МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ЯДЕРНЫЙ УНИВЕРСИТЕТ «МИФИ»
(НИЯУ МИФИ)

ИНСТИТУТ ЯДЕРНОЙ ФИЗИКИ И ТЕХНОЛОГИЙ
КАФЕДРА №40 «ФИЗИКА ЭЛЕМЕНТАРНЫХ ЧАСТИЦ»

УДК 004.021, 539.1.05

На правах рукописи

ОСЕТРОВ АЛЕКСАНДР ОЛЕГОВИЧ

МОДЕЛИРОВАНИЕ И РЕКОНСТРУКЦИЯ СОБЫТИЙ В МЮОННОЙ СИСТЕМЕ УСТАНОВКИ SPD

Направления подготовки:

09.04.04 «Программная инженерия»,

14.04.02 «Ядерные физика и технологии»

Диссертация на соискание степени магистра

Научный руководитель,

к.ф.-м.н. _____Е. Ю. Солдатов

Научный консультант,

к.ф.-м.н. _____А. Ю. Верхеев

Москва 2025

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА МАГИСТРА

МОДЕЛИРОВАНИЕ И РЕКОНСТРУКЦИЯ СОБЫТИЙ В МЮОННОЙ СИСТЕМЕ УСТАНОВКИ SPD

Студент _____А. О. Осетров

Научный руководитель,

к.ф.-м.н. _____Е. Ю. Солдатов

Научный консультант,

к.ф.-м.н. _____А. Ю. Верхеев

Рецензент,

к.ф.-м.н. _____И. В. Елецких

Рецензент _____Т. И. Комаров

Секретарь ГЭК,

к.ф.-м.н. _____Е. Ю. Солдатов

Зав. каф. №40,

д.ф.-м.н., проф. _____М. Д. Скорохватов

Рук. учеб. прог.,

к.ф.-м.н. _____Е. Ю. Солдатов

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

На сегодняшний день барионная материя кажется хорошо изученной по сравнению с основными компонентами Вселенной — тёмной материей и тёмной энергией. Несмотря на значительные успехи в описании взаимодействий кварков и глюонов в рамках пертурбативного подхода КХД, вопрос о том, почему нуклоны имеют именно такие свойства, какие мы наблюдаем, остаётся открытым. Понимание структуры и фундаментальных свойств адронов, исходя из динамики составляющих их кварков и глюонов, — одна из главных нерешённых задач КХД.

Нуклон обладает спином ħ/2, что определяет его магнитный момент, а также такие фундаментальные свойства природы, как существование различных фаз вещества при низких температурах и стабильность наблюдаемой Вселенной. Кварковая модель успешно предсказывает большинство основных свойств адронов, таких как заряд, чётность и изоспин. Однако она не позволяет объяснить спиновые свойства адронов в терминах их составных частей.

Научной установкой, предназначенной для изучения спиновой структуры нуклонов и других явлений, связанных со спином, является SPD (Spin Physics Detector) [1]. Её планируется разместить в одной из двух точек столкновения пучков коллайдера NICA, который строится в Международной межправительственной научной организации «Объединённый институт ядерных исследований» (Дубна, Россия).

Для проектирования установки, исследования возможности решения поставленных физических задач, а также разработки методов регистрации и идентификации процессов в эксперименте необходимо моделирование различных детекторных систем. Данная работа посвящена моделированию мюонной системы SPD с использованием программного пакета SpdRoot и последующей разработке программного обеспечения для обработки и анализа данных эксперимента.

Цель и задачи

Основной целью данной работы является разработка программного обеспечения для идентификации мюонов в мюонной системе экспериментальной установки SPD. В соответствии с поставленной целью определены следующие задачи:

1. **Моделирование прототипа системы**

Создание геометрической модели прототипа мюонной системы и моделирование событий для последующего тестирования алгоритмов. Полученные данные будут сравниваться с экспериментальными измерениями уже собранного физического прототипа.

2. **Кластеризация хитов**

Интеграция в программную среду SpdRoot и последующее применение алгоритма кластеризации пространственного распределения хитов с целью формирования треков частиц.

3. **Тестирование алгоритмов машинного обучения**

Сравнительный анализ различных подходов к идентификации частиц, включая ансамблевые методы и современные нейросетевые архитектуры, с акцентом на их эффективность в условиях мюонной системы SPD.

4. **Сравнительный анализ результатов**

Комплексная оценка полученных результатов для определения оптимальной архитектуры алгоритма идентификации частиц.

1 ОПИСАНИЕ ЭКСПЕРИМЕНТА SPD

1.1 Общее описание

В планах коллаборации SPD - установка детектора во второй точке взаимодействия строящегося коллайдера NICA (ОИЯИ, Дубна) для изучения спиновой структуры протона и дейтрона, а также других спиновых явлений. Планируется работа с поляризованными пучками частиц с энергией до 27 ГэВ в системе центра масс и светимостью до 10^{32} $см^{-2}$ $с^{-1}$.

Установка SPD, показанная на рисунке 1, представляет собой универсальный 4π-детектор с расширенными возможностями отслеживания и идентификации частиц [2].

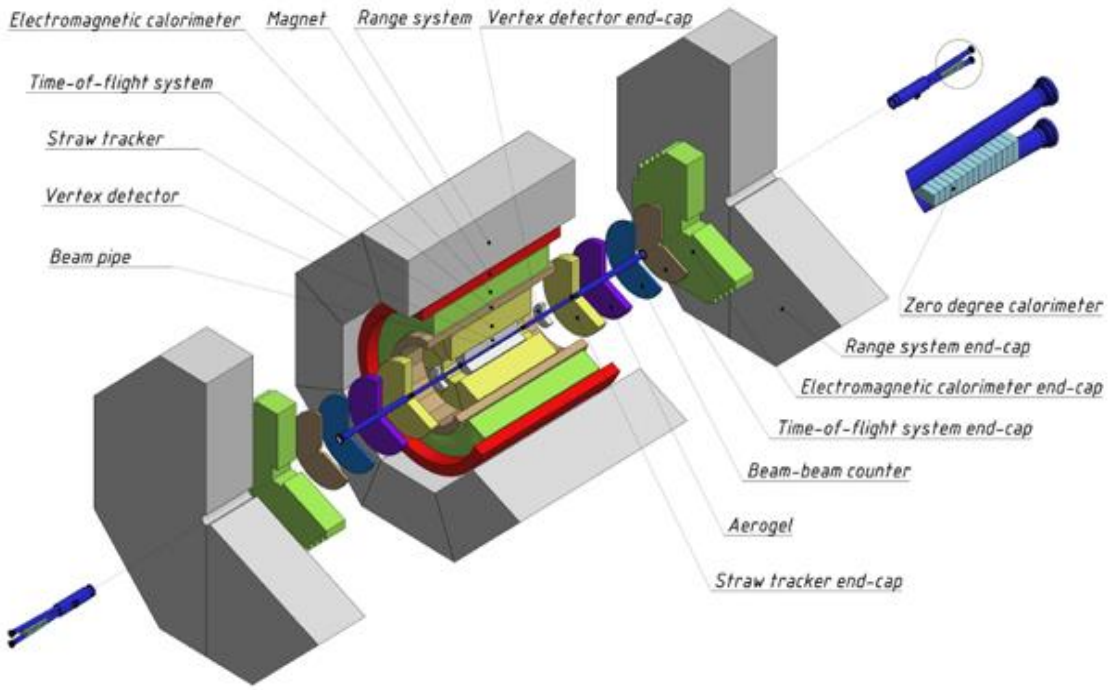

Рисунок 1 – Общая схема установки SPD

Прибор включает следующие детекторные системы:

- магнитную систему,
- вакуумную трубу,
- время-пролётную систему,
- вершинный детектор,
- трековый детектор,
- электромагнитный калориметр,
- счетчик столкновений пучков,
- калориметр нулевого угла,
- мюонную систему.

Сверхпроводящий магнит является одним из ключевых компонентов установки SPD. Вместе с трековой системой он обеспечивает измерение импульсов заряженных частиц с точностью около 2%.

Вакуумная труба отделяет среду вершинного детектора от ускорительного вакуума, что требует от неё высокой механической прочности при минимальной толщине для минимизации многократного рассеяния и радиационных эффектов.

Время-пролётная система предназначена для идентификации заряженных частиц различной массы в диапазоне импульсов до нескольких ГэВ. Её планируется использовать в событиях с множественными треками как для определения времени столкновения, так и для идентификации отдельных треков. Помимо идентификации частиц, система также обеспечивает временную привязку для работы трекового детектора.

Вершинный детектор SPD – кремниевая часть спектрометра, отвечающая за точное определение первичной вершины взаимодействия и реконструкцию вторичных вершин по распадам короткоживущих частиц (в первую очередь D-мезонов).

Трековый детектор предназначен для реконструкции треков первичных и вторичных частиц с высокой эффективностью и измерения их импульсов по кривизне траектории в магнитном поле. Он также участвует в идентификации частиц через измерение удельных энергопотерь (dE/dx).

Электромагнитный калориметр служит для регистрации и идентификации частиц, образующихся при адронных столкновениях в телесном угле 4π. Его задачи включают идентификацию одиночных фотонов и нейтральных пионов, а также отделение фотонных ливней, порожденных π^0-мезонами, для исключения фоновых событий.

Основными целями счетчика столкновения пучков являются локальная поляриметрия, мониторинг столкновений пучков и участие в точном определении времени столкновения в случаях, когда для этого неприменимы другие детекторы (например, при упругом рассеянии).

Калориметр нулевого угла будет установлен в зонах разделения пучков по обе стороны от точки взаимодействия SPD. Его основные задачи – измерение светимости пучка, локальная поляриметрия и генерация временных меток для разделения событий.

Эксперимент SPD заполнит существующий энергетический пробел между другими экспериментами.

1.2 Мюонная система

Мюонная (пробежная) система SPD служит для идентификации мюонов в присутствии значительного адронного фона и оценки энергии адронов (грубая адронная калориметрия). Она также является единственным субдетектором основной части установки SPD, способным идентифицировать нейтроны путем объединения отклика с сигналами от электромагнитного калориметра и внутренними трекерами. Идентификация мюонов осуществляется посредством распознавания фрагментов треков частиц и их последующим сопоставлением с треками внутри магнитной системы. Точное определение импульса мюона выполняется внутренними трекерами в магнитном поле.

Одной из основных задач пробежной системы является идентификация мюонов из распадов $J/\psi \rightarrow \mu^+\mu^-$ в присутствии фоновых пионов и продуктов их распада.

Схематичное изображение системы и её основные размеры (в мм) показаны на рисунке 2а. Схема мюонной системы, состоящей из слоёв поглотителя и детекторов, представлена на рисунке 2б.

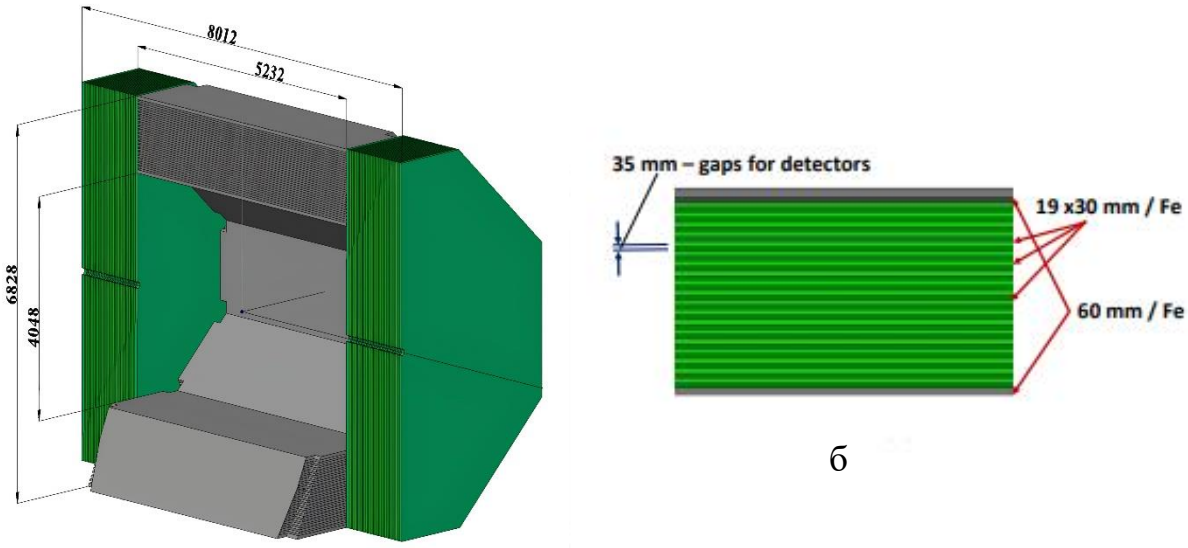

Рисунок 2 – (а) Схематичный вид (половина разреза) мюонной системы Барельная часть показана серым цветом, торцевые диски − зеленым; (б) Размеры слоев системы

Мюонная система состоит из восьмимодульного бареля ("бочки") и двух торцевых дисков. Наружные слои железа толщиной 60 мм служат для скрепления модулей между собой. В межслоевые зазоры помещаются детекторы Mini Drift Tubes (MDT) и считывающая электроника. Толщина основных поглощающих железных пластин (30 мм) выбиралась исходя из сравнения с длиной пробега мюонов в стали для оптимального разделения мюонов и адронов, а также для целей адронной калориметрии.

Общее количество детекторов MDT составляет около 8000 единиц. Они размещаются вдоль оси пучка в бареле и перпендикулярно ей в торцевых дисках. Общая толщина поглотителя выбрана равной четырём длинам ядерного взаимодействия λ_I, что обеспечивает равномерную фильтрацию мюонов во всех направлениях. Вместе с толщиной электромагнитного калориметра (~$0{,}5\lambda_I$) общая толщина установки SPD составляет около $4{,}5\lambda_I$.

Детектор MDT был первоначально разработан и изготовлен в ОИЯИ для мюонной системы эксперимента D0 в FNAL [3]. Позднее на его основе была создана мюонная система для эксперимента COMPASS в CERN [4]. Модификация с двухкоординатным считыванием для MDT с открытой геометрией катода и внешними съёмными электродами была предложена и принята коллаборацией PANDA [5] для мюонной системы их экспериментальной установки. Эта же новая версия MDT предлагается для проекта SPD, так как обладает всеми необходимыми характеристиками: радиационной стойкостью, координатным и временным разрешением, надёжностью, а также высоким уровнем уже проведённых НИОКР в рамках проекта PANDA.

Сечение детектора MDT с открытой геометрией катода показано на рисунке 3.

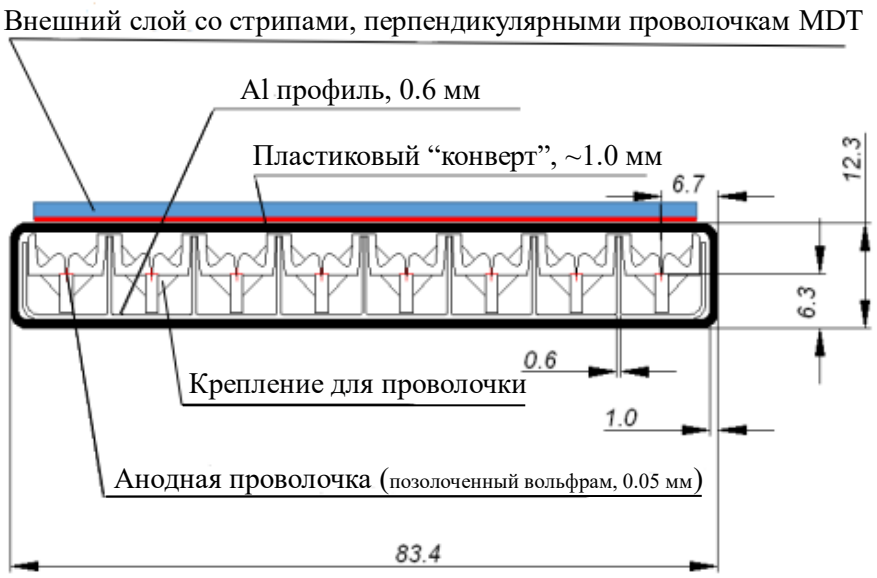

Рисунок 3 – Поперечное сечение детектора MDT

Детектор состоит из металлического катода (алюминиевый гребенчатый профиль из 8 ячеек), анодных проволок и пластиковой оболочки (норил) для обеспечения газонепроницаемости. Гребенчатый профиль катода позволяет получать отдельный сигнал от каждой анодной проволоки для определения одной из координат. Перпендикулярная стриповая плоскость используется для получения второй координаты. Форма наведённого сигнала повторяет исходный, имея противоположную полярность, но его амплитуда составляет около 15% от исходного сигнала, что требует дополнительного усиления и надлежащего экранирования от внешних электромагнитных полей.

Применение открытой геометрии катода позволяет реализовать двухкоординатное считывание, однако приводит к потере симметрии электрического поля в каждой из 8 ячеек детектора. В результате для достижения того же коэффициента газового усиления требуется меньшее приложенное напряжение по сравнению со стандартной геометрией (где катод закрыт крышкой из нержавеющей стали).

2 СОЗДАНИЕ ГЕОМЕТРИЧЕСКОЙ МОДЕЛИ ПРОТОТИПА МЮОННОЙ СИСТЕМЫ SPD И МОДЕЛИРОВАНИЕ СОБЫТИЙ

Первый этап работы включает описание и моделирование в программной среде ROOT [6] прототипа мюонной системы, который в настоящее время используется для тестирования электроники и сбора данных космических лучей. Полученная геометрическая модель прототипа представлена на рисунке 4.

Рисунок 4 – Геометрическая модель прототипа мюонной системы SPD

Сначала был смоделирован MDT-детектор из восьми ячеек, заполненных газовой смесью Ar+CO_2 (рабочее вещество газовых ячеек детектора). Затем шесть MDT-детекторов были объединены в одну детекторную плоскость. В железном корпусе размещено 16 детекторных плоскостей с железными поглотителями, расположенными с шагом 35 мм для размещения электроники. В начале системы расположен так называемый Bilayer, состоящий из двух детекторных плоскостей, каждая из которых содержит четыре MDT-детектора. За ними следует железный лист толщиной 60 мм. Созданная модель прототипа станет частью разрабатываемого

программного обеспечения для обработки и анализа данных планируемого эксперимента SPD.

Далее прототип был интегрирован в среду программного пакета SpdRoot с использованием методов объектно-ориентированного программирования. Для этого потребовалось подключить все необходимые библиотеки из репозитория SpdRoot software и настроить зависимости для переменных и параметров среды.

Следующим шагом было проведено моделирование событий в прототипе. Рабочий процесс можно описать следующей последовательностью:

- Задание имён выходных файлов для данных и параметров
- Создание объекта SpdRunSim, управляющего симуляцией
- Настройка симуляции: указание имени симулятора, используемых материалов, файла вывода данных и конфигурации распада частиц
- Инициализация геометрии
- Создание и настройка генератора SpdIsotropicGenerator для изотропной генерации частиц
- Установка глобальных параметров симуляции,
- Запуск симуляции с заданным числом событий nEvents и контролем времени выполнения
- Сохранение параметров и данных симуляции в выходные файлы для последующей обработки и анализа.

После генерации событий была выполнена их визуализация для наглядного представления происходящих процессов. На рисунках 5 и 6 показаны визуализации прохождения протона и мюона с энергиями 1 и 10 ГэВ через прототип. Условные обозначения:

- Сплошные линии: фиолетовая - протон, голубая - нейтрон, зеленая - электрон, желтая - позитрон, бирюзовая - мюон
- Красная пунктирная линия: гамма-квант
- Красные точки: хиты (события в газовом объеме детектора)

В представленных модельных событиях частицы попадают точно в центр детектора, тогда как в реальном эксперименте частицы будут приходить под различными углами.

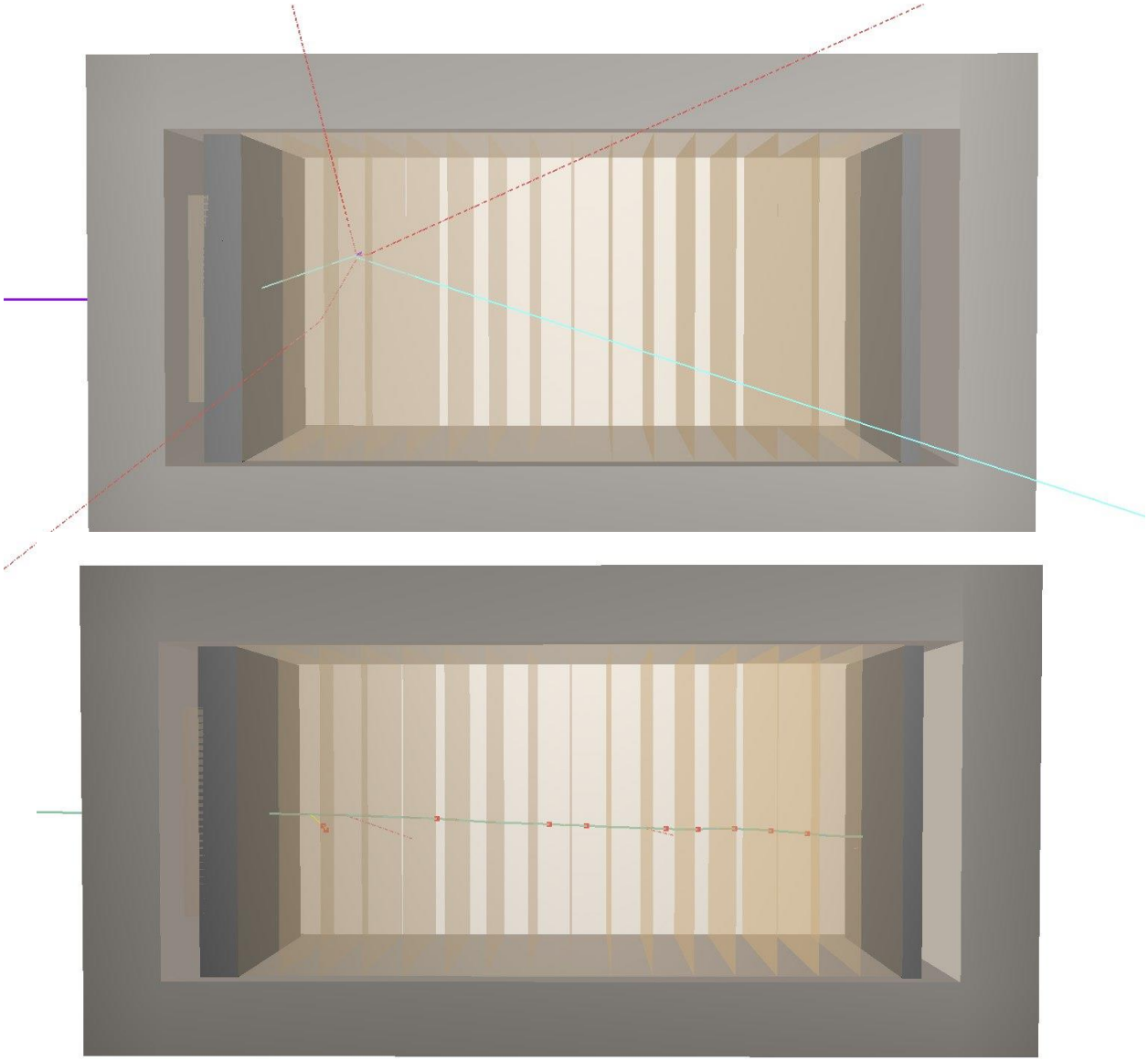

Рисунок 5 – Прохождения протона и мюона с энергией 1 ГэВ через прототип
сверху – протон, снизу – мюон

Анализ рисунков показывает, что в данном примере в результате взаимодействия протона с веществом детектора рождаются нейтроны и фотоны, покидающие пределы прототипа, в то время как при взаимодействии мюона с веществом такого не происходит – образующиеся электрон и гамма-квант поглощаются внутри прототипа.

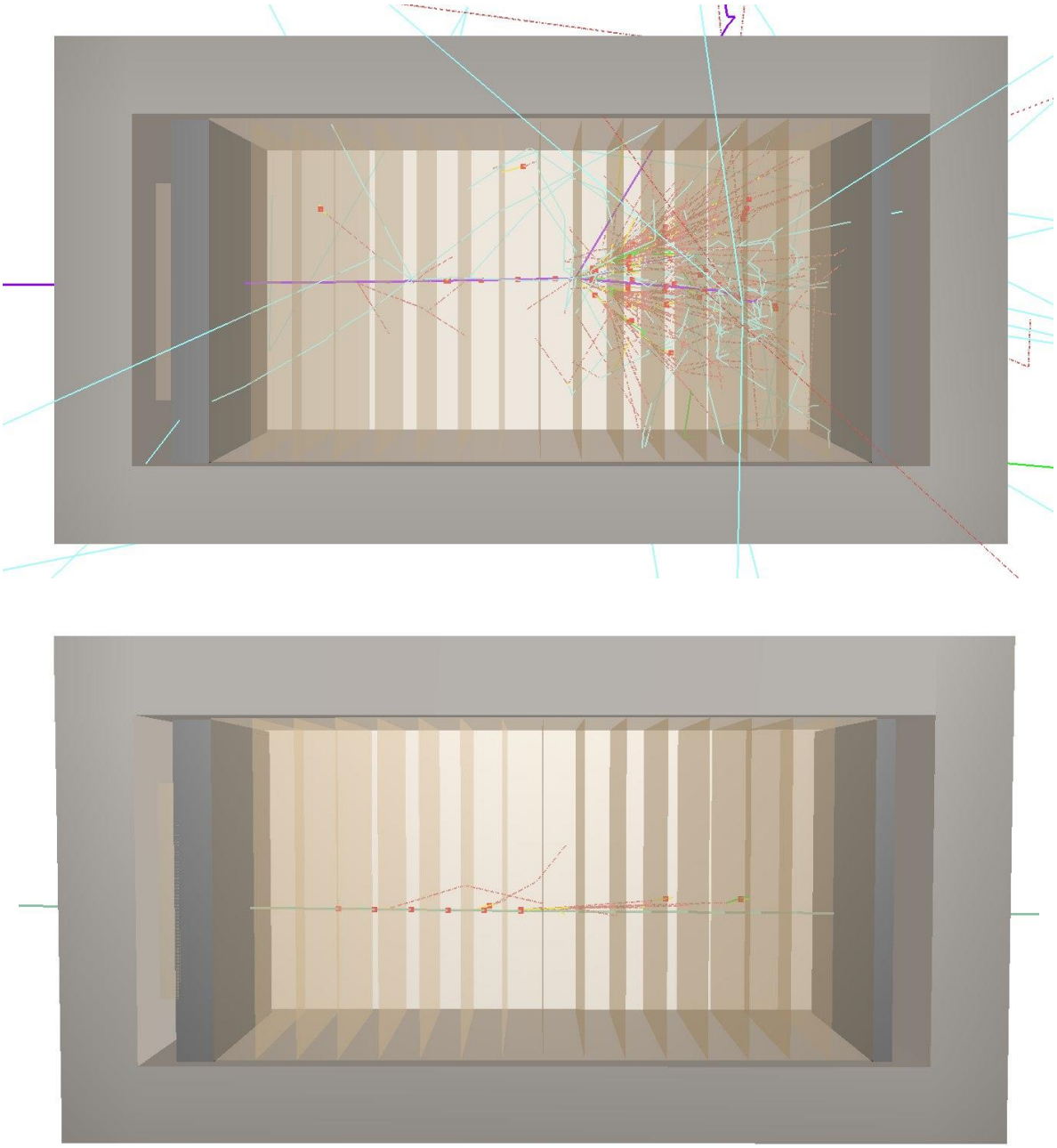

Рисунок 6 – Прохождения протона и мюона с энергией 10 ГэВ через прототип
сверху – протон, снизу – мюон

Протон с энергией 10 ГэВ вызывает каскадный процесс в детекторе, чего не наблюдается ни для протона с энергией 1 ГэВ, ни для мюона с энергией 10 ГэВ. Высокоэнергетический мюон генерирует в детекторе несколько заряженных лептонов и фотонов, которые не покидают пределы прототипа. В дальнейшем полученные результаты будут сравниваться с данными от космических лучей, проходящих через детектор.

3 КЛАСТЕРИЗАЦИЯ ХИТОВ ПРИ ПОМОЩИ АЛГОРИТМА DBSCAN

Для реконструкции событий по хитам в работе использовался алгоритм кластеризации DBSCAN [7]. Этот алгоритм относится к классу методов, выделяющих кластеры на основе плотности распределения точек в пространстве данных. Рассмотрим основные определения:

Пусть имеется база данных D точек в k-мерном пространстве S. Введем ключевые понятия:

- ε-окрестность точки p ($N_\varepsilon(p)$) = $\{q \in D \mid dist(p,q) \leq \varepsilon\}$, где dist(p,q) - расстояние между точками p и q.
- minPts (минимальное количество точек) - пороговое значение числа точек в ε-окрестности, необходимое для признания точки "основной" (Core point).
- Основная точка (Core Point) - точка, имеющая не менее minPts соседей в своей ε-окрестности.
- Пограничная точка (Border Point) - точка, не являющаяся основной, но попадающая в ε-окрестность какой-либо основной точки.
- Шумовая точка (Noise Point) - точка, не относящаяся ни к основным, ни к пограничным.

Алгоритм работает следующим образом. Сначала задаются параметры ε и minPts. В идеале эти параметры должны подбираться индивидуально для каждого кластера, но на практике отсутствует простой способ их априорного определения для всех кластеров. Однако существует эффективная эвристика (раздел 4.2 в [7]) для нахождения параметров ε и minPts наименее плотного кластера. Поэтому DBSCAN использует глобальные значения этих параметров для всех кластеров.

Алгоритм начинает работу с произвольной точки p, проверяя, является ли она основной. Если да, то все точки её ε-окрестности добавляются в текущий кластер. Каждая добавленная точка (кроме p) проверяется на принадлежность к основным. Если точка оказывается основной, её ε-окрестность также добавляется в кластер. Затем включаются все пограничные точки. Процесс продолжается до полного охвата всех точек, достижимых из начальной.

После завершения обработки первого кластера алгоритм переходит к следующей непосещенной точке, повторяя процедуру до полной обработки всех точек. Шумовые точки не включаются ни в один кластер и маркируются отдельно.

Эксперименты из работы [7] показывают, что результаты алгоритма остаются стабильными при $minPts \geq 2*n_{dim}$ (где n_{dim} - размерность пространства). Это позволяет сократить число параметров до одного, установив minPts = 6 для трехмерного случая.

На рисунке 7 представлены результаты работы DBSCAN на трех различных наборах данных в сравнении с алгоритмом CLARANS [8]. Как видно, DBSCAN обеспечивает более точную кластеризацию и, в отличие от CLARANS, корректно идентифицирует шумовые точки.

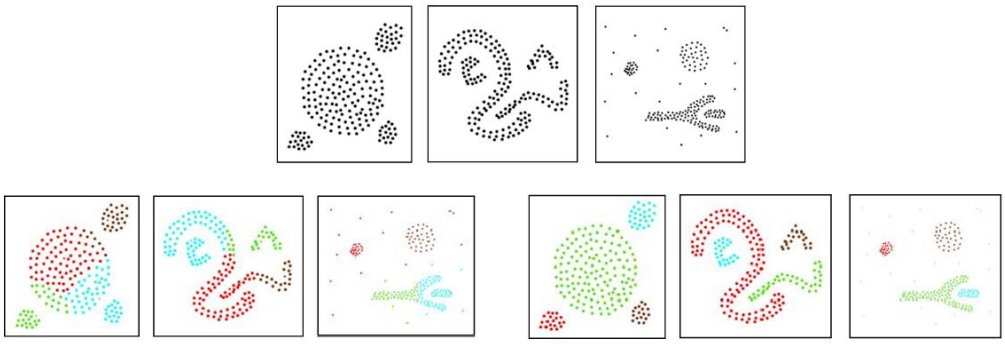

Рисунок 7 – Иллюстрация работы алгоритмов кластеризации
Сверху изображен оригинальный набор точек. Слева снизу – результат работы алгоритма CLARANS. Справа снизу – алгоритма DBSCAN

Так как стандартные библиотеки C++ (язык реализации SpdRoot) не содержат встроенной реализации DBSCAN, применялась переписанная на этот язык версия алгоритма. Для оптимизации работы с большими массивами данных использовалась структура kd-дерева из свободно распространяемой библиотеки nanoflann [9].

На рисунке 8 показано сравнение результатов C++ реализации DBSCAN с эталонной реализацией из библиотеки scikit-learn [10] (Python) на тестовом наборе точек. Многочисленные тесты на различных случайных выборках подтвердили полное совпадение результатов обеих реализаций.

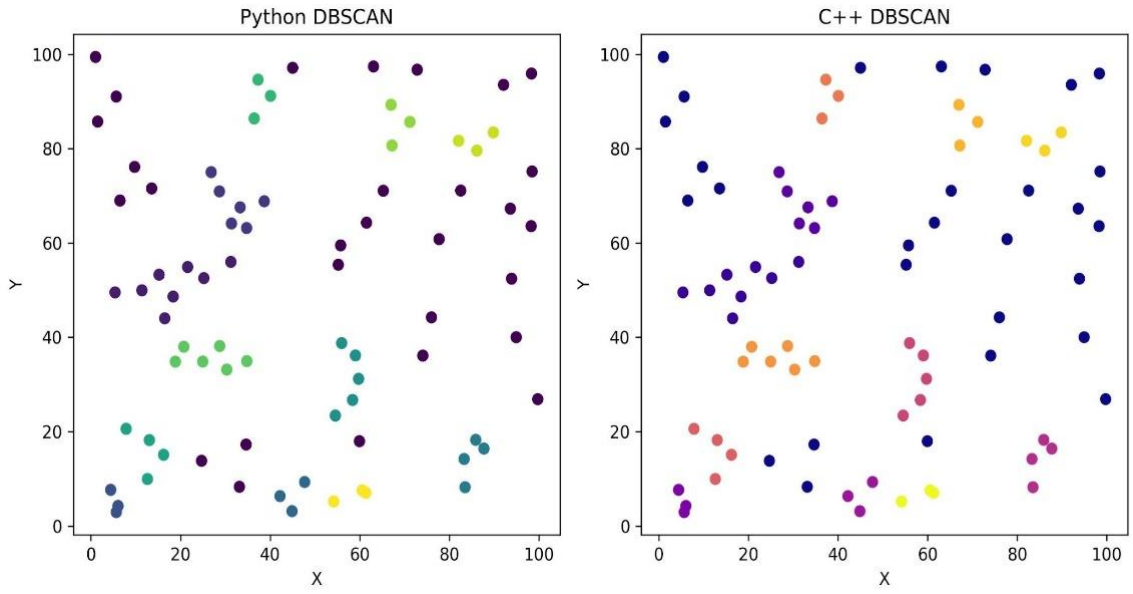

Рисунок 8 – Пример визуального сравнения работы двух реализаций алгоритма DBSCAN

Количество тестов n выбиралось из условия обеспечения доверительного уровня α=0.99 при вероятности ошибки p ≤ 0.01 в каждом тесте:

$$n \geq \frac{\ln(1-\alpha)}{\ln(1-p)} \approx 459 \qquad (1)$$

Адаптированный под C++ алгоритм DBSCAN был интегрирован в программный пакет SpdRoot для выполнения задач кластеризации хитов и классификации частиц в мюонной системе.

Данная разработка призвана заменить существующий алгоритм кластеризации, который в текущей реализации зависит от данных моделирования Монте-Карло, недоступных в реальных экспериментальных условиях.

Для объективной оценки эффективности нового алгоритма проведен детальный сравнительный анализ его работы с существующей реализацией. В ходе исследования были построены распределения количества кластеров в событии для мюонов и протонов с характерными энергиями 2 и 9 ГэВ, представленные на рисунках 9 и 10 соответственно. Дополнительно составлена сводная таблица (таблица 1), отражающая среднее количество формируемых кластеров $\bar{n}$ в широком диапазоне энергий от 2 до 10 ГэВ.

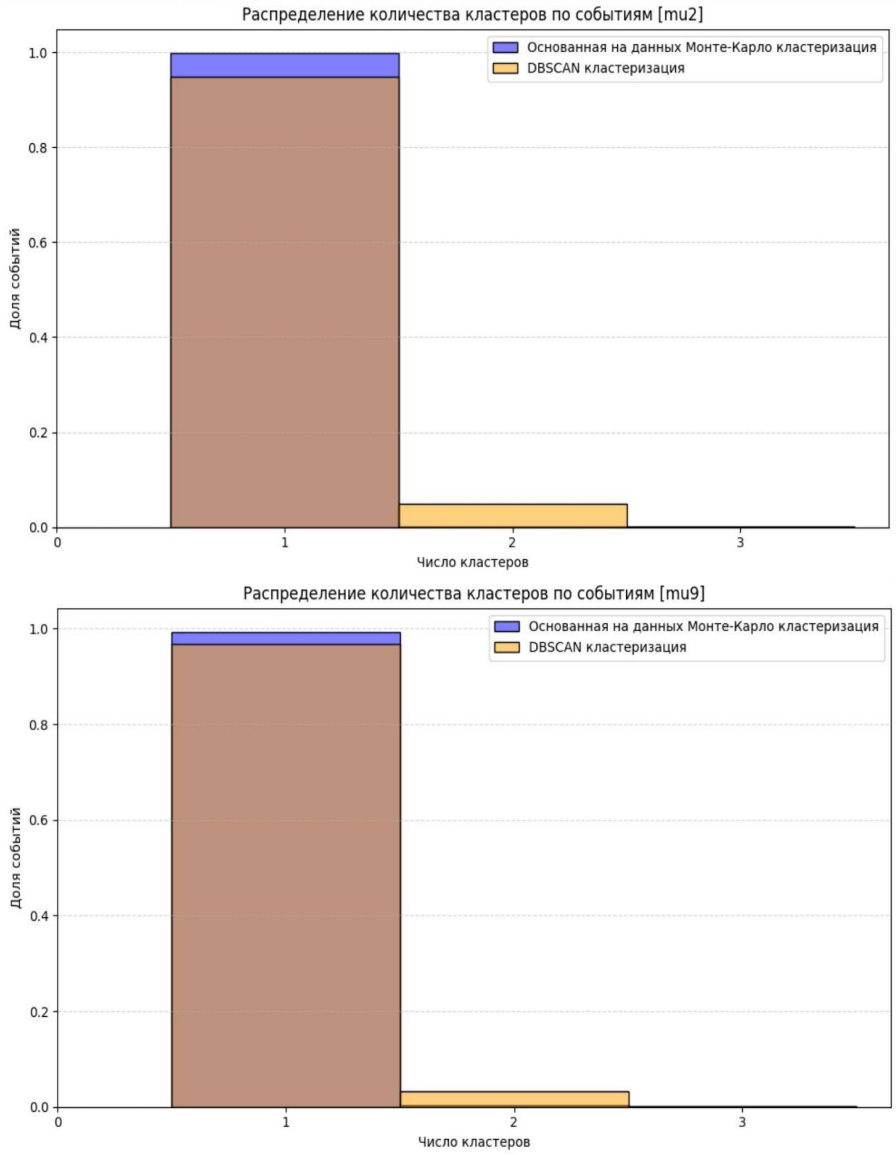

Рисунок 9 – Распределение по количеству сформированных кластеров для мюонов

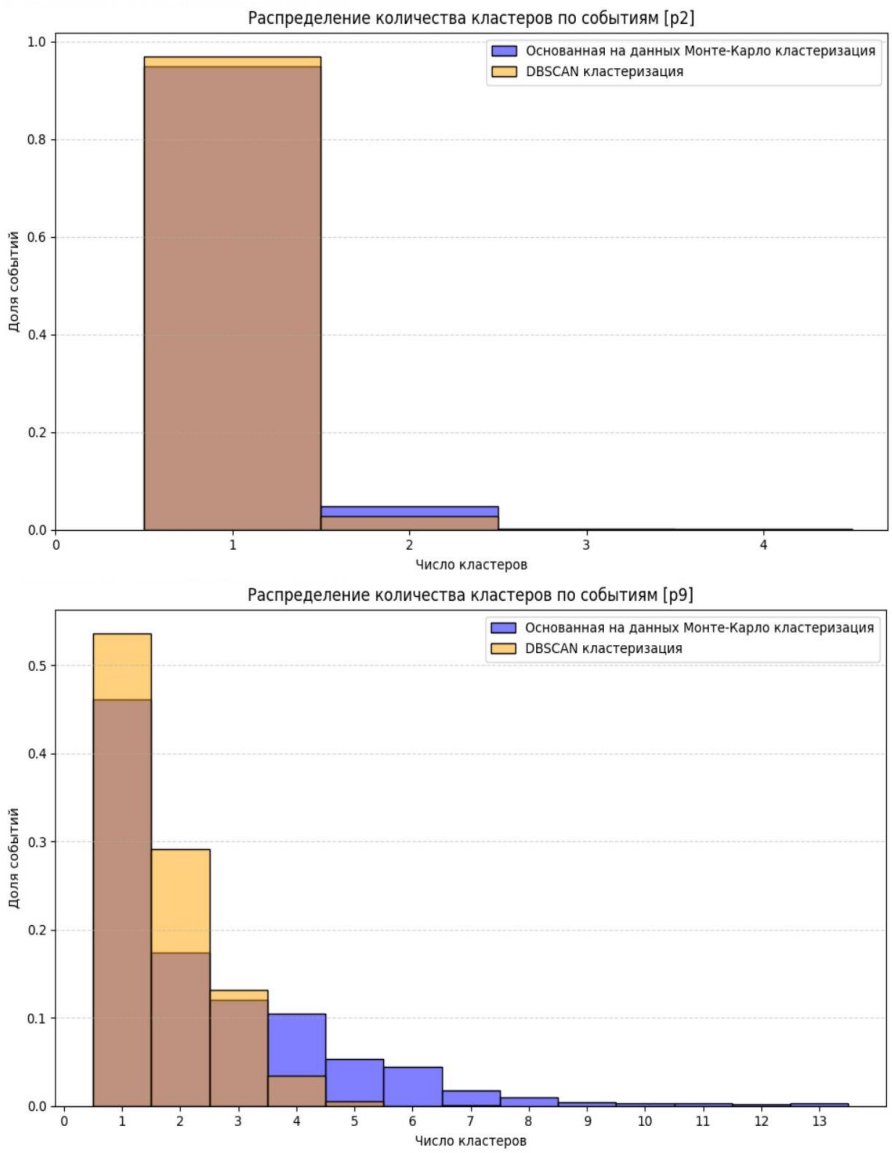

Рисунок 10 – Распределение по количеству сформированных кластеров для протонов

Таблица 1

Частица	Энергия, ГэВ	$\bar{n}_{\text{DBSCAN}}$	$\bar{n}_{\text{MC}}$	$\bar{n}_{\text{DBSCAN}}$ - $\bar{n}_{\text{MC}}$
μ	2	1.05	1.00	0.05
μ	3	1.04	1.00	0.04
μ	4	1.05	1.00	0.05
μ	5	1.04	1.00	0.04
μ	6	1.03	1.00	0.03
μ	7	1.04	1.00	0.04
μ	8	1.04	1.01	0.03

Продолжение таблицы 1

μ	9	1.03	1.01	0.02
μ	10	1.04	1.01	0.03
p	2	1.03	1.06	-0.03
p	3	1.13	1.21	-0.08
p	4	1.29	1.44	-0.15
p	5	1.36	1.62	-0.26
p	6	1.46	1.96	-0.50
p	7	1.60	2.05	-0.45
p	8	1.68	2.34	-0.66
p	9	1.69	2.48	-0.79
p	10	1.78	2.83	-1.05

Результаты проведенного сравнения демонстрируют, что алгоритм DBSCAN формирует статистически меньшее количество кластеров по сравнению с текущей реализацией для протонных событий. Это различие объясняется принципиально разными подходами к обработке данных: в то время как существующий алгоритм использует информацию о родительских частицах из моделирования Монте-Карло, DBSCAN опирается исключительно на геометрические параметры треков. В частности, близко расположенные треки вторичных частиц, которые текущий алгоритм разделяет благодаря информации о родительских частицах, в случае DBSCAN чаще объединяются в единые кластеры.

Полученные результаты подтверждают работоспособность DBSCAN в условиях, приближенных к экспериментальным, и обосновывают его применение в качестве перспективной альтернативы. В рамках дальнейших исследований планируется провести тонкую настройку параметров алгоритма для достижения оптимального баланса между физической интерпретацией событий и результатами кластеризации, что особенно важно для обработки реальных экспериментальных данных.

4 ОБЗОР АЛГОРИТМОВ МАШИННОГО ОБУЧЕНИЯ, ПОТЕНЦИАЛЬНО ПРИМЕНИМЫХ ДЛЯ РЕШЕНИЯ ЗАДАЧИ ИДЕНТИФИКАЦИИ МЮОНОВ

4.1 Нейронные сети: общий принцип работы

С целью классификации треков, полученных в результате кластеризации хитов, в работе предлагается использование различных алгоритмов машинного обучения, включая нейронные сети.

Математическая модель искусственного нейрона, имитирующего работу биологических нервных клеток, была впервые предложена Уорреном Мак-Каллоком и Уолтером Питтсом в работе [11]. Значительный вклад в развитие направления внес Фрэнк Розенблатт, разработавший первую обучаемую нейронную сеть - перцептрон, способный к обучению на примерах [12].

Искусственные нейронные сети (ИНС) представляют собой вычислительные модели, воспроизводящие принципы работы биологических нейронных сетей. Они состоят из взаимосвязанных искусственных нейронов, организованных в слои. Каждый нейрон обрабатывает входные сигналы и передает результат следующим нейронам (рисунок 11).

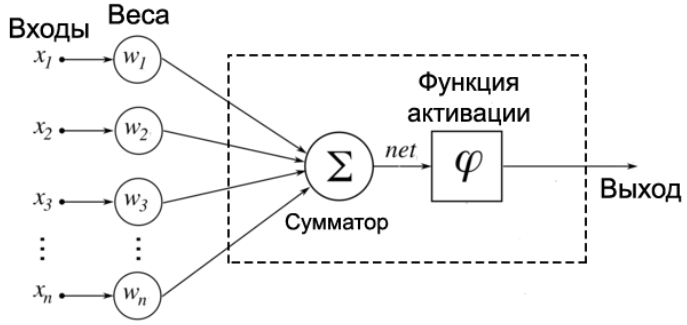

Рисунок 11 – Схема работы искусственного нейрона в нейронных сетях

Принцип работы отдельного нейрона включает:

1. Прием входных сигналов x_1, x_2, ..., x_n
2. Умножение каждого входа на соответствующий весовой коэффициент w_1, w_2, ..., w_n

3. Вычисление взвешенной суммы с добавлением смещения (bias) b
4. Применение нелинейной функции активации f к полученной сумме
5. Формирование выходного сигнала $y = f(\Sigma w_i x_i + b)$

Функция активации вводит в модель необходимую нелинейность, позволяя сети решать сложные задачи. Типичная архитектура ИНС включает:

- Входной слой (прием данных)
- Один или несколько скрытых слоев (обработка информации)
- Выходной слой (формирование результата)

Такая структура показана на рисунке 12.

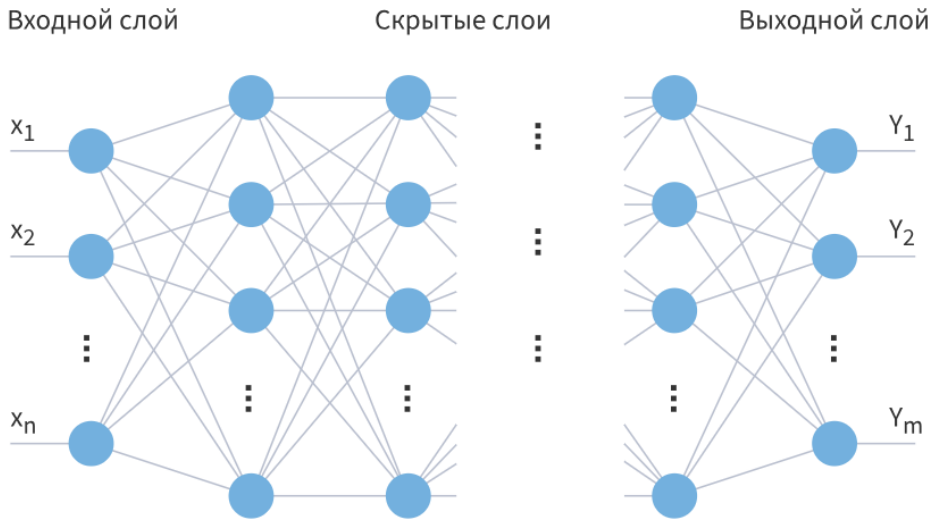

Рисунок 12 – Схема искусственной нейронной сети

Процесс обучения сети заключается в оптимизации весовых коэффициентов для минимизации ошибки предсказания. Наиболее распространенным методом обучения является алгоритм обратного распространения ошибки [13], который корректирует веса на основе разницы между предсказанными и эталонными значениями.

Таким образом, искусственные нейронные сети моделируют процессы обработки информации, характерные для биологических систем, и находят широкое применение в задачах классификации, распознавания образов и прогнозирования.

4.2 Свёрточные нейронные сети

Первым рассматриваемым методом решения задачи классификации выступают свёрточные нейронные сети (Convolutional Neural Networks, CNN), концепция которых была первоначально разработана Яном Лекуном и его коллегами [14]. Этот специализированный тип искусственных нейронных сетей создан для эффективной обработки данных с сеточной структурой, особенно изображений.

Основу CNN составляет операция свёртки, в ходе которой специальные фильтры (ядра свёртки) последовательно применяются к входным данным. Каждый такой фильтр представляет собой небольшую матрицу весовых коэффициентов, которая перемещается по всему входному изображению, вычисляя скалярное произведение между своими элементами и соответствующими участками входных данных. В результате этой операции формируется карта признаков, наглядно демонстрирующая наличие определённых характеристик в различных областях изображения, как показано на рисунке 13.

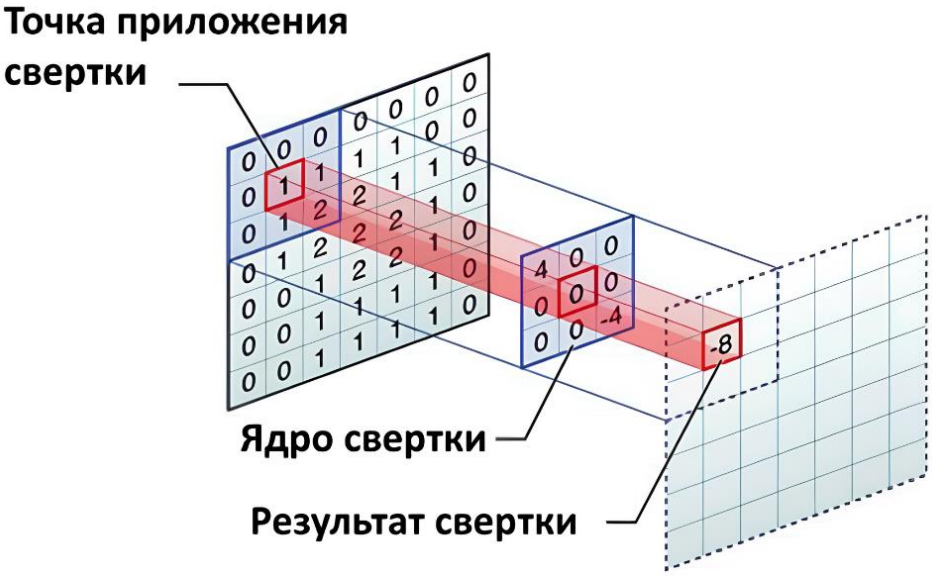

Рисунок 13 – Применение свёртки в свёрточном слое

Типичная архитектура CNN, представленная на рисунке 14, включает последовательность чередующихся свёрточных слоёв и слоёв подвыборки (пулинга), завершающуюся одним или несколькими полносвязными слоями. Свёрточные слои отвечают за извлечение локальных признаков различной сложности - от простых границ и углов до сложных текстур и паттернов. Слои пулинга выполняют важную функцию уменьшения размерности карт признаков, что позволяет снизить вычислительную нагрузку и повысить устойчивость модели к незначительным смещениям и искажениям входных данных. Завершающие полносвязные слои интегрируют все извлечённые признаки и выполняют окончательную классификацию.

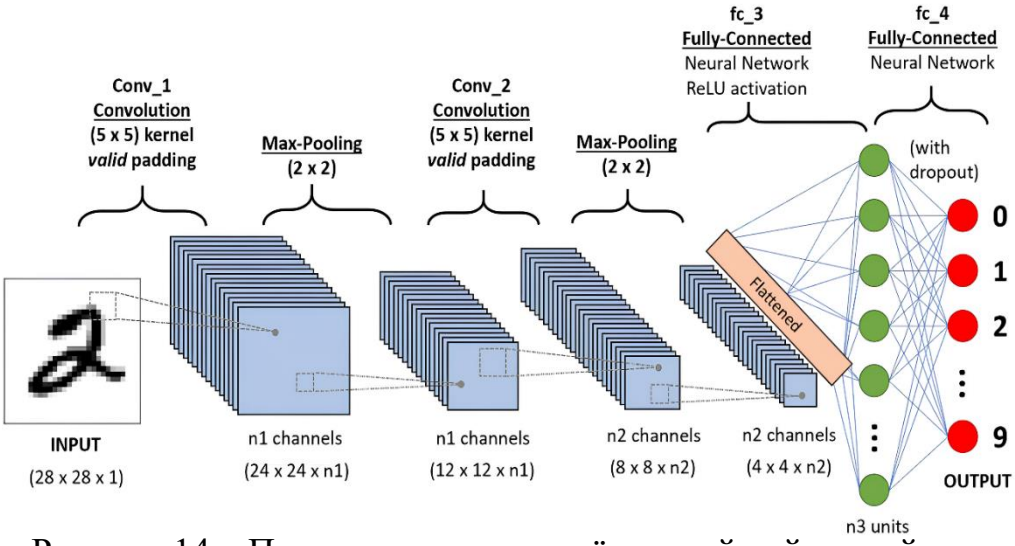

Рисунок 14 – Пример структуры свёрточной нейронной сети

Особая эффективность CNN в задачах классификации объясняется несколькими ключевыми особенностями. Во-первых, использование локальных связей между нейронами соответствует природной организации визуальных данных. Во-вторых, механизм разделения весов позволяет значительно сократить количество параметров модели по сравнению с полносвязными архитектурами. В-третьих, иерархическая организация слоёв обеспечивает поэтапное выделение признаков - от простых элементов к сложным композициям. Эти свойства делают CNN особенно подходящими для решения задач анализа и классификации изображений в физических экспериментах.

4.3 Дерево решений

Альтернативным подходом к решению задачи классификации выступает использование ансамблей деревьев решений. Деревья решений [15, 16] представляют собой непараметрический метод обучения с учителем, применяемый как для задач классификации, так и для регрессионного анализа. Основная идея метода заключается в построении предсказательной модели на основе последовательности логических правил, автоматически извлекаемых из признаков обучающей выборки.

Структурно дерево решений образует иерархическую древовидную схему, где каждый внутренний узел содержит правило вида "Если ..., то ...", формируемое в процессе обучения. Наглядное представление работы алгоритма приведено на рисунке 15. Процесс построения дерева предполагает рекурсивное бинарное разбиение входного множества примеров: на каждом узле выборка разделяется на два подмножества - объекты, удовлетворяющие условию узла, и объекты, не удовлетворяющие этому условию. Этот процесс продолжается до достижения критериев остановки алгоритма.

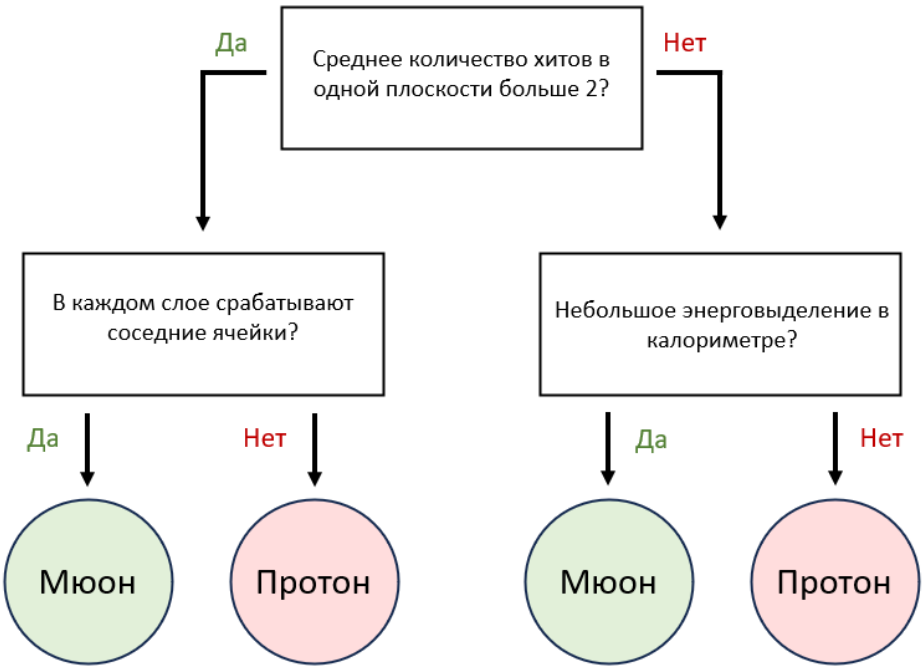

Рисунок 15 – Пример схемы работы дерева решений

Терминальные узлы (листья) дерева содержат конечное решение: для задач классификации - метку класса, для регрессионных задач - значение или интервал целевой переменной. Важной особенностью листьев является то, что они, в отличие от внутренних узлов, не осуществляют дальнейшего разбиения, а содержат подмножество объектов, удовлетворяющих всей цепочке правил от корня дерева до данного листа. При этом каждый объект может принадлежать только одному листу, что обеспечивает детерминированность предсказаний модели.

С увеличением количества признаков в данных модель получает возможность уточнять прогноз целевого значения из заданного множества классов. Полученные прогностические характеристики могут быть использованы в качестве входных параметров для моделей более высокого уровня. Особенностью деревьев решений является их способность работать с разнотипными данными, включая категориальные признаки, и автоматически определять наиболее информативные признаки для классификации. Кроме того, этот метод требует минимальной предварительной обработки данных и сохраняет прозрачность принимаемых решений, что особенно важно для интерпретации результатов в научных исследованиях.

4.4 Градиентный бустинг

Ансамблевые методы комбинируют предсказания множества базовых моделей, обученных по заданному алгоритму, для достижения лучшей обобщающей способности и устойчивости по сравнению с отдельными моделями. Классическим примером таких методов служит градиентный бустинг (Gradient Boosting) [17] – алгоритм последовательного построения линейной комбинации простых деревьев решений. На каждой итерации такой бустинг строит новое дерево, минимизирующее остаточные ошибки текущего ансамбля путем аппроксимации антиградиента функции потерь. Итоговый прогноз представляет собой линейную комбинацию предсказаний всех построенных деревьев. Ключевые преимущества алгоритма:

- Последовательная коррекция ошибок предыдущих моделей
- Возможность достижения высокой точности даже при использовании простых базовых моделей
- Естественная регуляризация за счет ограничения глубины деревьев

Хотя отдельные деревья решений легко интерпретировать благодаря возможности визуализации их структуры, модели градиентного бустинга, состоящие из сотен или тысяч деревьев, теряют это преимущество из-за своей сложности. Однако даже для таких сложных ансамблей существует возможность оценивать важность отдельных признаков, что позволяет сохранить хотя бы частичную интерпретируемость модели.

Ключевая идея заключается в том, что признаки, расположенные в верхних узлах деревьев, оказывают более существенное влияние на итоговый прогноз, поскольку они участвуют в разбиении для большей доли обучающих объектов. Напротив, признаки в глубоких узлах влияют лишь на небольшие подмножества данных и потому менее значимы.

Это наблюдение легло в основу метода MDI [16] (Mean Decrease in Impurity), который количественно оценивает важность признаков через усредненное уменьшение неопределенности (примеси) при разбиении по данному признаку во всех узлах всех деревьев ансамбля. Чем чаще признак используется для разбиения и чем выше расположены соответствующие узлы в деревьях, тем больше его итоговая важность.

Такой подход не только помогает понять относительный вклад признаков в предсказание модели, но и может быть полезен для отбора наиболее информативных признаков, что особенно ценно в задачах с высокой размерностью признакового пространства.

Градиентный бустинг, хотя и проигрывает нейронным сетям в способности выявлять сложные нелинейные закономерности и адаптироваться к различным типам данных, предлагает значительные практические преимущества при работе со структурированными данными. Главное достоинство метода заключается в его относительной простоте – для эффективной работы обычно достаточно настроить всего несколько ключевых гиперпараметров, таких как скорость обучения и глубина деревьев, в отличие от нейросетей, требующих сложной настройки архитектуры. Получаемые модели отличаются компактностью и низкими вычислительными затратами при сохранении конкурентной точности. Кроме того, градиентный бустинг предоставляет базовые средства интерпретации модели, включая оценку значимости признаков и анализ структуры отдельных деревьев, тогда как нейросети практически не поддаются содержательной интерпретации.

5 РАЗРАБОТКА МЕТОДА ИДЕНТИФИКАЦИИ МЮОНОВ

С целью идентификации мюонов на фоне адронных событий в работе были применены два алгоритма машинного обучения: сверточная нейронная сеть (CNN) и градиентный бустинг (Gradient Boosting) (приложение А). Обучающая выборка сформирована на основе данных Монте-Карло, которые моделировали прохождение протонов и мюонов через детектирующую систему. Энергии частиц варьировались в диапазоне от 1 до 10 ГэВ с шагом 1 ГэВ, при этом для протонов рассматривались энергии начиная с 2 ГэВ. Предварительное сравнительное тестирование продемонстрировало эффективность обоих методов в решении поставленной задачи. Однако для окончательного выбора оптимального алгоритма требуется проведение дополнительных исследований. В рамках дальнейшей работы планируется усовершенствовать обе модели. Для CNN это предполагает оптимизацию архитектуры сети, включая подбор количества слоев, типов фильтров и функций активации. В случае градиентного бустинга основное внимание будет уделено выбору наиболее информативных признаков и тонкой настройке гиперпараметров модели. После проведения указанных усовершенствований будет выполнено повторное сравнение характеристик CNN и градиентного бустинга. Это позволит сделать обоснованный выбор алгоритма, наиболее подходящего для решения задачи идентификации мюонов в условиях эксперимента.

Архитектура сверточной нейронной сети (CNN):

Сеть состоит из последовательности чередующихся сверточных и субдискретизирующих слоев с последующей полносвязной частью. В сверточных слоях используются ядра размером 3×3 пикселя с функцией активации ReLU, где количество фильтров последовательно увеличивается от 16 в первом слое до 32 во втором и 64 в третьем. Каждый сверточный слой сопровождается операцией пулинга (max-pooling) с окном 2×2 пикселя, что

обеспечивает постепенное уменьшение пространственной размерности признаков.

После завершения сверточной части выполняется преобразование тензора в одномерный вектор (операция "вытягивания" – flatten), который подается на вход полносвязным слоям.

Полносвязная часть сети включает:

- Один скрытый слой с 128 нейронами и активацией ReLU
- Выходной слой с 2 нейронами (соответствующими классам "протон" и "мюон") с линейной активацией

Для классификации на выходе сети применяется функция softmax, преобразующая линейные выходы в вероятности принадлежности к каждому классу. Сеть принимает на вход одноканальные изображения размером 128×128 пикселей в градациях серого. Каждое изображение представляет собой двумерную проекцию пространственного распределения хитов, зарегистрированных при прохождении частицы через детекторную систему. Примеры входных изображений приведены на рисунке 16.

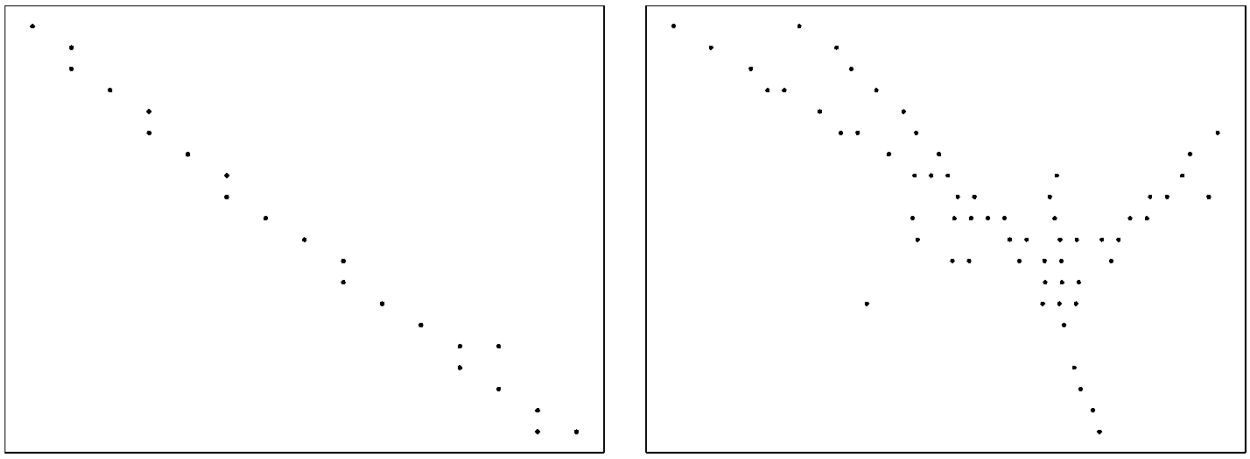

Рисунок 16 – Примеры входных изображений для CNN
слева – мюон, справа – протон

Таким образом, предложенная свёрточная нейронная сеть решает задачу классификации частиц путем анализа пространственных закономерностей в распределении срабатываний детекторной системы и последующего выявления характерных признаков в картине хитов, например наличие или отсутствие каскадных процессов.

В результате тестирования алгоритма получены следующие количественные характеристики его работы [18]:

- Точность (Accuracy): 0.92
- Полнота (Recall) для класса мюонов: 0.92
- Точность (Precision) для класса мюонов: 0.91
- F1-score для класса мюонов: 0.92

Также получены матрица ошибок (Confusion Matrix) [19] и ROC-кривые [20], представленные на рисунке 17 и рисунке 18 соответственно.

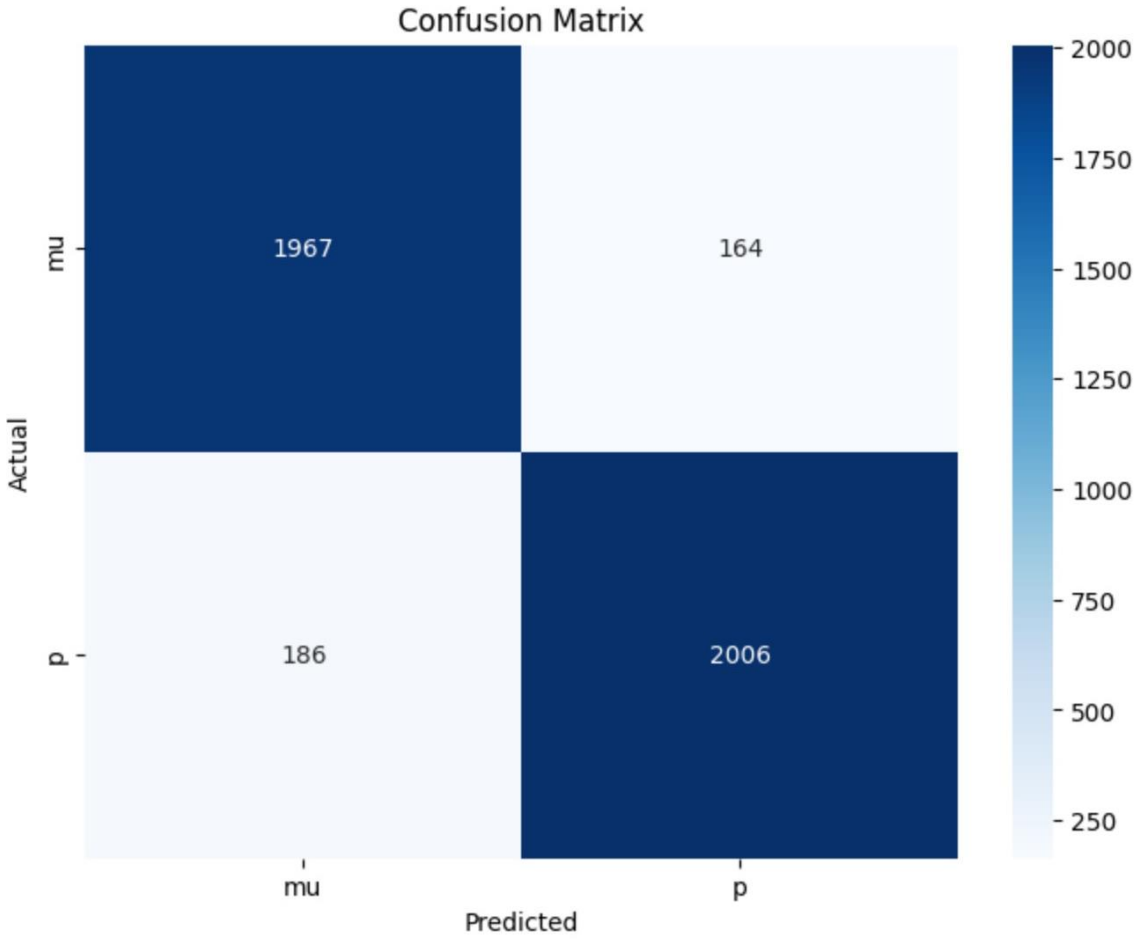

Рисунок 17 – Матрица ошибок для CNN

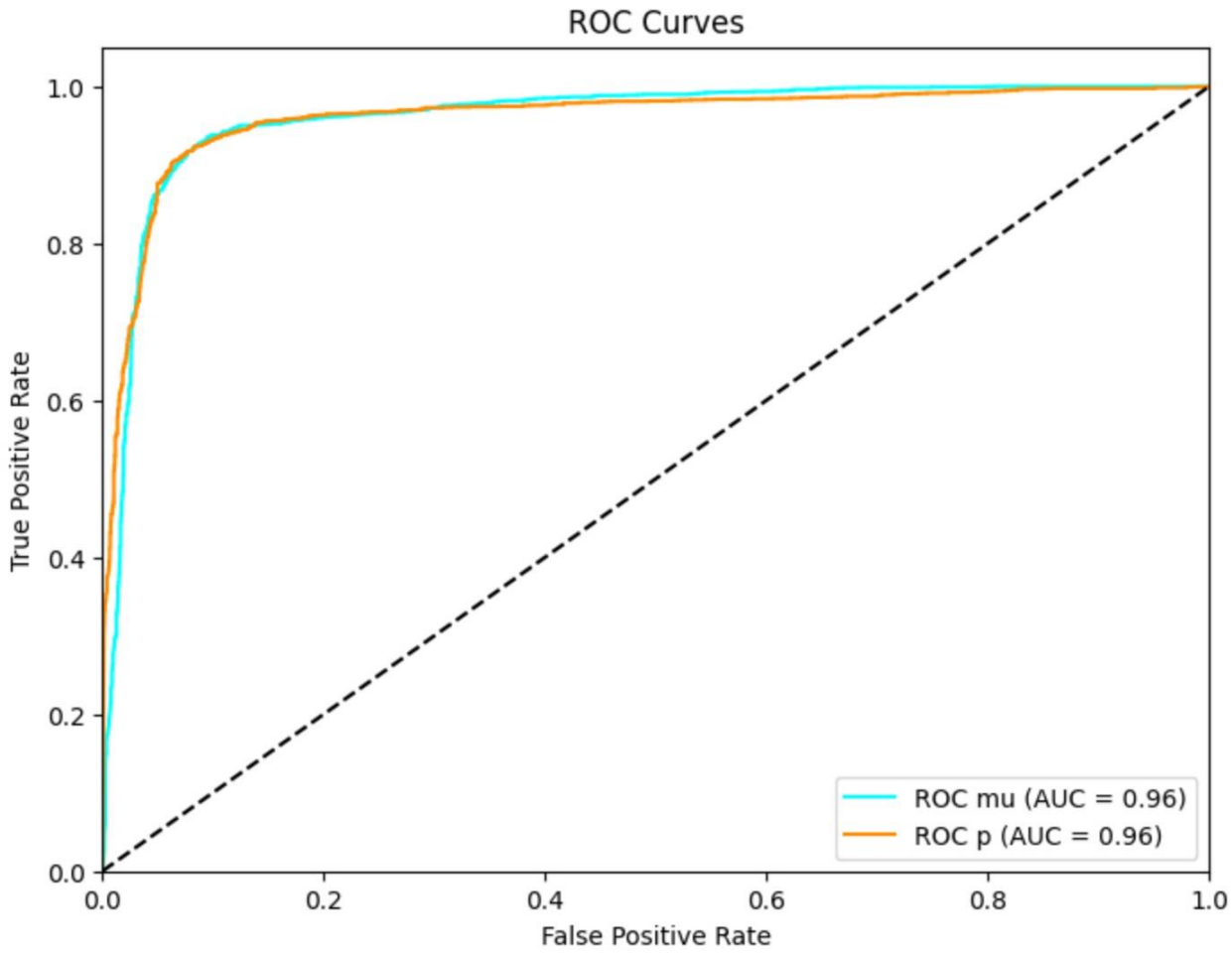

Рисунок 18 – ROC-кривые для CNN

В работе использовался алгоритм градиентного бустинга XGBoost со следующими настроенными гиперпараметрами:

- Количество деревьев в ансамбле: 100
- Максимальная глубина каждого дерева: 3 уровня
- Функция потерь: бинарная логистическая (binary:logistic)
- Метрика для валидации: логарифмические потери (logloss)

Ограничение глубины деревьев до 3 уровней обеспечивает:

- Выявление значимых признаков без переобучения
- Построение интерпретируемых правил классификации
- Эффективное вычисление предсказаний

Выбор бинарной логистической функции потерь оптимален для:

- Задачи разделения на два класса (мюоны/протоны)
- Получения вероятностных оценок принадлежности

Использование logloss в качестве метрики валидации:

- Чувствительно к качеству вероятностных предсказаний
- Жестко штрафует за уверенные ошибочные классификации

Дополнительные параметры (оставлены по умолчанию):

- Скорость обучения (learning rate): 0.3
- Доля случайных признаков: 100%
- Минимальное число образцов в листе: 1
- Коэффициент L2-регуляризации: 0

Такая конфигурация обеспечивает баланс между:

- Достаточной сложностью модели (100 деревьев)
- Устойчивостью к переобучению (ограниченная глубина)
- Интерпретируемостью результатов
- Вычислительной эффективностью

Алгоритм использовал три ключевые переменные для классификации частиц:
Интенсивность срабатываний детектора:

- Среднее количество хитов на детекторную плоскость (характеризует общую активность детектора)
- Максимальное количество хитов в одной плоскости (показывает локальные всплески активности)

Угловая характеристика трека (новый разработанный признак):

- Для каждого хита вычислялся угловой параметр в цилиндрической системе координат
- Для прямолинейного трека (характерного для мюонов) углы хитов локализуются вокруг одного значения
- Для каскадных процессов (типичных для адронов) наблюдается значительное разброс углов
- В качестве признака использовалось стандартное отклонение углового распределения σ_θ

Особенности вычисления углового признака:
Для треков в барельной области:

- Использовалась проекция XY ($\theta = \arctan(y/x)$)
- Учитывалась только σ_{xy}

Для треков в эндкапах:

- Вычислялись обе проекции: ZX (θ_{zx}) и ZY (θ_{zy})
- Выбиралось максимальное стандартное отклонение: $\max(\sigma_{zx}, \sigma_{zy})$

Выбор проекций для анализа треков обусловлен геометрией детектора и особенностями взаимодействия частиц с веществом. Поскольку ось z детектора сонаправлена с осью пучка, в баррельной области наиболее информативной оказывается проекция XY, перпендикулярная направлению пучка. В этой проекции четко проявляются характерные особенности мюонных треков – их прямолинейность и минимальное рассеяние, а также хорошо видно развитие адронных каскадов, происходящее преимущественно в поперечной плоскости. В торцевых же частях детектора (эндкапах) из-за специфического расположения детекторных элементов – проволочных камер и стриповых детекторов – пространственная картина взаимодействий может по-

разному проявляться в проекциях ZX и ZY. Поэтому для треков в эндкапах вычисляются угловые распределения в обеих проекциях, после чего выбирается наиболее показательная из них по величине стандартного отклонения. Такой подход позволяет максимально полно учитывать особенности пространственного распределения хитов как для прямолинейных мюонных треков, так и для разветвленных адронных каскадов во всех областях детектора.

В результате тестирования алгоритма также как и для CNN получены следующие количественные характеристики работы градиентного бустинга:

- Точность (Accuracy): 0.93
- Полнота (Recall) для класса мюонов: 0.96
- Точность (Precision) для класса мюонов: 0.91
- F1-score для класса мюонов: 0.93

На основе анализа работы алгоритма построена диаграмма важности признаков (рисунок 19), отражающая их относительный вклад в процесс классификации частиц.

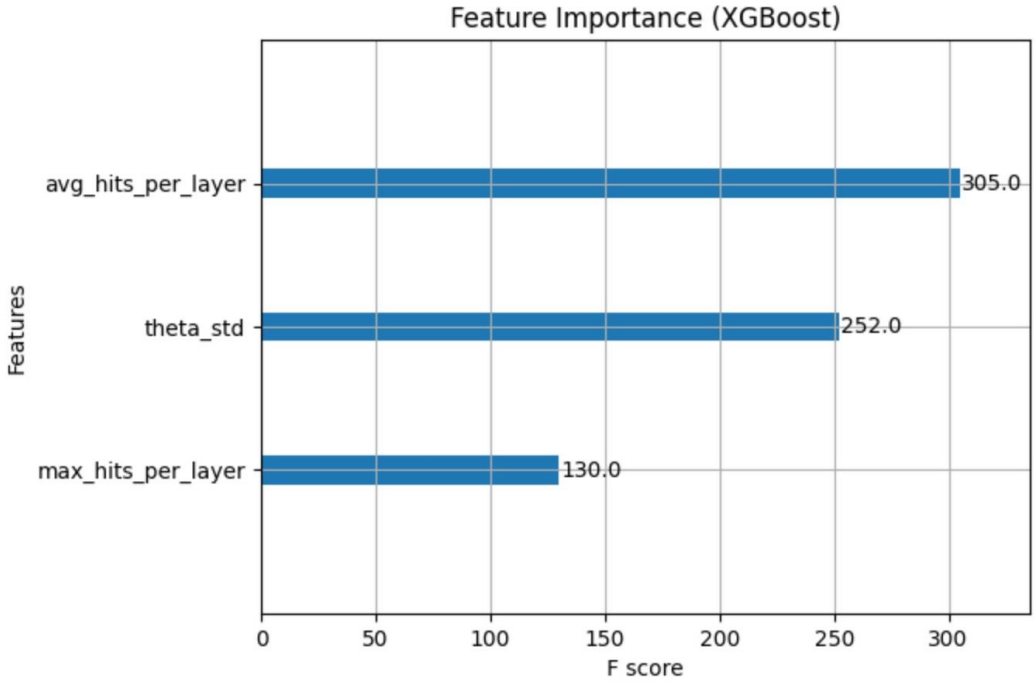

Рисунок 19 – Диаграмма важности признаков для модели бустинга

Как видно из диаграммы, наибольший вес имеет признак среднего количества срабатываний в одной детекторной плоскости, который позволяет различать частицы по общей активности их взаимодействия с детектором. Вторым по значимости оказалось стандартное отклонение углового распределения хитов, что подтверждает важную роль пространственных характеристик трека для разделения мюонов и адронного фона. Максимальное количество срабатываний, хотя и уступает двум другим признакам по важности, также вносит существенный вклад в классификацию, помогая эффективно идентифицировать локальные скопления хитов, характерные для адронных ливней. Полученное распределение важности признаков хорошо согласуется с физическими ожиданиями и подтверждает обоснованность выбранного подхода к формированию признакового пространства для решения задачи идентификации мюонов.

Также получены матрица ошибок (Confusion Matrix) и ROC-кривая, представленные на рисунке 20 и рисунке 21 соответственно.

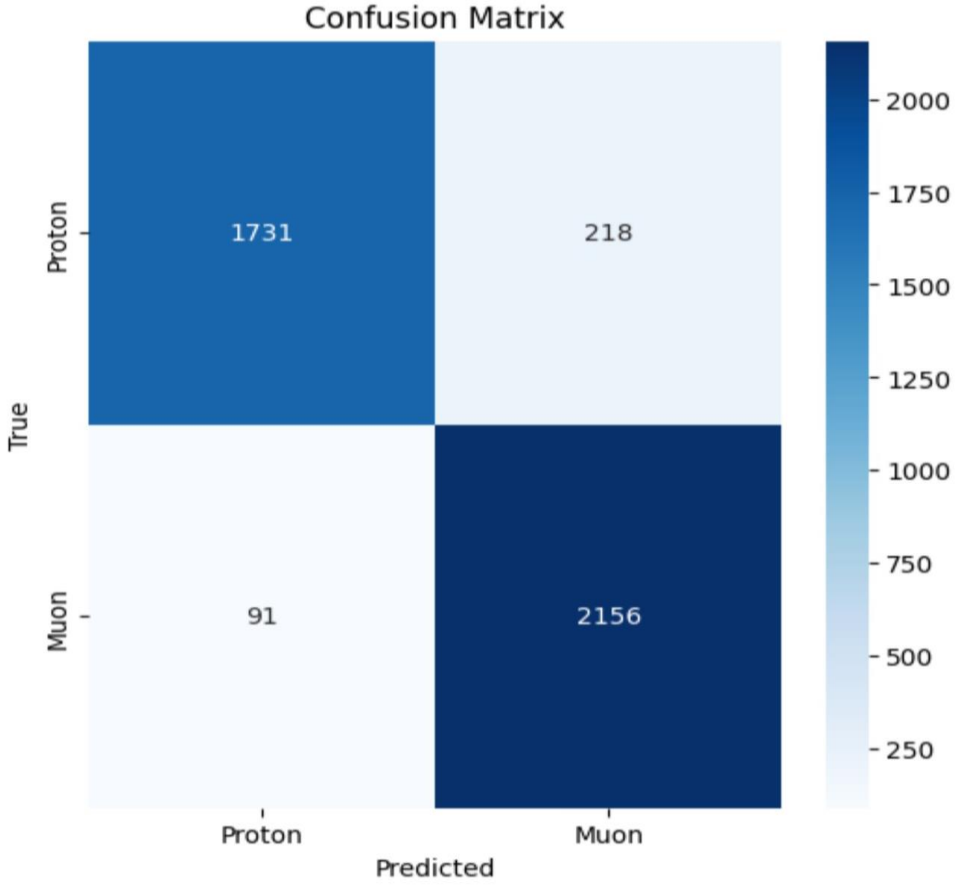

Рисунок 20 – Матрица ошибок для XGBoost

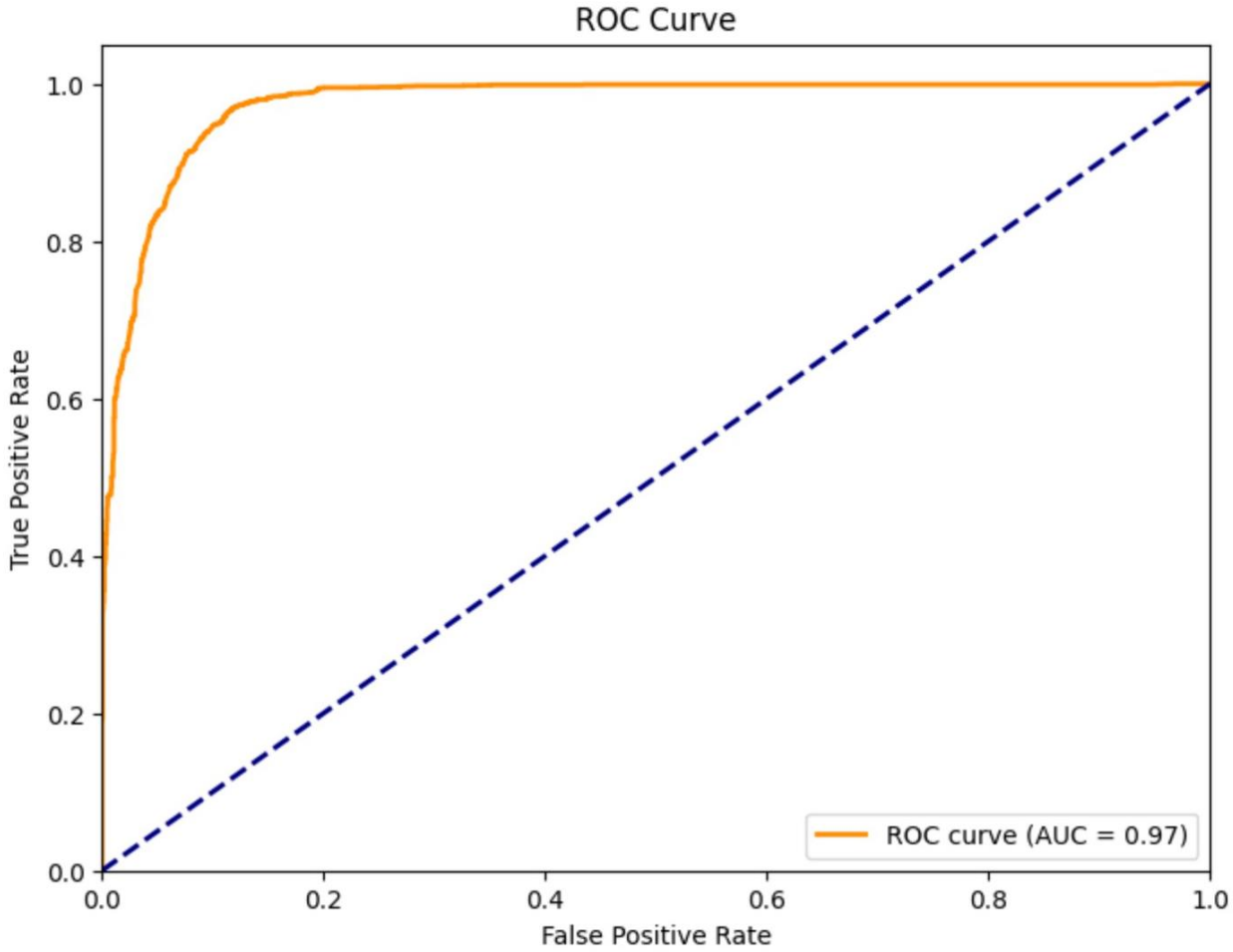

Рисунок 21 – ROC-кривая для XGBoost

Проведенный сравнительный анализ эффективности алгоритмов показал, что как свёрточная нейронная сеть (CNN), так и градиентный бустинг демонстрируют сопоставимую и достаточно высокую точность идентификации мюонов, однако имеется значительный потенциал для дальнейшего улучшения результатов классификации.

Для повышения качества распознавания мюонов планируется реализовать комплексный подход. В случае сверточной нейронной сети основное внимание будет уделено оптимизации архитектуры сети через систематический подбор гиперпараметров, включая количество и тип сверточных слоев, размеры ядер свертки и параметры функций активации. Это позволит выявить конфигурацию, обеспечивающую максимальную точность идентификации.

Для модели градиентного бустинга ключевым направлением станет исследование оптимальных комбинаций входных переменных, что должно привести к формированию более точных правил классификации. Особое значение при этом будет уделено анализу обновленной диаграммы важности признаков после проведения соответствующей оптимизации.

Важным аспектом дальнейшей работы станет детальное исследование зависимости качества классификации от энергии регистрируемых частиц. Это позволит выявить возможные систематические эффекты и скорректировать алгоритмы для различных энергетических диапазонов.

После реализации указанных улучшений планируется провести повторное сравнительное тестирование обоих алгоритмов. Перспективным направлением представляется также исследование возможностей гибридного подхода, который мог бы сочетать сильные стороны CNN и градиентного бустинга для достижения более высоких показателей точности.

Такой подход позволит получить более надежные и объективные результаты сравнения алгоритмов, а также выявить оптимальный подход к решению задачи идентификации частиц в различных энергетических диапазонах.

ЗАКЛЮЧЕНИЕ

В работе произведено моделирование прототипа мюонной системы эксперимента SPD с использованием программного пакета SpdRoot. В дальнейшем геометрическую модель планируется использовать для оптимизации алгоритмов идентификации частиц и анализа их взаимодействия с веществом детектора. Также произведено моделирование и визуализация событий в прототипе, получен набор хитов частиц. С использованием алгоритма кластеризации DBSCAN сформированы треки для дальнейшей обработки и анализа данных моделирования. Реализация алгоритма выполнена на языке C++, протестирована и сравнена с существующей реализацией, основанной на данных моделирования Монте-Карло, и интегрирована в программный пакет SpdRoot. В ходе работы также разработаны и сравнительно проанализированы предварительные варианты алгоритмов идентификации частиц на основе CNN и градиентного бустинга. Полученные результаты, несмотря на ограниченность исходной выборки данных, демонстрируют перспективность обоих подходов.

Основным направлением дальнейших исследований станет комплексная оптимизация обоих методов машинного обучения. Для свёрточной нейронной сети планируется провести детальный подбор архитектурных параметров, включая количество и конфигурацию слоев, размеры ядер свертки и параметры функций активации. Параллельно будет выполняться работа по отбору наиболее информативных входных переменных для модели градиентного бустинга с анализом их значимости.

Такая двусторонняя оптимизация позволит не только повысить точность каждой модели в отдельности, но и провести их более объективное сравнительное тестирование. Особый интерес представляет исследование возможностей создания комбинированного алгоритма, который мог бы интегрировать пространственную аналитическую способность CNN с

эффективностью работы с табличными данными, характерной для градиентного бустинга.

Важным аспектом исследований станет анализ поведения алгоритмов в различных энергетических диапазонах. Это включает изучение зависимости точности классификации от энергии частиц и разработку соответствующих корректирующих механизмов. Ожидается, что такой подход позволит создать более надежную и универсальную систему идентификации.

Необходима интеграция оптимизированного алгоритма в программную среду SpdRoot, что обеспечит его практическое применение в обработке экспериментальных данных.

СПИСОК ЛИТЕРАТУРЫ

1. *International spin physics collaboration at the collider NICA.* Technical design report of the Spin Physics Detector. — 12 February, 2023.

2. *Abazov V. M., Abramov V., Afanasyev L. G.* Conceptual design of the Spin Physics Detector. — arXiv:2102.00442, January 2021.

3. *Abazov V. M. et al.* The Muon system of the run II D0 detector // Nucl. Instrum. Meth. — 2005.

4. *Abbon P. et al.* The COMPASS experiment at CERN // Nucl. Instrum. Meth. — 2007.

5. *Novotny R. et al.* Technical Design Report for the PANDA Electromagnetic Calorimeter. — panda.gsi.de, August 2008.

6. Brun R., Rademakers F. ROOT — An object oriented data analysis framework // Nuclear Instruments and Methods in Physics Research Section A. — 1997. — Vol. 389, No. 1–2. — P. 81–86.

7. *Ester M. et al.* A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise // Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96). — 1996. — P. 226–231.

8. *Ng R. T., Han J.* CLARANS: A Method for Clustering Objects for Spatial Data Mining // IEEE Transactions on Knowledge and Data Engineering. — 1994. — Vol. 14, No. 5. — P. 1003–1016.

9. *Blanco J. L., Rai P. K.* nanoflann: A C++ header-only fork of FLANN, a library for Nearest Neighbor (NN) with KD-trees. — URL: https://github.com/jlblancoc/nanoflann.

10. Pedregosa F. et al. Scikit-learn: Machine learning in Python // Journal of Machine Learning Research. — 2011. — Vol. 12. — P. 2825–2830.

11. *McCulloch W., Pitts W.* A Logical Calculus of the Ideas Immanent in Nervous Activity // Bulletin of Mathematical Biophysics. — 1943.

12. *Rosenblatt F.* The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain // Psychological Review. — 1958.

13. *Rumelhart D., Hinton G., Williams R.* Learning Representations by Back-Propagating Errors // Nature. — 1986.

14. *LeCun Y. et al.* Backpropagation Applied to Handwritten Zip Code Recognition // Neural Computation. — 1989.

15. *Morgan J. N., Sonquist J. A.* Problems in the Analysis of Survey Data, and a Proposal // Journal of the American Statistical Association. — 1963. — Vol. 58, No. 302. — P. 415–434.

16. *Breiman L. et al.* Classification and Regression Trees. — Wadsworth, 1984.

17. *Friedman J. H.* Greedy Function Approximation: A Gradient Boosting Machine // Annals of Statistics. — 2001. — Vol. 29, No. 5. — P. 1189–1232.

18. *Sokolova M., Lapalme G.* A systematic analysis of performance measures for classification tasks // Information Processing & Management. — 2009. — Vol. 45, No. 4. — P. 427–437.

19. *Kohavi R., Provost F.* Glossary of Terms // Machine Learning. — 1998. — Vol. 30, No. 2–3. — P. 271–274.

20. *Fawcett T.* An introduction to ROC analysis // Pattern Recognition Letters. — 2006. — Vol. 27, No. 8. — P. 861–874.

ПРИЛОЖЕНИЕ А. РЕАЛИЗАЦИЯ АЛГОРИТМА ИДЕНТИФИКАЦИИ МЮОНОВ НА ОСНОВЕ CNN

Master_diploma URL: https://github.com/RozoStal/Master_diploma.git

```
import tensorflow as tf
from tensorflow.keras.utils import image_dataset_from_directory
from tensorflow.keras import layers, models
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import ModelCheckpoint
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix, roc_curve, auc
from itertools import cycle
import os

os.environ["CUDA_VISIBLE_DEVICES"] = "-1"
DATA_DIR = './LearningData/'
INPUT_SIZE = (128, 128)
BATCH_SIZE = 32
SEED = 42
VALIDATION_SPLIT = 0.2
EPOCHS = 10
LR = 0.001

raw_train_dataset = image_dataset_from_directory(
    directory=DATA_DIR,
    labels='inferred',
    label_mode='int',
    color_mode='grayscale',
    batch_size=BATCH_SIZE,
    image_size=INPUT_SIZE,
    shuffle=True,
    seed=SEED,
    validation_split=VALIDATION_SPLIT,
    subset='training',
)

class_names = raw_train_dataset.class_names
num_classes = len(class_names)
print("Class names:", class_names)

train_dataset = image_dataset_from_directory(
    directory=DATA_DIR,
    labels='inferred',
    label_mode='int',
    color_mode='grayscale',
    batch_size=BATCH_SIZE,
    image_size=INPUT_SIZE,
    shuffle=True,
    seed=SEED,
    validation_split=VALIDATION_SPLIT,
    subset='training',
)
```

```
val_dataset = image_dataset_from_directory(
    directory=DATA_DIR,
    labels='inferred',
    label_mode='int',
    color_mode='grayscale',
    batch_size=BATCH_SIZE,
    image_size=INPUT_SIZE,
    shuffle=True,
    seed=SEED,
    validation_split=VALIDATION_SPLIT,
    subset='validation',
)


def create_cnn_model(input_shape=(128, 128, 1), num_classes=2):
    inputs = tf.keras.Input(shape=input_shape)

    x = layers.Rescaling(1./255)(inputs)
    x = layers.Conv2D(16, 3, padding='same', activation='relu')(x)
    x = layers.MaxPooling2D()(x)
    x = layers.Conv2D(32, 3, padding='same', activation='relu')(x)
    x = layers.MaxPooling2D()(x)
    x = layers.Conv2D(64, 3, padding='same', activation='relu')(x)
    x = layers.MaxPooling2D()(x)

    x = layers.Flatten()(x)
    x = layers.Dense(128, activation='relu')(x)
    outputs = layers.Dense(num_classes)(x)

    return tf.keras.Model(inputs=inputs, outputs=outputs)

model = create_cnn_model(input_shape=(*INPUT_SIZE, 1),
num_classes=num_classes)
model.compile(
    optimizer=Adam(learning_rate=LR),
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
    metrics=['accuracy']
)

checkpoint = ModelCheckpoint(
    "model.keras",
    monitor='val_accuracy',
    save_best_only=True,
    mode='max',
    verbose=1
)

history = model.fit(
    train_dataset,
    epochs=EPOCHS,
    validation_data=val_dataset,
    callbacks=[checkpoint]
)

model.save("model.keras")
```

```
def evaluate_and_visualize(model, dataset, class_names):
    all_labels = []
    all_preds = []
    all_probs = []

    for images, labels in dataset:
        logits = model(images, training=False)
        probs = tf.nn.softmax(logits)
        preds = tf.argmax(probs, axis=1)

        all_labels.append(labels.numpy())
        all_preds.append(preds.numpy())
        all_probs.append(probs.numpy())

    all_labels = np.concatenate(all_labels)
    all_preds = np.concatenate(all_preds)
    all_probs = np.concatenate(all_probs)

    cm = confusion_matrix(all_labels, all_preds)
    plt.figure(figsize=(8, 6))
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
                xticklabels=class_names, yticklabels=class_names)
    plt.title('Confusion Matrix')
    plt.xlabel('Predicted')
    plt.ylabel('Actual')
    plt.show()

    plt.figure(figsize=(8, 6))
    colors = cycle(['aqua', 'darkorange', 'cornflowerblue'])

    for i, color in zip(range(len(class_names)), colors):
        fpr, tpr, _ = roc_curve((all_labels == i).astype(int), all_probs[:,
i])
        roc_auc = auc(fpr, tpr)
        plt.plot(fpr, tpr, color=color,
                 label=f'ROC {class_names[i]} (AUC = {roc_auc:.2f})')

    plt.plot([0, 1], [0, 1], 'k--')
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('ROC Curves')
    plt.legend(loc="lower right")
    plt.show()

evaluate_and_visualize(model, val_dataset, class_names)

tf.saved_model.save(model, "saved_model")
```

ПРИЛОЖЕНИЕ В. РЕАЛИЗАЦИЯ АЛГОРИТМА ИДЕНТИФИКАЦИИ МЮОНОВ НА ОСНОВЕ BDT

```
import pandas as pd
import numpy as np
import xgboost as xgb
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split

from sklearn.metrics import (
    accuracy_score,
    precision_score,
    recall_score,
    f1_score,
    roc_auc_score,
    classification_report,
    confusion_matrix,
    roc_curve,
    auc
)
import matplotlib.pyplot as plt
import seaborn as sns


def read_event_data(file_path):
    data = []
    current_event = None

    with open(file_path, 'r') as file:
        lines = [line.strip() for line in file if line.strip() != '']

    i = 0
    while i < len(lines):
        line = lines[i]
        if line.isdigit(): # Check if a string is an event number
            event_num = int(line)
            if i + 1 < len(lines) and lines[i + 1].isdigit(): # Check the
next line for another event number
                i += 1 # Skipping an empty event
                continue
            else:
                current_event = event_num
                i += 1
                while i < len(lines) and not lines[i].isdigit():
                    parts = [x.strip() for x in lines[i].split(',')]
                    if len(parts) == 9:
                        data.append([current_event] + parts)
                    i += 1
        else:
            i += 1

    columns = ['event', 'layer', 'x', 'y', 'z', 'cluster', 'pid',
'detectorID', 'isBarrel', 'isEC']
    df = pd.DataFrame(data, columns=columns)
    df['event'] = df['event'].astype(int)
    df['layer'] = df['layer'].astype(int)

    return df
```

```
mu_dfs = {f'mu{i}':
read_event_data(f'/home/user/Data/JINR/CNN_files/Monte_Carlo_files/mu/mu{i}.d
at')
          for i in range(1, 11)}

p_dfs = {f'p{i}':
read_event_data(f'/home/user/Data/JINR/CNN_files/Monte_Carlo_files/p/p{i}.dat
')
         for i in range(2, 11)}

dataframes = {**mu_dfs, **p_dfs}

# Function for cleaning and transforming data
def clean_and_convert(column, dtype):
    if dtype == 'float':
        return pd.to_numeric(column.str.strip(), errors='coerce')
    elif dtype == 'int':
        return pd.to_numeric(column.str.strip(),
errors='coerce').astype('Int64')
    elif dtype == 'bool':
        return column.str.strip().astype(int).astype('bool')
    else:
        return column


columns_to_convert = {
    'x': 'float',
    'y': 'float',
    'z': 'float',
    'cluster': 'int',
    'pid': 'int',
    'isBarrel': 'bool',
    'isEC': 'bool'
}

for df_name, df in dataframes.items():
    for col, dtype in columns_to_convert.items():
        df[col] = clean_and_convert(df[col], dtype)


BDT_dataframes = {}

def calculate_theta(row):
    if row['isBarrel']:
        return np.arctan2(row['y'], row['x'])
    else:
        theta_xz = np.arctan2(row['x'], row['z'])
        theta_yz = np.arctan2(row['y'], row['z'])
        return theta_xz

def optimize_theta_std(group):
    theta_xz = np.arctan2(group['x'], group['z'])
    theta_yz = np.arctan2(group['y'], group['z'])

    std_xz = np.std(theta_xz)
    std_yz = np.std(theta_yz)
```

```
    if std_xz < std_yz:
        group['theta_optimized'] = theta_xz
    else:
        group['theta_optimized'] = theta_yz

    return group


for key, df in dataframes.items():
    df_work = df.copy()

    event_counts = df_work.groupby('event').size()
    events_to_keep = event_counts[event_counts > 1].index
    df_work = df_work[df_work['event'].isin(events_to_keep)].copy()
    df_work['theta'] = df_work.apply(calculate_theta, axis=1)
    hits_per_layer = df_work.groupby(['event',
'layer']).size().reset_index(name='hits_per_layer')
    avg_hits_per_layer =
hits_per_layer.groupby('event')['hits_per_layer'].mean().reset_index(name='av
g_hits_per_layer')
    max_hits_per_layer =
hits_per_layer.groupby('event')['hits_per_layer'].max().reset_index(name='max
_hits_per_layer')
    df_endcap = df_work[~df_work['isBarrel']].copy()
    df_endcap = df_endcap.groupby('event',
group_keys=False).apply(optimize_theta_std)
    df_barrel = df_work[df_work['isBarrel']].copy()
    df_barrel['theta_optimized'] = np.arctan2(df_barrel['y'], df_barrel['x'])
    df_theta = pd.concat([df_barrel, df_endcap])
    theta_std =
df_theta.groupby('event')['theta_optimized'].std().reset_index(name='theta_st
d')

    result = pd.merge(avg_hits_per_layer, max_hits_per_layer, on='event')
    result = pd.merge(result, theta_std, on='event')

    BDT_dataframes[key] = result

mu_frames = [BDT_dataframes[f'mu{i}'] for i in range(1, 11)]
df_mu = pd.concat(mu_frames, ignore_index=True)

p_frames = [BDT_dataframes[f'p{i}'] for i in range(2, 11)]
df_p = pd.concat(p_frames, ignore_index=True)


df_p["label"] = 0  # Protons
df_mu["label"] = 1  # Muons

df = pd.concat([df_p, df_mu], ignore_index=True)
df = df.sample(frac=1).reset_index(drop=True)

X = df.drop(["event", "label"], axis=1)
y = df["label"]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

model = xgb.XGBClassifier(
    n_estimators=100,
```

```
    max_depth=3,
    objective="binary:logistic",
    eval_metric="logloss",
)
model.fit(X_train, y_train)

accuracy = model.score(X_test, y_test)
print(f"Accuracy: {accuracy:.4f}")

model.save_model("particle_classifier.json")


y_pred = model.predict(X_test)
y_pred_proba = model.predict_proba(X_test)[:, 1]

print("Accuracy:", accuracy_score(y_test, y_pred))
print("Precision:", precision_score(y_test, y_pred))
print("Recall:", recall_score(y_test, y_pred))
print("F1-Score:", f1_score(y_test, y_pred))
print("AUC-ROC:", roc_auc_score(y_test, y_pred_proba))

print("\nClassification Report:")
print(classification_report(y_test, y_pred))


cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6, 6))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
            xticklabels=["Proton", "Muon"],
            yticklabels=["Proton", "Muon"])
plt.xlabel("Predicted")
plt.ylabel("True")
plt.title("Confusion Matrix")
plt.show()


fpr, tpr, thresholds = roc_curve(y_test, y_pred_proba)
roc_auc = auc(fpr, tpr)

plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (AUC =
{roc_auc:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend(loc="lower right")
plt.show()


xgb.plot_importance(model, importance_type='weight')
plt.title("Feature Importance (XGBoost)")
plt.show()
```