch vorhanden. Das Generator-Level ist aber von theoretischer Natur und kann nicht mit realen Daten verglichen werden. Dies ändert sich im Rekonstruktions-Level, bei dem Detektoreffekte wie Verschmierung von Jets berücksichtigt werden.

Auf Rekonstruktions-Level liegt allerdings, wie bei realen Messdaten von Kollisionsereignissen, keine Information mehr über den Ursprung eines Jets vor. Allerdings ist es möglich diese Information vom Generator-Level auf das Rekonstruktions-Level zu übertragen. Liegen zwei Jets aus dem Generator-Level und dem Rekonstruktions-Level genügend nahe beieinander ($\Delta R \leq 0{,}4$), so kann angenommen werden, dass es sich um den gleichen Jet handelt. Auf diese Weise lassen sich $t\bar{t} + b\bar{b}$-Prozesse simulieren, bei denen die Information über den Ursprung eines Jets vorliegt. Abbildung 2.5 zeigt die Bezeichnung der verschiedenen Kategorien, in denen die Jets in dieser Arbeit eingeteilt werden. Die beiden zusätzlichen Jets mit B-Hadronen, auf denen der Fokus dieser Arbeit liegt, werden im Folgenden auch als „AddB-Jets“ (engl.: Additional B-Jets) bezeichnet. Weitere Jets, die in diesem Prozess auftreten können, werden in die Kategorie „Unbekannt“ eingeteilt.

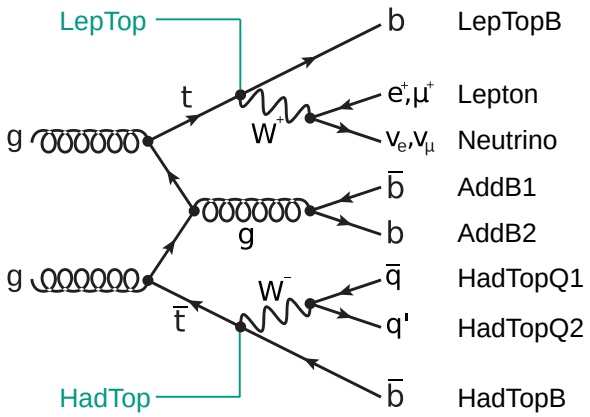


Abbildung 2.5: Jet-Kategorien im $t\bar{t} + b\bar{b}$-Prozess, verändert entnommen aus [1]: Bezeichnung der einzelnen Jet-Kategorien. Weitere Jets, die im Prozess auftreten können werden in die Kategorie Unbekannt eingeteilt.

Die in dieser Arbeit betrachteten Simulationsdaten unterliegen folgenden Selektionskriterien: Zum einen werden nur Ereignisse berücksichtigt, die auf Generator-Level dem $t\bar{t} + b\bar{b}$-Prozess entsprechen und somit genau zwei AddB-Jets besitzen, die jeweils ein B-Hadron enthalten. Ebenso müssen die Ereignisse auf Rekonstruktions-Level mindestens sechs Jets aufweisen, wovon mindestens vier Jets einen b-tag-Wert größer als 0,277 aufweisen.

Weiterhin werden zum Training in Kapitel 4 nur Ereignisse verwendet, bei denen die beiden AddB-Jets aus Abb. 2.5 auf Rekonstruktions-Level zugeordnet werden konnten, womit insgesamt 28.425 Ereignisse zur Verfügung stehen. In diesen 28.425 Ereignissen liegt also die wahre Information, welche beiden Jets die AddB-Jets sind, vor. Es wird allerdings nicht gefordert, dass alle Jet-Kategorien zugeordnet werden können, da die Anzahl der zur Verfügung stehenden Ereignisse sonst stark abnehmen würde. Da die Modelle in Kapitel 4 nur auf eine binäre Klassifikation trainiert werden, also ob ein Jet ein AddB-Jet ist oder nicht, ist dieses Vorgehen gerechtfertigt.

Dennoch werden in den Ereignissen auch die übrigen Kategorien HadTopB, LepTopB, HadTopQ und Unbekannt den verbleibenden Jets zugeteilt. Dafür werden die übrigen Jets nach der Größe des Transversalimpulses geordnet. Der Jet mit dem größten Transversalimpuls erhält die Kategorie HadTopB, der Jet mit dem zweitgrößten Transversalimpuls die Kategorie LepTopB und die nächsten zwei Jets die Kategorie HadTopQ. Falls weitere Jets im Ereignis rekonstruiert wurden, erhalten diese die Kategorie Unbekannt und haben von den übrigen Jets die kleinsten Transversalimpulse im Ereignis.

Dies führt dazu, dass viele der übrigen Jet-Kategorien nicht richtig zugeordnet werden. Dennoch gibt es eine Tendenz, dass zumindest ein großer Anteil der übrigen Jets die richtige Kategorie erhält. Dies liegt darin begründet, dass für die Erzeugung des Top-Quarks, aufgrund seiner hohen Ruhemasse, viel Energie aufgewendet werden muss. Das Top-Quark zerfällt anschließend in ein $W^{\pm}$-Boson und ein Bottom-Quark, welches eine deutlich kleinere Ruhemasse aufweist (s. Abb. 2.1). Dadurch geht ein Großteil der Energie in den Impuls des Bottom-Quarks über, was zu tendenziell höheren Transversalimpulsen bei den HadTopB- und LepTopB-Jets führt.

Der Umstand, dass die übrigen Jet-Kategorien nicht notwendigerweise korrekt zugeordnet werden, wird in in den Untersuchungen in Kapitel 5 aufgegriffen und dementsprechend behandelt. Für die restlichen Kapitel stellt dies allerdings, aufgrund der binären Klassifikation, kein Problem dar.

3 Graph Neural Networks

Kapitel 3 beschäftigt sich in Abschnitt 3.1 mit den Grundlagen der Graphentheorie, um darauf aufbauend die Ziele und Anwendungsmöglichkeiten von GNNs zu besprechen. Danach wird in Abschnitt 3.2 ein allgemeines Framework zur Beschreibung unterschiedlicher GNN-Varianten vorgestellt. In Abschnitt 3.3 wird die Funktionsweise von Gated Graph Sequence Neural Networks besprochen, eine GNN-Variante, die in dieser Arbeit eine besondere Rolle spielen wird. Abschließend werden in Abschnitt 3.4 wichtige Konzepte des maschinellen Lernens besprochen.

3.1 Grundlagen der Graphentheorie

3.1.1 Begriffe und Definitionen

Die folgende Einführung in die Graphentheorie orientiert sich an [22], wobei einige Definitionen für den Rahmen dieser Bachelorarbeit angepasst wurden. Im Zentrum der Graphentheorie steht der Graph selbst:

- **Definition 1**: Ein (*ungerichteter*) *Graph* G ist definiert als das geordnete Paar $G(V, E)$ mit der Menge $V = V(G)$, den *Knoten* von G und der Menge $E = E(G)$, den *Kanten* von G, bestehend aus den ungeordneten Paaren $e = [x, y]$ mit $x, y \in V$.

Neben ungerichteten Graphen existieren auch *gerichtete* Graphen, bei denen die Kanten zwischen zwei Knoten eine Richtung aufweisen. Im Abschnitt 3.2.1 wird die Definition des Begriffs „Graph“ erweitert, um diesem und weiteren Umständen Folge zu leisten. Da dieser Abschnitt allerdings dazu gedacht ist ein erstes, grundlegendes Verständnis für Graphen aufzubauen, beziehen sich die folgenden Definitionen zunächst auf ungerichtete Graphen. Die meisten Definitionen lassen sich aber intuitiv und ohne viel Aufwand auf gerichtete Graphen übertragen. Weiterhin werden in diesem Abschnitt zwei Einschränkungen vorausgesetzt:

1) Die Graphen sind *endlich*, was bedeutet, dass die Menge V (und somit auch die Menge E) endlich ist.

2) Die Graphen sind *einfach*, was bedeutet, dass der Graph keine *Mehrfachkanten* und keine *Schlingen* aufweist.

Der Begriff Schlinge beschreibt eine Kante $e = [x, y]$ für die $x = y$ gilt. Eine Mehrfachkante liegt vor, wenn zwei Knoten mit mehr als einer Kante verbunden sind. Graphen mit

Schlingen oder Mehrfachkanten heißen auch *Multigraphen*. Ist $e = [x, y]$ eine Kante, so heißen die Knoten x und y *Endpunkte* dieser Kante, beide Knoten sind dann *inzident* zu e und *benachbart* oder auch *adjazent* zueinander.

Weiterhin heißt ein ungerichteter Graph *vollständig*, wenn jeder Knoten mit jedem anderen Knoten durch eine Kante verbunden ist und der Graph keine Schlingen aufweist. Der Graph $G' = (V', E')$ heißt *Teilgraph* von G, falls $V' \subset V$ und $E' \subset E$ gilt. Abbildung 3.1 veranschaulicht die soeben genannten Begrifflichkeiten und Definitionen.

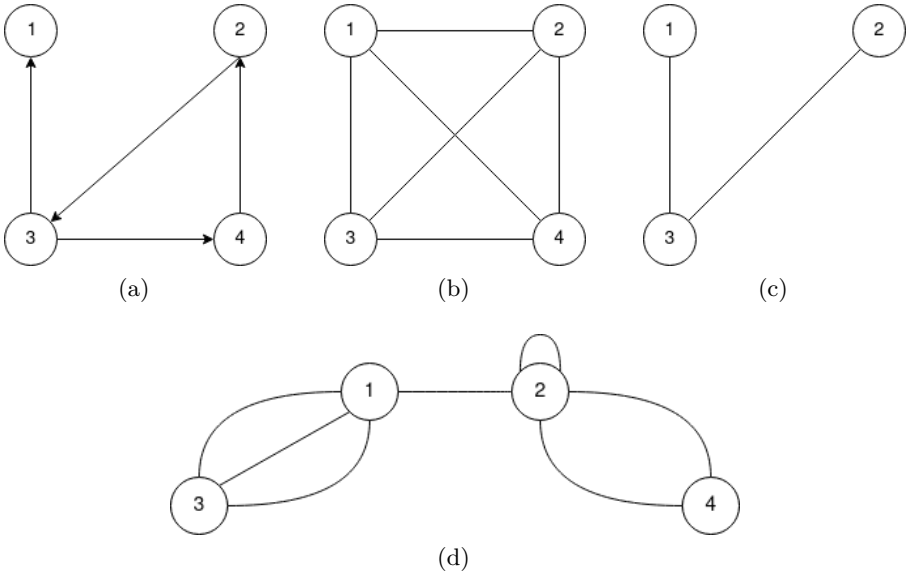


Abbildung 3.1: Beispiele für verschiedene Graphen: Bei Graph (a) handelt es sich um einen gerichteten Graphen. Die anderen Graphen sind ungerichtet. Graph (b) ist vollständig, da jeder Knoten mit jedem anderen Knoten benachbart ist. Graph (c) stellt einen Teilgraph von Graph (b) dar. Mathematisch wird Graph (c) beschrieben durch die Mengen $V = \{1, 2, 3\}$ und $E = \{[1, 3], [2, 3]\}$. Graph (d) ist ein Multigraph, er besitzt Mehrfachkanten und eine Schlinge am Knoten 2.

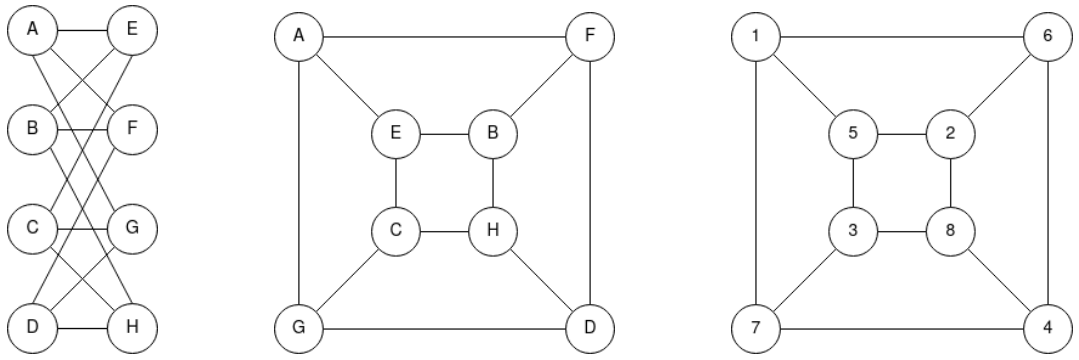
- **Definition 2**: Zwei Graphen $G(V, E)$ und $G'(V', E')$ heißen *isomorph* zueinander, falls eine bijektive Abbildung $\phi : V \to V'$ existiert, sodass gilt:

$$[x, y] \in E \Leftrightarrow [\phi(x), \phi(y)] \in E'$$

In anderen Worten bedeutet dies, dass zwei Graphen isomorph sind, falls sie sich nur in der Bezeichnung ihrer Knoten unterscheiden. Die Struktur der Graphen bleibt allerdings dieselbe, weshalb isomorphe Graphen für viele graphentheoretische Probleme als gleich angesehen werden können. Die Knoten eines Graphen besitzen also keine intrinsische Ordnung und eine Nummerierung der Knoten erfolgt willkürlich. Auf den ersten Blick sind zueinander isomorphe Graphen in der Regel schwer zu erkennen, wie Abb. 3.2 verdeutlicht.

Neben der graphischen Darstellung lässt sich ein Graph auch mit Hilfe der *Adjazenzmatrix* A repräsentieren. Hierfür werden die Knoten des Graphen von 1 bis n durchnummeriert. Die Einträge a_{ij} von A sind dann 1, wenn die Knoten i und j benachbart sind und sonst 0. Die Adjazenzmatrix des Graphen in Abbildung 3.1c hat zum Beispiel die Form:

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



(a) (b) (c)

Abbildung 3.2: Beispiele für isomorphe Graphen: Bei Graph (a) und Graph (b) handelt es sich um denselben Graphen. Sie unterscheiden sich lediglich in ihrer graphischen Darstellung. Graph (c) ist isomorph zu Graph (a) (und zu Graph (b) natürlich auch), sie haben also beide die gleiche Graphstruktur. Nur die Bezeichnung der Knoten ist unterschiedlich. Besonders deutlich wird die Isomorphie der beiden Graphen beim Vergleich von Graph (b) und Graph (c).

Die Adjazenzmatrix ungerichteter Graphen ist immer symmetrisch. Die Diagonalelemente sind 0, da der Graph keine Schlingen besitzt. Eine andere (genauso legitime) Nummerierung der Knoten des Graphen in Abb. 3.1c führt zu einer anderen Adjazenzmatrix. Strukturell unterscheiden sich die Graphen dadurch aber nicht. Auch gerichtete Graphen lassen sich mittels einer Adjazenzmatrix repräsentieren. Die Matrixelemente a_{ij} sind dann 1, wenn eine Kante von Knoten i zu Knoten j führt, sonst 0. Oft lässt sich also eine ungerichtete Kante auch als zwei entgegengesetzt gerichtete Kanten interpretieren.

- **Definition 3**: Der *Grad* eines Knoten x ist definiert als die Anzahl der mit x inzidenten Kanten, wobei Schlingen doppelt zählen, und wird mit $d(x)$ bezeichnet.

Jeder Knoten des Graphen in Abb. 3.1b hat zum Beispiel den Grad 3. Für gerichtete Graphen muss zwischen einfallenden und ausgehenden Kanten unterschieden werden.

- **Definition 4**: Sei G ein Graph mit den Knoten $x_1, x_2, ..., x_n$. Dann heißt eine Menge von Kanten, die x_1 mit x_n verbindet, *Kantenfolge* von x_1 bis x_n. Die Kantenfolge $\{[x_1, x_2], [x_2, x_3], ..., [x_{n-1}, x_n]\}$ soll als $(x_1x_2x_3...x_n)$ bezeichnet werden. Die Kantenfolge heißt *geschlossen*, falls $x_1 = x_n$ gilt, andernfalls *offen*.
  - Ein *Weg* von x nach y ist eine offene Kantenfolge von x nach y, in der kein Knoten mehrmals vorkommt.
  - Der Graph G heißt *zusammenhängend*, falls alle seine Knoten paarweise durch einen Weg verbunden sind. Es existieren also keine *isolierten* Knoten.
  - Die *Länge* $l = l((x_1x_2...x_n)) = n - 1$ einer Kantenfolge entspricht der Anzahl der Kanten in der Kantenfolge.
  - Der *Abstand* $a(x, y)$ zweier Knoten ist die Länge des kürzesten Weges zwischen den Knoten.
  - Der *Durchmesser* $D(G)$ eines zusammenhängenden Graphen entspricht dem größten Abstand zwischen zwei seiner Knoten.

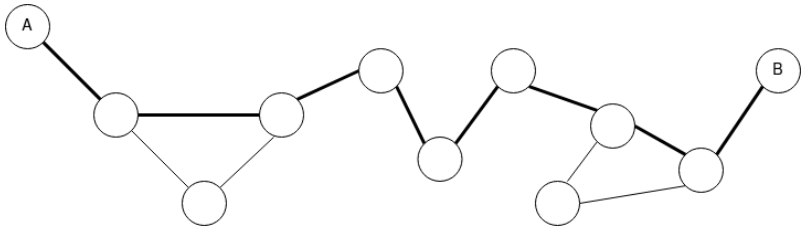


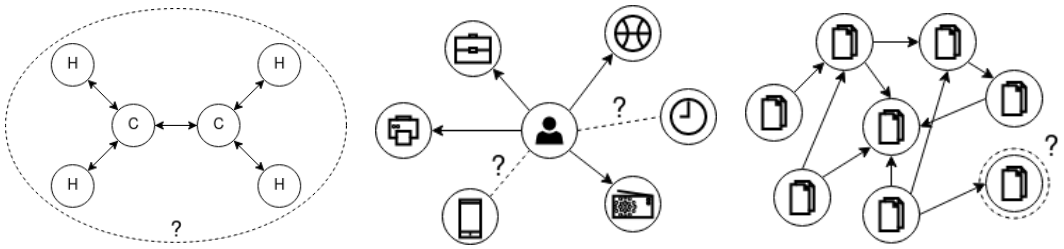
Abbildung 3.3: Durchmesser eines zusammenhängenden Graphen: Die Knoten A und B im dargestellten Graph haben den Abstand $a(\mathrm{A}, \mathrm{D}) = 8$, was der Länge des kürzesten Weges zwischen den beiden Knoten entspricht. Da die Knoten A und B am weitesten voneinander entfernt sind, hat auch der Durchmesser des Graphen den Wert $D(G) = 8$.

Ein Beispiel für den Durchmesser eines Graphen ist in Abb. 3.3 dargestellt. Bei gerichteten Graphen kann ein Weg oder eine Kantenfolge zwischen zwei Knoten nur entlang der Richtung der Kanten erfolgen.

- **Definition 5**: Ein Graph G heißt *bewertet*, falls allen Kanten $e = [x, y]$ von G ein *Gewicht* $w(x, y) \in \mathbb{R}$ zugeordnet ist. Für bewertete Graphen ist die Länge einer Kantenfolge die Summe der Gewichte aller Kanten in der Kantenfolge.

3.1.2 Anwendung von Graph Neural Networks

Die Knoten eines Graphen können viele verschiedene Entitäten repräsentieren, wobei die Kanten diese Entitäten in Beziehung zueinander setzen. Somit lassen sich viele Problemstellungen durch Graphen abbilden. Das Ziel von Graph Neural Networks ist es dabei, Aussagen über diesen Graphen zu treffen. Dabei wird unter Aussagen auf Graph-, Knoten- und Kanten-Ebene unterschieden. Typische Beispiele hierfür sind in Abb. 3.4 dargestellt.



(a) (b) (c)

Abbildung 3.4: Aussagen über Graphen: Graph (a) stellt ein Molekül dar. Eine wichtige Fragestellung in der Bioinformatik ist zum Beispiel, ob es sich bei einem Molekül um eine Droge handelt. Dies entspricht einer Aussage auf Graph-Ebene. Graph (b) repräsentiert das Konsumverhalten eines Kunden. Für Unternehmen ist es interessant zu wissen, welche weiteren Produkte dem Kunden gefallen könnten. Dies stellt eine Aussage auf Kanten-Ebene dar. In Graph (c) ist ein Zitationsnetzwerk abgebildet. Es besteht aus wissenschaftlichen Arbeiten, die aufeinander verweisen. Interessant ist hier zum Beispiel die Frage, ob eine bestimmte Arbeit seriös ist oder nicht, was einer Aussage auf Knoten-Ebene entspricht.

Es liegt nahe anzunehmen, dass sich die verschiedenen Bestandteile des Graphen gegenseitig beeinflussen. Um also eine Aussage über einen Graphen, Knoten oder eine Kante zu treffen,

müssen diese im Kontext des gesamten Graphen betrachtet werden. Die einzelnen Bestandteile müssen also Informationen miteinander austauschen, um sinnvolle und verlässliche Aussagen erhalten zu können. Deutlich wird dies mit den Beispielen aus Abb. 3.4: Um festzustellen, ob es sich bei einem Molekül um eine Droge handelt, muss die gesamte chemische Struktur betrachtet werden, bestehend aus allen Atomen und chemischen Verbindungen. Die Frage, welches Produkt einem Kunden gefallen könnte, hängt stark davon ab, was der Kunde schon gekauft hat. Und um festzustellen, ob eine wissenschaftliche Arbeit seriös ist, kann ein Zitationsnetzwerk hilfreich sein, da seriöse Arbeiten in so einem Graphen tendenziell besser vernetzt sind, also öfters zitieren aber vor allem auch öfters zitiert werden.

Wie dieser Informationsaustausch mathematisch verwirklicht werden kann, wird im folgenden Abschnitt erläutert.

3.2 Allgemeine Funktionsweise von Graph Neural Networks

Graph Neural Networks werden seit einigen Jahren immer weiter erforscht und stets weiterentwickelt. Da es mittlerweile eine Vielzahl von Anwendungen und Methoden gibt, ist es schwierig, eine allgemeingültige Funktionsweise für GNNs zu definieren. Dennoch soll in diesem Abschnitt das *Graph Networks Framework* aus [23] vorgestellt werden, welches eine Vielzahl von Ansätzen vereint.

3.2.1 Erweiterung des Graph-Begriffs

Zunächst wird der Begriff des Graphen gemäß [23] erweitert:

- **Definition 1 (erweitert)**: Ein *Graph* G ist definiert als das Tripel $G(\boldsymbol{u}, V, E)$. Dabei ist $\boldsymbol{u}$ das *globale Attribut* von G. $V = \{\boldsymbol{v}_i\}_{i=1:N^v}$ beschreibt die Menge der *Knoten* von G (mit Mächtigkeit N^v), wobei jedes $\boldsymbol{v}_i$ das *Attribut* eines Knoten ist. $E = \{(\boldsymbol{e}_k, r_k, s_k)\}_{k=1:N^e}$ ist die Menge der Tripel $e_k = (\boldsymbol{e}_k, r_k, s_k)$, den *Kanten* von G (mit Mächtigkeit N^e). Dabei ist $\boldsymbol{e}_k$ das *Attribut*, r_k der Index des *Empfängerknotens* und s_k der Index des *Senderknotens* einer Kante.

In dieser Definition werden neben der Zulassung von gerichteten Multigraphen die Knoten, Kanten und der Graph selbst zusätzlich mit Attributen versehen. Attribute stehen hier für jegliche Eigenschaften eines Knotens, einer Kante oder des Graphen, die als Skalar, Vektor, Menge oder sogar als eigener Graph kodiert werden können. Als globales Attribut werden Eigenschaften auf Graph-Ebene bezeichnet. Besonders häufig wird einem Knoten ein Vektor als Attribut zugeordnet, welcher auch *Feature-Vektor* genannt wird. Die Zusammenfassung $\tilde{V} = (\boldsymbol{v}_1, \boldsymbol{v}_2, ...)^{\mathrm{T}}$ der Feature-Vektoren wird auch *Node-Embedding* genannt. Die erweiterte Definition des Graph-Begriffs ist in Abbildung 3.5 illustriert.

Es gibt viele Beispiele aus der realen Welt, die sich mit so einem Graphen beschreiben lassen. Ein typisches Beispiel sind soziale Netzwerke, bei denen Personen als Knoten dargestellt werden und mit Attributen wie Alter, Geschlecht, politischer Einstellung etc. versehen werden können. Die Kanten stellen zwischenmenschliche Beziehungen dar und gewichtete Kanten könnten den Grad der Zuneigung repräsentieren. Ein globales Attribut könnte besondere Begebenheiten einer Gesellschaft repräsentieren.

3.2.2 Graph Networks Framework

Aufbauend auf der erweiterten Definition aus 3.2.1 wird nun das Graph Networks Framework vorgestellt. Den Kern des Frameworks bildet der *Graph Network-Block* (GN-Block), ein Modul, welches einen Graphen als Input nimmt, Berechnungen basierend auf der Graphstruktur ausführt, und einen neuen Graphen als Output zurück gibt. In den meisten Fällen

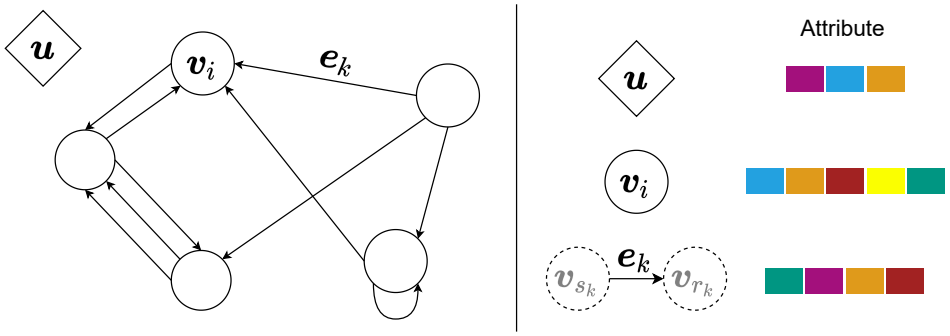


Abbildung 3.5: Erweiterte Definition eines Graphen, eigene Darstellung nach [23]: In dieser Definition von Graph sind gerichtete Kanten, Mehrfachkanten und Schlingen zugelassen. Die Knoten sind mit Attributen $\boldsymbol{v}_i$ und die Kanten mit den Attributen $\boldsymbol{e}_k$ versehen. Zusätzlich ist dem gesamten Graph das globale Attribut $\boldsymbol{u}$ zugewiesen. Die Attribute können jegliche Informationen darstellen, die sich als Skalar, Vektor, Menge oder auch als eigener Graph kodieren lassen. Die unterschiedlichen Attribute der Bestandteile des Graphen sind symbolisch als farbige Rechtecke dargestellt.

hat der Output-Graph dieselbe Struktur wie zuvor, ist allerdings mit neuen Attributen versehen. Ein GN-Block enthält drei *Updatefunktionen* ϕ und drei *Aggregatfunktionen* ρ:

$$\boldsymbol{e}'_k = \phi^e(\boldsymbol{e}_k, \boldsymbol{v}_{r_k}, \boldsymbol{v}_{s_k}, \boldsymbol{u}) \qquad \bar{\boldsymbol{e}}'_i = \rho^{e\to v}(E'_i) \tag{3.1}$$

$$\boldsymbol{v}'_i = \phi^v(\bar{\boldsymbol{e}}'_i, \boldsymbol{v}_i, \boldsymbol{u}) \qquad \bar{\boldsymbol{e}}' = \rho^{e\to u}(E') \tag{3.2}$$

$$\boldsymbol{u}' = \phi^u(\bar{\boldsymbol{e}}', \bar{\boldsymbol{v}}', \boldsymbol{u}) \qquad \bar{\boldsymbol{v}}' = \rho^{v\to u}(V') \tag{3.3}$$

wobei $E'_i = \{(\boldsymbol{e}'_k, r_k, s_k)\}_{r_k=i,\ k=1:N^e}$, $E' = \bigcup_i E'_i = \{(\ \boldsymbol{e}'_k, r_k, s_k)\}_{k=1:N^e}$ und $V' = \{\boldsymbol{v}'_i\}_{i=1:N^v}$ gilt. Die Funktionen ϕ^e und ϕ^v werden parallel auf alle Kanten bzw. Knoten angewandt und aktualisieren deren Attribute. Die Funktion ϕ^u wird hingegen nur einmal als globale Updatefunktion angewandt. Die Funktionen ρ nehmen eine Menge als Input, aggregieren die Information, und geben ein einzelnes Element zurück. Eine wichtige Eigenschaft der Aggregatfunktionen ist, dass sie *permutationsinvariant* sein müssen, da ein Graph, wie in 3.1.1 erläutert, keine intrinsische Ordnung besitzt. Die Aggregatfunktionen müssen also unabhängig von der Reihenfolge ihrer Argumente sein.

Die Berechnungen innerhalb eines GN-Blocks sind in folgende Schritte unterteilt und in Abbildung 3.6 dargestellt:

(A) ϕ^e wird parallel auf alle Kanten $e_k = (\boldsymbol{e}_k, r_k, s_k)$ einzeln angewandt und berechnet mit Hilfe der Attribute $\boldsymbol{e}_k$ der Kante, den Attributen des Empfänger- und Senderknoten $\boldsymbol{v}_{r_k}$ und $\boldsymbol{v}_{s_k}$, sowie des globalen Attributs $\boldsymbol{u}$ den Output $\boldsymbol{e}'_k$ dieser Kante. In der Menge $E'_i = \{(\boldsymbol{e}'_k, r_k, s_k)\}_{r_k=i,\ k=1:N^e}$ werden alle am Knoten i einfallenden Kanten mit den aktualisierten Attributen zusammengefasst. Die Menge $E' = \bigcup_i E'_i = \{(\ \boldsymbol{e}'_k, r_k, s_k)\}_{k=1:N^e}$ enthält alle Kanten des Graphen mit aktualisierten Attributen.

(B) $\rho^{e\to v}$ nimmt die Menge E'_i, bestehend aus den Outputs aller am Knoten i einfallenden Kanten aus (A), und aggregiert die Information in $\bar{\boldsymbol{e}}'_i$.

(C) ϕ^v wird parallel auf alle Knoten einzeln angewandt und berechnet mit Hilfe der im Schritt (B) aggregierten Information der einfallenden Kanten $\bar{\boldsymbol{e}}'_i$ eines Knoten, den

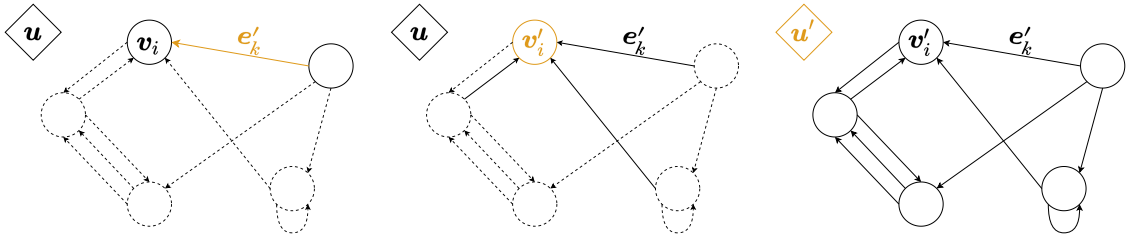
Attributen $\boldsymbol{v}_i$ des Knotens und des globalen Attributs $\boldsymbol{u}$ die neuen Attribute $\boldsymbol{v}'_i$ des Knotens. Die Menge $V' = \{\boldsymbol{v}'_i\}_{i=1:N^v}$ enthält alle aktualisierten Attribute der Knoten.

(D) $\rho^{e\to u}$ nimmt die Menge E' aller aktualisierten Kanten und aggregiert die Information in $\bar{\boldsymbol{e}}'$.

(E) $\rho^{v\to u}$ nimmt die Menge V' aller aktualisierten Knoten und aggregiert die Information in $\bar{\boldsymbol{v}}'$.

(F) ϕ^u nimmt die im Schritt (D) und (E) aggregierten Informationen $\bar{\boldsymbol{e}}'$, $\bar{\boldsymbol{v}}'$ und das globale Attribut $\boldsymbol{u}$ und berechnet daraus das aktualisierte, globale Attribut $\boldsymbol{u}'$.

Die hier vorgegebene Reihenfolge der Berechnungsschritte ist nicht bindend und nicht jeder Schritt muss zwingend durchgeführt werden. Die Funktionen ϕ werden oft mittels neuronaler Netze mit erlernbaren Parametern umgesetzt, können aber auch mit anderen Funktionen implementiert werden. Aus diesem Grund verzichten die Autoren von [23] auch auf den Begriff „Neural“ in „Graph Networks Framework“.



(a) Kanten Aktualisierung: ϕ^e (b) Knoten Aktualisierung: ϕ^v (c) Globale Aktualisierung: ϕ^u

Abbildung 3.6: Berechnungsschritte in einem GN-Block, eigene Darstellung nach [23]: In orange ist das Element hervorgehoben, welches im jeweiligen Berechnungsschritt aktualisiert wird. Alle Elemente des Graphen, die für den jeweiligen Berechnungsschritt genutzt werden, sind mit durchgezogenen Linien gekennzeichnet. Dabei ist zu beachten, dass bei jedem Berechnungsschritt auch der vorherige Wert der zu aktualisierenden Größe in die Aktualisierungsfunktion einfließt. Gestrichelte Elemente spielen im jeweiligen Berechnungsschritt keine Rolle. Die Berechnungsschritte in (a) werden auf alle Kanten parallel angewandt. Ebenso werden danach die Berechnungsschritte in (b) parallel auf alle Knoten angewandt. Der Berechnungsschritt in (c) wird nur einmal auf den gesamten Graphen angewandt.

3.3 Gated Graph Sequence Neural Networks

Dieser Abschnitt beschäftigt sich mit der Funktionsweise von *Gated Graph Sequence Neural Networks* (GGS-NNs) [24], welche in den Untersuchungen in Kapitel 4 eine zentrale Rolle einnehmen werden. Die Funktionsweise lässt sich auch im Rahmen des Graph Networks Frameworks aus Abschnitt 3.2.2 erklären.

Die hier verwendete Ausführung des GGS-NNs nimmt einen Graphen mit Feature-Vektoren $\boldsymbol{v}_i^{(0)}$ und Gewichten $e_{i,j}$ als Input und gibt einen Graphen mit derselben Struktur, allerdings mit neuen Feature-Vektoren $\boldsymbol{h}_i^{(L)}$ zurück. Dabei ist $L \in \mathbb{N}$, die Sequenzlänge des GGS-NNs, ein Hyperparameter und bestimmt, wie viele GN-Blöcke hintereinander ausgeführt werden. Die einzelnen GN-Blöcke sind strukturell identisch und aktualisieren jedes Mal die Feature-Vektoren aller Knoten. Dafür nimmt der l-te GN-Block sowohl den Output

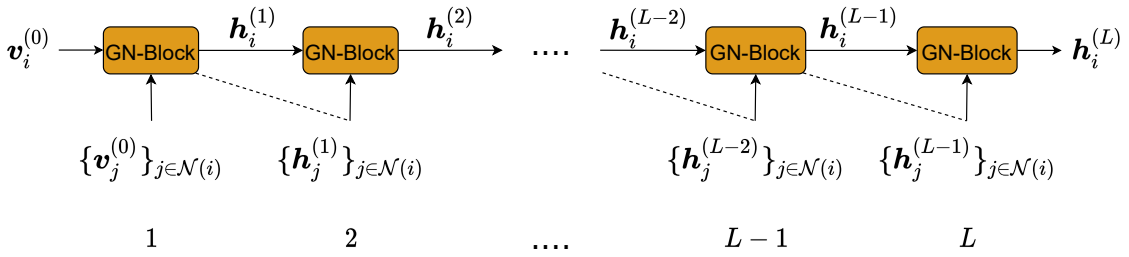


Abbildung 3.7: Architektur des Gated Graph Sequence Neural Networks: Schematische Darstellung des GGS-NNs aus der Sicht des Knotens i. Jeder GN-Block nimmt den Feature-Vektor des zu aktualisierenden Knoten i aus dem vorherigen GN-Block, sowie die Feature-Vektoren aller Nachbarn von i aus dem vorherigen GN-Block als Input. Der Output eines GN-Blocks ist der aktualisierte Feature-Vektor des Knotens i. Es werden L GN-Blöcke hintereinander geschaltet, wobei L der Sequenzlänge entspricht. Dieser Prozess läuft für alle Knoten des Graphen parallel ab.

des vorangegangenen GN-Blocks $\boldsymbol{h}_i^{(l-1)}$ als auch $\{\boldsymbol{h}_j^{(l-1)}\}_{j\in\mathcal{N}(i)}$, die Menge der Feature-Vektoren der Nachbarn von i aus dem vorherigen Berechnungsschritt, als Input und gibt den neuen Feature-Vektor $\boldsymbol{h}_i^{(l)}$ des Knotens i zurück. Die Berechnungen eines GN-Blocks finden parallel für alle Knoten im Graphen statt. Die Dimension n der neuen Feature-Vektoren $\boldsymbol{h}_i^{(l)}$ ist ebenfalls ein Hyperparameter und muss größer oder gleich der Dimension der Feature-Vektoren $\boldsymbol{v}_i^{(0)}$ sein. Die Architektur des GGS-NNs ist in Abb. 3.7 dargestellt.

Ein GN-Block führt die Berechnungsschritte (A), (B) und (C) aus dem Graph Networks Framework aus (s. Abschnitt 3.2.2). Lediglich im ersten GN-Block wird zu Beginn noch der Berechnungsschritt (A0) durchgeführt. Abbildung 3.8 veranschaulicht die folgenden Berechnungsschritte. Der l-te GN-Block nimmt folgende Struktur an:

(A0) Im ersten GN-Block wird die Dimension aller Feature-Vektoren auf n vergrößert und die Feature-Vektoren mit 0 aufgefüllt (sogenanntes Zero-Padding):

$$\boldsymbol{h}_i^{(0)} = \boldsymbol{v}_i^{(0)} || \, \boldsymbol{0}. \tag{3.4}$$

Falls n der Dimension von $\boldsymbol{v}_i^{(0)}$ entspricht, gilt:

$$\boldsymbol{h}_i^{(0)} = \boldsymbol{v}_i^{(0)}. \tag{3.5}$$

(A & B) Die beiden Berechnungsschritte mit der Update-Funktion ϕ^e und der Aggregatfunktion $\rho^{e\to v}$ können in einer Formel zusammengefasst werden. Die Aggregatfunktion kann verschiedene Formen annehmen und wird im Rahmen dieser Untersuchung einmal als Summe und einmal als Mittelwert implementiert:

$$\begin{aligned} \boldsymbol{m}_i^{(l)} &= \sum_{j\in\mathcal{N}(i)} e_{i,j} \cdot \Theta^{(l-1)} \cdot \boldsymbol{h}_j^{(l-1)} \\ \boldsymbol{m}_i^{(l)} &= \underbrace{\frac{1}{|\mathcal{N}(i)|} \sum_{j\in\mathcal{N}(i)}}_{\rho^{e\to v}} \underbrace{e_{i,j} \cdot \Theta^{(l-1)} \cdot \boldsymbol{h}_j^{(l-1)}}_{\phi^e}. \end{aligned} \tag{3.6}$$

Hierbei beschreibt $\mathcal{N}(i)$ die Menge der Knoten, von denen eine Kante bei Knoten i einfällt. Die Einträge der Gewichtsmatrix Θ mit Dimension $n \times n$ werden im Verlaufe des Trainings erlernt und unterscheiden sich in jedem GN-Block.

(C) Die Update-Funktion ϕ^v wird mittels einer Gated Recurrent Unit (GRU) umgesetzt:

$$\boldsymbol{h}_i^{(l)} = \text{GRU}\left(\boldsymbol{m}_i^{(l)}, \boldsymbol{h}_i^{(l-1)}\right). \tag{3.7}$$

Die GRU-Zelle selbst führt folgende Berechnungen durch:

$$\begin{aligned}
&1) \quad \boldsymbol{r} = \sigma\left(W_{ir}\boldsymbol{m}_i^{(l)} + \boldsymbol{b}_{ir} + W_{hr}\boldsymbol{h}_i^{(l-1)} + \boldsymbol{b}_{hr}\right)\\
&2) \quad \boldsymbol{z} = \sigma\left(W_{iz}\boldsymbol{m}_i^{(l)} + \boldsymbol{b}_{iz} + W_{hz}\boldsymbol{h}_i^{(l-1)} + \boldsymbol{b}_{hz}\right)\\
&3) \quad \boldsymbol{n} = \tanh\left(W_{in}\boldsymbol{m}_i^{(l)} + \boldsymbol{b}_{in} + \boldsymbol{r} * \left(W_{hn}\boldsymbol{h}_i^{(l-1)} + \boldsymbol{b}_{hn}\right)\right)\\
&4) \quad \boldsymbol{h}_i^{(l)} = (1 - \boldsymbol{z}) * \boldsymbol{n} + \boldsymbol{z} * \boldsymbol{h}_i^{(l-1)}.
\end{aligned} \tag{3.8}$$

Dabei beschreibt σ die Sigmoid-Funktion und $*$ das Hadamard-Produkt (elementweise Multiplikation). Die Matrizen W und Vektoren $\boldsymbol{b}$ sind die zu trainierenden Gewichte der GRU. Je nachdem, ob ein Bias in der GRU gewünscht ist, können die Vektoren $\boldsymbol{b}$ auch entfernt werden. Im Folgenden werden die vier Funktionen kurz erklärt:

Funktion 1) und 2) berechnen das Reset-Gate $\boldsymbol{r}$ und Update-Gate $\boldsymbol{z}$. Strukturell sind beide Funktionen identisch und nehmen als Input sowohl die aggregierte Information $\boldsymbol{m}_i^l$ der Nachbarknoten aus dem momentanen Berechnungsschritt als auch die Information $\boldsymbol{h}_i^{(l-1)}$ des Knotens selbst aus dem vorherigen Berechnungsschritt. In Funktion 3) werden mit Hilfe des Reset-Gates relevante Informationen aus $\boldsymbol{h}_i^{(l-1)}$ herausgefiltert und mit $\boldsymbol{m}_i^l$ zur Größe $\boldsymbol{n}$ kombiniert. Diese stellt den Kandidat-Aktivierungsvektor für den neuen Feature-Vektor des Knotens dar. Letztlich berechnet Funktion 4) mit Hilfe des Update-Gates $\boldsymbol{z}$, des Kandidat-Aktivierungsvektors $\boldsymbol{n}$ und des vorherigen Feature-Vektors $\boldsymbol{h}_i^{(l-1)}$ den neuen Feature-Vektor $\boldsymbol{h}_i^{(l)}$ des Knotens. Das Update-Gate bestimmt dabei, bis zu welchem Grad der Feature-Vektor seine Einträge aktualisiert.

Gated Recurrent Units wurden im Jahr 2014 in [25] vorgeschlagen. Sie besitzen den Vorteil, dass sie gegenüber herkömmlichen Recurrent Neural Networs (RNNs) weniger stark am Problem verschwindender Gradienten leiden [26]. Aufgrund des Gating-Mechanismus sind sie außerdem sehr gut dazu geeignet, Langzeit-Abhängigkeiten in sequentiellen Daten zu erkennen. Eine genauere Beschreibung zur Funktionsweise von GRUs kann in den soeben genannten Quellen nachgelesen werden.

In jedem GN-Block tauschen die Knoten ihre Information ein Mal mit ihren Nachbarn aus. Eine wichtige Frage bei GNNs ist es, wie oft dieser Nachrichtenaustausch durchgeführt werden soll. Für das GGS-NN entspricht dies der Frage nach der Anzahl der Sequenzen L. Einen groben Anhaltspunkt liefert hierfür oft der Durchmesser D eines Graphen. Wird der Nachrichtenaustausch D mal durchgeführt, hat jeder Knoten im Graphen die Information jedes anderen Knoten erhalten. Allerdings kann es auch durchaus sinnvoll sein, diesen Nachrichtenaustausch noch weitere Male zu wiederholen. In Kapitel 4 und 5 wird ein vollständiger Graph mit Durchmesser $D = 1$ verwendet, was bedeutet, dass der größte Abstand zweier Knoten lediglich einer Kante entspricht. Trotzdem werden dort Sequenzlängen verwendet, die ein Vielfaches von D sind. Dies liegt darin begründet, dass das GGS-NN es den Knoten ermöglicht, ihren eigenen Feature-Vektor im Kontext der Feature-Vektoren aller

anderen Knoten zu betrachten und dann entsprechend zu aktualisieren. Diese aktualisierten Feature-Vektoren stellen im Optimalfall eine verbesserte Repräsentation der Knoten dar. Jeder Knoten enthält nun Informationen über sich und alle seine Nachbarn. Dennoch hat dabei jeder Knoten unterschiedliche Informationen erhalten (die seiner jeweiligen Nachbarn) und daraus eine neue Information geschaffen (den aktualisierten Feature-Vektor), die den anderen Knoten noch nicht bekannt ist. Daher ist es sinnvoll, den Nachrichtenaustausch mit den aktualisierten Feature-Vektoren erneut zu wiederholen. Da sich nun jeder Knoten mit seiner verbesserten Repräsentation im Kontext der verbesserten Repräsentationen der anderen Knoten betrachtet, stellen die neuen aktualisierten Feature-Vektoren im Optimalfall wieder eine verbesserte Repräsentation der Knoten dar. Wie oft dieser Vorgang letztendlich wiederholt werden muss, also die Bestimmung der Sequenzlänge L, ist Gegenstand der Hyperparameteranalyse in Abschnitt 4.3.

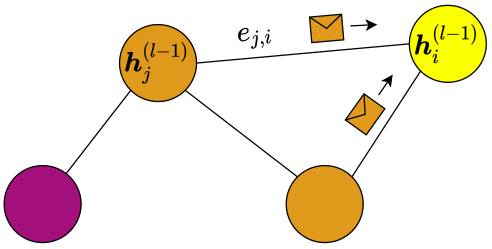
$$\boldsymbol{v}_i^{(0)} = (a_1, \ldots, a_m)^{\mathrm{T}}$$

$$\downarrow$$

$$\boldsymbol{h}_i^{(0)} = (a_1, \ldots, a_m, 0, \ldots, 0)^{\mathrm{T}}$$

$$\boldsymbol{h}_i^{(0)} \in \mathbb{R}^n$$

(a) Berechnungsschritt (A0)



$$= e_{j,i}\Theta^{(l-1)}\boldsymbol{h}_j^{(l-1)}$$

$$\boldsymbol{m}_i^{(l)} = \mathrm{AGGR}(\quad)$$

(b) Berechnungsschritt (A & B)

$$\boldsymbol{h}_i^{(l)} = \mathrm{GRU}(\boldsymbol{m}_i^{(l)}, \boldsymbol{h}_i^{(l-1)})$$

(c) Berechnungsschritt (C)

Abbildung 3.8: Berechnungsschritte in einem GN-Block des GGS-NNs: (A0) wird lediglich im ersten GN-Block ausgeführt und füllt den Feature-Vektor mit Nullen auf, sodass er die Dimension n hat. In (A & B) erhält der betrachtete Knoten (gelb) die Nachrichten der Nachbarknoten (orange), welche dann aggregiert werden. Mit Hilfe der aggregierten Information der Nachbarknoten und des Feature-Vektors des Knotens selber, wird der aktualisierte Feature-Vektor des Knotens berechnet. Diese Berechnungsschritte laufen für alle Knoten des Graphen parallel ab.

3.4 Konzepte des maschinellen Lernens

Für das bessere Verständnis des Kapitels 4 werden im Folgenden zwei wichtige Konzepte aus dem Bereich des maschinellen Lernens besprochen: Batches und das Training neuronaler Netze mittels Rückpropagierung und Loss-Funktion.

- **Batches**: Beim Trainieren von GNNs werden wie in herkömmlichen neuronalen Netzen Batches, eine Ansammlung von mehreren Graphen, an das GNN übergeben. Hierfür werden die einzelnen Adjazenzmatrizen der n Graphen im Batch in einer weiteren Adjazenzmatrix $\tilde{A}_{\text{Batch}}$ zusammengefasst. Ebenso werden die einzelnen Node-Embeddings der Graphen in einem neuen Node-Embedding $\tilde{V}_{\text{Batch}}$ zusammengefasst [27]:

$$\tilde{A}_{\text{Batch}} = \begin{pmatrix} A_1 & & \\ & \ddots & \\ & & A_n \end{pmatrix}, \qquad \tilde{V}_{\text{Batch}} = \begin{pmatrix} \tilde{V}_1 \\ \vdots \\ \tilde{V}_n \end{pmatrix}. \tag{3.9}$$

  Dieses Batch entspricht einem neuen Graphen, der aus n isolierten Teilgraphen besteht und durch die Adjazenzmatrix $\tilde{A}_{\text{Batch}}$ und dem Node-Embedding $\tilde{V}_{\text{Batch}}$ beschrieben wird. Dieser neue Graph wird an das GNN übergeben. Da der Informationsaustausch in einem GNN weiterhin nur entlang der Kanten erfolgt, findet kein Informationsaustausch zwischen den isolierten Teilgraphen statt.

- **Training neuronaler Netze**: Das Trainieren der Gewichte in GNNs erfolgt analog zu herkömmlichen neuronalen Netzen. Der Trainingsprozess läuft dabei in drei Schritten ab [28], zunächst erfolgt der Vorwärtsdurchlauf. Dabei werden die Inputs an ein Modell übergeben, welches durch seine Gewichte $\boldsymbol{\mathcal{P}}$ parametrisiert ist. Für das GGS-NN handelt es sich bei diesen Gewichten um die Einträge der Gewichtsmatrizen $\Theta^{(l-1)}$ (s. Gleichung 3.6) sowie den Gewichtsmatrizen W und Gewichtsvektoren $\boldsymbol{b}$ in der GRU (s. Gleichung 3.8). Das Modell führt nun Berechnungen basierend auf den Inputs und seiner Gewichte durch und gibt eine Vorhersage $\boldsymbol{Y}_{\text{pred}}$ zurück. Im Falle des GGS-NNs muss diese Vorhersage, entsprechend des Anwendungsfalles, aus dem vom GGS-NN zurückgegebenen finalen Node-Embedding $\tilde{V}^{\text{final}}$ berechnet werden (s. Abschnitt 4.3.1). Hierfür werden oft weitere neuronale Netze verwendet, deren Gewichte ebenfalls in $\boldsymbol{\mathcal{P}}$ repräsentiert werden. In einem zweiten Schritt wird diese Vorhersage $\boldsymbol{Y}_{\text{pred}}$ dann mit der wahren Information $\boldsymbol{Y}_{\text{gt}}$ (engl.: ground truth information) verglichen. Dieser Vergleich erfolgt mittels einer Loss-Funktion $\mathcal{L}(\boldsymbol{\mathcal{P}})$, die eine positive Zahl zurückgibt. Diese Zahl wird auch Loss-Wert genannt und stellt ein Maß für die Abweichung zwischen Vorhersage und wahrer Information dar. Je kleiner der Loss-Wert, desto geringer sind diese Abweichungen. Das Ziel ist es nun, die Gewichte so anzupassen, dass der Loss-Wert minimiert wird. Dies führt zum dritten Schritt. Mit Hilfe des Gradienten $\nabla_{\boldsymbol{\mathcal{P}}}\mathcal{L}$ kann der Loss-Wert nun zurück propagiert werden. Dabei zeigt der Gradient in die Richtung des größten Anstiegs der Loss-Funktion. Somit zeigt $-\nabla_{\boldsymbol{\mathcal{P}}}\mathcal{L}$ in die Richtung des größten Abstiegs. Die alten Gewichte $\boldsymbol{\mathcal{P}}$ des Modells werden dabei so angepasst, dass für die neuen Gewichte $\boldsymbol{\mathcal{P}}'$ folgendes gilt:

$$\boldsymbol{\mathcal{P}}' = \boldsymbol{\mathcal{P}} - \lambda \nabla_{\boldsymbol{\mathcal{P}}}\mathcal{L}. \tag{3.10}$$

  Dabei ist $\lambda \in (0, 1)$ die Lernrate des Modells und bestimmt, wie stark die Gewichte angepasst werden. Dieser dritte Schritt wird auch Rückpropagierung genannt. Der soeben beschriebene Lernprozess wird mehrmals wiederholt bis ein Minimum der Loss-Funktion gefunden ist.

4 Klassifikation der AddB-Jets mittels Graph Neural Networks

Wie in Kapitel 2.4 besprochen, stellt die Untersuchung der zusätzlichen Jets mit B-Hadronen, im Folgenden auch AddB-Jets genannt, im $t\bar{t} + b\bar{b}$-Prozess eine wichtige Aufgabe dar. Hierfür ist es essenziell, die AddB-Jets von den anderen Jets im Ereignis unterscheiden zu können. Auf der Entwicklung solcher Klassifikationsmethoden mittels GNNs liegt der Fokus dieses Kapitels. Dabei werden für das Training simulierte Daten (s. Abschnitt 2.5) verwendet, die die Information über die Herkunft eines Jets beinhalten.

In Abschnitt 4.1 wird zunächst beschrieben, wie dieses Klassifikationsproblem als graphentheoretisches Problem aufgefasst werden kann. Danach wird in Abschnitt 4.2 der Trainings- und Evaluationsprozess diskutiert. Anschließend erfolgt die Auswertung drei verschiedener Modelle in den Abschnitten 4.3 - 4.5.

4.1 Darstellung als graphentheoretisches Problem

Das Klassifizieren der beiden AddB-Jets im $t\bar{t} + b\bar{b}$-Prozess kann als graphentheoretisches Problem aufgefasst werden. Dabei repräsentieren die Knoten des Graphen die rekonstruierten Detektorobjekte des $t\bar{t} + b\bar{b}$-Prozesses, also die Jets und das Lepton. Das Ziel ist es, alle Knoten in einem Ereignis in die beiden Kategorien 1 (entspricht einem AddB-Jet) und Kategorie 0 (entspricht keinem AddB-Jet) einzuteilen. Dies stellt eine binäre Klassifikation auf Knoten-Ebene dar.

Die Kanten zwischen den Knoten legen die Kommunikationswege fest und bestimmen, welche Knoten Nachrichten miteinander austauschen. Weiterhin eignen sich die Kanten auch dazu, Informationen über paarweise Eigenschaften von Knoten im Graphen festzuhalten. Daher wird hier folgende Graphstruktur festgelegt:

Die Detektorobjekte des $t\bar{t} + b\bar{b}$-Prozess werden als vollständiger, ungerichteter und statischer Graph dargestellt. Jede ungerichtete Kante kann im Folgenden aber auch als zwei entgegengesetzt gerichtete Kanten interpretiert werden. Dies führt dazu, dass jeder Knoten Informationen mit jedem anderen Knoten austauscht und sich die Graphstruktur nicht verändert. Jeder Kante zwischen zwei Knoten i und j wird der geometrische Abstand ΔR (s. Formel 2.3) der von i und j repräsentierten Jets (oder Lepton) als Gewicht zugeordnet. Weiterhin wird jedem Knoten i ein initialer Feature-Vektor $\boldsymbol{v}_i^{(0)}$ zugewiesen:

$$\boldsymbol{v}_i^{(0)} = \left(p_\mathrm{T}, \phi, \eta, M, E, \tilde{b}\right)^\mathrm{T}. \tag{4.1}$$

Dieser besteht aus den kinematischen Größen (s. Abschnitt 2.2.3) und seinem b-tag-Wert $\tilde{b}$ (s. Abschnitt 2.3) und besitzt die Dimension $m = 6$. Ein beispielhafter Graph, der ein Ereignis mit sieben Jets repräsentiert, ist in Abb. 4.1 abgebildet. Die Zahl der Knoten im Graph entspricht der Anzahl der Jets plus ein Knoten für das Lepton. Hier kommt ein Vorteil von GNNs zum Tragen: Da die Graphstruktur nicht fest vorgegeben sein muss, können Ereignisse mit unterschiedlicher Jet-Multiplizität an das GNN übergeben werden.

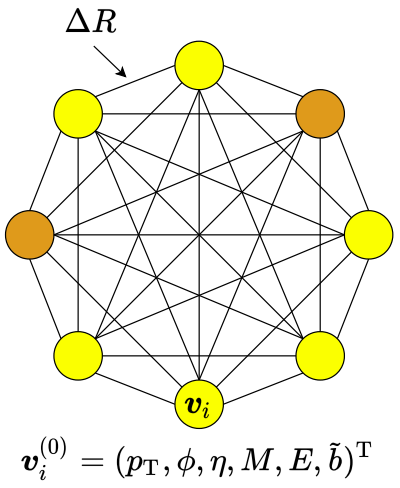


Abbildung 4.1: Detektorobjekte des $\mathrm{t\bar{t}} + \mathrm{b\bar{b}}$-Prozesses als vollständiger Graph: Die Knoten repräsentieren die rekonstruierten Jets, wobei einer der Knoten dem Lepton entspricht. Jeder Verbindung wird der entsprechende ΔR-Wert als Gewicht zugeordnet. Jeder Knoten erhält einen initialen Feature-Vektor $\boldsymbol{v}_i^{(0)}$ mit seinen kinematischen Größen und dem b-tag-Wert. Die beiden orangefarbenen Knoten repräsentieren die AddB-Jets, die gelben Knoten die anderen Jets und das Lepton. Die Anzahl der Knoten im Graph hängt von der Jet-Multiplizität des Ereignisses ab. Herauszufinden, welche zwei Knoten den AddB-Jets entsprechen, ist Aufgabe des GNNs.

Die Aufgabe des GNNs ist es nun, durch diverse Berechnungsschritte die initialen Feature-Vektoren der Knoten so zu transformieren, dass jeder Knoten nur noch durch eine Zahl zwischen 0 und 1 repräsentiert wird. Diese Zahl soll als Maß dafür dienen, ob es sich bei dem entsprechenden Knoten um einen AddB-Jet handelt. Hierfür wird ein Graph mit seinem initialen Node-Embedding $\tilde{V}^{(0)} = (\boldsymbol{v}_1^{(0)}, \boldsymbol{v}_2^{(0)}, ...)^\mathrm{T}$ an das GNN übergeben, welches einen Graphen mit gleicher Graphstruktur, allerdings mit aktualisiertem Node-Embedding $\tilde{V}^{\mathrm{final}}$ zurückgibt (s. Abb. 4.2). Die beiden Knoten mit den höchsten Zahlenwerten werden dann als AddB-Jets klassifiziert.

Der Erfolg des GNNs wird anhand folgender Kriterien bewertet:

- **TPR**: Die True-Positive-Rate (TPR) des GNNs beschreibt, welchen Prozentsatz der wahren AddB-Jets das GNN auch als AddB-Jets klassifiziert.
- **TNR**: Die True-Negative-Rate (TNR) des GNNs beschreibt, welchen Prozentsatz der wahren nicht AddB-Jets das GNN auch als nicht AddB-Jets klassifiziert.
- **2/2**: Diese Größe gibt an, in wie viel Prozent der Fälle das GNN beide AddB-Jets eines Kollisionsereignisses identifiziert hat.

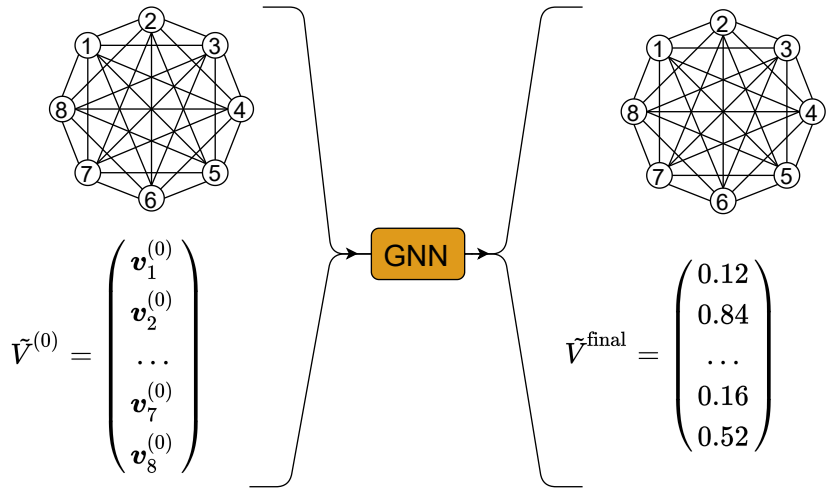


Abbildung 4.2: Klassifikation der AddB-Jets mittels GNNs: das hier verwendete GNN nimmt einen Graphen als Input und gibt einen Graphen mit gleicher Struktur zurück. Das ursprüngliche Node-Embedding $\tilde{V}^{(0)}$ wird allerdings durch verschiedene Berechnungen zum finalen Node-Embedding $\tilde{V}^{\text{final}}$ transformiert. In diesem Anwendungsfall soll jeder Knoten nach Durchlaufen des GNNs nur noch durch eine Zahl repräsentiert werden, die als Maß dafür dient, ob es sich bei einem Knoten um den AddB-Jet handelt.

- **1/2**: Diese Größe gibt an, in wie viel Prozent der Fälle das GNN einen der beiden AddB-Jets eines Kollisionsereignisses identifiziert hat.
- **0/2**: Diese Größe gibt an, in wie viel Prozent der Fälle das GNN keinen der beiden AddB-Jets eines Kollisionsereignisses identifiziert hat.

Im Folgenden wird das Modell als „am besten“ bezeichnet, welches bei vergleichbarer TNR die beste TPR aufweist, da dieses Modell dann insgesamt die meisten AddB-Jets findet. Sollte sich die Bezeichnung „bestes Modell“ auf eine andere Größe beziehen, wird dies explizit erwähnt.

4.2 Trainings- und Evaluationsprozess

Das Training und die Evaluation verlaufen folgenderweise:

1) Für eine bessere Reproduzierbarkeit der Ergebnisse werden die Startwerte der relevanten Zufallsgeneratoren festgelegt.

2) Alle Input-Größen des Graphen werden zu Beginn normiert. Dadurch nehmen die verschiedenen Größen im Feature-Vektor, die sich über unterschiedliche Wertebereiche erstrecken, nach der Normierung den gleichen Wertebereich ein. Die relativen Abstände bleiben dabei jedoch erhalten. Dadurch kann das Modell die Größen besser vergleichen. Alle initialen Feature-Vektoren werden so normiert, dass die einzelnen Größen im Feature-Vektor einen Mittelwert von 0 und eine Standardabweichung von 1 aufweisen:

$$x' = \frac{x - \bar{x}}{\sigma}. \tag{4.2}$$

Dabei ist x' die transformierte Größe, $\bar{x}$ der Mittelwert und σ die Standardabweichung von x. Alle Gewichte von Kanten durchlaufen eine lineare Normierung, womit sich die Werte zwischen 0 und 1 bewegen:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}. \tag{4.3}$$

3) Der Datensatz (28.425 Ereignisse) wird durchmischt und in Trainings-, Validations- und Test-Datensatz eingeteilt. Dabei werden 60 % der Graphen dem Trainings-Datensatz und je 20 % dem Validations- und Test-Datensatz zugeordnet.

4) Das Training umfasst 100 Epochen. Die Graphen werden in Batches in das Modell geladen und die Zusammensetzung der Batches jede Epoche neu durchmischt. Die Gewichte werden nach jedem Batch via Rückpropagierung so aktualisiert, dass die Loss-Funktion minimiert wird (s. Abschnitt 3.4). Für die Loss-Funktion wird die Binary Cross Entropy verwendet. Der Loss-Wert eines Knoten i ergibt sich somit aus

$$l_i = -w_i \left(y_i \cdot \ln h_i + (1 - y_i) \cdot \ln 1 - h_i \right). \tag{4.4}$$

Dabei ist $y_i \in \{0, 1\}$ die wahre Information und $h_i \in (0, 1)$ die Vorhersage des Modells. Knoten, die einen AddB-Jet repräsentieren, werden entsprechend dem Verhältnis zu den anderen Knoten im aktuellen Batch, mehrfach gezählt. Dies wird durch das Gewicht w_i berücksichtigt. Wurden alle Batches aus dem Trainings-Datensatz in das Modell geladen, gilt die Epoche als beendet. Als Optimierungsalgorithmus wird Adam [29] verwendet, der für viele Anwendungen im Bereich des maschinellen Lernens Standard ist.

5) Nach jeder Epoche wird der Validations-Datensatz an das Modell übergeben und ausgewertet. Eine Aktualisierung der Gewichte im Modell findet nicht statt. Die Gewichte des Modells werden gespeichert, falls die TPR des Validations-Datensatzes einen neuen Höchstwert erreicht hat.

6) Anpassung der Lernrate: Die Lernrate beträgt zu Beginn 0,01 und wird um Faktor 0,1 verkleinert, wenn es nach 10 Epochen keine Verbesserung von 0,5 % auf die TPR des Validations-Datensatzes gegeben hat. Diese Werte haben sich im Verlauf dieser Studie als besonders vielversprechend erwiesen.

7) Verfrühter Stopp: Das Training wird verfrüht abgebrochen, wenn es nach 15 Epochen keine Verbesserung von 0,01 % auf die TPR des Validations-Datensatzes gegeben hat.

8) Nach Beenden des Trainings wird der Test-Datensatz an das Modell übergeben und ausgewertet. Dafür wird das abgespeicherte Modell aus der besten Epoche verwendet (Epoche mit höchster TPR auf dem Validations-Datensatz).

4.3 Auswertung GGS-NN-Modell

Im Folgenden wird das Gated Graph Sequence-Neural Network-Modell (GGS-NN-Modell) vorgestellt und ausgewertet. Den Kern des Modells bildet das GGS-NN, im Folgenden auch GGS-NN-Modul genannt, wie es in Abschnitt 3.3 beschrieben wird.

4.3.1 GGS-NN-Modell

Das GGS-NN-Modul kann nur Feature-Vektoren zurückgeben, deren Dimension größer oder gleich der initialen Feature-Vektoren ist. Um dem Lösungsansatz aus Abschnitt 4.1 gerecht zu werden, besteht das GGS-NN-Modell also aus einem GGS-NN-Modul mit einer anschließenden linearen Transformation der vom GGS-NN-Modul ausgegebenen Feature-Vektoren $\boldsymbol{h}_i^{(L)} \in \mathbb{R}^n$. Diese lineare Transformation sorgt dafür, dass jeder Knoten nur noch durch eine Zahl repräsentiert wird. Die lineare Transformation wird mit einer voll verbundenen Lage (engl.: Fully Connected Layer) aus herkömmlichen neuronalen Netzen (NN) durchgeführt und operiert nicht auf einem Graphen wie das GGS-NN-Modul, sondern nur auf den Feature-Vektoren selbst:

$$h_i' = A \cdot \boldsymbol{h}_i^{(L)} \tag{4.5}$$

Dabei ist A eine Matrix der Größe $1 \times n$, deren Einträge im Verlauf des Trainings erlernt werden und n ist ein Hyperparameter des GGS-NN-Moduls. Dies sorgt dafür, dass nun jeder Knoten nur noch durch eine Zahl $h_i' \in \mathbb{R}$ beschrieben wird. Abschließend wird auf h_i' die Sigmoid-Funktion angewandt, um die finale Repräsentation $h_i \in (0, 1)$ des Knotens zu erhalten:

$$h_i = \sigma(h_i') = \frac{1}{1 + e^{-h_i'}} \tag{4.6}$$

Eine Darstellung des Modells ist in Abb. 4.3 gegeben.

4.3.2 Hyperparameter

Im Folgenden werden alle Kombinationen der verschiedenen Hyperparameter getestet:

- **Batch-Größe**: Die Batch-Größe bestimmt, wie viele Graphen in das Modell geladen werden, bevor der Loss-Wert berechnet und die Gewichte aktualisiert werden:

$$\text{Batch-Größe} \in \{200, 500, 800\} \tag{4.7}$$

- **Dimension n**: Die Dimension n bestimmt die Dimension der Feature-Vektoren beim und nach Durchlaufen des GGS-NN-Moduls:

$$\text{Dimension } n \in \{6, 12, 18, 24\} \tag{4.8}$$

- **Sequenzlänge L**: Die Sequenzlänge L bestimmt, wie viele GN-Blöcke das GGS-NN-Modul ausführt:

$$\text{Sequenzlänge } L \in \{6, 12, 18\} \tag{4.9}$$

- **Aggregatfunktion**: Die Funktion, welche die Nachrichten der Nachbarknoten aggregiert (s. Gleichungen 3.6)

$$\text{Aggregatfunktion} \in \{\text{Summe}, \text{Mittelwert}\} \tag{4.10}$$

- **Bias**: Implementiert die zusätzlichen Vektoren $\boldsymbol{b}$ als Bias in der Gated Recurrent Unit (s. Gleichungen 3.8):

$$\text{Bias} \in \{\text{True}, \text{False}\} \tag{4.11}$$

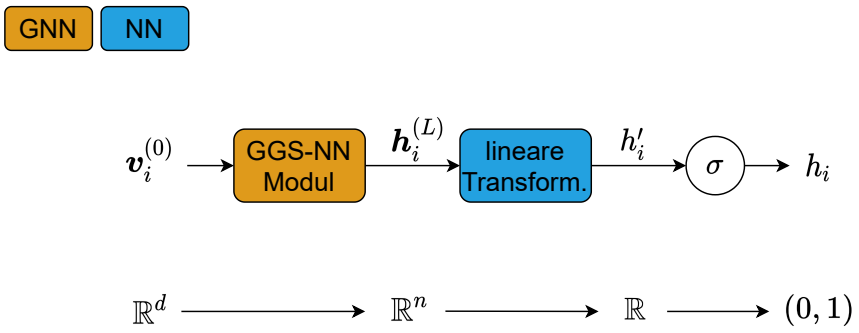


Abbildung 4.3: Darstellung des GGS-NN-Modells: Das GGS-NN-Modell nimmt die initialen Feature-Vektoren der Dimension m und leitet sie zunächst in das GGS-NN-Modul, welches die aktualisierten Feature-Vektoren $\mathbf{h}_i^{(L)} \in \mathbb{R}^n$ mit $n \geq m$ zurückgibt. Die aktualisierten Feature-Vektoren durchlaufen eine lineare Transformation, sodass jeder Knoten durch eine Zahl $h_i' \in \mathbb{R}$ repräsentiert wird. Als Aktivierungsfunktion dient die Sigmoid-Funktion, welche die finale Repräsentation $h_i \in (0,1)$ des Knotens ausgibt. Der orangefarbene Block repräsentiert ein GNN, welches auf einen Graphen angewendet wird und somit Berechnungen basierend auf der Graphstruktur und den Graphattributen durchführt. Der blaue Block hingegen repräsentiert ein herkömmliches neuronales Netz, das auf jeden Feature-Vektor einzeln angewendet wird.

4.3.3 Auswertung

Die Auswertung basiert auf dem Trainings- und Evaluationsprozess, der in 4.2 beschrieben ist. Jede der 144 Hyperparameterkombinationen durchläuft den Trainings- und Evaluationsprozess 20 mal, um Mittelwerte und Standardabweichungen der Evaluationsgrößen zu bestimmen.

In Abbildung 4.4 ist dargestellt, zu welchen durchschnittlichen TPRs die unterschiedlichen Hyperparameterkombinationen führen. Es ist deutlich zu erkennen, dass der wichtigste Hyperparameter die Dimension n ist. Die besten Ergebnisse werden mit $n = 24$ erzielt und nehmen mit sinkendem n ab. Somit erscheint es auf den ersten Blick logisch, noch größere Werte für n zu wählen. Allerdings hat die beste Hyperparameterkombination mit $n = 24$ durchschnittlich 71,85 % TPR erreicht. Mit $n = 18$ konnte bis zu 71,65 % TPR, mit $n = 12$ bis zu 71,38 % TPR und mit $n = 6$ bis zu 70,37 % TPR erreicht werden. Die Verbesserungen nehmen also mit steigendem n zwar zu, werden allerdings auch immer kleiner. Weiterhin beginnen die Modelle mit größerem n die Daten zu übertrainieren. Die Trainingsdaten werden immer besser bewertet, während die Ergebnisse der Validationsdaten schlechter werden. Aus diesen beiden Gründen werden keine größeren Werte für n verwendet.

Weiterhin erzielt auch eine kleine Batch-Größe von 200 bessere Ergebnisse als große Batch-Größen. Allerdings konnte mit einer Batch-Größe von 200 maximal 71,85 % TPR, mit 500 maximal 71,66 % TPR und mit 800 maximal 71,65 % TPR erreicht werden. Der Einfluss ist also deutlich geringer als der der Dimension n. Aus diesem Grund, und um die Trainingsdauer nicht noch weiter zu erhöhen, wird auch keine kleinere Batch-Größe verwendet. Bei der Sequenzlänge L, der Aggregatfunktion und dem Bias werden die Unterschiede in den Ergebnissen immer geringer.

Dies spiegelt sich auch in Tabelle 4.1 wieder, welche die Mittelwerte und Standardabweichungen der sechs besten und sechs schlechtesten Ergebnisse der verschiedenen Hyperparameterkombinationen zeigt. Die besten Ergebnisse haben alle eine Batch-Größe von 200 und die Dimension $n = 24$. Die schlechtesten Ergebnisse haben eine Batch-Größe von 800 und $n = 6$. Weiterhin haben die sechs besten Kombinationen tendenziell eher größere Sequenzlängen. Allerdings wird es im oberen Bereich schwierig, zwischen schlechteren und besseren Hyperparameterkombinationen zu unterscheiden. Die Unterschiede der verschiedenen Evaluationsgrößen liegen sehr nah beieinander und innerhalb der einzelnen Standardabweichungen. Weiterhin ist durch lediglich 20 Wiederholungen auch keine große statistische Signifikanz gegeben. Dennoch sollen in den weiteren Untersuchungen die Hyperparameter aus der zweiten Zeile in Tabelle 4.1 verwendet werden, da hier durchschnittlich die größten 2/2-Werte erzielt worden sind.

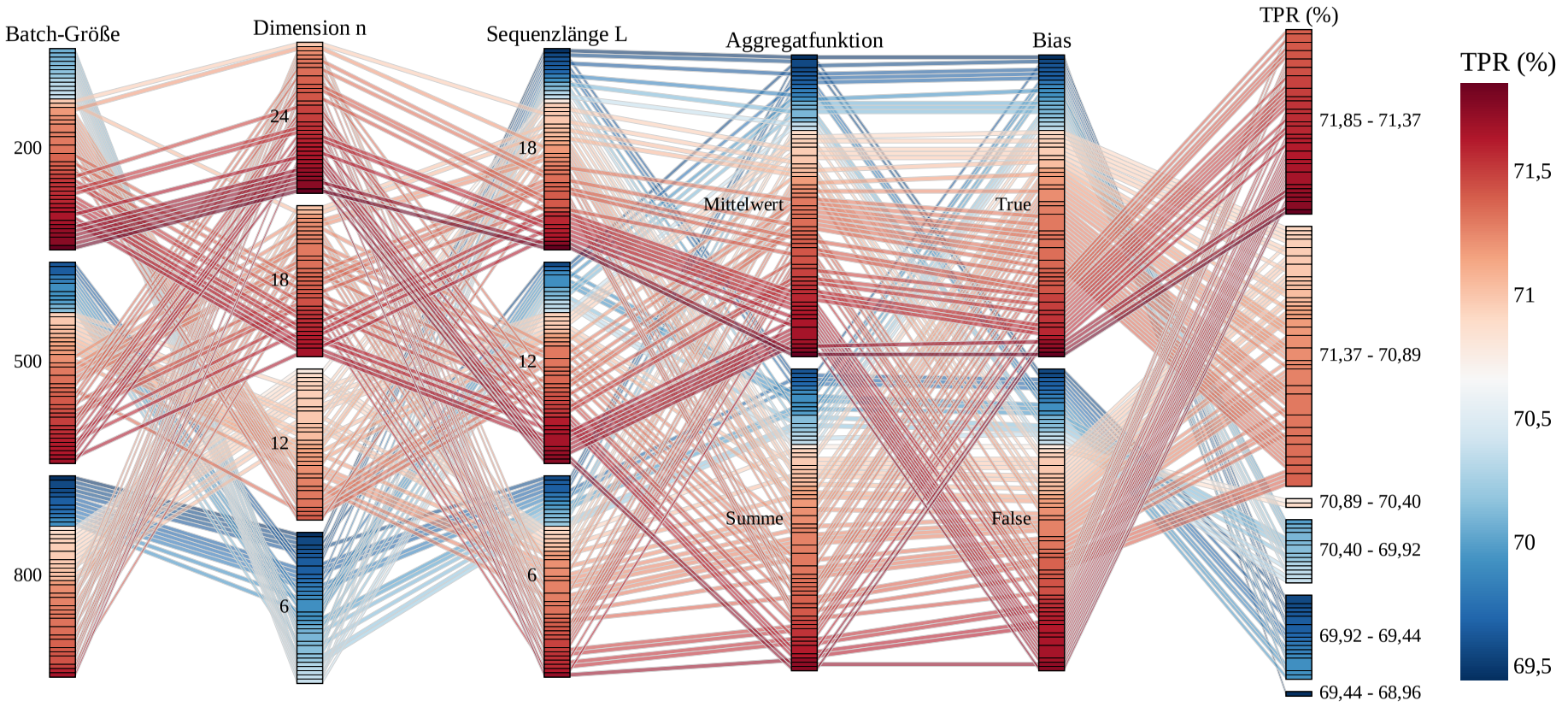


Abbildung 4.4: Hyperparameterkombinationen des GGS-NN-Modells: Die Abbildung beschreibt, welche Hyperparameterkombinationen bei 20-facher Wiederholung zu welcher durchschnittlichen TPR führen. Jede Box repräsentiert einen Hyperparameter. Jede Linie stellt eine Hyperparameterkombination dar und verläuft durch die entsprechenden Boxen. Dabei erreichen rot eingefärbte Linien höhere TPRs als blau eingefärbte Linien. Je mehr rote Linien eine Box aufweist, desto öfter erreicht das Modell mit diesem Hyperparameter (in Kombination mit anderen Hyperparametern) eine höhere TPR.

Tabelle 4.1: Durchschnittliche Ergebnisse mit Standardabweichung des GGS-NN-Modells für verschiedene Hyperparameterkombinationen. Dargestellt sind die sechs besten und sechs schlechtesten der 144 Hyperparameterkombinationen sortiert nach TPR.

	(Batch-Größe, n, L, Aggregatfunktion, Bias)	TPR (%)	TNR (%)	2/2 (%)	1/2 (%)	0/2 (%)
1	(200, 24, 18, Mittelwert, True)	71,85 ± 0,31	90,50 ± 0,11	52,01 ± 0,44	39,68 ± 0,51	8,31 ± 0,37
2	(200, 24, 18, Mittelwert, False)	71,74 ± 0,29	90,47 ± 0,10	52,06 ± 0,39	39,36 ± 0,42	8,58 ± 0,33
3	(200, 24, 12, Summe, True)	71,74 ± 0,36	90,47 ± 0,12	51,93 ± 0,52	39,62 ± 0,48	8,45 ± 0,31
4	(200, 24, 12, Summe, False)	71,72 ± 0,29	90,46 ± 0,10	51,88 ± 0,48	39,68 ± 0,48	8,44 ± 0,24
5	(200, 24, 6, Summe, True)	71,71 ± 0,31	90,46 ± 0,11	51,76 ± 0,53	39,90 ± 0,50	8,34 ± 0,21
6	(200, 24, 12, Mittelwert, True)	71,70 ± 0,44	90,45 ± 0,15	51,76 ± 0,59	39,89 ± 0,50	8,35 ± 0,41
⋮						
139	(500, 6, 18, Mittelwert, True)	69,62 ± 0,55	89,75 ± 0,19	48,50 ± 0,99	42,22 ± 0,92	9,27 ± 0,23
140	(800, 6, 18, Summe, True)	69,61 ± 0,47	89,75 ± 0,16	48,57 ± 0,81	42,08 ± 0,90	9,35 ± 0,43
141	(800, 6, 6, Mittelwert, False)	69,56 ± 0,36	89,73 ± 0,12	48,21 ± 0,57	42,69 ± 0,57	9,10 ± 0,30
142	(800, 6, 18, Mittelwert, False)	69,55 ± 0,49	89,73 ± 0,16	48,45 ± 0,73	42,21 ± 0,60	9,35 ± 0,36
143	(800, 6, 12, Mittelwert, True)	69,55 ± 0,50	89,73 ± 0,17	48,41 ± 0,84	42,29 ± 0,81	9,30 ± 0,33
144	(800, 6, 18, Mittelwert, True)	69,44 ± 0,70	89,69 ± 0,24	48,38 ± 1,19	42,13 ± 1,02	9,50 ± 0,28

4.4 Auswertung GGS-NN-Modell mit MLP

4.4.1 GGS-NN-Modell mit MLP

Bei der Wahl der Architektur verschiedener GNN-Modelle kann es hilfreich sein, diese mit herkömmlichen neuronalen Netzen zu kombinieren. Dies wird bereits im GGS-NN-Modell mit einer linearen Transformation umgesetzt, die auf jeden Knoten angewandt wird. Aber auch die Verwendung einer Lage zur Vorprozessierung (engl.: Pre-Process Layer) in Form eines Multi-Lagen-Perzeptrons (engl.: Multilayer Perceptron, MLP) kann eine sinnvolle Ergänzung für GNNs sein [30]. Dabei durchlaufen die initialen Feature-Vektoren $\boldsymbol{v}_i^{(0)}$ jedes Knotens ein MLP, das die Vektoren $\boldsymbol{v}_i'^{(0)}$ zurückgibt. Der Graph mit den neuen Feature-Vektoren $\boldsymbol{v}_i'^{(0)}$ wird dann an das GGS-NN-Modell übergeben (s. Abb. 4.5).

Dieses Vorgehen ermöglicht den einzelnen Knoten, die kinematischen Größen in ihrem Feature-Vektor zunächst untereinander zu kombinieren und daraus High-Level-Features zu extrahieren, ohne durch die Feature-Vektoren der anderen Knoten beeinflusst zu werden. Im Optimalfall enthält die neue Repräsentation jedes Knoten $\boldsymbol{v}_i'^{(0)}$ zusätzliche und tiefergreifende Informationen über den Knoten selbst und besitzt eine größere Aussagekraft über die Zugehörigkeit zur Kategorie AddB-Jet.

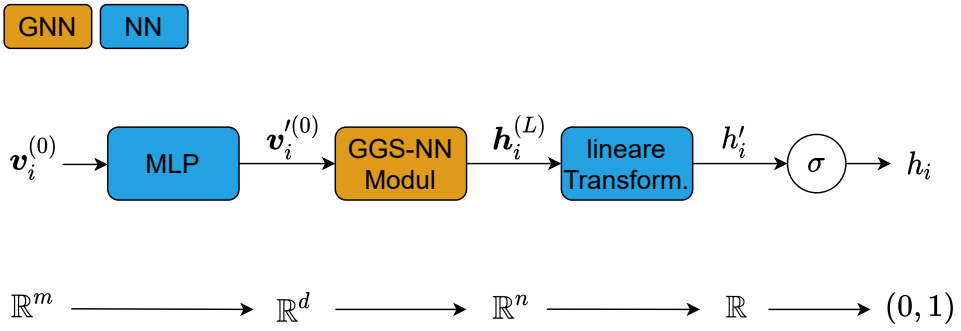


Abbildung 4.5: Darstellung des GGS-NN-Modells mit MLP: Hier schaltet das Modell eine Lage zur Vorprozessierung in Form eines Multi-Lagen-Perzeptrons vor das GGS-NN-Modell.

Das MLP besteht aus drei voll verbundenen Lagen der Form

$$\boldsymbol{y} = A \cdot \boldsymbol{x} + \boldsymbol{b}. \tag{4.12}$$

In diesem Fall ist A in der ersten Lage eine Matrix der Dimension $d \times m$ und in der zweiten und dritten Lage von der Dimension $d \times d$. Bei d handelt es sich um die MLP-Dimension, einen frei wählbaren Hyperparameter, der die Dimension der Feature-Vektoren nach Durchlaufen des MLPs bestimmt. Ein zusätzlicher Bias wird durch den Vektor $\boldsymbol{b}$ beschrieben. Eine beispielhafte Darstellung des MLPs für $d = 4$ ist in Abb. 4.6 gegeben. Nach jedem der drei Lagen durchlaufen die Einträge der Vektoren $\boldsymbol{y}$ eine Aktivierungsfunktion. In der Auswertung werden verschiedene Aktivierungsfunktionen und Werte für d getestet.

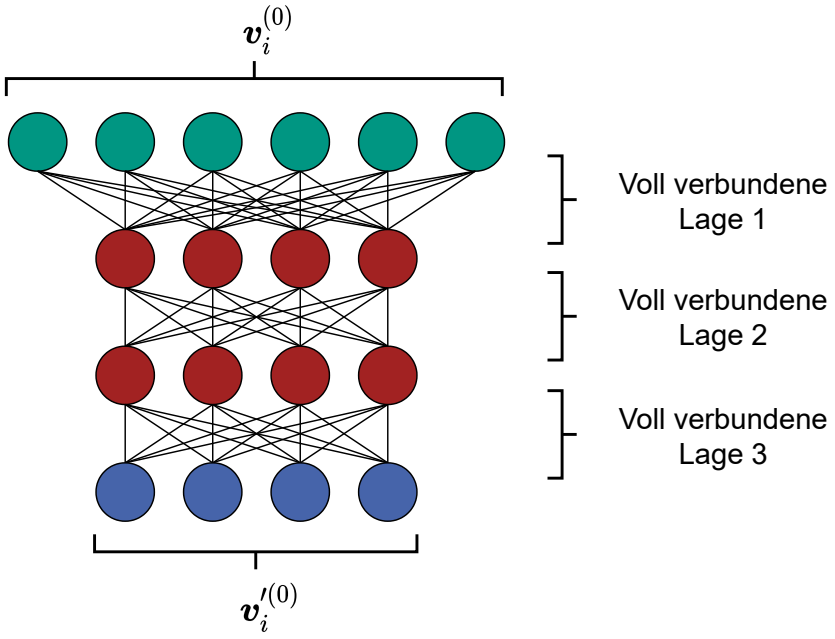


Abbildung 4.6: Multi-Lagen-Perzeptron: Das MLP dient als Lage zur Vorprozessierung für das GGS-NN-Modell. Es besteht aus drei voll verbundenen Lagen mit einem Bias. Nach jeder Lage durchlaufen die Vektoren eine Aktivierungsfunktion. Die Dimension der Output-Vektoren $\boldsymbol{v}_i'^{(0)}$ kann beliebig gewählt werden, in diesem Fall beträgt sie vier.

4.4.2 Hyperparameter

Basierend auf der Auswertung in Abschnitt 4.3.1 werden die Hyperparameter des GGS-NN-Modells verwendet, welches durchschnittlich die besten 2/2-Werte erreicht hat. Diese sind in Tabelle 4.1 in der zweiten Zeile zu finden: Batch-Größe = 200, Dimension $n = 24$, Sequenzlänge $L = 18$, Aggregatfunktion = Mittelwert und Bias = False (bezieht sich nur auf den Bias in der GRU des GGS-NN-Modells und nicht auf den Bias im MLP).

Somit werden im Folgenden zwei Hyperparameter für das MLP getestet:

- **Aktivierungsfunktion**: Die Aktivierungsfunktion, welche nach jeder voll verbundenen Lage angewendet wird:

$$\text{Aktivierungsfunktion} \in \{\text{Sigmoid}, \tanh, \text{ReLU}, \text{ELU}\} \tag{4.13}$$

  Die Aktivierungsfunktionen werden elementweise auf die Vektoren $\boldsymbol{y}$ angewendet:

$$\begin{aligned}
\sigma(\boldsymbol{y}) &= \frac{1}{1+e^{-\boldsymbol{y}}} \\
\tanh(\boldsymbol{y}) &= \frac{e^{\boldsymbol{y}} - e^{-\boldsymbol{y}}}{e^{\boldsymbol{y}} + e^{-\boldsymbol{y}}} \\
\text{ReLU}(\boldsymbol{y}) &= \max(0, \boldsymbol{y}) \\
\text{ELU}(\boldsymbol{y}) &= \begin{cases} \boldsymbol{y}, & \text{falls } \boldsymbol{y} > 0 \\ e^{\boldsymbol{y}} - 1, & \text{sonst} \end{cases}
\end{aligned}$$

- **MLP-Dimension *d***: Bestimmt die Größen der Matrizen A in den drei voll verbundenen Lagen und somit die Dimension des Feature-Vektor nach Durchlaufen des MLPs:

$$\text{MLP-Dimension } d \in \{4, 6, 12\} \tag{4.14}$$

4.4.3 Auswertung

Die Auswertung erfolgt analog zum vorherigen Abschnitt. Tabelle 4.2 stellt die Ergebnisse für die zwölf verschiedenen MLPs dar. Während das GGS-NN-Modell mit den hier verwendeten Hyperparametern durchschnittlich eine TPR von 71,74 % und einen 2/2-Wert von 52,06 % erreicht hat, erzielt dieses GGS-NN-Modell mit MLP durchschnittlich nur maximal 71,35 % TPR und 51,26 % beim 2/2-Wert. Somit schneidet das Modell mit MLP geringfügig schlechter ab.

Mit dem MLP in seiner bisherigen Form ist also nicht möglich, relevante Informationen aus der initialen Repräsentation eines einzelnen Knotens herauszufiltern. Das bedeutet, dass die Repräsentation eines Knotens durch die Jet-Kinematiken bereits sehr aussagekräftig ist und nicht durch Berechnungen basierend auf dem Feature-Vekor selbst verbessert werden kann, zumindest nicht mit den MLPs, wie sie hier verwendet werden.

Tabelle 4.2: Durchschnittliche Ergebnisse mit Standardabweichung des GGS-NN-Modells mit MLP für verschiedene Hyperparameterkombinationen.

(d, Aktivierung)	TPR (%)	TNR (%)	2/2 (%)	1/2 (%)	0/2 (%)
(12, ReLU)	71,35 ± 0,56	90,34 ± 0,19	51,23 ± 0,86	40,25 ± 0,68	8,52 ± 0,34
(4, ELU)	71,28 ± 0,70	90,31 ± 0,24	51,26 ± 0,96	40,03 ± 0,61	8,71 ± 0,49
(12, ELU)	71,22 ± 0,68	90,29 ± 0,23	51,18 ± 1,01	40,08 ± 0,73	8,75 ± 0,42
(6, ELU)	71,22 ± 0,76	90,29 ± 0,26	51,12 ± 0,99	40,21 ± 0,58	8,67 ± 0,59
(12, tanh)	71,11 ± 0,78	90,25 ± 0,26	50,98 ± 0,96	40,25 ± 0,48	8,77 ± 0,64
(6, ReLU)	71,07 ± 0,55	90,24 ± 0,19	50,89 ± 0,80	40,36 ± 0,62	8,75 ± 0,40
(12, Sigmoid)	71,06 ± 0,77	90,24 ± 0,26	50,90 ± 0,92	40,32 ± 0,47	8,78 ± 0,67
(6, tanh)	70,84 ± 1,16	90,16 ± 0,39	50,66 ± 1,54	40,35 ± 0,88	8,99 ± 0,85
(4, ReLU)	70,64 ± 0,65	90,09 ± 0,22	50,42 ± 0,82	40,44 ± 0,56	9,14 ± 0,58
(4, tanh)	70,63 ± 1,46	90,09 ± 0,49	50,44 ± 1,87	40,38 ± 0,92	9,18 ± 1,09
(6, Sigmoid)	69,77 ± 0,56	89,80 ± 0,19	49,26 ± 0,75	41,03 ± 0,48	9,72 ± 0,44
(4, Sigmoid)	69,47 ± 0,31	89,70 ± 0,11	48,98 ± 0,50	40,98 ± 0,46	10,04 ± 0,23

4.5 Auswertung Multi-GGS-NN-Modell

4.5.1 Multi-GGS-NN-Modell

Als letztes wird das Multi-GGS-NN-Modell untersucht. Dieses schaltet drei GGS-NN-Module hintereinander und kann mit oder ohne Skip-Connection implementiert werden. Ohne Skip-Connection erfolgt die Informationsverarbeitung analog zum GGS-NN-Modell aus Abschnitt 4.3.1 bis auf die Verwendung von drei GGS-NN-Modulen. Bei Implementierung der Skip-Connection wird der ursprüngliche Feature-Vektor $\boldsymbol{v}_i^{(0)} \in \mathbb{R}^m$ dem Output Feature-Vektor $\boldsymbol{h}_i^{(L_1+L_2)} \in \mathbb{R}^{n_2}$ des zweiten GGS-NN-Modul angehängt:

$$\boldsymbol{h}_i'^{(L_1+L_2)} = \left(\boldsymbol{h}_i^{(L_1+L_2)}, \boldsymbol{v}_i^{(0)}\right)^{\mathrm{T}} \quad \text{mit} \quad \boldsymbol{h}_i'^{(L_1+L_2)} \in \mathbb{R}^{n_2+m}. \tag{4.15}$$

Danach wird der Feature-Vektor $\boldsymbol{h}_i'^{(L_1+L_2)}$ an das dritte GGS-NN-Modul übergeben. Dies hat den Vorteil, dass das dritte GGS-NN-Modul nun die bereits verarbeiteten Informationen aus den vorangegangenen GGS-NN-Modulen nochmal mit den initialen Feature-Vektoren kombinieren kann. Dadurch soll sichergestellt werden, dass relevante Informationen der initialen Repräsentation im Modell nicht verloren gehen. Für die Dimensionen n_k der GGS-NN-Module mit $k \in \{1, 2, 3\}$ gilt $n_1 \leq n_2 \leq n_3$ (ohne Skip-Connection) bzw. $n_1 \leq n_2$ und $n_2 + m \leq n_3$ (mit Skip-Connection). Eine graphische Darstellung des Modells ist in Abb. 4.7 gegeben.

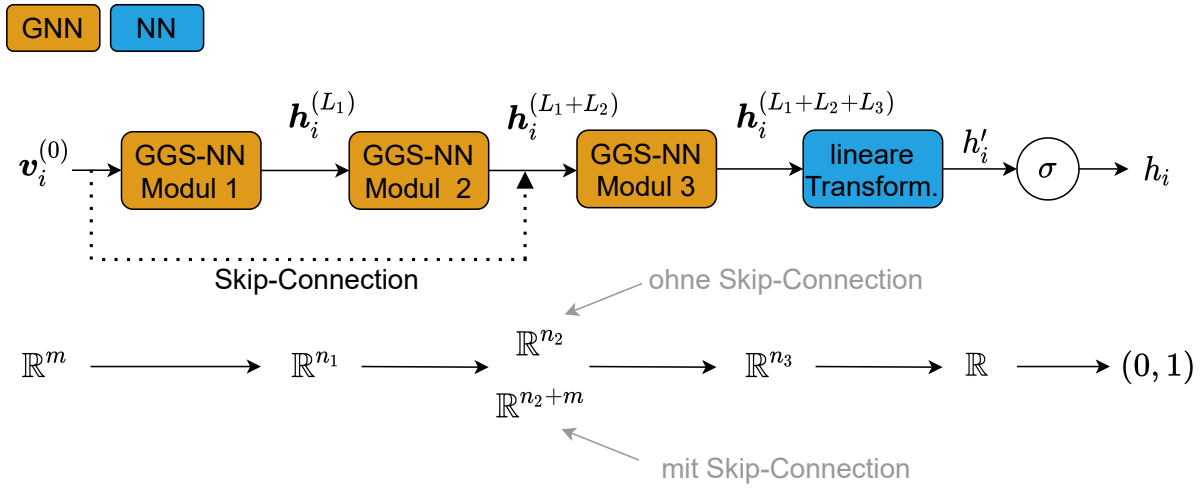


Abbildung 4.7: Darstellung des Multi-GGS-NN-Modells mit und ohne Skip-Connection: Das Multi-GGS-NN-Modell ohne Skip-Connection schaltet drei GGS-NN-Module hintereinander, bevor sie an die lineare Transformation und die Sigmoid-Funktion übergeben werden. Bei Implementierung der Skip-Connection wird der initiale Feature-Vektor dem Output des zweiten GGS-NN-Moduls angehängt und zusammen an das dritte GGS-NN-Modul übergeben.

4.5.2 Hyperparameter

Auch hier werden basierend auf den besten Ergebnissen bezüglich des 2/2-Wertes aus Abschnitt 4.3.3 folgende Hyperparameter verwendet: Batch-Größe = 200, Aggregatfunktion

= Mittelwert, Bias = False und Dimensionen $n_k = 24$ (Hyperparameter gelten für alle drei GGS-NN-Module).

Weiterhin werden folgende Hyperparameter getestet:

- **Skip-Connection**: Bestimmt, ob die Skip-Connection implementiert wird oder nicht:

$$\text{Skip-Connection} \in \{\text{True}, \text{False}\} \tag{4.16}$$

  Je nachdem, ob eine Skip-Connection implementiert wird, hat dies Einfluss auf die Dimensionen n_3 des dritten GGS-NN-Moduls. Diese Dimension beträgt bei Implementierung der Skip-Connection $n_3 = 24 + m$.

- **Sequenzlängen** L_k: Die Sequenzlängen L_k der einzelnen GGS-NN-Module:

$$\text{Sequenzlängen } L_k \in \{6, 18\} \tag{4.17}$$

  Die Idee der beiden Sequenzlängen ist folgende: Für $L_k = 6$ entsprechen die hier verwendeten GGS-NN-Module dem GGS-NN-Modul des besten GGS-NN-Modells bezüglich 2/2-Wert (s. Tabelle 4.1) aus Abschnitt 4.3, nur dass nach sechs Sequenzen die GRU andere Gewichte besitzt. Für $L_k = 18$ wird das GGS-NN-Modul des besten GGS-NN-Modells drei Mal hintereinander geschaltet (ebenfalls mit drei eigenen GRUs).

4.5.3 Auswertung

In Tabelle 4.3 sind die durchschnittlichen Ergebnisse des Multi-GGS-NN-Modells dargestellt. Das Einführen einer Skip-Connection trägt zu keiner Verbesserung der Ergebnisse bei. Ebenso schneidet das Multi-GGS-NN-Modell ohne Skip-Connection und mit $L_k = 18$ am schlechtesten ab. Das Multi-GGS-NN-Modell ohne Skip-Connection und $L_k = 6$ schneidet mit TPR = 71,87 % und 2/2 = 52,22 % unter den getesteten Kombinationen am besten ab. Damit ist es auch insgesamt das bisher beste Modell, sowohl bezüglich der TPR, TNR und des 2/2-Wertes. Dieses Modell ist dem GGS-NN-Modell mit den besten 2/2-Werten (Tabelle 4.1, Zeile 2) sehr ähnlich. Der einzige Unterschied besteht darin, dass hier drei GRUs mit eigenen Gewichten verwendet werden. Das Modell könnte auch als GGS-NN-Modell beschrieben werden, nur dass nach sechs Sequenzen die GRU andere Gewichte besitzt. Auffällig sind noch die hohen Standardabweichungen beim Multi-GGS-NN-Modell ohne Skip-Connection und $L_k = 18$. Diese kommen allerdings dadurch zustande, dass ein paar Durchläufe deutlich schlechtere Ergebnisse erzielt haben. Wie auch zuvor sind hier die Aussagen über die Güte eines Modells mit Vorsicht zu genießen, da die einzelnen Ergebnisse innerhalb der Standardabweichungen liegen und nur 20 Wiederholungen durchgeführt worden sind.

Tabelle 4.3: Durchschnittliche Ergebnisse mit Standardabweichung des Multi-GGS-NN-Modells für verschiedene Hyperparameterkombinationen.

(Skip-Conn., L_k)	TPR (%)	TNR (%)	2/2 (%)	1/2 (%)	0/2 (%)
(False, 6)	71,87 ± 0,31	90,51 ± 0,11	52,22 ± 0,42	39,30 ± 0,41	8,48 ± 0,32
(True, 18)	71,77 ± 0,33	90,48 ± 0,11	52,02 ± 0,48	39,51 ± 0,49	8,47 ± 0,33
(True, 6)	71,62 ± 0,31	90,42 ± 0,10	51,57 ± 0,50	40,10 ± 0,54	8,33 ± 0,29
(False, 18)	69,03 ± 6,96	89,55 ± 2,35	48,92 ± 7,11	40,22 ± 0,79	10,86 ± 6,84

5 Weiterführende Studien

In Kapitel 4 wurden verschiedene Modelle für die Klassifikation der zusätzlichen Jets mit B-Hadronen (AddB-Jets) getestet und optimiert. Aufbauend auf diesen Ergebnissen wird das beste Modell nun weiteren Studien unterzogen. Hierfür werden in Abschnitt 5.1 die Verwechslungsraten der verschiedenen Jet-Kategorien genauer untersucht. Im Abschnitt 5.2 erfolgen weitere, physikalisch motivierte Tests, um das bisher beste Modell weiter zu optimieren. In Abschnitt 5.3 wird dann die finale Auswertung und ein Vergleich der Ergebnisse mit der bisherigen Klassifikationsmethode aus [1] vorgenommen. Abschließend werden die Ergebnisse in Abschnitt 5.4 zusammengefasst und ein Ausblick für zukünftige Untersuchungen gegeben.

5.1 Verwechslungsraten der Jet-Kategorien

Das Multi-GGS-NN-Modell ohne Skip-Connection und $L_k = 6$ aus Abschnitt 4.5.1 hat durchschnittlich sowohl die besten Ergebnisse auf die TPR und den 2/2-Werten erzielt (s. Tabelle 4.3, Zeile 1):

TPR (%)	TNR (%)	2/2 (%)	1/2 (%)	0/2 (%)
71,87 ± 0,31	90,51 ± 0,11	52,22 ± 0,42	39,3 ± 0,41	8,48 ± 0,32

Nun soll genauer untersucht werden, welche Jet-Kategorien dabei besonders häufig mit den AddB-Jets verwechselt werden. Im Folgenden wird das Lepton nicht als Kategorie betrachtet, da das Modell dieses immer von den AddB-Jets unterscheiden kann. Für diese Untersuchung muss nun allerdings beachtet werden, dass nicht alle übrigen Jet-Kategorien beim Label-Vorgang richtig zugeordnet werden. Da die übrigen Jets allerdings nach absteigendem Transversalimpuls die Labels HadTopB, LepTopB und HadTopQ erhalten, ist zumindest ein Teil der Labels richtig zugeordnet (s. Abschnitt 2.5). Aus diesem Grund werden alle Ereignisse nun in drei Unterklassen eingeteilt: HadTopB-Unterklasse, LepTopB-Unterklasse und HadTopQ-Unterklasse, im Folgenden auch HTB-, LTB- oder HTQ-Unterklasse genannt. In diesen Unterklassen befinden sich nur Ereignisse, in denen die Kategorie der jeweiligen Unterklasse richtig zugeordnet ist. Richtig heißt in diesem Fall, dass der rekonstruierte Jet in einem Ereignis mit der Kategorie der jeweiligen Unterklasse, einen Abstand von $\Delta R < 0{,}4$ zu dem Generator-Jet mit dieser Kategorie aufweist. Somit sind z.B. in der HTQ-Unterklasse beide AddB- und beide HadTopQ-Jets richtig zugeordnet. Für diese Unterklasse lässt sich nun eine Aussage darüber treffen, wie oft die HadTopQ-Jets im

Vergleich zu allen anderen Jets mit den AddB-Jets verwechselt werden. Analog kann diese Aussage für die anderen Unterklassen getroffen werden.

Dafür wird in diesem Abschnitt der gesamte Datensatz mit 28.425 Ereignissen betrachtet, um die Größe der Unterklassen zu maximieren. Da die Trainingsdaten nicht übertrainiert worden sind, ist dies gerechtfertigt. Die folgende Erläuterung bezieht sich auf die HTQ-Unterklasse. Analog können auch die anderen Unterklassen beschrieben werden (s. Tabelle 5.1).

Die HTQ-Unterklasse besteht aus 1.068 Ereignissen. Davon sind 461 Ereignisse auch in der HTB-Unterklasse und 470 Ereignisse in der LTB-Unterklasse enthalten. Die HTQ-Rest-Unterklasse enthält die 27.357 Ereignisse, bei denen nur einer oder kein HadTopQ-Jet richtig zugeordnet ist. Durchschnittlich weist die HTQ-Unterklasse pro Ereignis 7,51 Knoten auf. Da zwei der Knoten die AddB-Jets sind, ein Knoten das Lepton und zwei Knoten die beiden HadTopQ-Jets, ergeben sich somit 2,51 andere Jets pro Ereignis. Das Verhältnis von anderen Jets zu HadTopQ-Jets beträgt somit 1,26. Nun kann für diese Unterklasse die Verwechslungsrate der HadTopQ-Jets berechnet werden, die beschreibt, wie oft die HadTopQ-Jets als AddB-Jet klassifiziert werden. Ebenso kann die Verwechslungsrate berechnet werden, wie oft einer der anderen Jets mit den AddB-Jets verwechselt wird. Da ein Ungleichgewicht in der Anzahl der Jets herrscht, ist die Wahrscheinlichkeit theoretisch um den Faktor 1,26 höher, dass ein anderer Jet als AddB-Jet klassifiziert wird. Daher wird auch eine eine angepasste Verwechslungsrate der anderen Jets berechnet. Diese ergibt ergibt sich durch das Teilen der Verwechslungsrate durch 1,26, also dem Verhältnis von anderen Jets zu HadTopQ-Jets.

Tabelle 5.1: Beschreibung der Unterklassen.

	Gesamt	HTB-Unterkl.	LTB-Unterkl.	HTQ-Unterkl.
Anzahl Ereignisse	28.425	7.223	7.341	1.068
Ereignisse in Rest-Unterkl.	0	21.202	21.084	27.357
auch in HTB-Unterkl.	7.223	-	2.764	461
auch in LTB-Unterkl.	7.341	2.764	-	470
auch in HTQ-Unterkl.	1.068	461	470	-
∅ Anzahl Knoten	7,90	7,69	7,75	7,51
Verhältnis $\frac{\text{Andere Jets}}{\text{Kategorie-Jets}}$	-	3,69	3,75	1,26

In Abb. 5.1 sind die so berechneten Verwechslungsraten für die verschiedenen Unterklassen abgebildet. Bei der Verwechslungsrate der AddB-Jets handelt es sich um die TPR der Unterklasse. Es ist deutlich zu erkennen, dass die Verwechslungsrate der HadTopQ-Jets in der entsprechenden Unterklasse sehr gering im Vergleich zur angepassten Verwechslungsrate der anderen Jets ist. Dies entspricht auch den Erwartungen, da die HadTopQ-Jets aufgrund des geringen b-tag-Wertes gut von den AddB-Jets zu trennen sind. Umgekehrt verhält es sich bei der Verwechslungsrate der HadTopB-Jets und der LepTopB-Jets in der entsprechenden Unterklasse. Diese werden in den jeweiligen Unterklassen häufiger als andere Kategorien mit den AbbB-Jets verwechselt. Auch dies entspricht den Erwartungen, da diese Jets den AbbB-Jets aufgrund ihrer b-Artigkeit deutlich ähnlicher sind als die HadTopQ-Jets.

Bei dieser Auswertung müssen allerdings einige Punkte beachtet werden. Zunächst gelten die Aussagen über die Verwechslungsraten nur für die jeweilige Unterklasse und nicht für den gesamten Datensatz. Ein Vergleich der Verwechslungsraten zwischen den Unterklassen ist ebenfalls nicht ohne Weiteres möglich. Dies liegt daran, dass die einzelnen Unterklassen Überschneidungen aufweisen und teilweise dieselben Ereignisse beinhalten. Ebenso gilt zu beachten, dass die Ereignisse in den Unterklassen durchschnittlich weniger Knoten

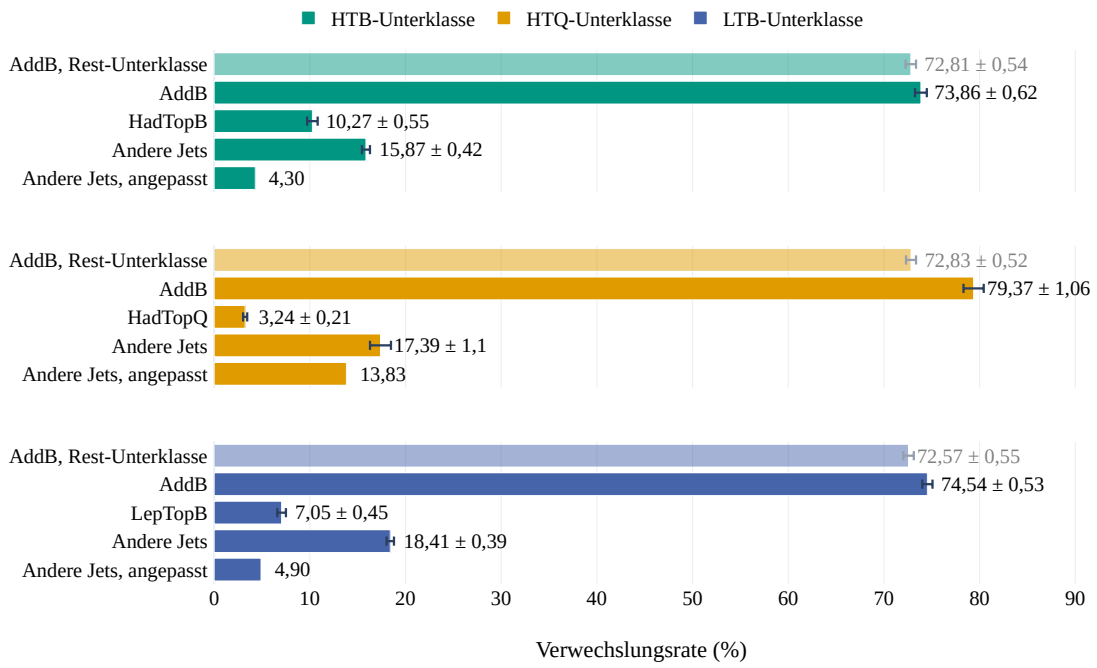


Abbildung 5.1: Verwechslungsraten der Jet-Kategorien in den verschiedenen Unterklassen: Dargestellt sind die jeweiligen Unterklassen und die erreichte TPR (entspricht der Verwechslungsrate der AddB-Jets) und die erreichte TPR der jeweiligen Rest-Unterklasse. Zusätzlich ist die Verwechslungsrate der jeweiligen Kategorie im Vergleich zur Verwechslungsrate aller anderen Jets dargestellt. Die angepasste Verwechslungsrate berücksichtigt das Ungleichgewicht von Jets der untersuchten Kategorie und anderen Jets im Ereignis.

aufweisen als der gesamte Datensatz. Der Grund liegt in der höheren Wahrscheinlichkeit, dass die richtige Kategorie zugewiesen wird, wenn die Jet-Multiplizität des Ereignisses geringer ist. Somit weisen die Unterklassen auch zwangsweise eine bessere TPR als ihre jeweiligen Rest-Unterklassen auf, da das Modell bei Ereignissen mit weniger Jets leichter die beiden AddB-Jets finden kann. Die TPR wird aber auch dadurch erhöht, da nun auch Trainingsdaten in die Auswertung einfließen, die das Modell besser einordnen kann.

5.2 Physikalisch motivierte Untersuchungen

In diesem Abschnitt werden weitere, physikalisch motivierte Tests durchgeführt, um das bisherige beste Model weiter zu optimieren. Die bisherigen Ergebnisse des besten Modells dienen dabei als Vergleichswerte.

5.2.1 Kombination von zwei Graphen

In der bisherigen Untersuchung wurde stets der Graph aus Abschnitt 4.1 verwendet. Dabei haben die Kanten den jeweiligen ΔR-Wert als Gewicht erhalten. Entsprechend dem Berechnungsschritt (A & B) im GGS-NN-Modul (s. Gleichung 3.6), skaliert dieses Gewicht die ausgetauschten Nachrichten zwischen den Knoten mit den Faktor ΔR. Dies ist durchaus sinnvoll, da die AddB-Jets eine hohe Wahrscheinlichkeit haben, nahe beieinander zu liegen [1]. Dennoch wird dem Modell dadurch aber auch die Möglichkeit verwehrt, einen unvoreingenommenen Blick auf die kinematischen Größen der Feature-Vektoren zu werfen. Das Modell soll nun also zusätzlich in der Lage sein, nur die Feature-Vektoren zu vergleichen, ohne durch den Abstand in der η, ϕ-Ebene beeinflusst zu werden. Dies ist

besonders dann wichtig, wenn in einem Ereignis der Abstand zwischen zwei AddB-Jets größer ausfällt als normal, oder andere Jets näher beieinander liegen.

Um dies zu gewährleisten, wird in diesem Test ein Modell verwendet, dass aus zwei Multi-GGS-NN-Modellen ohne Skip-Connection besteht. Die finale Vorhersage für einen Knoten besteht dann aus den Informationen beider Multi-GGS-NN-Modelle. Das erste Multi-GGS-NN-Modell erhält den Graphen entsprechend Abschnitt 4.1 mit Gewichten. Das zweite erhält den gleichen Graphen, allerdings ohne Gewichte (bzw. alle Gewichte sind 1). Die Vorhersagen der beiden Modelle werden dann zusammengeführt. Bevor die Sigmoid-Funktion als Aktivierungsfunktion angewendet wird, werden die skalaren Repräsentationen der Knoten $h'_{i,1}$ des ersten Modells und $h'_{i,2}$ des zweiten Modells addiert (s. Abb. 5.2).

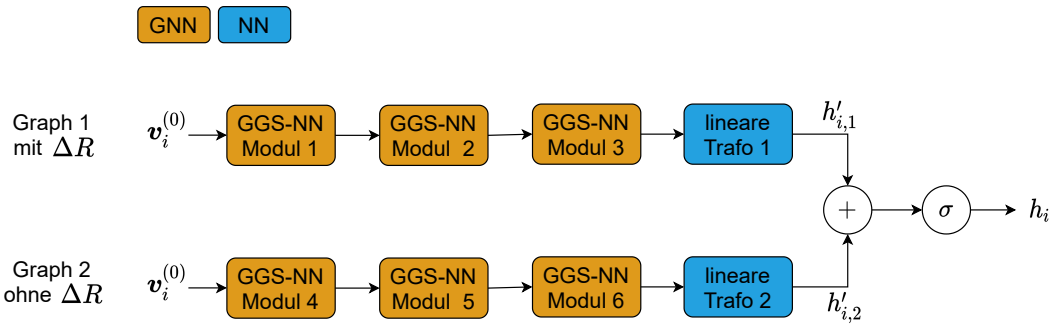


Abbildung 5.2: Multi-GGS-NN-Modell für den Zwei-Graphen-Test: bei diesem Test wird dem ersten Multi-GGS-NN-Modell der Graph 1 übergeben, dessen Kanten die ΔR-Werte als Gewicht zugeordnet werden. Das zweite Modell erhält den Graph 2, dessen Kanten keine Gewichte besitzen (bzw. alle Gewichte sind 1). Die finale Repräsentation des Knotens und somit die Vorhersage, ob es sich um einen AddB-Jet handelt, enthält die Information beider Multi-GGS-NN-Modelle.

Auswertung: Zwei-Graphen-Test

Der Zwei-Graphen-Test erzielt im Vergleich zum bisherigen besten Modell eine Verbesserung von 0,27 % auf die TPR und 0,33 % auf den 2/2-Wert (s. Tabelle 5.2). Dennoch liegen die durchschnittlich erzielten Werte von TPR = 72,14 % und 2/2 = 52,55 % innerhalb der Standardabweichungen der Vergleichswerte. Somit hilft es dem Modell zwar, die Feature-Vektoren der Knoten auch ohne Beeinflussung durch die ΔR-Werte zu betrachten, allerdings fällt diese Verbesserung eher gering aus.

Tabelle 5.2: Ergebnisse des Zwei-Graphen-Tests.

Test	TPR (%)	TNR (%)	2/2 (%)	1/2 (%)	0/2 (%)
Vergleich	71,87 ± 0,31	90,51 ± 0,11	52,22 ± 0,42	39,30 ± 0,41	8,48 ± 0,32
Zwei Graphen	72,14 ± 0,38	90,60 ± 0,13	52,55 ± 0,50	39,18 ± 0,43	8,27 ± 0,37

5.2.2 Verwendung eines invertierten Gewichts

In diesem Test wird der Einfluss eines invertierten Gewichts untersucht. Dafür beträgt das Gewicht $e_{i,j}$ einer Kante zwischen zwei Knoten i und j nun

$$e_{i,j} = \Delta R_{\max} - \Delta R_{i,j}. \tag{5.1}$$

Dabei ist $\Delta R_{\max}$ der größte Abstand im gesamten Datensatz in der η, ϕ-Ebene und $\Delta R_{i,j}$ der Abstand der Objekte i und j. Somit entspricht nun ein großer Gewichtswert einer hohen Wahrscheinlichkeit, dass es sich bei zwei Knoten um die AddB-Jets handelt.

Wie im vorangegangenen Test besprochen, werden die Gewichte mit den Nachrichten der Knoten multipliziert. Somit haben Nachrichten, die zwischen Knoten mit geringem Abstand ausgetauscht werden, bisher tendenziell kleinere Werte angenommen, obwohl diese beiden Knoten eine höhere Wahrscheinlichkeit haben, die AddB-Jets zu sein. Vor dem Hintergrund, dass Knoten mit hohen Zahlenwerten am Ende als AddB-Jets klassifiziert werden, könnte es nun sinnvoll erscheinen, invertierte Gewichte zu verwenden. Dies ist allerdings nicht der Fall, da hier ein vollständiger Graph verwendet und somit jeder Knoten mit jedem anderen verbunden ist. Dies führt dazu, dass jeder Knoten auch bei invertierten Gewichten sowohl Kanten mit kleinen als auch großen Gewichten besitzt. Dadurch bleiben die Wertebereiche der aggregierten Nachrichten für jeden Knoten auch weiterhin ungefähr gleich.

Die Verwendung eines invertierten Gewichts ist allerdings aus anderer Sicht interessant. Wie bereits beschrieben kommunizieren Knotenpaare, die aufgrund des ΔR-Wertes eine höhere Wahrscheinlichkeit haben, die beiden AddB-Jets zu sein, bisher weniger stark miteinander als andere Knotenpaare. Durch ein invertiertes Gewicht wird dieser Sachverhalt nun umgekehrt: Knotenpaare, die mit höherer Wahrscheinlichkeit AddB-Jets sind, kommunizieren nun stärker miteinander. Geometrisch bedeutet das: Zuvor haben Knoten stärker mit miteinander kommuniziert, die einen großen Abstand in der η, ϕ-Ebene haben. Nun kommunizieren Knoten stärker, die nahe beieinander in der η, ϕ-Ebene liegen.

Auswertung: $\Delta R_{\max} - \Delta R$-Test

Auch der $\Delta R_{\max} - \Delta R$-Test führt durchschnittlich nur zu sehr kleinen Verbesserungen von 0,15 % auf die TPR. Allerdings schneidet der 2/2-Wert hier um 0,16 % schlechter ab. Die Ergebnisse liegen sehr nahe an den Vergleichswerten und innerhalb deren Standardabweichungen (s. Tabelle 5.3). Demnach hat es keinen großen Einfluss, ob Knotenpaare mit großem oder kleinem ΔR-Wert stärker miteinander kommunizieren.

Trotzdem ist es wichtig, dass eine Unterscheidung in der Kommunikationsstärke durch die ΔR-Werte stattfindet. Denn wird nur ein Graph ohne Gewichte an das Multi-GGS-NN-Modell ohne Skip-Connection übergeben, wird durchschnittlich nur eine TPR von 67,65 % erreicht. Der ΔR-Wert als Gewicht spielt also eine zentrale Rolle im Graphen, dabei ist es aber irrelevant, ob dieser invertiert wird oder nicht.

Tabelle 5.3: Ergebnisse des $\Delta R_{\max} - \Delta R$-Tests.

Test	TPR (%)	TNR (%)	2/2 (%)	1/2 (%)	0/2 (%)
Vergleich	71,87 ± 0,31	90,51 ± 0,11	52,22 ± 0,42	39,30 ± 0,41	8,48 ± 0,32
$\Delta R_{\max} - \Delta R$	72,02 ± 0,34	90,56 ± 0,12	52,06 ± 0,50	39,93 ± 0,40	8,01 ± 0,25
keine Gewichte	67,65 ± 0,84	89,09 ± 0,28	44,69 ± 1,22	45,92 ± 0,89	9,39 ± 0,56

5.2.3 Verwendung der invarianten Masse

Die invariante Masse eines Systems von n Teilchen ist eine Erhaltungsgröße unter Lorentztransformationen und folgendermaßen definiert:

$$M^{\text{inv}} = \sqrt{\left(\sum_n E_n\right) - \left|\sum_n \boldsymbol{p}_n\right|^2} \tag{5.2}$$

wobei E_n die Energie und $\boldsymbol{p}_n$ der vektorielle Impuls des Teilchen n ist. Für den Graphen bedeutet das, dass diese Größe eine Eigenschaft von zwei oder mehreren Knoten darstellt. Trotzdem kann diese Größe auch im Feature-Vektor einzelner Knoten berücksichtigt werden. Durch die Verwendung der invarianten Masse konnten bereits in vorherigen Klassifikationsmethoden deutliche Verbesserung erzielt werden [1]. Die Motivation liegt darin begründet, dass die invariante Masse $M^{\text{inv}}_{\text{Kategorie, Lepton}}$ von einer Jet-Kategorie (AddB, HadTopQ, LepTopB oder HadTopB) und dem Lepton je nach Kategorie unterschiedlichen Verteilungen folgt. Diese Verteilungen sind in Abb. 5.3 dargestellt und wurden mit den Daten der entsprechenden Unterklassen aus Abschnitt 5.1 erstellt. Somit ist diese Größe selbst schon ein Indikator dafür, zu welcher Kategorie ein Knoten gehört. Aufbauend auf dieser Idee werden zwei Tests durchgeführt:

- **1) $\boldsymbol{M}^{\text{inv}}$ 1**: Jedem Knoten i wird die invariante Masse $M^{\text{inv}}_{i,\text{ Lepton}}$ für die Kombination des Knotens i und dem Lepton als zusätzliche Größe im Feature-Vektor mitgegeben. Damit beträgt die Dimension des Feature-Vektors nun $m = 7$.
- **2) $\boldsymbol{M}^{\text{inv}}$ 2**: Jedem Knoten i wird wie im vorherigen Test die Größe $M^{\text{inv}}_{i,\text{ Lepton}}$ mitgegeben, sowie für alle vier Jet-Kategorien die Größe:

$$p_{\text{Kategorie},i} = \max\left(M^{\text{inv}}_{\text{Kategorie, Lepton}}\right) - \left|\bar{M}^{\text{inv}}_{\text{Kategorie, Lepton}} - M^{\text{inv}}_{i,\text{ Lepton}}\right|. \tag{5.3}$$

  Dabei beschreibt $\bar{M}^{\text{inv}}_{\text{Kategorie, Lepton}}$ die durchschnittliche invariante Masse von Knoten einer bestimmten Jet-Kategorie mit dem Lepton. Kleine Werte für die Größe $\left|\bar{M}^{\text{inv}}_{\text{Kategorie, Lepton}} - M^{\text{inv}}_{i,\text{ Lepton}}\right|$ sprechen somit also für eine höhere Wahrscheinlichkeit, dass ein Knoten i zu dieser Kategorie gehört. Da die Größen im Feature-Vektor verwendet werden sollen, ist es sinnvoll hier große Werte für eine hohe Wahrscheinlichkeit zu verwenden. Daher wird dieser Wert vom maximalen Wert für $M^{\text{inv}}_{\text{Kategorie, Lepton}}$ subtrahiert. Somit sprechen nun große Werte für $p_{\text{Kategorie},i}$ dafür, dass ein Knoten i zu dieser Kategorie gehört. Die Mittelwerte und Maximalwerte der einzelnen Verteilungen betragen (bezogen auf die jeweiligen Unterklassen):

$$\begin{aligned}
\bar{M}^{\text{inv}}_{\text{AddB, Lepton}} &\approx 181{,}15\,\text{GeV} & \max(M^{\text{inv}}_{\text{AddB, Lepton}}) &\approx 2066{,}05\,\text{GeV}\\
\bar{M}^{\text{inv}}_{\text{HadTopB, Lepton}} &\approx 216{,}71\,\text{GeV} & \max(M^{\text{inv}}_{\text{HadTopB, Lepton}}) &\approx 2106{,}88\,\text{GeV}\\
\bar{M}^{\text{inv}}_{\text{HadTopQ, Lepton}} &\approx 128{,}06\,\text{GeV} & \max(M^{\text{inv}}_{\text{HadTopQ, Lepton}}) &\approx 716{,}29\,\text{GeV}\\
\bar{M}^{\text{inv}}_{\text{LepTopB, Lepton}} &\approx 102{,}67\,\text{GeV} & \max(M^{\text{inv}}_{\text{LepTopB, Lepton}}) &\approx 473{,}39\,\text{GeV}
\end{aligned}$$

Auswertung: $\boldsymbol{M}^{\text{inv}}$-Test

Durch das alleinige Verwenden der invarianten Masse $M^{\text{inv}}_{i,\text{ Lepton}}$ im Feature-Vektor verbessern sich die Ergebnisse über die Standardabweichungen der Vergleichswerte hinaus. Bei Verwendung der invarianten Masse $M^{\text{inv}}_{i,\text{ Lepton}}$ und der Werte $p_{\text{Kategorie},i}$ verbessern sich die Ergebnisse zwar ebenfalls, aber weniger stark (s. Tabelle 5.4). Bei der alleinigen Verwendung der invarianten Masse $M^{\text{inv}}_{\text{Kategorie, Lepton}}$ kommt es zu einer Verbesserung von durchschnittlich 0,47 % auf der TPR und um 0,61 % bei den 2/2-Werten. Werden auch die Größen $p_{\text{Kategorie},i}$ verwendet, verbessern sich die Ergebnisse ebenfalls, aber nur um 0,23 % bei der TPR. Dies könnte daran liegen, dass die Größen $p_{\text{Kategorie},i}$ nur mit Hilfe der Unterklassen berechnet worden sind und somit nicht den gesamten Datensatz widerspiegeln. Weiterhin sind die maximalen Werte für die invarianten Massen teilweise deutlich größer als die durchschnittlichen Werte. Somit nimmt $p_{\text{Kategorie},i}$ selten kleine Werte nahe 0 an und häufig große Werte näher beim Maximum. Dabei werden die relativen Abstände zwischen

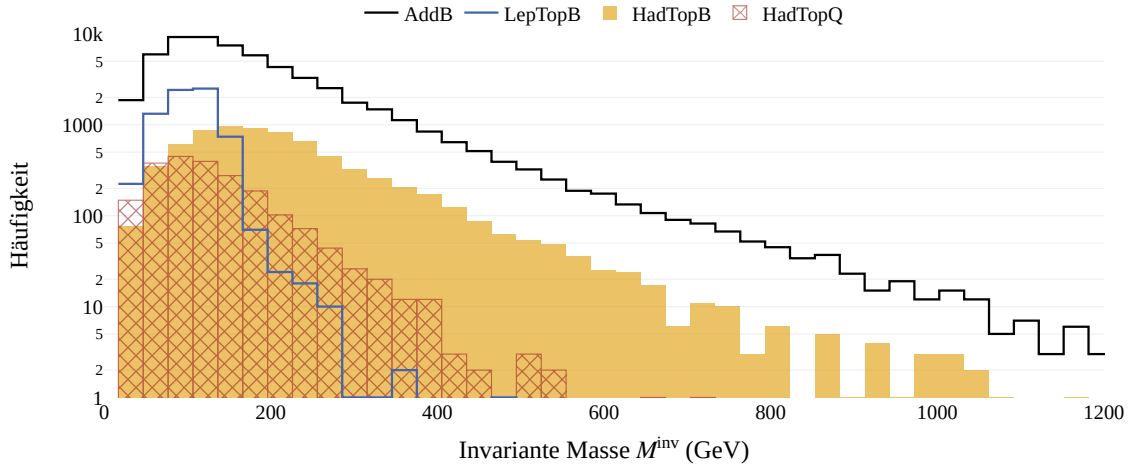


Abbildung 5.3: Verteilung der invarianten Masse $M^{\text{inv}}_{\text{Kategorie, Lepton}}$ für die Kombination des Leptons und Jets einer Kategorie. Die Daten beziehen sich auf die Unterklassen, in denen die wahre Information über die jeweilige Kategorie vorhanden ist.

den größeren Zahlenwerten für $p_{\text{Kategorie},i}$ immer geringer im Vergleich zum gesamten Wertebereich von $p_{\text{Kategorie},i}$. Dies erschwert eine zuverlässige Unterscheidung basierend auf hohen Werten für $p_{\text{Kategorie},i}$.

Tabelle 5.4: Ergebnisse des M^{inv}-Tests.

Test	TPR (%)	TNR (%)	2/2 (%)	1/2 (%)	0/2 (%)
Vergleich	71,87 ± 0,31	90,51 ± 0,11	52,22 ± 0,42	39,30 ± 0,41	8,48 ± 0,32
M^{inv} 1	72,34 ± 0,29	90,67 ± 0,10	52,83 ± 0,48	39,03 ± 0,56	8,15 ± 0,31
M^{inv} 2	72,10 ± 0,27	90,59 ± 0,09	52,44 ± 0,47	39,33 ± 0,56	8,24 ± 0,29

5.2.4 Fokus auf b-tagged Jets

In diesem Test wird dem Modell explizit verboten, Knoten als AddB-Jet zu klassifizieren, die einen b-tag-Wert kleiner als 0,277 haben. Dadurch soll dem Modell geholfen werden, vermehrt b-tagged Jets als AddB-Jets zu klassifizieren.

Auswertung: b-tagged Jets-Test

Wird als explizites Kriterium für die Klassifikation von AddB-Jets ein b-tag-Wert größer 0,277 gefordert, verschlechtern sich die Ergebnisse leicht (s. Tabelle 5.5). Dies liegt daran, dass das Multi-GGS-NN-Modell bereits sehr gut darin ist, nur Knoten mit einem größeren b-tag-Wert als AddB-Jet zu klassifizieren. So befinden sich im gesamten Test-Datensatz 28.018 Jets, die keine AddB-Jets sind. Von den fälschlicherweise als AddB-Jet klassifizierten Jets weisen allerdings nur 27 einen b-tag-Wert kleiner als 0,277 auf. Dabei hat das Modell aber 22 AddB-Jets korrekt identifiziert, die einen b-tag-Wert kleiner 0,277 aufweisen (s. Abb. 5.4). Somit könnten durch diesen Test im Optimalfall ohnehin nur maximal fünf weitere AddB-Jets im Test-Datensatz gefunden werden.

Weiterhin ist in Abb. 5.4 deutlich zu erkennen, dass es ab einem b-tag-Wert von 0,277 zu einem starken Anstieg von fälschlicherweise als AddB-Jet klassifizierten Jets kommt. Das Modell hat diese Grenze für B-Jets also bereits selbst erlernt. Dies liefert auch eine mögliche

Erklärung dafür, warum niemals ein Lepton als AddB-Jet klassifiziert wird, denn durch seinen b-tag-Wert von 0 kann das Modell dieses Objekt eindeutig als keinen AddB-Jet klassifizieren.

Tabelle 5.5: Ergebnisse des b-tagged Jets-Tests.

Test	TPR (%)	TNR (%)	2/2 (%)	1/2 (%)	0/2 (%)
Vergleich	71,87 ± 0,31	90,51 ± 0,11	52,22 ± 0,42	39,30 ± 0,41	8,48 ± 0,32
b-tagged Jets	71,83 ± 0,29	90,45 ± 0,10	52,09 ± 0,40	39,49 ± 0,40	8,42 ± 0,30

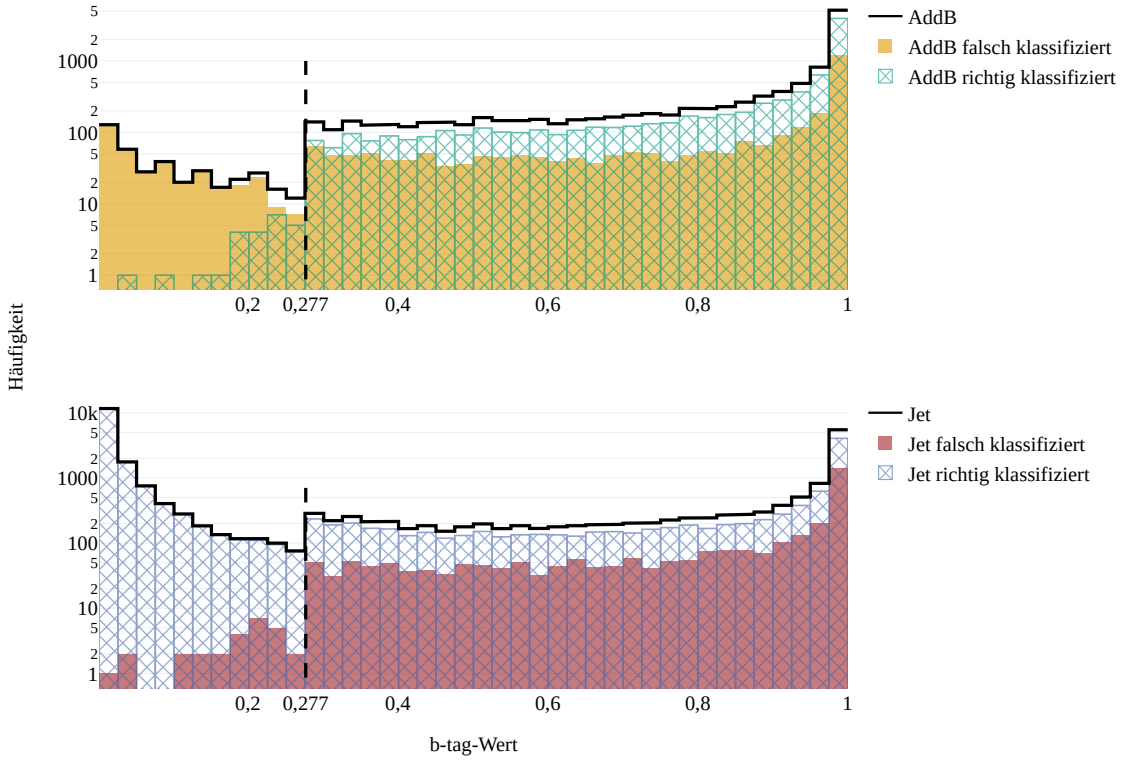


Abbildung 5.4: Verteilung der b-tag-Werte für falsch und richtig klassifizierte AddB-Jets und für andere Jets: die obere Darstellung zeigt die Verteilungen des b-tag-Wertes für alle AddB-Jets im Vergleich zu den Verteilungen der falsch und richtig klassifizierten AddB-Jets. Die untere Darstellung stellt den gleichen Sachverhalt für die anderen Jets dar. Die Daten beziehen sich auf den Test-Datensatz des besten Durchlaufs des Multi-GGS-NN-Modells ohne Skip-Connection und mit $L_k = 6$.

5.2.5 Verwendung der Jet-Ladung

In bisherigen Untersuchungen [1] und in dieser Arbeit wurde die Jet-Ladung nicht berücksichtigt. Diese Größe könnte dem Modell allerdings helfen, da aufgrund der Ladungserhaltung die Gesamtladung bestimmter Gruppen von Knoten der Ladung des Teilchens entsprechen muss, aus dem sie zerfallen sind. Andererseits können geladene Teilchen, die nicht in einem Jet enthalten sind, für weitere Unsicherheiten bei dieser Größe führen. Die Jet-Ladung kann als weiterer Eintrag im Feature-Vektor repräsentiert werden. Das Lepton erhält dafür seine Ladung $Q = \pm 1\,e$. Für die Dimension des Feature-Vektors gilt nun $m = 7$.

Auswertung: Jet-Ladung-Test

Die Verwendung der Jet-Ladung verschlechtert das Ergebnis der TPR durchschnittlich um 0,3 % und liegt damit gerade noch innerhalb der Standardabweichung der Vergleichswerte (s. Tabelle 5.6). Bei Betrachtung der Verteilung der Jet-Ladung für die AddB- und andere Jets (Abb. A.7) ist erkennbar, dass die Verteilungen sehr ähnlich sind. Somit kann das Modell allein aufgrund der Jet-Ladung keine tiefergreifenden Erkenntnisse über die Kategorie eines Knotens ziehen. Die Jet-Ladung könnte sehr effektiv sein, wenn das Modell in der Lage wäre, Jet-Ladungen von einzelnen Knotengruppen zu vergleichen. Zum Beispiel wäre jede Knotenkombination, die zusammen eine neutrale Jet-Ladung aufweist, ein potentieller Kandidat für die beiden AddB-Jets. Aber auch die Information, ob z.B. die Gesamtladung einer Gruppe von drei Knoten der des hadronisch zerfallenden Top-Quarks entspricht, könnte dem Modell helfen, diese Gruppe als die beiden HadTopQs und das HadTopB zu identifizieren. Aufgrund der vollständigen Graphstruktur ist so ein direkter Vergleich allerdings nicht möglich, da jeder Knoten die aggregierten Nachrichten von jedem anderen Knoten erhält. Weiterhin ist das Modell auch nicht in der Lage, Eigenschaften einer Gruppe von mehr als zwei Knoten zu repräsentieren.

Tabelle 5.6: Ergebnisse des Jet-Ladungs-Tests.

Test	TPR (%)	TNR (%)	2/2 (%)	1/2 (%)	0/2 (%)
Vergleich	71,87 ± 0,31	90,51 ± 0,11	52,22 ± 0,42	39,30 ± 0,41	8,48 ± 0,32
Jet-Ladung	71,57 ± 0,31	90,41 ± 0,10	51,80 ± 0,47	39,55 ± 0,53	8,66 ± 0,33

5.2.6 Verwendung der fehlenden Transversalenergie

Dieser Test berücksichtigt die fehlende Transversalenergie (MET). Dies geschieht durch einen weiteren Knoten im Graphen, dessen Feature-Vektor die fehlende Transversalenergie $\not{E}_{\mathrm{T}}$ und den Winkel ϕ enthält, die Richtung, in der die Energie im Ereignis in der Transversalebene fehlt. Die restlichen Einträge im Feature-Vektor werden auf 0 gesetzt.

Auswertung: MET-Test

Die Verwendung der fehlenden Transversalenergie erzielt die größte Verschlechterung der Ergebnisse. Hier sinkt die TPR um 0,74 % und der 2/2-Wert und 1,09 % (s. Tabelle 5.7). Die erzielten Werte liegen auch außerhalb der Standardabweichung der Vergleichswerte. Auffällig ist allerdings die Verbesserung der TNR. Dies liegt daran, dass die fehlende Transversalenergie durch einen zusätzlichen Knoten im Graphen repräsentiert wird. Das Modell ist aber in der Lage diesen Knoten, so wie den Knoten des Leptons, fehlerfrei von den AddB-Jets zu trennen. Somit steigt die TNR.

Tabelle 5.7: Ergebnisse des MET-Tests.

Test	TPR (%)	TNR (%)	2/2 (%)	1/2 (%)	0/2 (%)
Vergleich	71,87 ± 0,31	90,51 ± 0,11	52,22 ± 0,42	39,30 ± 0,41	8,48 ± 0,32
MET	71,13 ± 0,36	91,64 ± 0,10	51,13 ± 0,64	40,01 ± 0,68	8,87 ± 0,27

5.3 Finale Auswertung

In den vorherigen Untersuchungen wurden bisher immer Durchschnittswerte bei 20-facher Wiederholung betrachtet. Für den finalen Test soll allerdings der Trainingsdurchlauf untersucht werden, der die besten Ergebnisse erzielt hat. Der Durchlauf mit dem besten 2/2-Wert entstammt dem Zwei-Graphen-Test aus Abschnitt 5.2.1. Dieser Durchlauf erzielt folgende Ergebnisse:

TPR (%)	TNR (%)	2/2 (%)	1/2 (%)	0/2 (%)
72,87	90,85	53,49	38,75	7,76

Der Verlauf des Loss-Wertes und der TPR ist in Abb. 5.5 dargestellt. Es ist zu erkennen, dass es ab Epoche 27 zu einem Übertraining kommt. Die Werte des Trainings-Datensatzes verbessern sich immer weiter während die TPR des Validations-Datensatzes immer schlechter wird. In Epoche 27 hat das Modell die beste TPR auf den Validations-Datensatz erzielt, weshalb auch der Test-Datensatz in dieser Epoche getestet wird, bevor es zum Übertraining kommt.

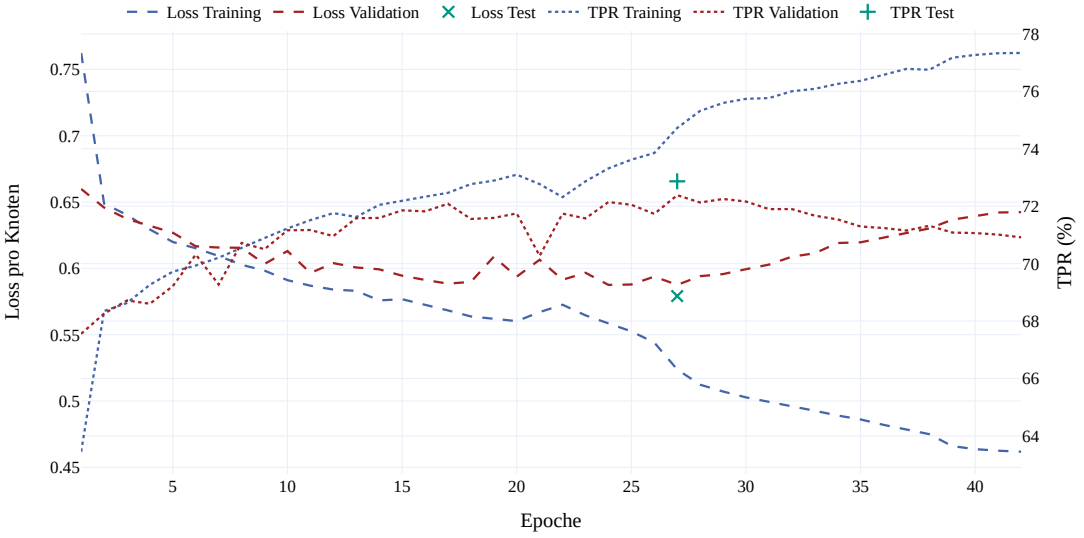


Abbildung 5.5: Verlauf des Loss-Wertes und der TPR während des Trainingsprozesses des besten Durchlaufs.

Um diese Ergebnisse letztendlich mit denen aus [1] vergleichen zu können, muss das Modell noch einmal auf demselben Datensatz getestet werden. Dieser Datensatz entspricht der Selektion aus Abschnitt 2.5, nur dass jetzt auch Ereignisse verwendet werden, bei denen keine vollständige Zuordnung der AddB-Jets von Generator-Level auf Rekonstuktions-Level möglich ist. Weiterhin werden auch nur noch Ereignisse aus dem $t\bar{t} + b\bar{b}$-Sample verwendet (s. Abschnitt 2.5). Insgesamt enthält diese Auswertung 18.115 Ereignisse. In [1] werden für den finalen Test nach dem Training nochmals alle Ereignisse mit der soeben beschriebenen Selektion in das Modell geladen und ausgewertet, womit auch Ereignisse aus den ehemaligen Trainings- und Validations-Datensätzen in die Auswertung mit einfließen. Das hier verwendete Modell erkennt zwar die Trainingsdaten besser als die Validations- und Test-Daten, hat allerdings noch nicht mit dem Übertraining begonnen, was dieses Vorgehen rechtfertigt. Der Vergleich ist in Tabelle 5.8 dargestellt. Die Bezeichnungen „2/2“, „1/2“ und „0/2“ entsprechen nun nicht mehr exakt den zuvor eingeführten Definitionen, da es auch Kollisionsereignisse in den Simulationsdaten gibt, bei denen nicht alle AddB-Jets zugeordnet werden konnten. Somit gibt es nicht immer zwei Jets der Kategorie „AddB-Jet“.

Die Bezeichnungen beziehen sich nun schlicht darauf, wie viele AddB-Jets pro Ereignis gefunden worden sind, unabhängig davon ob zwei, einer oder kein AddB-Jet im Ereignis vorhanden ist. Dadurch, dass nun auch Ereignisse in die Auswertung einfließen, bei denen keine zwei AddB-Jets gefunden werden können, sinkt der 2/2-Wert etwas.

Tabelle 5.8: Vergleich der Ergebnisse des DNNs und des GNNs.

Vergleich	2/2 (%)	1/2 (%)	0/2 (%)
DNN aus [1]	45,93	37.58	16,49
Multi-GGS-NN-Modell	52,08	38,85	9,08

Wie aus Tabelle 5.8 hervorgeht, stellen Graph Neural Networks eine effektive Methode dar, um die AddB-Jets zu klassifizieren. So findet das hier verwendete Multi-GGS-NN-Modell in 52,08 % der Fälle zwei AddB-Jets, womit es rund 6 % besser als das DNN abschneidet. In 90,92 % der Fälle findet es mindestens einen AddB-Jet, was ebenfalls eine Verbesserung von mehr als 7 % gegenüber dem DNN darstellt.

5.4 Diskussion und Ausblick

Die vorangegangenen Untersuchungen haben gezeigt, dass GNNs sehr effektiv darin sind, die AddB-Jets in $t\bar{t} + b\bar{b}$-Ereignissen zu klassifizieren. Dabei konnten mit GNNs deutlich bessere Ergebnisse als mit DNNs erzielt werden. Eine Verbesserung gegenüber DNNs konnte auch schon in Kapitel 4 ausgemacht werden. In diesem Kapitel wurden lediglich grundlegende Jet-Kinematiken verwendet, während das DNN auch mit weiteren Features wie der invarianten Masse gearbeitet hat [1]. In dieser Diskussion soll auf die Vorteile von GNNs eingegangen werden, die diese Verbesserung der Ergebnisse erklären könnten. Ebenso werden verschiedene Verfahren vorgeschlagen, wie der Einfluss dieser Vorteile auf die Verbesserungen der Ergebnisse überprüft werden könnte. Abschließend werden weitere Ideen für zukünftige Untersuchungen besprochen.

5.4.1 Vorteile von GNNs

Ein Vorteil von GNNs ist, dass Graphen keine intrinsische Ordnung besitzen (Graphisomorphismus, s. Abschnitt 3.1). Dies kommt der Natur des Kollisionsereignisses sehr entgegen, da auch die Teilchen keine intrinsische Ordnung besitzen (Permutationsinvarianz). Somit ist es egal, welcher Knoten welchen Jet repräsentiert, die Outputs für die Knoten bleiben immer dieselben. Dies ist ein Vorteil, der die Handhabung der Daten stark vereinfacht. Dennoch ist dies kein Vorteil in dem Sinne, dass herkömmliche neuronale Netze und die DNNs in [1] dieses Problem nicht umgehen könnten. So verändert sich zwar der Output eines DNNs bei Veränderung der Reihenfolge der Inputs, allerdings kann durch die Verwendung von Hypothesen, bei denen alle Permutationen ausprobiert werden, dieses Problem umgangen werden.

Ein weiterer Vorteil von GNNs ist die variable Graphstruktur. Bei herkömmlichen neuronalen Netzen ist die Größe der Inputs fest vorgegeben und kann nicht verändert werden. Bei GNNs ist dies etwas anders. Je nach GNN-Variante müssen die Dimensionen der Attribute des Graphen eventuell immer gleich sein, die Graphstruktur selbst kann aber beliebig variieren. Somit können bei Ereignissen mit höherer Jet-Multiplizität auch weiterhin alle Jets gleichzeitig betrachtet werden und Informationen miteinander austauschen. Es ist also immer die gesamte Information eines Ereignisses vorhanden, was bei DNNs nicht der Fall ist.

Einer der größten Vorteile von GNNs ist der Graph mit seiner spezifischen Graphstruktur selbst. Durch die Kanten im Graphen werden die Kommunikationswege festgelegt, wodurch

nur Informationen zwischen den gewünschten Knoten ausgetauscht werden. Dieser explizite Ausdruck von Beziehungen unter den Knoten wird auch induktiver Bias genannt. In der hier verwendeten, vollständigen Graphstruktur geht dieser Vorteil allerdings etwas verloren, da alle Knoten mit allen Knoten verbunden sind. Dennoch wird auch hier ein induktiver Bias insofern berücksichtigt, dass die Kommunikationsstärke zwischen Knotenpaaren durch die ΔR-Werte beeinflusst werden.

5.4.2 Überprüfung der Vorteile von GNNs

Im Folgenden werden Verfahren diskutiert, mit denen der Einfluss der Vorteile von GNNs auf die Verbesserung der Ergebnisse genauer untersucht werden kann. Hierfür ist es wichtig, die einzelnen Vorteile isoliert zu betrachten und möglichst gleiche Rahmenbedingungen für das GNN und das DNN zu schaffen.

Zunächst müssten das DNN und das GNN die exakt gleichen Jet-Kinematiken als Input-Größen erhalten. In der jetzigen Form ist dies nicht gegeben, da das DNN z.B. mit der invarianten Masse von Kombinationen von mehr als zwei Jets arbeitet und der Graph alle ΔR-Werte erhält und nicht nur den kleinsten ΔR-Wert von den b-tagged Jets wie das DNN.

Erhalten beide Modelle die gleichen Input-Größen, müssten dann die Vorteile von GNNs isoliert betrachtet werden. Der Vorteil, dass der Graph keine intrinsische Ordnung besitzt, lässt sich nicht beseitigen. Allerdings ist dies kein Problem, da das DNN durch Hypothesen mit dieser Problematik umgehen kann. Der Vorteil, dass bei GNNs immer die gesamte Information eines Ereignisses vorhanden ist, kann allerdings isoliert werden. Hierfür muss das GNN und das DNN nur auf Ereignisse mit einer festen Jet-Multiplizität trainiert und ausgewertet werden. Dabei muss das DNN entsprechend genügend Inputs aufweisen, um alle Jets zu berücksichtigen. Kann das GNN unter diesen Bedingungen bessere Ergebnisse vorweisen, ist dies auf die Graphstruktur und die verwendete GNN-Variante zurückzuführen. Damit kann allerdings nicht unbedingt gezeigt werden, dass die Graphstruktur selbst grundsätzlich besser geeignet ist. Denn dieselbe Graphstruktur, übergeben an andere GNN-Varianten kann auch deutlich schlechtere Ergebnisse als das DNN liefern. Vor diesem Hintergrund kann auch die Verwendung von Gated Recurrent Units in Kombination mit herkömmlichen neuronalen Netzen interessant sein, da die GRU eines der Hauptmerkmale des GGS-NNs darstellt und diese GNN-Variante ebenfalls verantwortlich für die Verbesserungen ist.

Der Einfluss des Nachteils von DNNs, dass diese nicht immer die gesamte Information eines Ereignisses zur Verfügung haben, kann folgendermaßen überprüft werden: Der Datensatz wird in Unterklassen aufgeteilt, wobei jede Unterklasse nur Ereignisse mit einer festen Jet-Multiplizität aufweist. Das DNN wird dann auf jeder dieser Unterklassen einzeln trainiert und ausgewertet (Experiment 1). Dabei muss bei jedem Training die Anzahl der Inputs entsprechend an die Jet-Multiplizität angepasst werden. Somit können in den einzelnen Trainings und Auswertungen immer die gesamte Information eines Ereignisses betrachtet werden. Es ist stark anzunehmen, dass das DNN, so wie das GNN in Abschnitt 5.1, bei Ereignissen mit höherer Jet-Multiplizität schlechtere Ergebnisse erzielt, da es immer schwieriger wird, die beiden AddB-Jets zu finden. Nun muss das DNN nochmals auf den gesamten Datensatz trainiert werden. Die Auswertung des DNNs erfolgt allerdings getrennt auf den Unterklassen mit fester Jet-Multiplizität (Experiment 2). Dabei hat das DNN während des Trainings und der Auswertung aber nicht mehr immer die gesamte Information eines Ereignisses. Nun kann verglichen werden, wie stark sich die Ergebnisse des DNNs in Experiment 1 und Experiment 2 bei steigender Jet-Multiplizität verschlechtern. Verschlechtern sich die Ergebnisse bei Experiment 2 stärker, kann davon ausgegangen werden, dass es ein Nachteil für das DNN ist, nicht die gesamte Information eines Ereignisses nutzen zu können.

Ein analoger Test für das GNN ist nicht ohne Weiteres möglich, da sich die Tatsache, dass das GNN immer die gesamte Information eines Ereignisses erhält, nicht einfach ausschalten lässt. Es könnte probiert werden, dem GNN in Experiment 2 nur verschiedene Teilgraphen zu übergeben, bei denen bestimmte Jets fehlen. Allerdings stellen sich dann weitere Fragen, z.B. nach welchen Kriterien nun die beiden AddB-Jets ausgewählt werden sollen.

Ebenfalls interessant ist ein Vergleich der Ereignisse, bei denen die beiden AddB-Jets von einem Modell gefunden worden sind, vom anderen aber nicht. Hieraus können weitere Schlüsse darüber gezogen werden, ob Ereignisse mit bestimmten Eigenschaften von einem der Modelle besser verstanden werden.

Letztlich bedarf es also einer umfassenden Analyse, um die tatsächlichen Gründe für das bessere Abschneiden der GNNs einzeln zu identifizieren. Dies liegt zum einen daran, dass GNNs und DNNs eine grundsätzlich andere Beschaffenheit und Funktionsweise aufweisen, was einen direkten Vergleich sehr schwierig macht. Zusätzlich ist es nicht immer möglich, die verschiedenen Vorteile und Eigenheiten von GNNs isoliert zu betrachten, was eine Aussage über einzelne Aspekte von GNNs erschwert. Zum jetzigen Zeitpunkt kann allerdings festgehalten werden, dass die Kombination der Vorteile von GNNs und die verwendete GNN-Variante für eine Verbesserung der Ergebnisse sorgen. Dabei spielt der induktive Bias, ausgedrückt durch die Anpassung der Kommunikationsstärke zwischen Knoten basierend auf dem ΔR-Wert, eine sehr wichtige Rolle. Denn ohne eine Anpassung der Kommunikationsstärke, also bei Verwendung eines Graphen ohne Gewichte, schneidet das GNN beim 2/2-Wert mit $2/2 = 44{,}69\,\%$ schlechter aber als das DNN mit $2/2 = 45{,}93\,\%$. Dennoch findet das GNN ohne Gewichte insgesamt 67,65 % der AddB-Jets und das DNN nur 64,72 % (s. Tabelle 5.3 und Tabelle 5.8, TPR für das DNN ergibt sich aus dem Erwartungswert der 2/2- und 1/2-Werte). Somit spielen auch weitere Aspekte von GNNs in die Verbesserung der Ergebnisse mit ein.

5.4.3 Zukünftige Untersuchungen

Auch wenn das bisherige Multi-GGS-NN-Modell und die verwendete Graphstruktur sehr gute Ergebnisse erzielt haben, kann das GNN-Modell noch weiter verbessert werden.

Der induktive Bias kann noch verstärkt berücksichtigt werden, indem z.B. die Kanten zwischen bestimmten Knoten in unterschiedliche Kategorien eingeteilt werden. Dass eine solche Unterteilung sinnvoll sein kann, zeigt der ΔR-Wert: durch Betrachtung der ΔR-Werte können in 42 % der Fälle beide AddB-Jets identifiziert werden, wenn hierfür die beiden b-tagged Jets mit dem kleinsten ΔR-Wert gewählt werden [1]. Über die Betrachtung des kleinsten ΔR-Wertes von allen Jet- und Lepton-Kombinationen hingegen, können nur noch in 17 % der Fälle die beiden AddB-Jets gefunden werden. Dadurch, dass also jede Kante in der hier verwendeten Graphstruktur gleich behandelt wird und den entsprechenden ΔR-Wert erhält, sinkt dessen Aussagekraft. Für das hier verwendete Modell stellt dies allerdings kein großes Problem dar, da weiterhin fast nur Knoten mit einem b-tag-Wert größer 0,277 als AddB-Jet klassifiziert werden (s. Abschnitt 5.2.4). Dennoch könnte eine Unterscheidung der Kanten sinnvoll sein. Mögliche Kantentypen wären z.B. Kanten zwischen b-tagged Jets, Kanten zwischen b-tagged und nicht b-tagged Jets oder Kanten von und zu dem Lepton oder auch dem MET-Knoten in Abschnitt 5.2.6. Je nach Kategorie einer Kante könnten diesen andere Gewichte mitgegeben werden. Ebenso könnte der Informationsaustausch entlang Kanten einer bestimmten Kategorie auf unterschiedliche Art und Weise erfolgen. Hierfür müssten allerdings neue GNN-Varianten, wie z.B. Relational Graph Convolutional Networks aus [31], herangezogen werden.

Ein weiterer Nachteil der vollständigen Graphstruktur ist, dass keine einzelnen Teilgraphen betrachtet werden können. Wie die Auswertung des Jet-Ladungs-Tests in Abschnitt 5.2.5

allerdings gezeigt hat, könnte dies die Ergebnisse deutlich verbessern. So ist die Information über die Gesamtladung von Gruppen von Knoten ein wichtiger Indikator dafür, zu welchen Kategorien die Knoten der Gruppe gehören können. Ebenso kann die invariante Masse von einer Gruppen von mehr als zwei Knoten dabei helfen, die Jets dieser Gruppe zu identifizieren. Das hier verwendete Modell ist allerdings nicht in der Lage, Berechnungen nur für einen Teilgraph durchzuführen. Ebenso ist das Modell auch nicht in der Lage, Eigenschaften von Gruppen von mehr als zwei Knoten angemessen zu repräsentieren. Eine Möglichkeit, Berechnungen nur für bestimmte Gruppen von mehr als zwei Knoten durchzuführen sind Hypergraphen. Mit diesen können mehrere Knoten durch eine Kante verbunden werden, wodurch ein Informationsaustausch zwischen solchen Knotengruppen möglich wird [32].

6 Zusammenfassung

Im Zentrum dieser Arbeit stand der $t\bar{t} + b\bar{b}$-Prozess, der unter anderen bei Proton-Proton-Kollisionen am Large Hadron Collider erzeugt wird. Die genaue Untersuchung dieses Prozesses stellt eine wichtige Aufgabe dar, um das Standardmodell der Teilchenphysik besser verstehen zu können. Eines der Experimente, welches diese Kollisionsereignisse genauer untersucht ist das CMS-Experiment. Für die Weiterentwicklung der bisherigen Analysemethoden am CMS-Experiment hat sich diese Arbeit mit der Entwicklung einer neuen Klassifikationsmethode für die zusätzlichen B-Jets im $t\bar{t} + b\bar{b}$-Prozess beschäftigt. Hierfür wurden in den Abschnitten 4.3 - 4.5 verschiedene Modelle rund um Gated Graph Sequence Neural Networks untersucht. Es wurde zunächst eine Minimalversion eines GNN-Modells betrachtet, das GGS-NN-Modell. Dieses besteht lediglich aus einem GGS-NN-Modul und der zwingend notwendigen linearen Transformation und Aktivierungsfunktion. Diese Minimalvariante wurde dann mit weiteren Komponenten versehen und komplexer gestaltet. So wurde der Einfluss eines Multi-Lagen-Perzeptrons als Lage zur Vorprozessierung untersucht, sowie mehrere GGS-NN-Module mit und ohne Skip-Connection hintereinandergeschaltet.

Dabei hat die Auswertung gezeigt, dass allein durch das Modifizieren der Architektur des GGS-NN-Modells, keine signifikanten Verbesserungen mehr erzielt werden konnten. Während das Verwenden von MLPs die Ergebnisse sogar verschlechtert hat, konnte nur das Multi-GGS-NN-Modell ohne Skip-Connection Ergebnisse erzielen, die leicht besser als die der Minimalvariante sind. Dieses Modell ist der Minimalvariante sehr ähnlich, womit sich die simpleren Modelle hier also als effektiver erwiesen haben.

Dass mit diesem Modell allerdings ein gewisses Plateau erreicht worden ist, bestätigen auch die Ergebnisse der weiterführenden Tests in Abschnitt 5.2. Trotz verschiedener, physikalisch motivierter Überlegungen, konnten auch hier die Ergebnisse des besten Modells nicht mehr signifikant verbessert werden. Lediglich die Verwendung der invarianten Masse (s. Abschnitt 5.2.3) erzielte in den Tests eine Verbesserung über die Standardabweichung der Vergleichswerte hinaus. In den weiterführenden Untersuchungen hat sich auch eine Schwachstelle des Modells herauskristallisiert: der Graph, und somit auch das Modell, ist nicht in der Lage, explizit Informationen zu Gruppen von mehr als zwei Knoten zu speichern. Somit können wertvolle Informationen wie die invariante Masse oder Jet-Ladung nicht vollständig genutzt werden.

Dennoch haben die Ergebnisse der Kapitel 4 und 5 deutlich gezeigt, dass Graph Neural Networks sehr gut für die Klassifikation der AddB-Jets im $t\bar{t} + b\bar{b}$-Prozess geeignet sind.

Das hier verwendete GNN erzielt eine Verbesserung von über 6 % auf den 2/2-Wert im Vergleich zu der bisherigen Methode mittels Deep Neural Networks aus [1]. Ein möglicher Vorteil von GNNs, der diese Verbesserungen erklären könnte, ist die Graphstruktur selbst, die durch die gewichteten Kanten einen induktiven Bias berücksichtigt. Weiterhin kann bei GNNs immer die vollständige Information eines Ereignisses betrachtet werden. Allerdings wird auch die verwendete GNN-Variante selbst eine zentrale Rolle bei den Verbesserungen der Ergebnisse spielen. Insgesamt ist es also sehr schwierig zu sagen, was letztendlich für die Verbesserungen verantwortlich ist, da GNNs und DNNs von grundsätzlich unterschiedlicher Natur sind, was einen Vergleich in vielerlei Hinsicht erschwert.

Insgesamt hat diese Arbeit gezeigt, dass GNNs ein vielversprechender, neuer Zweig im Bereich des maschinellen Lernens in der Teilchenphysik ist. Daher lohnt sich der Fokus auf die Entwicklung weiterer Klassifikationsmethoden basierend auf Graphen. Um mit GNNs weitere Verbesserungen zu erzielen, müssen in künftigen Untersuchungen allerdings neue Ansätze herangezogen werden. Solche Ansätze könnten z.B. die Verwendung von Hypergraphen oder die Verwendung von verschiedenen Kantentypen sein.

Literatur

[1] E. L. Pfeffer. *Studies on $t\bar{t} + b\bar{b}$ production at the CMS experiment*. Masterthesis. Karlsruher Institut für Technologie (KIT), 2021.

[2] D. J. Griffiths. *Introduction to elementary particles*. 2., revised edition. Physics textbook. Weinheim: Wiley-VCH Verlag GmbH Co. KGaA, 2008. ISBN: 3527406018; 9783527406012.

[3] O. Philipsen. *Quantenfeldtheorie und das Standardmodell der Teilchenphysik : Eine Einführung*. SpringerLinkSpringer eBook Collection. Berlin: Springer Spektrum, [2018]. ISBN: 9783662578209.

[4] *Standard-Modell der Elementarteilchen*. Aufgerufen am 04.09.2021. URL: https://commons.wikimedia.org/wiki/File:Standard_Model_of_Elementary_Particles-de.svg#/media/Datei:Standard_Model_of_Elementary_Particles-de.svg.

[5] P. W. Higgs. „Broken Symmetries and the Masses of Gauge Bosons“. In: *Phys. Rev. Lett.* 13 (16 Okt. 1964), S. 508–509. DOI: 10.1103/PhysRevLett.13.508.

[6] 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), S. 1–29. ISSN: 0370-2693. DOI: https://doi.org/10.1016/j.physletb.2012.08.020.

[7] The CMS Collaboration. „Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC“. In: *Physics Letters B* 716.1 (2012), S. 30–61. ISSN: 0370-2693. DOI: https://doi.org/10.1016/j.physletb.2012.08.021.

[8] CERN. *Facts and figures about the LHC*. Aufgerufen am 09.09.2021. URL: https://home.cern/resources/faqs/facts-and-figures-about-lhc#:.

[9] The ATLAS Collaboration. „The ATLAS Experiment at the CERN Large Hadron Collider“. In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08003–S08003. DOI: 10.1088/1748-0221/3/08/s08003.

[10] The ALICE Collaboration. „The ALICE experiment at the CERN LHC“. In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08002–S08002. DOI: 10.1088/1748-0221/3/08/s08002.

[11] The LHCb Collaboration. „The LHCb Detector at the LHC“. In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08005–S08005. DOI: 10.1088/1748-0221/3/08/s08005.

[12] The CMS Collaboration. „The CMS experiment at the CERN LHC“. In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08004–S08004. DOI: 10.1088/1748-0221/3/08/s08004.

[13] E. Mobs. *The CERN accelerator complex - 2019. Complexe des accélérateurs du CERN - 2019*. General Photo. Juli 2019. URL: https://cds.cern.ch/record/2684277.

[14] S. R. Davis. *Interactive Slice of the CMS detector*. Aug. 2016. URL: http://cds.cern.ch/record/2205172.

[15] The CMS Collaboration. „Particle-flow reconstruction and global event description with the CMS detector“. In: *Journal of Instrumentation* 12.10 (Okt. 2017), P10003–P10003. DOI: 10.1088/1748-0221/12/10/p10003.

[16] M. Cacciari, G. P. Salam und G. Soyez. „The anti-k_t jet clustering algorithm“. In: *Journal of High Energy Physics* 2008.04 (Apr. 2008), S. 063–063. DOI: 10.1088/1126-6708/2008/04/063.

[17] The CMS Collaboration. *Performance of the DeepJet b tagging algorithm using 41.9/fb of data from proton-proton collisions at 13TeV with Phase 1 CMS detector*. Nov. 2018. URL: https://cds.cern.ch/record/2646773.

[18] The CMS Collaboration. „Measurement of the cross section for $\mathrm{t\bar{t}}$ production with additional jets and b jets in pp collisions at $\sqrt{s} = 13\,\mathrm{TeV}$“. In: *Journal of High Energy Physics* 2020.7 (Juli 2020), S. 125. ISSN: 1029-8479. DOI: 10.1007/JHEP07(2020)125.

[19] T. Sjöstrand u. a. „An introduction to PYTHIA 8.2“. In: *Computer Physics Communications* 191 (Juni 2015), S. 159–177. ISSN: 0010-4655. DOI: 10.1016/j.cpc.2015.01.024.

[20] S. Alioli u. a. „A general framework for implementing NLO calculations in shower Monte Carlo programs: the POWHEG BOX“. In: *Journal of High Energy Physics* 2010.6 (Juni 2010). ISSN: 1029-8479. DOI: 10.1007/jhep06(2010)043.

[21] F. Maltoni, G. Ridolfi und M. Ubiali. „b-initiated processes at the LHC: a reappraisal“. In: *Journal of High Energy Physics* 2012 (Juli 2012). ISSN: 1029-8479. DOI: 10.1007/JHEP07(2012)022.

[22] P. Hartmann. *Mathematik für Informatiker: ein praxisbezogenes Lehrbuch*. 7. Auflage. Springer eBook Collection. Wiesbaden: Springer Vieweg, 2019. ISBN: 9783658265243.

[23] P. W. Battaglia u. a. „Relational inductive biases, deep learning, and graph networks“. In: *CoRR* abs/1806.01261 (2018). arXiv: 1806.01261.

[24] Y. Li u. a. *Gated Graph Sequence Neural Networks*. 2017. arXiv: 1511.05493 [cs.LG].

[25] K. Cho u. a. *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*. 2014. arXiv: 1406.1078 [cs.CL].

[26] J. Chung u. a. *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*. 2014. arXiv: 1412.3555 [cs.NE].

[27] Pytorch Geometric. *Advanced Mini-Batching*. Aufgerufen am 29.09.2021. URL: https://pytorch-geometric.readthedocs.io/en/latest/notes/batching.html.

[28] L. Boué. *Deep learning for pedestrians: backpropagation in CNNs*. 2018. arXiv: 1811.11987 [cs.LG].

[29] D. P. Kingma und J. Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[30] J. You, R. Ying und J. Leskovec. *Design Space for Graph Neural Networks*. 2021. arXiv: 2011.08843 [cs.LG].

[31] M. Schlichtkrull u. a. *Modeling Relational Data with Graph Convolutional Networks*. 2017. arXiv: 1703.06103 [stat.ML].

[32] Y. Feng u. a. *Hypergraph Neural Networks*. 2019. arXiv: 1809.09401 [cs.LG].

Anhang

A Verteilungen der Jet-Kinematiken

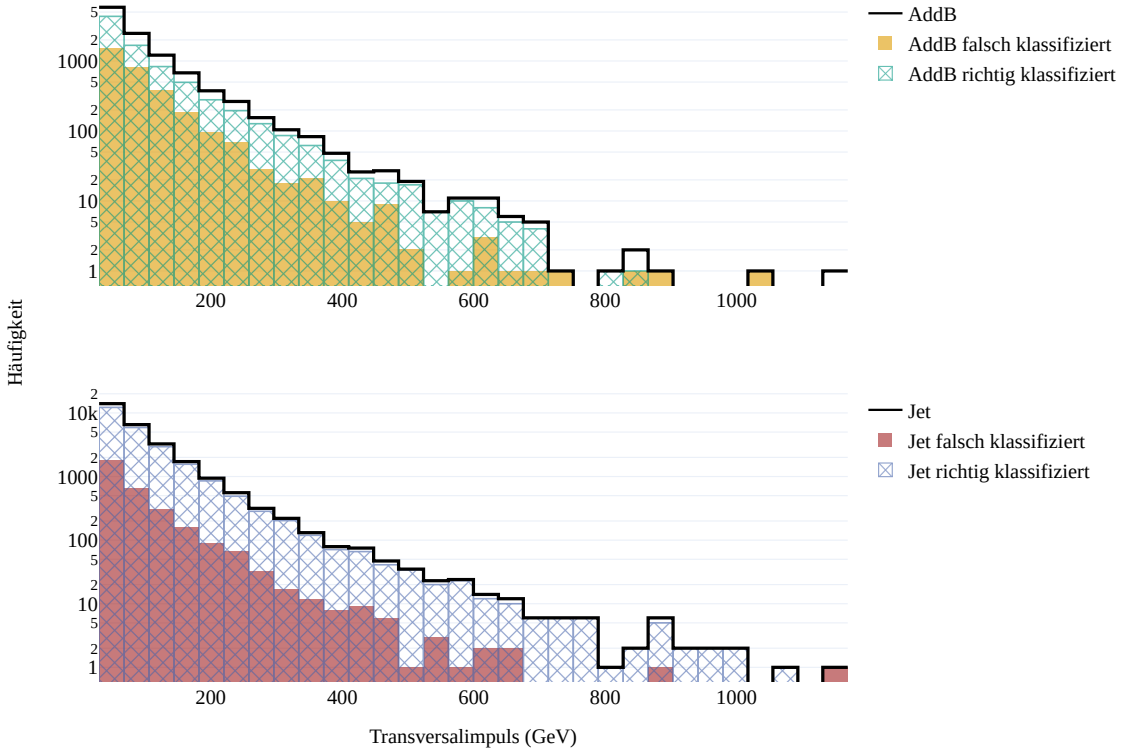


Abbildung A.1: Verteilung des Transversalimpulses für falsch und richtig klassifizierte AddB-Jets und für andere Jets: die obere Darstellung zeigt die Verteilungen des Transversalimpulses für alle AddB-Jets im Vergleich zu den Verteilungen der falsch und richtig klassifizierten AddB-Jets. Die untere Darstellung stellt den gleichen Sachverhalt für die anderen Jets dar. Die Daten beziehen sich auf den Test-Datensatz des besten Durchlaufs des Multi-GGS-NN-Modells ohne Skip-Connection und mit $L_k = 6$.

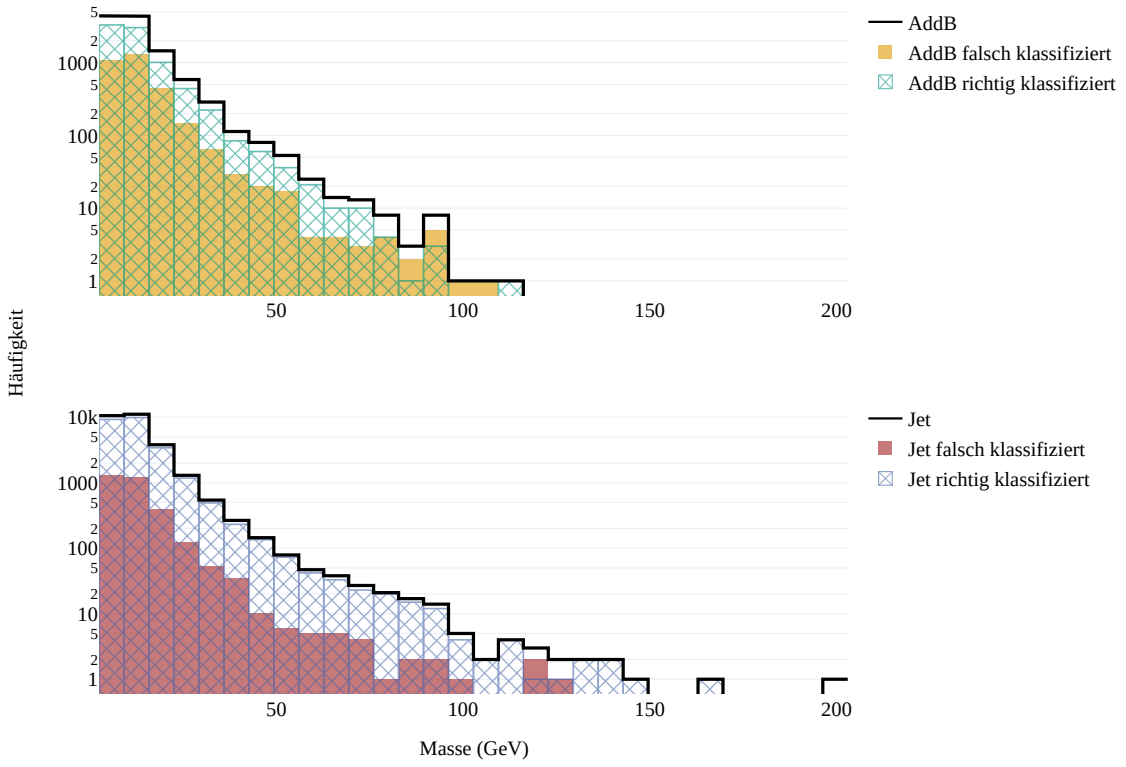


Abbildung A.2: Verteilung der Masse für falsch und richtig klassifizierte AddB-Jets und für andere Jets: die obere Darstellung zeigt die Verteilungen der Massen für alle AddB-Jets im Vergleich zu den Verteilungen der falsch und richtig klassifizierten AddB-Jets. Die untere Darstellung stellt den gleichen Sachverhalt für die anderen Jets dar. Die Daten beziehen sich auf den Test-Datensatz des besten Durchlaufs des Multi-GGS-NN-Modells ohne Skip-Connection und mit $L_k = 6$.

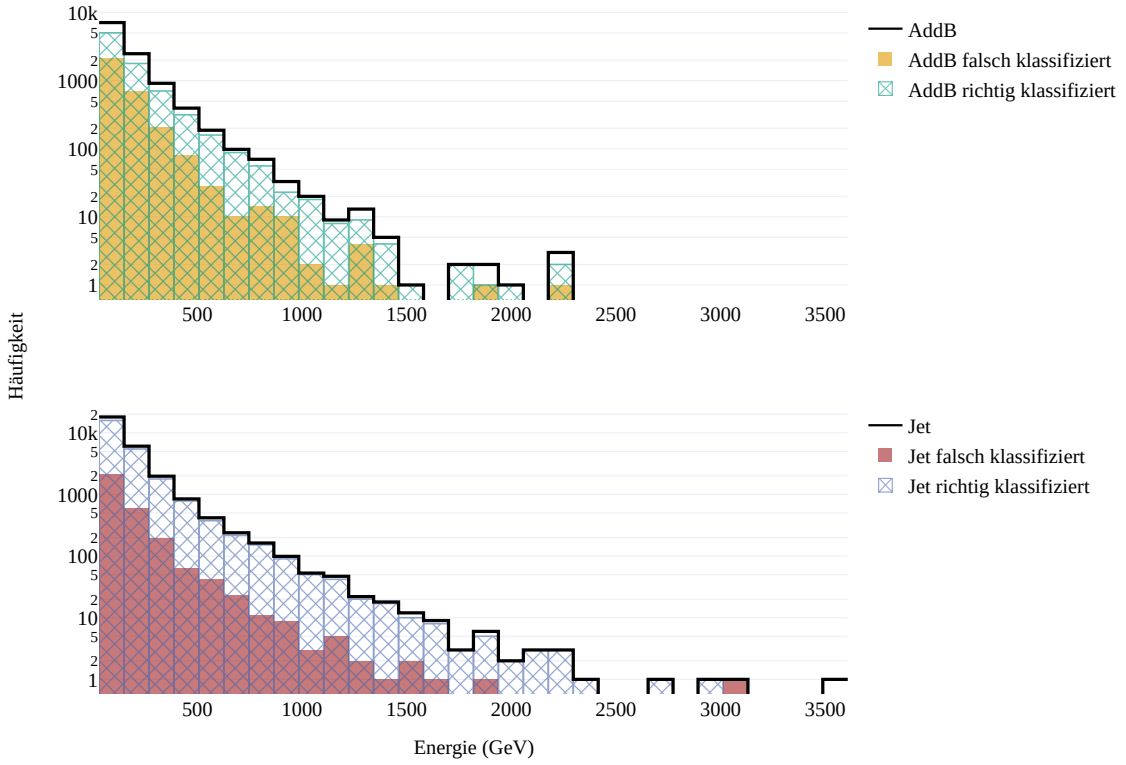


Abbildung A.3: Verteilung der Energie für falsch und richtig klassifizierte AddB-Jets und für andere Jets: die obere Darstellung zeigt die Verteilungen der Energie für alle AddB-Jets im Vergleich zu den Verteilungen der falsch und richtig klassifizierten AddB-Jets. Die untere Darstellung stellt den gleichen Sachverhalt für die anderen Jets dar. Die Daten beziehen sich auf den Test-Datensatz des besten Durchlaufs des Multi-GGS-NN-Modells ohne Skip-Connection und mit $L_k = 6$.

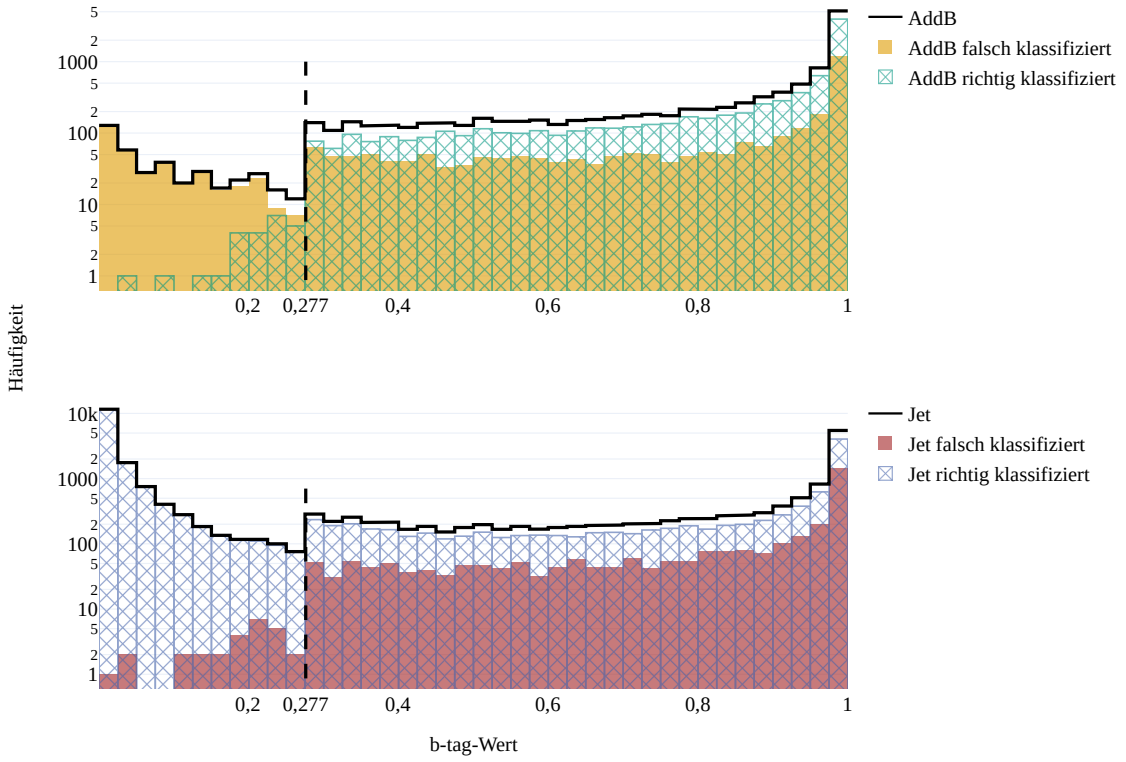


Abbildung A.4: Verteilung der b-tag-Werte für falsch und richtig klassifizierte AddB-Jets und für andere Jets: die obere Darstellung zeigt die Verteilungen der b-tag-Werte für alle AddB-Jets im Vergleich zu den Verteilungen der falsch und richtig klassifizierten AddB-Jets. Die untere Darstellung stellt den gleichen Sachverhalt für die anderen Jets dar. Die Daten beziehen sich auf den Test-Datensatz des besten Durchlaufs des Multi-GGS-NN-Modells ohne Skip-Connection und mit $L_k = 6$.

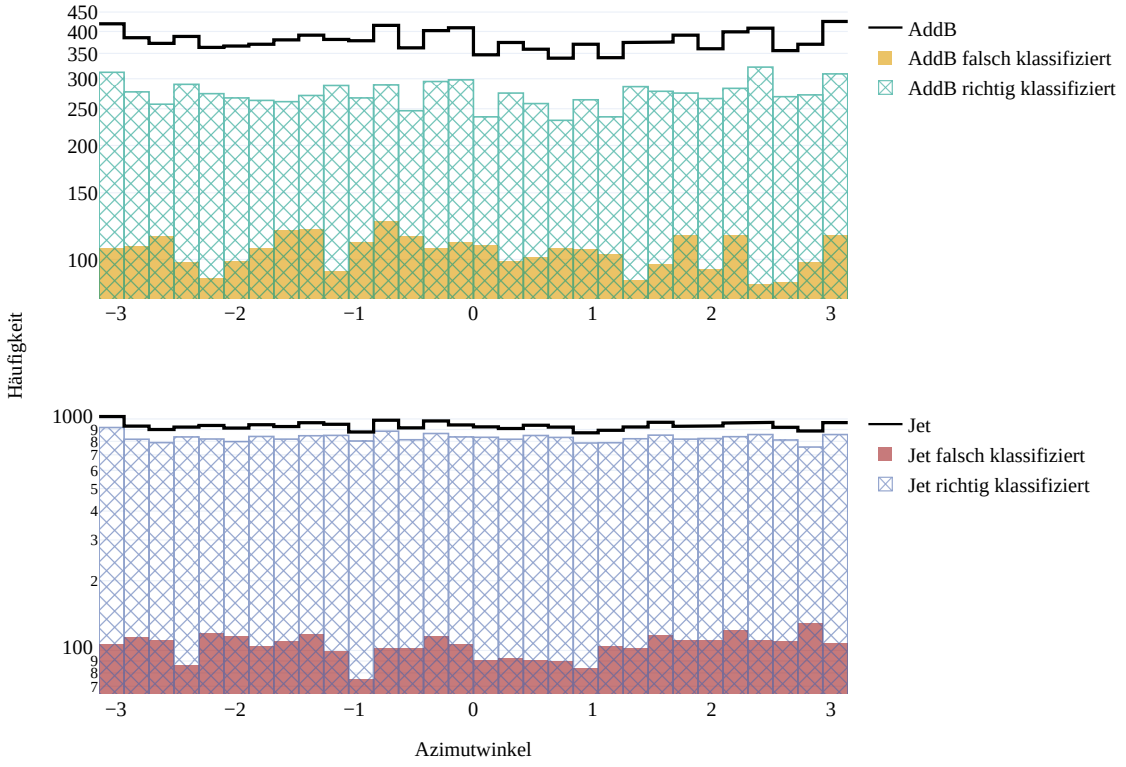


Abbildung A.5: Verteilung des Azimutwinkels für falsch und richtig klassifizierte AddB-Jets und für andere Jets: die obere Darstellung zeigt die Verteilungen des Azimutwinkels für alle AddB-Jets im Vergleich zu den Verteilungen der falsch und richtig klassifizierten AddB-Jets. Die untere Darstellung stellt den gleichen Sachverhalt für die anderen Jets dar. Die Daten beziehen sich auf den Test-Datensatz des besten Durchlaufs des Multi-GGS-NN-Modells ohne Skip-Connection und mit $L_k = 6$.

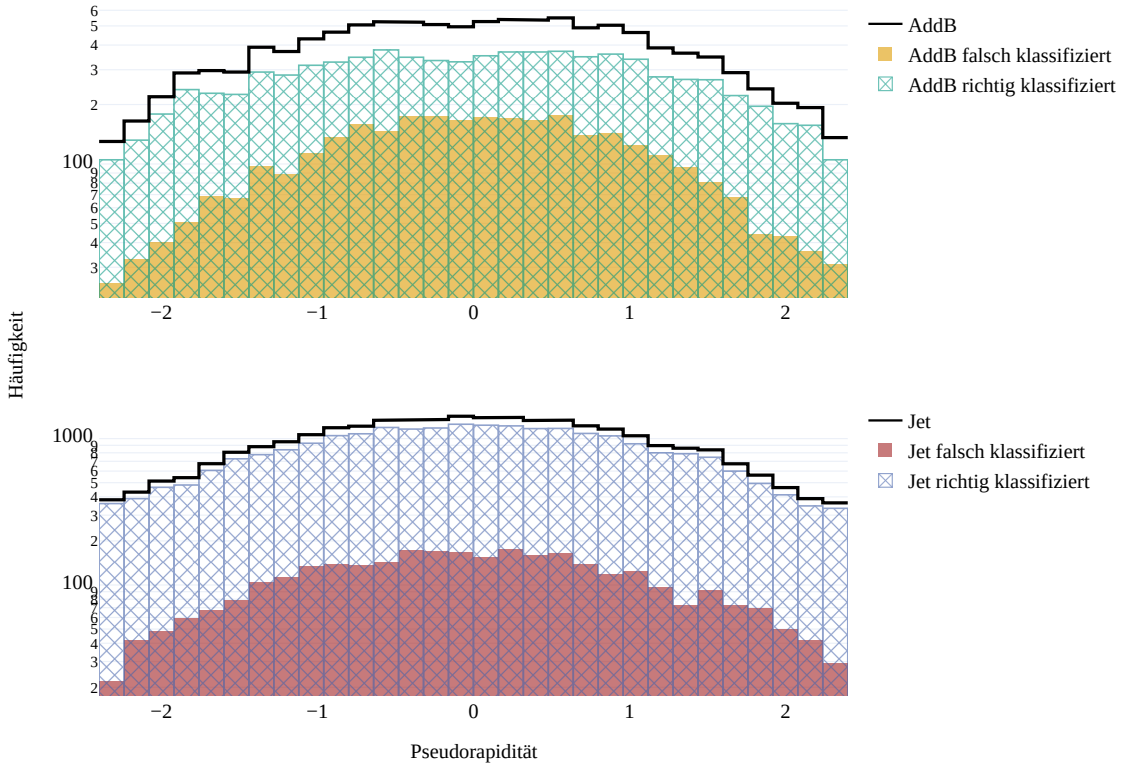


Abbildung A.6: Verteilung der Pseudorapidität für falsch und richtig klassifizierte AddB-Jets und für andere Jets: die obere Darstellung zeigt die Verteilungen der Pseudorapidität für alle AddB-Jets im Vergleich zu den Verteilungen der falsch und richtig klassifizierten AddB-Jets. Die untere Darstellung stellt den gleichen Sachverhalt für die anderen Jets dar. Die Daten beziehen sich auf den Test-Datensatz des besten Durchlaufs des Multi-GGS-NN-Modells ohne Skip-Connection und mit $L_k = 6$.

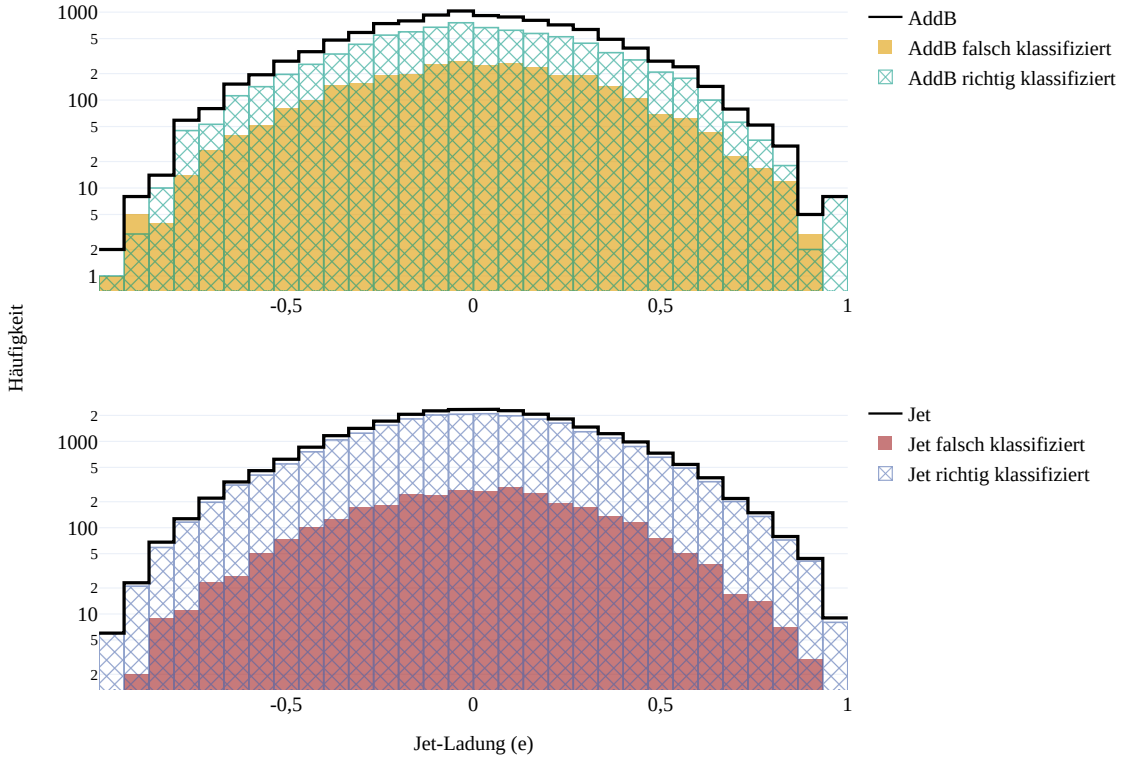


Abbildung A.7: Verteilung der Jet-Ladung für falsch und richtig klassifizierte AddB-Jets und für andere Jets: die obere Darstellung zeigt die Verteilungen der Jet-Ladung für alle AddB-Jets im Vergleich zu den Verteilungen der falsch und richtig klassifizierten AddB-Jets. Die untere Darstellung stellt den gleichen Sachverhalt für die anderen Jets dar. Die Daten beziehen sich auf den Jet-Ladungs-Test.

Abbildungsverzeichnis

Tabellenverzeichnis