Санкт-Петербургский государственный университет

Кафедра компьютерного моделирования и многопроцессорных систем

ЩАВЕЛЕВ Егор Михайлович

Применение глубоких нейронных сетей к задаче трекинга для детектора GEM в эксперименте BM@N мегапроекта NICA

Магистерская диссертация

Направление 02.04.02

Фундаментальная информатика и информационные технологии

Основная образовательная программа магистратуры

Вычислительные технологии

Научные руководители:
доцент, кафедра компьютерного
моделирования и многопроцессорных систем
к. т. н. Гришкин Валерий Михайлович

профессор,
главный научный сотрудник ОИЯИ,
д.ф.-м.н. Ососков Геннадий Алексеевич

Санкт-Петербург
2019

Оглавление

Введение

В ходе решения задач любого уровня сложности всегда появляется необходимость работать с информацией. В современном мире зачастую правильная обработка и интерпретация данных является ключом к решению многих проблем. Особенно актуально это будет и в ближайшем будущем, так как близится новая эпоха намного более значительных как по объему, так и по трудоемкости вычислений задач. И именно из-за этого необходимо использовать самые современные способы и инструменты для работы с такими огромными массивами данных.

В частности, экспериментальная физика высоких энергий и ядерная физика требуют не только гигантских комплексов для распределенных вычислений и соответствующих сетевых инфраструктур, но и того, что является наиболее важным в контексте настоящей работы – подходов машинного обучения для поиска неочевидных закономерностей в данных для получения наиболее вероятного прогноза изучаемых явлений. Искусственные нейронные сети с их способностью к обучению и самообучению являются эффективными инструментами машинного обучения, поэтому физики накопили довольно обширный опыт применения различных нейронных сетей во многих экспериментах: для распознавания треков заряженных частиц, колец черенковского излучения, физических проверок гипотез и обработки изображений. Одной из основных составляющих частей современных экспериментов физики высоких энергий является задача трекинга частиц в детекторах. При решении такой задачи исследователи сталкиваются с большим количеством проблем, возникающих как на этапе работы с оборудованием детектора, так и с проблемами отслеживания траекторий частиц.

Проблема восстановления траекторий пролёта элементарных частиц, образовавшихся при взаимодействиях пучков в современных экспериментах физики высоких энергий и ядерной физики, по данным, полученным в результате регистрации следов каждой частицы элементами электронных детекторов, является особенно важной и сложной для физиков-экспериментаторов. Наиболее актуальными сейчас являются

эксперименты с тяжелыми ионами, позволяющие проверить физические теории о свойствах барионной материи и ненайденным пока ее состоянием, называемым кварк-глюонной плазмой. Такие эксперименты бывают двух типов: в одних пучок тяжелых ионов, разогнанных ускорителем до субсветовой скорости, ударяет в мишень так, что продукты взаимодействия в виде сотен и тысяч траекторий, называемых треками, летят вперед в узком конусе и регистрируются в специальных трековых детекторах. К такому типу относится эксперимент BM@N, обработке трековых данных с которого посвящена настоящая диссертация.

Другой тип экспериментов – коллайдерные, когда мишени нет, а соударяются ионы разных зарядов, пучки которых разгоняются ускорителем-коллайдером в противоположных направлениях так, чтобы после разгона пучки могли столкнуться с удвоенной скоростью. К такому типу относятся коллайдерные эксперименты, проводимые в ЦЕРНе на большом адронном коллайдере (БАК) и планируемые эксперименты MPD и SPD, которые будут работать на коллайдере NICA в ОИЯИ. В отличии от экспериментов с фиксированной мишенью, треки частиц, образовавшихся при столкновении ионов, разлетаются в пространстве внутри установки во все стороны под углами, распределенными в 4π.

Трековые детекторы в обоих типах экспериментов помещены внутрь сильных магнитов, чтобы по искривлению траектории каждой частицы, образовавшейся в результате столкновения, определить ее импульс, позволяющий идентифицировать эту частицу.

Трековые детекторы в экспериментах с фиксированной мишенью состоят из координатных плоскостей, образованных электронными элементами, в которых появляется сигнал, наведенный пролетевшей рядом частицей. В случае пиксельного детектора эти элементы состоят из малых ячеек – педов, регистрирующих двумерные координаты сигнала. Однако из-за большой дороговизны таких детекторов, более популярными стали детекторы, состоящие из чувствительных линейных элементов - проволочек в электростатическом поле или тонких полосок – стрипов на силиконовой подложке. Пролетевшая частица "зажигает",

т. е. активирует линию в координатной плоскости, а для получения второй координаты нужна еще одна близкая плоскость с другим направлением стрипов или проволочек. Пересечение зажженных линий и даст нам координаты места близко от которого пролетела частица. Следует сразу же отметить большое неудобство, которым приходится платить за выбор более дешевых стриповых детекторов: когда частиц много, то помимо реальных пересечений, соответствующих месту пролета частицы, появится еще на порядок больше ложных пересечений, очень и очень засоряющих результаты измерений.

Таким образом, задача восстановления траекторий частиц в современных стриповых детекторах актуальна и в наши дни. Огромное количество «ложных» срабатываний при регистрации частиц в детекторах такого типа, а отсюда необходимость их обработки с максимальной точностью наиболее эффективным образом, является одной из основных проблем, которые будут возникать в ходе использования GEM детектора эксперимента BM@N мегапроекта NICA. Решение такой задачи и является целью настоящей работы.

1. Постановка задачи

В рамках настоящей магистерской диссертации ставились следующие задачи.

1. Выполнить обзор существующих подходов к задаче трекинга частиц.

2. Разработать алгоритм препроцессинга модельных данных сгенерированных для GEM детектора.

3. Разработать и протестировать подход к задаче трекинга с использованием глубоких нейронных сетей.

2. Обзор существующих решений

Определение истинных точек траекторий пролета частиц в детекторе среди множества фейковых и близких соседних точек вырождается в одну из классических проблем в области распознавания образов – для некоторого множества точек, имеющем в себе как шумовую, так и истинную компоненту, необходимо определить подлинные кривые траекторий каких-либо объектов с максимальной точностью. Особенно актуальной и сложной такая проблема оказывается в ядерной физике высоких энергий, где, исходя из данных полученных на детекторах, необходимо как определить траекторию конкретной заряженной частицы среди огромного количества других частиц, так и дать оценку ее параметров, включая вычисление импульса частицы с максимально возможной точностью, который, в свою очередь, определяется с помощью определения кривизны траектории данной частицы. Хотя такая проблема и имеет более чем полувековую историю, методы ее решений радикально менялись по ходу совершенствования систем и детекторов – начиная от пузырьковых камер, где всего несколько треков фиксировались на стереофотографиях, и заканчивая современными экспериментами с тяжелыми ионами, в ходе которых в результате столкновений ионов рождаются тысячи и десятки тысяч треков, фиксируемых различными электронными детекторами, которые хранят обработанную информацию в виде массивов данных в памяти компьютеров, занимающихся обработкой таких событий. Так, например, знаменитый Большой Адронный Коллайдер (LHC) в настоящее время находится на пороге перехода в новый этап своего развития – к 2026 году учёными детектор будет переведен в High Luminocity фазу, в ходе которой количество столкновений частиц будет увеличено на несколько порядков, что, в свою очередь, повлечет необходимость работы с огромными объемами информации.

Именно поэтому один из главных критериев применимости алгоритмов и систем, использующихся в современных экспериментах: необходимо обеспечить максимальную скорость вычислений при предельно

достижимой их точности, при этом сохраняя высокую эффективность методов оценки физических параметров, интересующих экспериментаторов. Реализация этих требований при наличии особенно сложной текстуры и зашумленности данных естественным образом натолкнулась на ограниченность традиционных применяемых подходов, например таких базовых, как кластерный анализ и метод наименьших квадратов, так как в таких условиях они не могли обеспечить одновременно и точности, и скорости вычислений, и высокой эффективности оценок параметров.

В настоящей главе выполнен краткий обзор методов, применяемых в реконструкции данных и в распознавании траекторий частиц.

2.1. Конформное отображение

Метод конформного отображения [9] для нахождения треков основан на том факте, что окружности, проходящие через начало двумерной системы координат $x - y$ преобразуются в прямые линии в системе координат $u - v$ с помощью преобразования

$$u = \frac{x}{x^2 + y^2}, v = \frac{y}{x^2 + y^2}, \tag{1}$$

где окружности определены с помощью уравнения окружности $(x-a)^2+(y-b)^2 = r^2 = a^2 + b^2$. Прямые линии в плоскости $u - v$ определяются как

$$v = \frac{1}{2b} - u\frac{a}{b}. \tag{2}$$

Для больших значений r или, иначе говоря, треков с высоким импульсом, прямые линии проходят близко к началу координат, и кандидаты в треки могут быть получены путем преобразования измерений из плоскости $u - v$ в сферическую систему координат. Затем, собрав данные по угловой составляющей в виде гистограмм, поиск треков отслеживается путем поиска пиков в этой гистограмме.

2.2. Преобразование Хафа

В случае, если прямые линии не проходят близко к началу координат (т. е. присутствуют треки с большим диапазоном возможных импульсов), необходим более общий подход для определения местоположения линий. Преобразования Хафа[10] хорошо подходит для такой задачи. Его идея основана на простом преобразовании уравнения прямой линии в плоскости $x - y$, $y = cx + d$, в другую прямую линию в плоскости $c - d$, $d = -xc + y$. Точки на прямой в плоскости $c - d$ соответствуют всем возможным линиями, проходящим через точку (x, y) в плоскости $x - y$. Следовательно, точки, лежащие вдоль прямой линии в плоскости $x - y$, имеют тенденцию создавать линии в плоскости $c - d$, пересекающиеся в точке, которая определяет параметры этой линии в плоскости $x - y$. Что касается метода конформного отображения, положение пиков на гистограмме предоставляет информацию о параметрах линий в пространстве $x - y$. В отличие от одномерного пространства параметров метода конформного отображения, пространство параметров в этом случае двумерно. Преобразование Хафа быстро теряет эффективность поиска треков, если попытаться перейти в пространство параметров с размером больше 2.

Для нахождения трека в дрейфовых трубках (drift tubes), дрейфовые круги, обеспечиваемые известным дрейфовым расстоянием каждого из измерений, могут быть преобразованы в синусоидальные кривые в сферической системе координат путем применения преобразования Лежандра[1]. Пики на пересечениях нескольких синусоидальных кривых в этой системе координат дают не только набор дрейфовых трубок, зарегистрировавших одну и ту же частицу, но также и решения некоторых неопределенностей, присущей этому типу детекторной системы. Иллюстрация показана на рисунке 1.

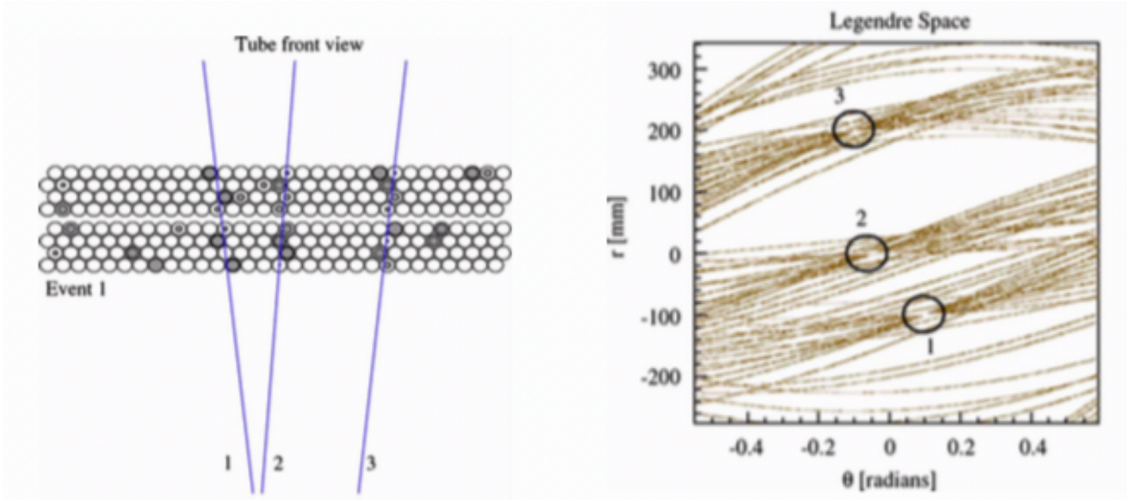

Рис. 1: Изображение из статьи[1].

2.3. Прослеживание трека

Примером локального подхода к поиску пути является так называемый метод отслеживания пути (track road). Он начинается с набора измерений, которые могли быть созданы конкретной заряженной частицей. Модель трека, например форма траектории, может использоваться для интерполяции измерений и создания «коридора» вокруг траектории. Точки внутри границ коридора являются кандидатом на трек. Количество точек и качество последующей посадки треков используются для оценки правильности гипотезы треков.

Похожим подходом является подход «прослеживания» трека, который начинается со специальных «начальных» значений, сидов (track seed). В большинстве случаев сиды представляют собой короткий отрезок пути, построенный из нескольких точек. Кроме того, его можно ограничить указанием на область взаимодействия. Сиды могут быть сконструированы во внутренней области детектора вблизи области взаимодействия, где измерения часто имеют очень высокую точность, или во внешней области, где плотность измерений ниже. Из начального числа трек экстраполируется на следующий слой детектора, содержащий измерение. Точка, наиболее близкая к прогнозируемому треку, включается в число кандидатов на трек.

Еще один из базовых подходов к трекингу – «подгонка» трека (track fitting). Фитирование трека направлено на оценку множества или вектора параметров, представляющих кинематическое состояние заряженной частицы, на основе информации, содержащейся в различных измерениях позиций трек-кандидатов. Поскольку эти позиции являются стохастическими величинами с неопределенностями, оценка представляет собой статистическую процедуру. В дополнение к оценочным значениям параметров трека, представляется возможным также получить меру неопределенности этих значений в терминах ковариационной матрицы вектора параметров трека. Большинство методов оценки могут быть разложены на набор базовых компонентов, и основные методы определения траекторий отличаются по логике того, как эти компоненты объединяются.

2.4. Метод наименьших квадратов

Подавляющее большинство экспериментальных реализаций используют какой-либо линейный метод наименьших квадратов для задачи трекинга. Метод линейных глобальных наименьших квадратов является оптимальным, если модель треков является линейной, то есть если описывающая трек функция $f_{k|i}$ от слоя i детектора до уровня k детектора является линейной функцией вектора состояния q_i, и если все плотности вероятностей, встречающиеся в течение процедуры оценки – гауссовы. Если функция f является нелинейной, линейный метод наименьших квадратов по-прежнему применим. Однако, хотя оценки методом наименьших квадратов легко вычислить, им не хватает робастности[13].

Начальным шагом для использования метода наименьших квадратов является получения функциональной зависимости между начальным состоянием q_0 частицы и вектором измерений m_k на слое k:

$$m_k = d_k(q_0) + \gamma_k, \tag{3}$$

где d_k – композиция измерений модельной функции $m_k = h_k(q_k)$ и функ-

ции f, описывающей трек:

$$d_k = h_k \circ f_{k|k-1} \circ \cdots \circ f_{2|1} \circ f_{1|0}. \tag{4}$$

Член γ_k является стохастическим и содержит все кратные кулоновские рассеяния вплоть до слоя k, а также погрешность измерения m_k. Линейная оценка требует линеаризованной модели трека, и для этого необходим якобиан D_k, состоящий из d_k,

$$D_k = H_k F_{k|k-1} \cdots F_{2|1} F_{1|0}, \tag{5}$$

где H – есть якобиан h, а F якобиан f. Наблюдения m_k, функции d_k, якобианы D_k, и шумовые компоненты γ_k могут быть представлены в одном векторе или матрице,

$$m = \begin{pmatrix} m_1 \\ \vdots \\ m_n \end{pmatrix}, d = \begin{pmatrix} d_1 \\ \vdots \\ d_n \end{pmatrix}, D = \begin{pmatrix} D_1 \\ \vdots \\ D_n \end{pmatrix}, \gamma = \begin{pmatrix} \gamma_1 \\ \vdots \\ \gamma_n \end{pmatrix}, \tag{6}$$

где n – общее число слоёв измерений. Модель представима в виде:

$$m = d(q_0) + \gamma; \tag{7}$$

в линеаризованном виде:

$$m = D(q_0) + c + \gamma,$$

где c – константа. Глобальная оценка метода наименьших квадратов состояния q_0 оценивается:

$$\tilde{q_0} = D^T G D^{-1} D^T G (m - c), \tag{8}$$

где $V = G^{-1}$ является недиагональной ковариационной матрицей γ. Если имеется значительное многократное рассеяние, оцениваемый трек может значительно отклоняться от реального трека. За фактическим треком можно следить более точно путем явной оценки двух проецируемых углов рассеяния на каждом слое детектора или на наборе вирту-

альных точек останова (breakpoints) внутри непрерывного рассеивателя[11][4].

2.5. Фильтр Калмана

Большое количество измерений или точек останова приводит к высокой вычислительной стоимости этих методов из-за необходимости инвертировать большие матрицы. Рекурсивное представление метода наименьших квадратов, фильтр Калмана, требует инверсии только небольших матриц и обладает той же особенностью, что и метод точки останова, при котором довольно точно отслеживается фактический трек[3][7], с тем преимуществом, что такие эффекты, как многократное рассеяние и потеря энергии, можно обрабатывать локально. Фильтр Калмана работает путем чередования этапов прогнозирования и обновления (Рис.2).

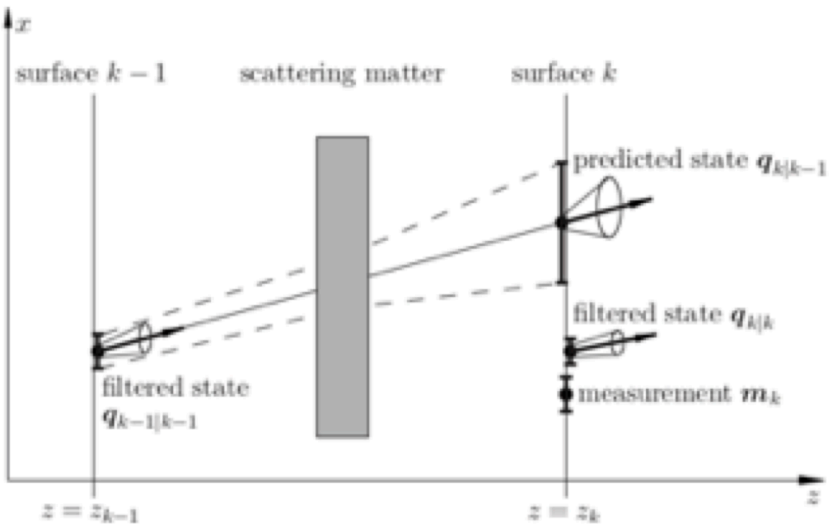

Рис. 2: Прогноз и шаг фильтра Калмана. Распространение происходит в направлении z, в то время как измеряется координата x.

Этап прогнозирования распространяет оцениваемый параметр трека $q_{k-1|k-1}$ из слоя детектора $k-1$ на следующий слой, содержащий измерение

$$q_{k|k-1} = f_{k|k-1}(q_{k-1|k-1}), \tag{9}$$

так же как и соответствующую ковариацинную матрицу

$$C_{k|k-1} = F_{k|k-1} C_{k-1|k-1} F_{k|k-1}^T + Q_k, \tag{10}$$

где Q_k – ковариационная матрица многократного рассеивания после слоя $k-1$ до k включительно. Часть Q_k, возникающая в результате рассеяния между слоями, должна распространяться на слой k соответствующим якобианом. Этап обновления корректирует прогнозируемый вектор состояния, используя информацию из измерения на уровне k,

$$q_{k|k} = q_{k|k-1} + K_k[m_k - h_k(q_{k|k-1})], \tag{11}$$

где матрица усиления K_k:

$$K_k = C_{k|k-1} H_k^T (V_k + H_k C_{k|k-1} H_k^T)^{-1}, \tag{12}$$

и V_k – ковариационная матрица m_k. Ковариационная матрица обновляется

$$C_{k|k} = (I - K_k H_k) C_{k|k-1}. \tag{13}$$

2.6. Подходы на основе фильтра Калмана в наши дни

Использование фильтра Калмана для задач трекинга даёт превосходные результаты и активно применяется как 20 лет назад, так и в наши дни. Однако один из основных его недостатков – в современных экспериментах количество точек и измерений настолько велико, что скорости работы алгоритмов, использующих такой подход совершенно недостаточно для полной обработки событий в детекторах. Современные узлы вычислений в своей основе многопроцессорные и распределенные, что требует от алгоритмов иметь свойство хорошо распараллеливаться по разным процессорам и узлам.

В настоящее время учёные и исследователи всё чаще обращают внимание на набирающие популярность и активно применяющиеся в ра-

боте с большими объёмами данных алгоритмы на основе нейронных сетей. Их отличительной особенностью является возможность их эффективного распараллеливания, при этом они достигают необходимой точности при решении всевозможных видов задач, начиная от задач кластеризации, и заканчивая сложными задачами распознавания образов.

2.7. Подходы, использующие нейронные сети

В наши дни человечество генерирует огромные массивы данных, и вместе с ростом объемов информации, приобретает популярность и актуальность такая область науки как наука о больших данных (Big Data) и машинное обучение (Machine Learning). «Переоткрытые» заново нейронные сети стремительно развиваются с начала 2010-ых годов, с каждым годом решая всё более невообразимые по сложности задачи. Учитывая современные вычислительные потребности задач физики высоких энергий, широко развиваются применения нейронных сетей для задачи трекинга частиц в детекторах.

Детектор BM@N мегакомплекса NICA не является исключением. Так, например в статье[15] описывается оригинальный подход к решению задачи трекинга, использующий реккурентные нейронные сети. Из-за особенностей микростripового детектора, при регистрации события в детекторе помимо истинных измерений(«хитов»), появляется огромное количество ложных («фейковых») срабатываний детектора. Количество фейковых хитов превосходит количество истинных хитов на несколько порядков, что значительно усложняет задачу трекинга. Предыдущие и зарекомендовавшие себя алгоритмы используют 2-ух шаговый подход:

1. Выполняется сбор так называемых «сидов» (трек-кандидатов) среди первых нескольких слоёв детектора с помощью алгоритма K-d дерева

2. На основе сидов предсказывается дальнейшая траектория пролёта частиц (с помощью нейронных сетей или фильтра Калмана)

Однако производительность такого подхода оставляет желать лучшего, так как основное время алгоритм занимается построением вышеупомянутого K-d дерева. Для преодоления такой проблемы было принято решение отказаться от первого шага данного алгоритма и использовать одну рекуррентную нейронную сеть (RNN). Её структура приводится на рисунке 3.

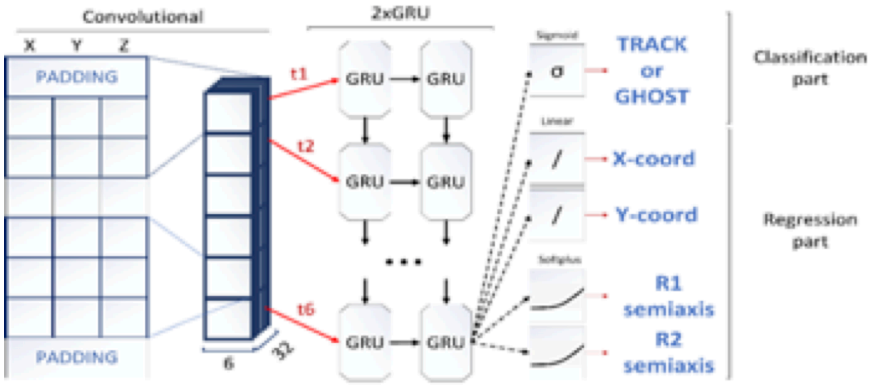

Рис. 3: Основная схема RNN.

На вход нейронной сети подаются пространственные координаты хитов (X, Y, Z) на соответствующих слоях детектора начиная со слоя 1 до слоя N. Сеть предсказывает область на $N + 1$ слое детектора в виде эллипса (x и y координаты центра вместе с полуосями R_1 и R_2), и значение σ полученное через сигмоидную функцию $\sigma(x) = \frac{1}{1+e^x}$ принимающее «1» в случае, если данная точка принадлежит треку, и «0» в противном случае. Такой подход продемонстрировал многообещающие результаты (Табл. 1). при достижении скорости порядка $3 * 10^6$ трек-

	3 points	**4 points**	**5 points**
Recall	98.2%	99.0%	98.3%
Precision	49.0%	57.0%	70.0%
Площадь эллипса	1.67 см^2	1.64 см^2	1.91 см^2

Таблица 1: Результаты работы сети.

кандидатов/секунду, что позволяет обрабатывать большее количество событий детектора в единицу времени по сравнению с подходами, основанными на фильтрах Калмана и K-d деревьях.

Однако стоит отметить, что данный подход практически не учитывает общую совокупность всех точек в одном событии, а рассматривает только конкретную кривую трек-кандидата, что и является одной из причин не идеальной точности (precision) даже для треков, состоящих из 5-ти точек.

В свою очередь, исследователи, работающие с детекторами на Большом Адронном Коллайдере(LHC), тоже активно разрабатывают и применяют различные подходы в области больших данных и глубоких нейронных сетей. В работе [5] предлагается оригинальный подход, основанный на графовых нейронных сетях(GNN). Впервые графовые нейронные сети были представлены в работе [8], и данный подход был взят за основу при решении задачи трекинга.

Событие представляется в виде полносвязного графа (Рис. 4).

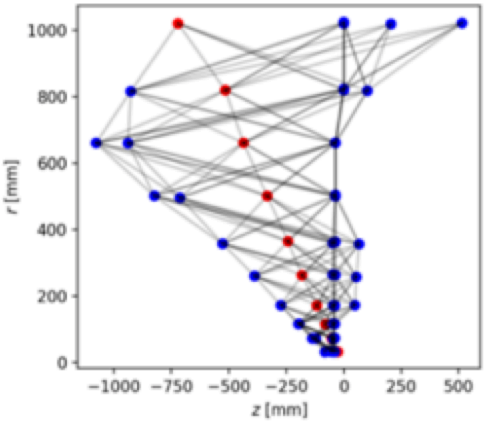

Рис. 4: Событие представлено в виде графа, где регистрации частиц – вершины графа, а ребра соединяют точки от первого до последнего слоя детектора.

Результаты такого подхода многообещающие, демонстрируется высокая точность вкупе с эффективностью. Однако заявленный подход хорошо работает только для частиц имеющих высокую энергию (> 1 GeV), к тому же такой подход работает только для детекторов с низким уровнем шума (”пиксельные” детекторы), что является большим препятствием для решения поставленных в настоящей работе задач.

2.8. Вывод

Таким образом, применение машинного обучения в задачах трекинга является важнейшей составляющей современных экспериментов в области физики высоких энергий. Глубокие нейронные сети помогают учёным и исследователям обрабатывать огромные массивы данных, генерируемые современными детекторами, что в результате ложится в основу современных и будущих открытий. Именно поэтому, исследования и результаты, достигнутые в ходе настоящей работы, являются актуальными и необходимыми в области физики, в частности для успешного проведения эксперимента BM@N мегапроекта NICA.

3. Основная часть

В этой главе будет описаны используемые технологии, процесс подготовки и фильтрации данных, решение задачи на основе обучения двух различных моделей глубоких нейронных сетей и результаты обучения.

3.1. Используемые технологии

В настоящее время успешное решение задач обработки данных и машинного обучения почти невозможно представить без языка Python[16]. Язык Python – один из наиболее гибких языков программирования в настоящее время, поддерживающий множество парадигм программирования. Ещё одной его отличительной особенностью является огромное количество библиотек, способных помогать исследователям решать свои задачи не только быстро, но ещё и сохраняя такие важные параметры, как эффективность и производительность. Для обработки данных в настоящей работе была использована библиотека Pandas[12].

3.1.1. Pandas

Pandas – программная библиотека на языке Python для обработки и анализа данных. Работа Pandas с данными строится поверх библиотеки NumPy, являющейся инструментом более низкого уровня. Используя данную библиотеку, был выполнен один из важнейших этапов настоящей работы – подготовка и препроцессинг данных. Именно благодаря удобным методам для работы с данными, уже реализованным в этой библиотеке, таким как:

- `DataFrame.groupby` – позволяет разбивать данные по указанным столбцам;
- `DataFrame.filter` – позволяет очищать данные по строкам, в зависимости от заданного условия;

- `DataFrame.to_csv/read_csv` – позволяет сохранять/загружать данные в текстовом формате,

и другим, препроцессинг данных был очень быстро реализован, оставаясь при этом эффективным и производительным.

3.1.2. PyTorch и TensorFlow

В современном мире реализация алгоритмов для глубоких нейронных сетей трудно осуществима без использования специальных библиотек, таких как TensorFlow или PyTorch. Они обеспечивают эффективную работу при реализации структуры и процессов обучения сетей и позволяют автоматически осуществлять распараллеливание получившихся программ. В настоящей работе использовались две наиболее распространенных в настоящее время библиотеки для работы с нейросетями и машинным обучением: PyTorch и Tensorflow. Обе библиотеки имеют свои преимущества и недостатки:

PyTorch[2] – библиотека машинного обучения для языка Python с открытым исходным кодом. Разработана и поддерживается компанией Facebook. Относительно Tensorflow, библиотека более молодая, однако является достойным конкурентом, как в плане производительности, так и в плане удобства использования.

Tensorflow[14] – библиотека машинного обучения для языка Python с открытым исходным кодом. Разработана и поддерживается компанией Google. Одним из главных недостатков этой библиотеки является практическая неудобность её использования – необходимо либо писать много низкоуровневого кода, либо использовать специальные обёртки, например Keras. Но ввод ещё одного уровня абстракции только усложняет общую логику программы, что сказывается как на общей гибкости, так и скорости работы программы. Также в Tensorflow неестественный процесс отладки, что тоже сказывается на удобстве работы с библиотекой.

3.2. Описание эксперимента BM@N мегапроекта NICA

NICA (Рис. 5) (Nuclotron based Ion Collider fAcility) – это новый ускорительный комплекс, который создаётся на базе Объединённого института ядерных исследований (Дубна, Россия) с целью изучения свойств плотной барионной материи [NICA]. После запуска коллайдера учёные рассчитывают воссоздать в лабораторных условиях особое состояние вещества – кварк-глюонную плазму.

Рис. 5: Мегапроект NICA.

BM@N (Baryonic Matter at Nuclotron, Рис. 6) – эксперимент проекта NICA на фиксированной мишени с выведенным из Нуклотрона пучком частиц для изучения свойств барионной материи образуемой в результате столкновения тяжелых ионов при энергиях пучка от 2 до 6 A·GEV.

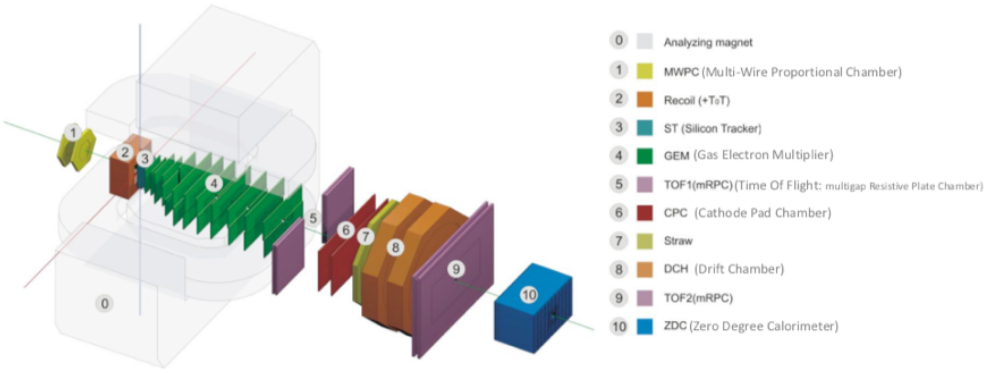

Рис. 6: Полная конфигурация детекторных систем эксперимента BM@N.

Одна из составных частей установки – микростриповый GEM (Gas Electron Multiplier) детектор, основной трековый детектор в эксперименте BM@N. Полная конфигурация детектора состоит из 12 независимых станций, расположенных вдоль оси пучка (ось Z) на определенном расстоянии от мишени. В настоящей работе рассматривается сокращённая конфигурация GEM детектора, состоящая из 6 станций.

3.2.1. Схема работы микрострипового детектора

Схему регистрации событий GEM детектора можно описать следующим образом (Рис. 7).

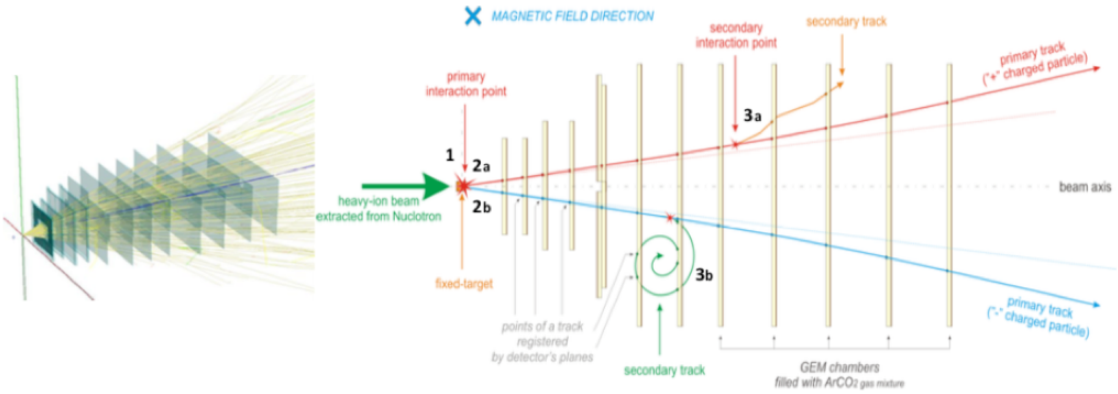

Рис. 7: Слева – визуализация события Au+Au. Справа – Схема регистрации треков (вид сверху).

1. Выведенный из Нуклотрона пучок тяжелых ионов сталкивается с фиксированной мишенью.

2. В результате этого взаимодействия образуются различные частицы, треки которых регистрируются плоскостями детектора.Под действием магнитного поля траектории частиц искривляются. По кривизне траектории можно определить импульс частицы. Более того, частицы с положительным зарядом закручиваются в одну сторону (2a), а с отрицательным – в другую (2b) (под действием силы Лоренца).

3. В результате вторичных взаимодействий могут рождаться новые (вторичные) частицы с различной энергией, треки которых также фиксируются детектором. Заряд и значение импульса этих частиц определяют траекторию их движения (3a, 3b).

3.2.2. Особенности реконструкции координат точек трека

Считывающая плоскость GEM детектора состоит из двух микростриповых слоёв (Рис. 8):

- Слой вертикальных слоев;
- Слой наклонных стрипов.

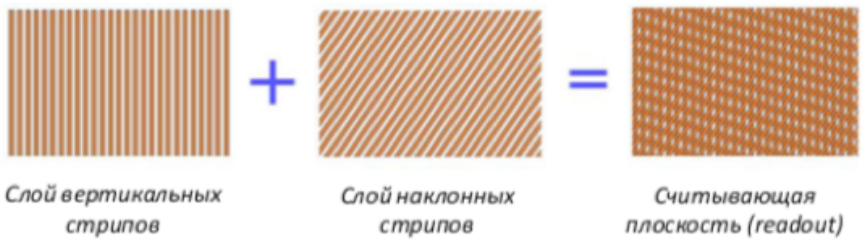

Рис. 8: Считывающая плоскость.

Точкой пересечения стрипа одного слоя со стрипом другого слоя является точкой регистрации частицы. Таким образом, существенный недостаток восстановления координат со стриповых плоскостей – появление ложных пересечений («фейков») при регистрации двух и более треков

одной независимой плоскостью (Рис. 9). Сокращение числа «фейков» можно добиться путём размещения стрипов в плоскостях под малым углом друг к другу. В BM@N GEM детекторе используется угол = 15°.

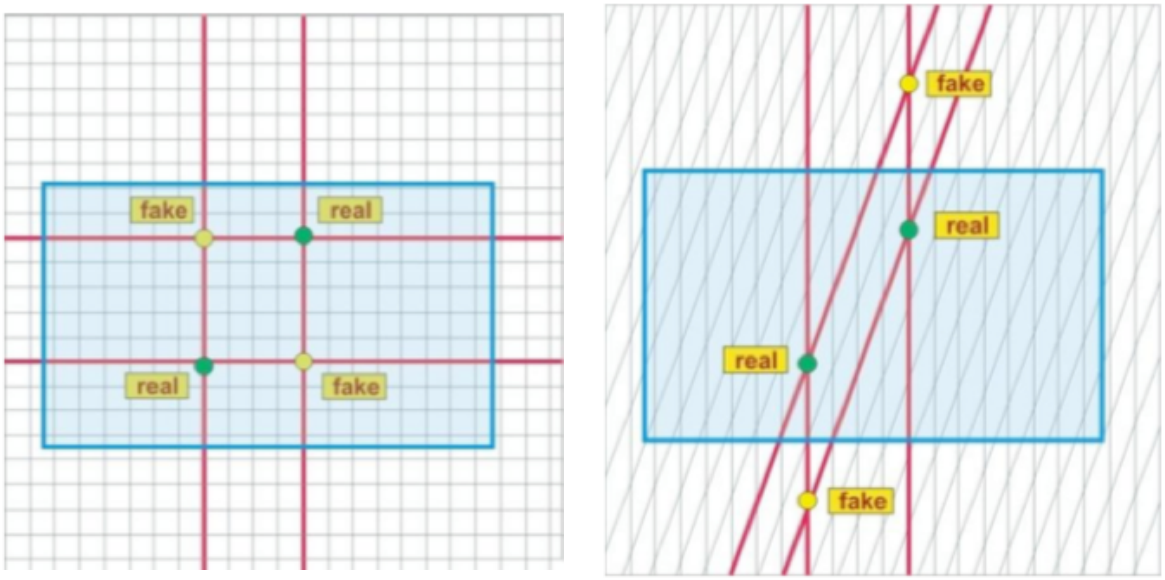

Рис. 9: Сокращение числа фейков.

3.3. Препроцессинг данных

В рамках настоящей работы рассматривались данные для событий C+C.

1. Методом Монте-Карло были симулированы 550 тыс. событий энергией 4 GeV используя генератор LAQGSM.

2. Сильно закручивающиеся – треки, чье присутствие было более чем 1 раз на конкретной станции – были убраны из сгенерированных данных, так как такие треки ученых не интересуют.

3. Короткие – треки, которые отметились менее чем на 3-ёх станциях – были убраны по той же причине.

Для подготовки данных необходимо решить задачу сопоставления сгенерированного физического представления трека и регистраций детектора (Рис. 10).

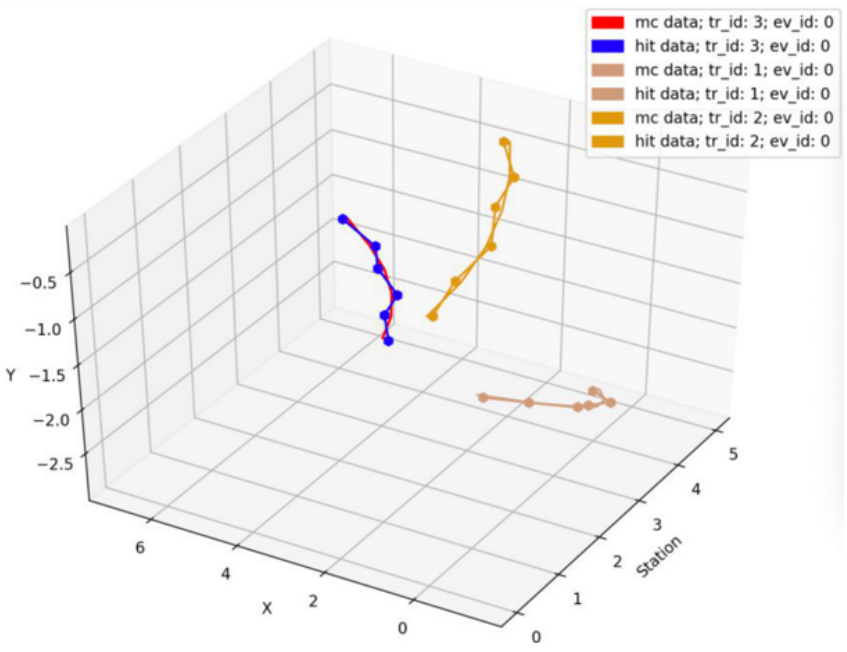

Рис. 10: Сплошные кривые («mc data») – физическая траектория частицы, Монте-Карло точка; ломаная линия («hit data») – траектория, образованная регистрацией этой же частицы в детекторе.

При обработке данных в наборе данных было найдено несколько проблем:

1. Для некоторых случаев две и более частицы могут пролететь через одно «пересечение» стрипов (Рис. 11). Таких событий ничтожно мало, но их необходимо исключить из набора данных.

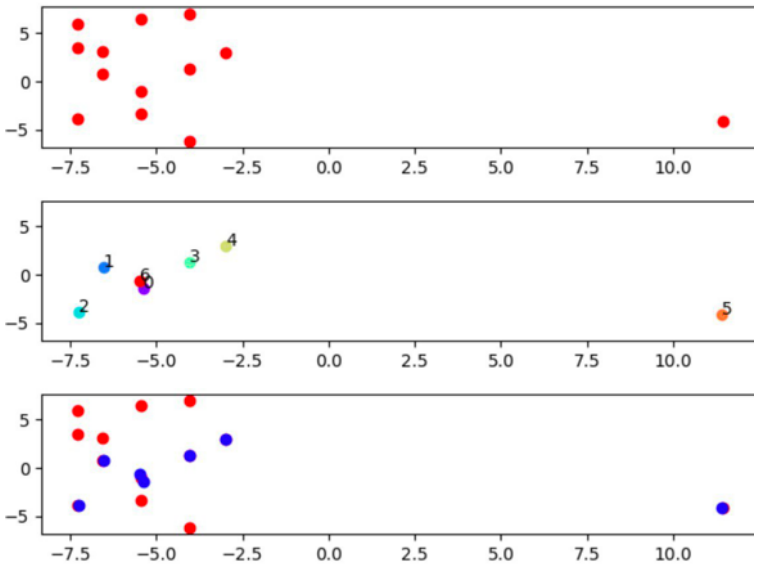

Рис. 11: Первая ось – хиты детектора, вторая ось – Монте-Карло точки, третья – сопоставление. Треки №6 и №0 проходят через один хит детектора.

2. Для некоторых случаев регистрация хита находится слишком «далеко» от реального места полета частицы (Рис. 12). Таких событий также ничтожно мало, и они исключаются из набора данных.

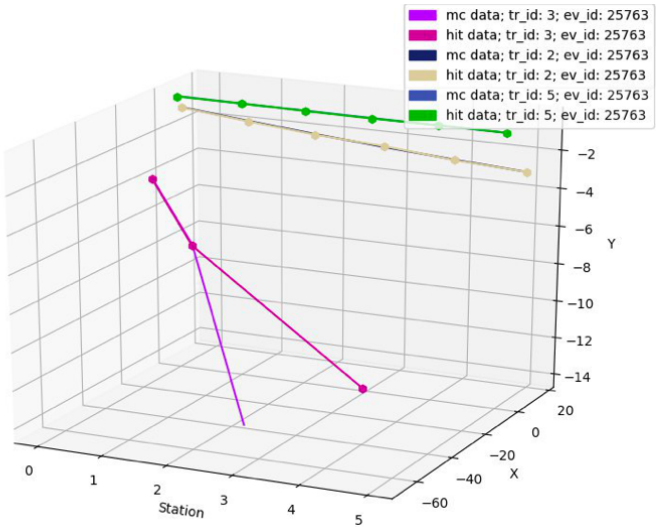

Рис. 12: Расстояние от Монте-Карло точки до хита слишком велико.

Исключив из данных такие случаи, был реализован алгоритм матчинга Монте-Карло точек и хитов детектора на основе двухмерного KD-

дерева. Также была реализована программа, позволяющая визуализировать как события целиком, так и произвольное количество треков для любого события из обработанных данных, пример работы можно увидеть на рисунке 13.

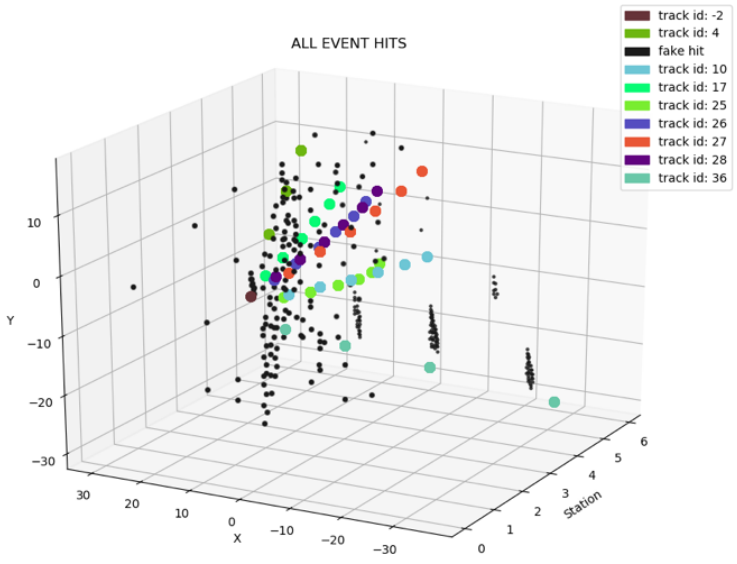

Рис. 13: Визуализация события.

3.4. Модификация рекуррентной сети для трекинга

В рамках настоящей работы были внесены улучшения в существующий подход, описанный в статье[15]. Также этот подход был опробован на данных, полученных из описанной выше симуляции, прошедшей весь этап препроцессинга и фильтрации.

Рассмотрим ключевые составляющие подхода. Как уже было описано ранее (Рис. 3), модель сети предсказывает координаты и двухмерную область, в которую должны попасть частицы на следующем слое детектора. Необходимо отметить, что в предложенном подходе используется двухмерная система координат, а третьей «пространственной» координатой по сути является размерность входных данных.

Одной из главных частей процесса обучения нейронной сети является функция ошибки. В ходе реализации описанного подхода, данные, полученные после препроцессинга содержали сопоставимое описанному в работе отношению истина/шум: на 1 «истинный» трек, прихо-

дится около 120 ложных. Проблема несбалансированности количества экземпляров классов решается использованием специальной функции, называемой «Focal Loss»[6]. Эта функция позволяет сконцентрировать внимание нейросети на «сложных» примерах, штрафуя нейросеть за переобучение на более представленных количественно «лёгких» примерах. Итоговый вид функции ошибки:

$$J = max(\lambda_1, 1-p)FL(p,p') + p\left(\lambda_2\sqrt{\left(\frac{x-x'}{R1}\right)^2 + \left(\frac{y-y'}{R2}\right)^2} + \lambda_3 R1R2\right), \tag{14}$$

где λ_{1-3} – веса для каждой части выражения; p – значение, показывающее является ли множество точек, переданных на вход нейросети истинной траекторий или ложной; p' – вероятность того, что множество истинно/ложно, предсказанное сетью; x', y' – координаты центра эллипса, предсказанные сетью; x, y – координаты следующей точки из траектории истинного трека; $R1$, $R2$ – полуоси эллипса; $FL(p,p')$ – вышеописанная функция Focal Loss.

В ходе обработки ложных треков не нужно учитывать ошибку предсказания размеров эллипса и координат его центра, поэтому параметр p расположен только перед второй частью выражения. Следовательно, когда его значение равно 0 (т. е. трек фейковый), регрессионная часть сократится. Напротив, $max(\lambda_1, 1-p)$ используется чтобы сфокусировать сеть на ошибке классификации, когда регрессионная часть отсутствует.

В рамках настоящей работы была реализована и протестирована следующая гипотеза: во входной слой нейросети необходимо добавить четвертый параметр – Z-координату точки траектории на следующем слое детектора. Гипотеза была успешно реализована и протестирована (Табл. 2). Обучение выполнялось в интерактивной среде Google Collaboratory и доступно по ссылке[17].

Подход, реализованный в рамках настоящей работы, продемонстрировал улучшение как точности работы нейросети, так и уменьшение площади предсказанного эллипса, хотя и потерял менее 1% Recall.

	Старый подход	**Новый подход**
Recall	98.4%	97.8%
Precision	54%	56%
Ellipse square	1.74 см^2	1.56 см^2

Таблица 2: Сравнение результатов обучения сети.

3.5. Графовая нейронная сеть (GNN)

Идея представления массива данных в виде графа для последующей обработки специальной нейронной сетью была представлена в статье [8] в 2016 году. Эта идея была адаптирована для решения задачи трекинга проектом HEPTrkX[5]. Рассмотрим вышеописанное представление события в детекторе как графа подробнее.

Представим событие как граф, в котором узлы – срабатывания детектора. Ребрами соединим все вершины между слоями. На одном слое вершины не соединяются. GNN состоит из трех основных частей: Input Network, Node Network и Edge Network. Событие представляется в виде 4-ёх матриц:

- X – матрица «фичей» (features) вершин графа размера $N \times M$, где N – количество вершин, а M количество фичей. Координаты хита используются как фичи, поэтому $M = 3$;
- Ri – матрица ребер, которые заканчиваются в соответствующих вершинах размером $N \times E$ (E – количество вершин). В этой матрице Ri$[i, j]$=1 если вершина с индексом j входит в вершину с индексом i, и равна 0 в противном случае;
- Ro – матрица рёбер, которые исходят из вершин. Обладает теми же свойствами, что и Ri, но описывает исходящие вершины;
- Y – класс ребра размером $(1 \times E)$. Y$[j]$=1 если ребро с индексом j принадлежит реальному треку, и 0 в противном случае.

«Input network» – Многослойный персептрон (MLP) с тангенциальной функцией активации (Tanh). На вход Input Network подаётся мат-

рица X. Выход Input Network подается на итерации ‘Edge-Node’ сетей (Рис. 14).

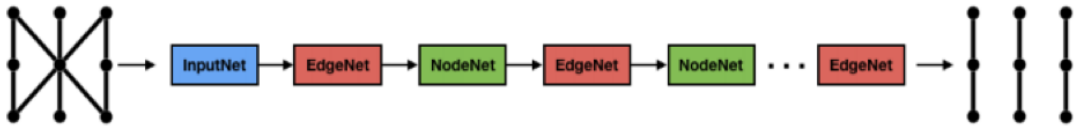

Рис. 14: Графовое представление поступает на вход Input Network и проходит цепочки Edge-Node итераций.

3.5.1. Итерации Edge-Node

Edge network представляет собой двухслойную полносвязную нейросеть. Между слоями сети функция активации та же – Tanh, но на выходном слое используется сигмоида, которая определяет является ли ребро графа сегментом траектории истинного трека или нет. По сути своей Edge network вычисляет веса для ребер графа. Для каждого ребра она выбирает фичи ассоциированных с ней вершин (путем умножения на матрицы Ri и Ro) и подает эту информацию на вход MLP.

Node network – такой же двухслойный MLP с Tanh активациями. Эта сеть пересчитывает фичи вершин графа. Для каждой вершины она собирает фичи соседних вершин (отдельно на стороне входа и выхода) и комбинирует их с предыдущими фичами этой вершины подавая эту инфомарцию на вход MLP.

Эти сети могут быть расположены прямо друг за другом в виде итераций. Количество таких итераций является одним из гиперпараметров модели.

3.5.2. Сравнение использования GNN для GEM детектора и LHC

Основное различие между детекторами большого адронного коллайдера (LHC) и GEM детектором: детекторы, используемые в LHC, являются пиксельными, и поэтому не имеют такого количества шума,

как GEM детекторы. На рисунке 15 можно наблюдать колоссальную разницу в количестве шума.

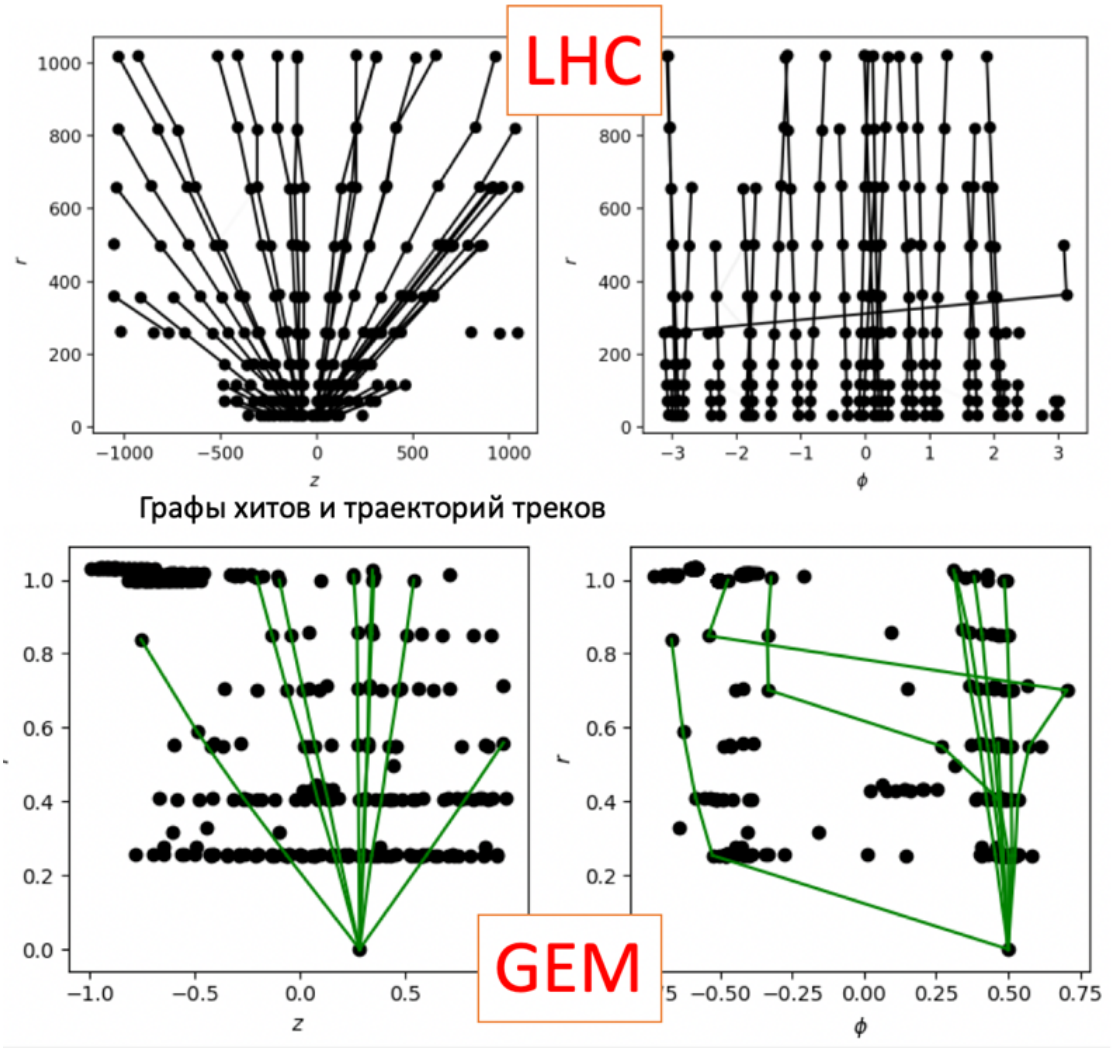

Рис. 15: Сверху – данные с детектора LHC. Снизу – данные с GEM детектора. На нижнем изображении зеленым показаны траектории треков, на верхнем – все траектории являются истинными траекториями частиц.

Применив напрямую GNN подход для данных с GEM детектора, были получены неудовлетворительные результаты. Из-за огромной несбалансированности классов (на 1 истинный сегмент траектории приходится 120 фейковых сегментов) сеть либо переобучается, предсказывая все сегменты как ложные, либо при введении разного веса для истинных и ложных рёбер не может достигнуть даже 20% precision. Полностью

пронаблюдать представление одного события как графа для GEM детектора можно на рисунке 16.

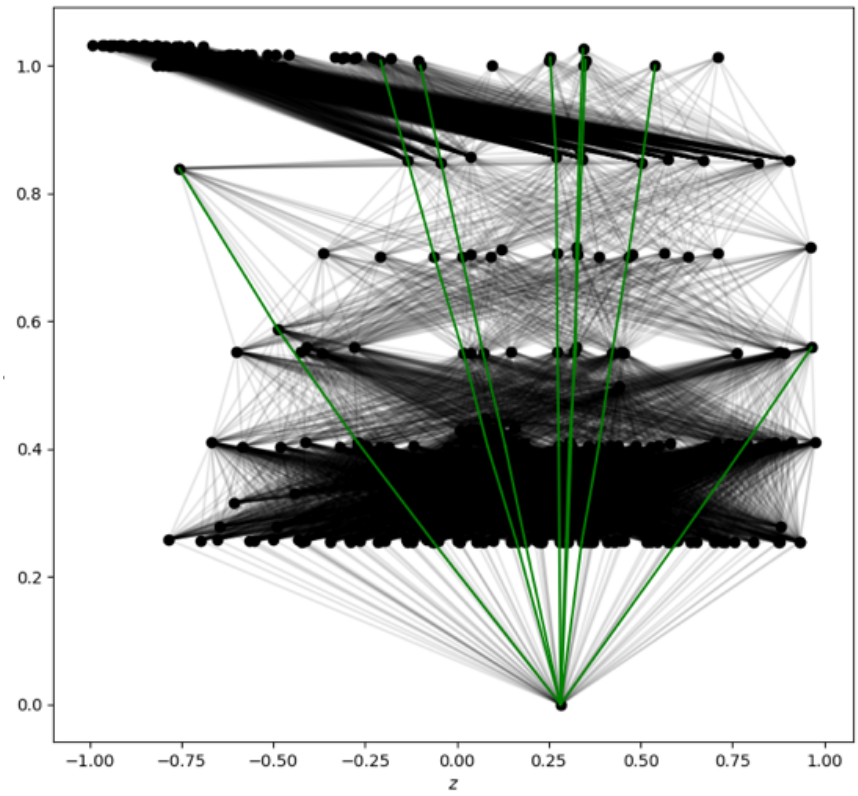

Рис. 16: «Вид» сверху на событие. Черным обозначаются ложные ребра, зеленым – рёбра, являющиеся частью истинных траекторий треков.

Для успешного обучения сети совершенно точно необходимо понизить количественное отношение между классами. Как можно наблюдать, некоторое количество ложных рёбер можно отсеять по углу. Для этих целей на наборе данных необходимо провести статистический анализ. Также на наборе данных выполняется нормализация и перевод в другую систему координат.

3.6. Статистический анализ данных

Первым этапом выполним нормализацию данных и переведем их в цилиндрические координаты следующим образом:

$$\begin{aligned} x_{norm} &= \frac{2 * (x - x_{min})}{x_{max} - x_{min}} - 1, \\ y_{norm} &= \frac{2 * (y - y_{min})}{y_{max} - y_{min}} - 1, \\ z_{norm} &= \frac{z - z_{min}}{z_{max} - z_{min}}. \end{aligned} \tag{15}$$

x_{min},x_{max},y_{min},y_{max},z_{min},z_{max} известны исходя из конфигурации детектора.

Далее переведем координаты в сферические для более удобной работы с углами.

$$\begin{aligned} r &= \sqrt{x_{norm}^2 + z_{norm}^2}, \\ \phi &= arctan(y_{norm}, x_{norm}), \\ z &= y. \end{aligned} \tag{16}$$

Далее возьмем 50 тыс. событий и визуализируем значения $\Delta\phi$ и Δz (Рис. 17).

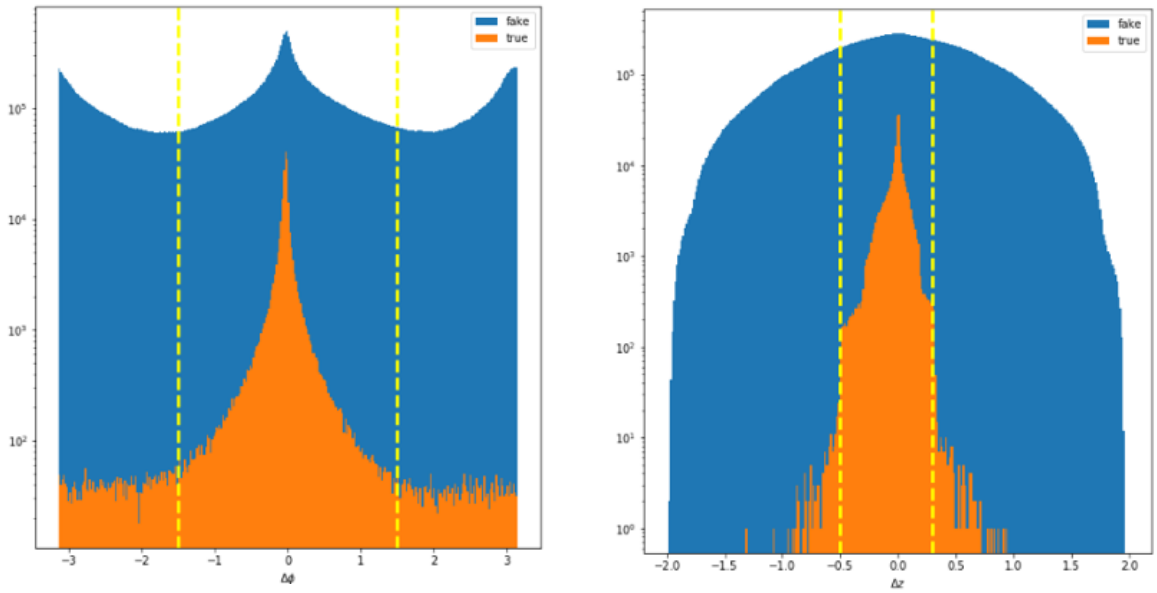

Рис. 17: Статистика для $\Delta\phi$ и Δz. Шкала логарифмическая.

Как можно видеть, можно отсеять большое количество фейковых

рёбер путём введения следующих ограничений:

- От -1.5 до 1.5 для $\Delta\phi$,
- От -0.5 до 0.3 для Δz.

После применения вышеописанного ограничения, соотношение ложных и истинных рёбер в графе уменьшилось в 2 раза с 1/120 до 1/60 (Рис. 18). Однако GNN всё еще не может обучиться на таких данных. Необходимо радикально сократить количество ложных рёбер.

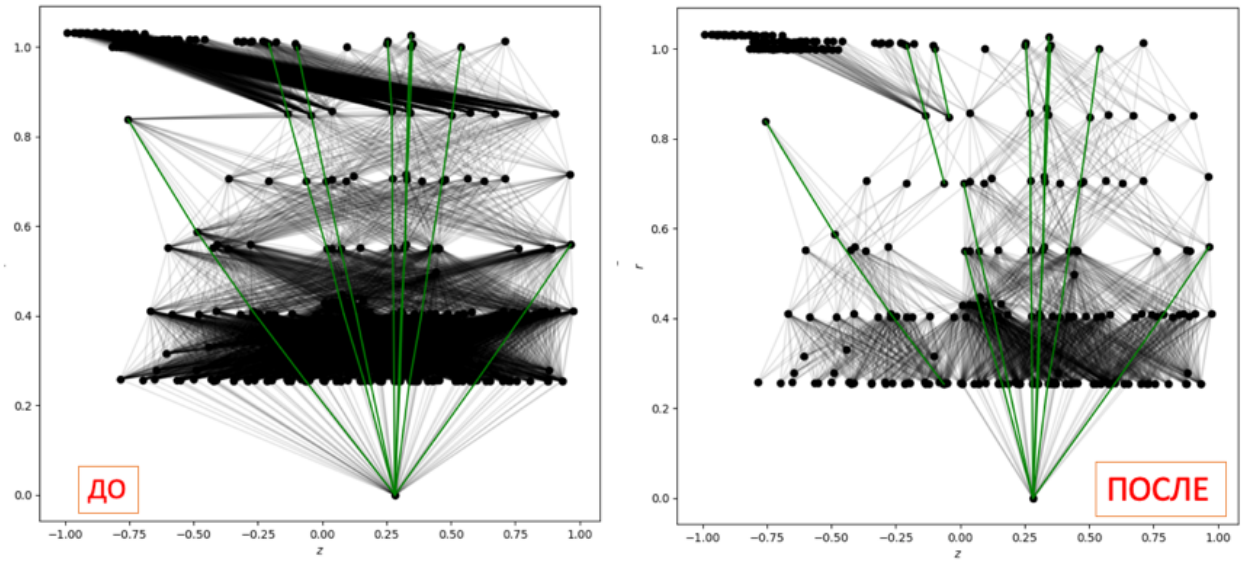

Рис. 18: Число фейковых рёбер снизилось, но недостаточно.

3.7. Минимальное остовное дерево

Остовное дерево – минимальное подмножество рёбер графа, таких, что из любой вершины графа можно попасть в любую другую вершину, двигаясь по этим рёбрам. Если каждому ребру в графе присвоить вес, то нахождением оптимального остовного дерева, которое минимизирует сумму весов входящих в него рёбер, занимаются многочисленные алгоритмы нахождения минимального остовного дерева.

Минимальное остовное дерево (Рис. 19) – это остовное дерево этого графа, имеющее минимальный возможный вес, где под весом дерева понимается сумма весов входящих в него рёбер.

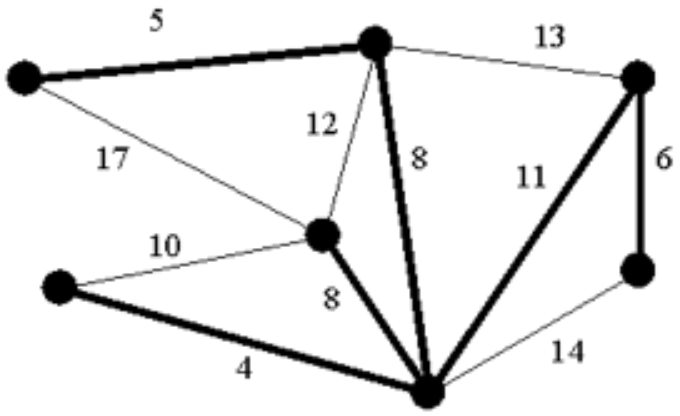

Рис. 19: Пример минимального остовного дерева.

Применим алгоритм минимального остовного дерева для графа события. Для этого необходимо каждому ребру присвоить какой-либо вес. Введем вес как:

$$cost = \frac{cos^m(\phi_i - \phi_j)}{(r_i - r_j)^n}, \tag{17}$$

где:

- i, j – индексы хитов на соседних слоях;
- ϕ, r – координаты в цилиндрической системе координат;
- m, n – произвольные целочисленные константы.

Результаты работы алгоритма представлены на рисунке 20.

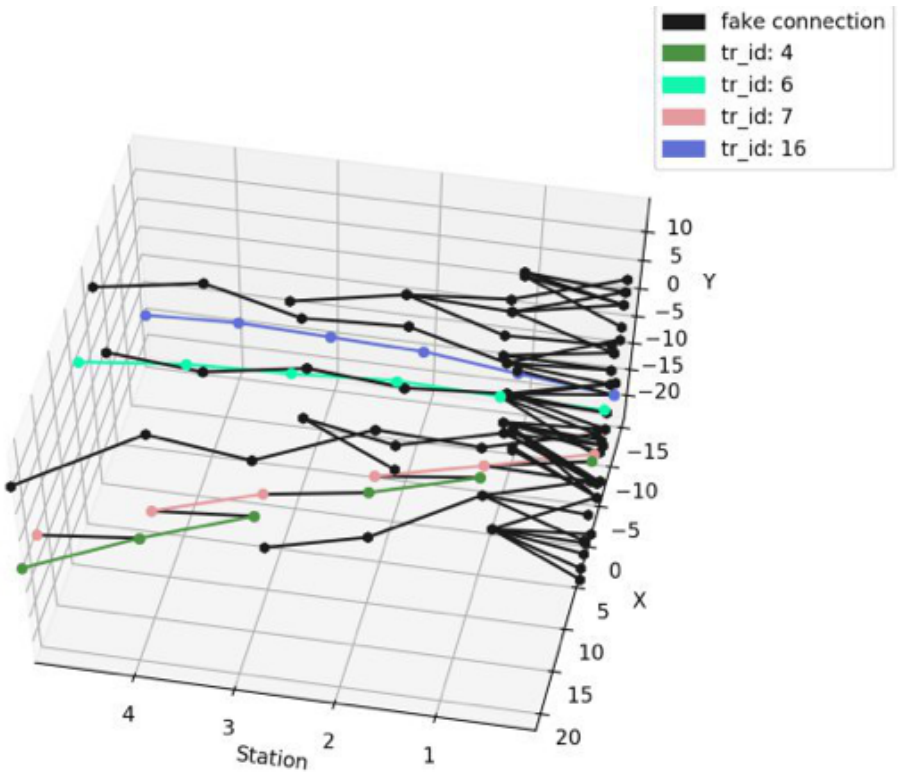

Рис. 20: Минимизированное остовное дерево демонстрирует удовлетворительный результат.

Алгоритм минимизации дерева сразу же дал многообещающие результаты. Однако, как можно видеть на рисунке 20, алгоритм «соединяет назад» некоторые вершины. Это просходит в силу того, что алгоритм старается «выбрать» наиболее прямой участок для пути графа. Такую проблему легко решить, введя направление для рёбер графа. Теперь рёбра в графа направлены со станции n на станцию $n + 1$. Минимизированное остовное дерево, в котором рёбра имеют направление называется Minimum branching tree. Результаты работы алгоритма представлены на рисунке 21.

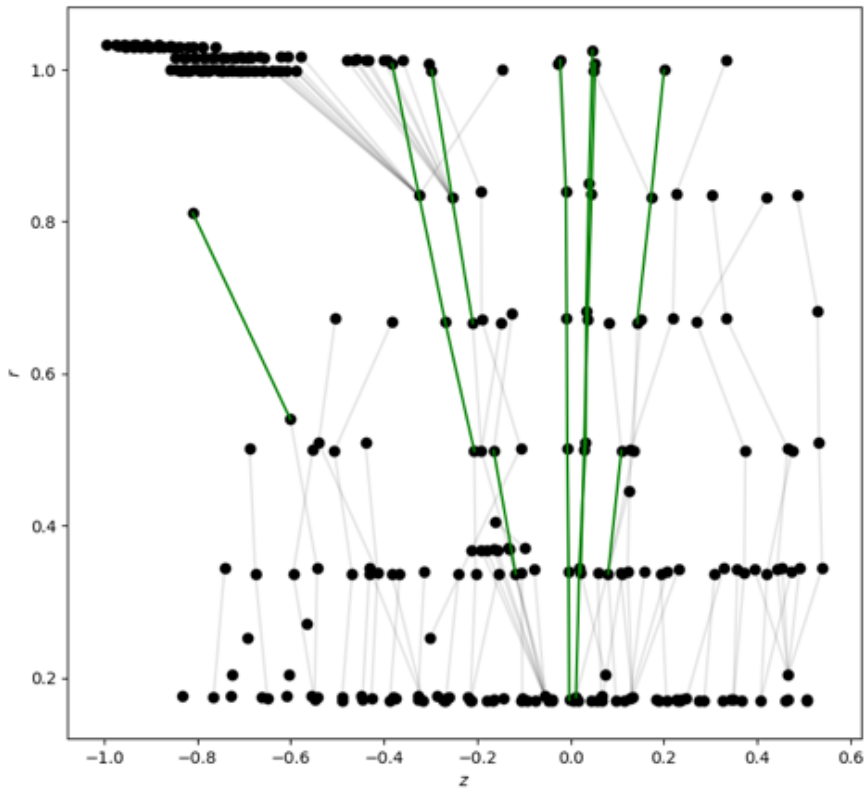

Рис. 21: Результат работы алгоритма MBT.

После вышеописанного препроцессинга и работы алгоритма MBT соотношение ложных рёбер к истинным уменьшилось в 12 раз. «Чистота» (purity) препроцессинга (количество сохраненных истинных рёбер) составляет 82%.

Необходимо отметить, что последние работы, применяющие алгоритмы на основе минимальных остовных деревьев, были опубликованы в 70-ых годах прошлого века. Они решали более тривиальные проблемы на более простых детекторах. Именно поэтому подход, предлагаемый в настоящей работе, отличается своей актуальностью и новизной.

3.8. Результаты обучения

Итоговый алгоритм препроцессинга выглядит следующим образом:

1. Нормализация данных;

2. Перевод точек в цилиндрические координаты;

3. Фильтрация данных по найденному в ходе стат. анализа критерию;

4. Применение алгоритма МВТ.

После выполнения все вышеописанных шагов, данные можно передать на вход нейросети. Сеть была обучена с перевзвешиванием весов для истинных рёбер в том же соотношении, что и итоговое соотношение истинных рёбер к ложным:

$$\begin{aligned} \tilde{w}_{false} &= weight_{false} * 0.1, \\ \tilde{w}_{true} &= weight_{true} * 0.9. \end{aligned} \tag{18}$$

Для обучения использовался оптимизатор `Adam`,`learning rate`=0.001, функция ошибки – `torch.nn.functional.binary_cross_entropy`, `batch_size` = 10 (10 событий), для обучения использовалось 32 тыс. событий, для валидации – 1024. Количество итераций Node-Edge = 3. Количество эпох = 128.

На рисунке 22 можно наблюдать успешное обучение сети.

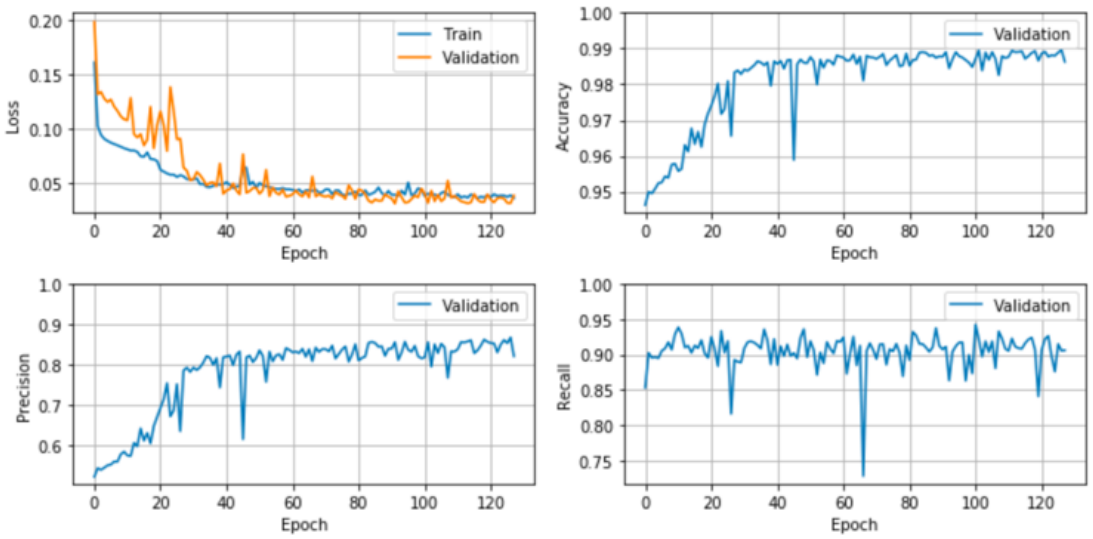

Рис. 22: Метрики Loss, Accuracy, Precision, Recall.

Для тестирования работы сети из набора данных не задействованным в обучении было случайным образом выбрано 1 тыс. событий. На рисунке 23 можно наблюдать итоговые результаты тестирования. Метрика AUC = 0.983.

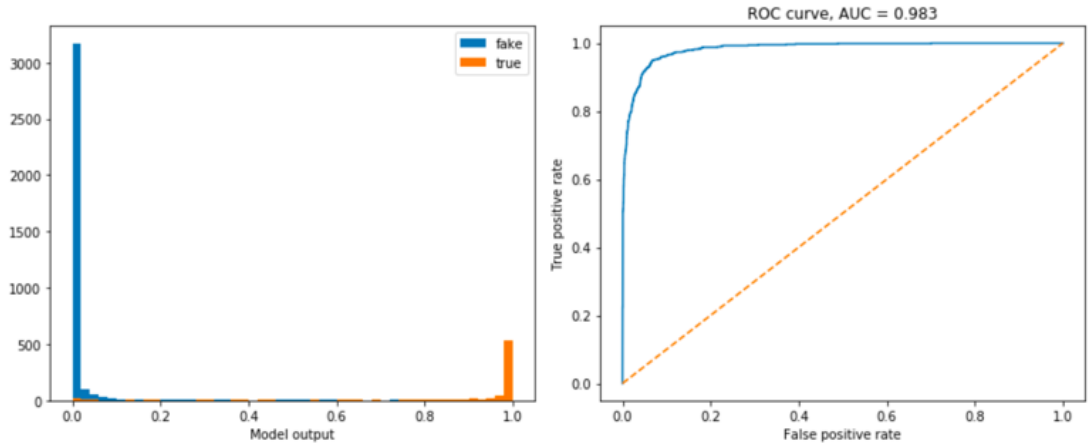

Рис. 23: Результаты тестирования работы сети.

3.8.1. Вывод

Представленный в настоящей работе подход – работающее решение задачи трекинга для GEM-детектора[18]. Он отличается своим быстродействием благодаря использованию векторизованных алгоритмов и нейронных сетей, а также высокой точностью. Использование MBT дало мгновенные удовлетворительные результаты, что показывает перспективность такого подхода.

Настоящая работа была впервые представлена на конференции AYSS-2019, где была отмечена наградой ”Best Report”, статья на её основе принята к публикации в журнале, индексируемом в SCOPUS.

Заключение

В рамках магистерской диссертации были выполнены все поставленные задачи.

1. Выполнен обзор существующих подходов к задаче трекинга частиц.

2. Разработан алгоритм препроцессинга модельных данных сгенерированных для GEM детектора.

3. Разработан и протестирован подход к задаче трекинга с использованием глубоких нейронных сетей.

Список литературы

[1] Alexopoulos T. Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment. — 2008. — Vol. 592. — P. 456–462. — URL: http://dx.doi.org/10.1016/j.nima.2008.04.038.

[2] Automatic differentiation in PyTorch / Adam Paszke, Sam Gross, Soumith Chintala et al. — 2017.

[3] Billoir P. Nuclear Instruments and Methods in Physics Research. — 1984. — Vol. 225. — P. 352–366. — URL: http://dx.doi.org/10.1016/0167-5087(84)90274-6.

[4] Eichinger H., Regler M. Technical Report No. CERN 81-06. — 1981.

[5] Farrell Steven et al. Novel deep learning methods for track reconstruction // 4th International Workshop Connecting The Dots 2018 (CTD2018) Seattle, Washington, USA, March 20-22, 2018. — 2018. — 1810.06111.

[6] Focal Loss for Dense Object Detection / Tsung-Yi Lin, Priya Goyal, Ross B. Girshick et al. // CoRR. — 2017. — Vol. abs/1708.02002. — 1708.02002.

[7] Frühwirth R. Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment. — 1987. — Vol. 262. — P. 444–450. — URL: http://dx.doi.org/10.1016/0168-9002(87)90887-4.

[8] Geometric deep learning: going beyond Euclidean data / Michael M. Bronstein, Joan Bruna, Yann LeCun et al. // CoRR. — 2016. — Vol. abs/1611.08097. — 1611.08097.

[9] Hansroul M. H. Jeremie, Savard D. Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors

and Associated Equipment. — Vol. 270. — P. 498–501. — URL: http://dx.doi.org/10.1016/0168-9002(88)90722-X.

[10] Hough. Proceedings of the International Confer- ence on High Energy Accelerators and Instrumentation. — 1959.

[11] Laurikainen P. W. G. Moorhead, Matt W. Nuclear Instruments and Methods. — 1972. — Vol. 98. — P. 349–359. — URL: http://dx.doi.org/10.1016/0029-554X(72)90116-4.

[12] McKinney Wes. Data Structures for Statistical Computing in Python // Proceedings of the 9th Python in Science Conference / Ed. by Stéfan van der Walt, Jarrod Millman. — 2010. — P. 51 – 56.

[13] Rousseeuw P. J., Leroy A. M. Robust Regression and Outlier Detection. — 1987.

[14] Abadi Martín, Agarwal Ashish, Barham Paul et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. — 2015. — Software available from tensorflow.org. URL: http://tensorflow.org/.

[15] Baranov Dmitriy, Mitsyn Sergey, Goncharov Pavel, Ososkov Gennady. The particle track reconstruction based on deep learning neural networks. — 2018. — 1812.03859.

[16] van Rossum G. Python tutorial. — 1995.

[17] Щавелев Егор. Проверка гипотезы для RNN сети // Google Collaboratory. — 2018. — URL: https://colab.research.google.com/drive/1vCWK7qLOjgScNUN477jeBlqwnvVaZWVR (online; accessed: 15.11.2018).

[18] Щавелев Егор. Имплементация подхода с использованием MBT и графовой нейросети // Github. — 2019. — URL: https://github.com/gooldan/gem-gnn-network.