КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. И. РАЗЗАКОВА

На правах рукописи

УДК: 004.4’22:004.89(043.3)

Лян Чжаньхао

РАЗРАБОТКА И ИССЛЕДОВАНИЕ КОМПЬЮТЕРНЫХ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ В АВТОМАТИЗИРОВАННЫХ СИСТЕМАХ

Специальность 05.13.06 - Автоматизация и управление технологическими процессами и производствами

Диссертация на соискание учёной степени кандидата наук

Научный руководитель:

доктор технических наук, профессор

Батырканов Жеңиш Исакунович

Бишкек 2025

ОГЛАВЛЕНИЕ

ПЕРЕЧЕНЬ УСЛОВНЫХ ОБОЗНАЧЕНИЙ 4

ВВЕДЕНИЕ 5

ГЛАВА 1. ОБЗОР КОМПЬЮТЕРНЫХ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ В СИСТЕМАХ АВТОМАТИЗАЦИИ 13

1.1. Компьютерные автоматизированные системы..... 13

1.2. Исследование применения автоматизации в области обнаружения падений пожилых людей 17

ВЫВОДЫ 23

ГЛАВА 2. ТЕОРИЯ ОБНАРУЖЕНИЯ ЦЕЛЕЙ И АНАЛИЗА ВРЕМЕННЫХ РЯДОВ НА ОСНОВЕ ГЛУБОКОГО ОБУЧЕНИЯ 25

2.1. Обзор модели методологии обнаружения целей 25

2.2. Разработка и принципы работы семейства алгоритмов YOLO..... 31

2.3. Алгоритм обнаружения целей YOLOv8 40

2.4. Анализ временных рядов с помощью LSTM-моделей 46

ВЫВОДЫ 53

ГЛАВА 3. АЛГОРИТМ ОБНАРУЖЕНИЯ ПАДЕНИЙ НА ОСНОВЕ УЛУЧШЕННОГО YOLOV8 С LSTM 55

3.1. Анализ и ограничения оригинального алгоритма YOLOv8..... 55

3.2. Разработка улучшенной программы 56

3.3. Экспериментальные результаты и анализ 66

ВЫВОДЫ 91

ГЛАВА 4. ИНТЕЛЛЕКТУАЛЬНАЯ СИСТЕМА МОНИТОРИНГА ДЛЯ ОБНАРУЖЕНИЯ ПАДЕНИЙ НА ОСНОВЕ УЛУЧШЕННОГО YOLOV8-LSTM 93

4.1. Системный анализ 93

4.2. Общий дизайн архитектуры системы 99

4.3. Выбор среды разработки программного обеспечения и инструментов 103

4.4. Реализация функциональности системы 111

ВЫВОДЫ 118

ЗАКЛЮЧЕНИЕ 119

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 120

ПРИЛОЖЕНИЕ 136

ПЕРЕЧЕНЬ УСЛОВНЫХ ОБОЗНАЧЕНИЙ

YOLOv8 — Модель обнаружения объектов "You Only Look Once", версия 8

LSTM — Долгосрочная и краткосрочная память (Long Short-Term Memory)

SVM — Метод опорных векторов (Support Vector Machine)

CNN — Сверточная нейронная сеть (Convolutional Neural Network)

Haar — Характеристики Хаара для распознавания объектов (Haar Features for Object Recognition)

HOG — Гистограмма ориентированных градиентов (Histogram of Oriented Gradients)

ECA — Эффективное канальное внимание (Efficient Channel Attention)

GSConv — Групповая сверточная операция (Group Shuffle Convolution)

IoU — Пересечение над объединением (Intersection over Union)

FPN — Сеть пирамид признаков (Feature Pyramid Network)

PAN — Сеть согласования признаков (Path Aggregation Network)

GAP — Глобальный усредненный пуллинг (Global Average Pooling)

URFD — Набор данных UR для обнаружения падений(UR Fall Detection Dataset)

MCFD — Набор данных падений с нескольких камер(Multiple Cameras Fall Dataset)

LFD — Набор данных Le2i для обнаружения падений(Le2i Fall Detection Dataset)

VOC — Формат разметки данных (Visual Object Classes)

ВВЕДЕНИЕ

В нынешнюю стремительно развивающуюся технологическую эпоху технология автоматизации стала важной движущей силой во всех сферах жизнедеятельности. Будь то промышленное производство, медицинский мониторинг или создание "Умных" домов, системы автоматизации играют незаменимую роль. Благодаря интеграции многопрофильных передовых технологий, современные системы автоматизации больше не ограничиваются повторяющимися механизированными операциями, а направлены на более интеллектуальное и адаптивное русло.

Суть системы автоматизации заключается в ее способности автономно выполнять сложные задачи в соответствии с установленными правилами и процедурами. С внедрением искусственного интеллекта, аналитики больших данных и технологий IoT сфера и глубина применения автоматизации значительно расширились. Например, промышленные роботы могут точно выполнять задачи по сборке на производственных линиях, устройства "Умного дома" могут регулировать параметры окружающей среды в зависимости от привычек пользователя, а в области медицины системы мониторинга здоровья обеспечивают точную поддержку данных для профилактики заболеваний и ухода за пациентами.

В последние годы использование автоматизированных технологий в мониторинге здоровья стало особенно значительным, особенно в инновационных исследованиях, направленных на обеспечение безопасности

пожилых людей. Среди пожилых людей падения являются распространенным и серьезным риском, который может привести к серьезным последствиям для здоровья или даже к опасным для жизни ситуациям. Поэтому существует острая необходимость в разработке интеллектуальных систем, способных отслеживать и распознавать случаи падения в режиме реального времени. Это может не только значительно улучшить качество жизни пожилых людей, но и обеспечить своевременную помощь медицинскими работникам в чрезвычайных ситуациях.

Традиционные методы обнаружения падений в основном опираются на сенсорные подходы, такие как носимые устройства или датчики окружающей среды. Однако эти методы имеют множество ограничений в практическом применении, таких как неудобство ношения и высокое влияние помех окружающей среды. С развитием компьютерного зрения и алгоритмов искусственного интеллекта методы обнаружения падений, основанные на глубоком обучении, постепенно превратились в горячую точку исследований. Эти методы не только способны эффективно анализировать поведение человека на основе видеоданных, но и демонстрируют высокую точность обнаружения и устойчивость в сложных условиях.

Учитывая вышеизложенное, данная работа посвящена исследованиям применения технологий автоматизации в области обнаружения падений, путем теоретического анализа и совершенствования алгоритмов искусственного интеллекта, для повышения точности и эффективности обнаружения падений, и разрабатывает комплекс интеллектуальных системы обнаружения и мониторинга

падений, которая задает направление для исследования и применения будущей интеллектуальной системы мониторинга здоровья.

Цель и задачи исследования.

Основная цель диссертации - предложить алгоритм обнаружения падений на основе улучшенных моделей YOLOv8 и LSTM, а также разработать интеллектуальную систему мониторинга как применение технологий компьютерного интеллекта в автоматизированных системах в области обнаружения падений. Сочетание методов обнаружения целей и анализа временных рядов позволяет повысить точность и эффективность обнаружения падений в реальном времени, что обеспечивает эффективную поддержку в предотвращении падений и своевременном вмешательстве.

Задачи исследований:

1. Проанализировать текущее состояние дел в области применения методов обнаружения целей и анализа временных рядов для обнаружения падений, а также обсудить преимущества и проблемы глубокого обучения в этой области;
2. Разработать и совершенствовать алгоритм обнаружения целей YOLOv8, сочетающего механизм внимания ECA и технологию GSConv для повышения точности и эффективности обнаружения целей;
3. Реализовать применение модели LSTM для анализа временных рядов при обнаружении падений, улучшить распознавание поведения при падении путем объединения улучшенного алгоритма YOLOv8 и модели LSTM;

4. Спроектировать и разработать интеллектуальную систему мониторинга для обнаружения падений пожилых людей на основе улучшенной модели YOLOv8-LSTM, завершить функциональный анализ, архитектурный дизайн и реализацию системы, чтобы убедиться, что система оснащена функциями обнаружения, отображения результатов и записи в режиме реального времени;

Новизной научных результатов является:

Инновационный алгоритм, сочетающий YOLOv8 и LSTM: В этом исследовании алгоритм обнаружения целей YOLOv8 улучшен за счет внедрения механизма внимания ECA и модуля GSConv, а также объединен с моделью анализа временных рядов LSTM для оптимизации динамического сценарев и временных характеристик в задаче обнаружения падения, что эффективно повышает точность обнаружения и устойчивость модели к временным изменениям в сложных условиях.

Разработка и внедрение интеллектуальной системы мониторинга для автоматизированных компьютерных систем на основе улучшенной модели YOLOv8-LSTM: Разработка интеллектуальной системы мониторинга с интегрированным обнаружением, отображением в реальном времени и регистрацией данных, обеспечивающих комплексное решение для обнаружения падений пожилых людей в режиме реального времени.

Практическая значимость полученных научных результатов заключается в следующем:

- Разработанная система мониторинга, объединяет передовые технологии обнаружения целей и анализа поведения, которые могут отслеживать ситуацию с падением пожилых людей в режиме реального времени и обеспечивают своевременную обратную связь для создания эффективных механизмов раннего предупреждения для семей и в учреждении по уходу, что может помочь уменьшить травмы, вызванные падениями, и улучшить качество жизни пожилых людей.
- Данное исследование не только оптимизирует существующие методы обнаружения целей, но и создает новую идею интеллектуальной системы мониторинга на основе глубокого обучения, внедрения анализ временных рядов в область мониторинга здоровья, что является важным импульсом для области автоматизированного мониторинга здоровья.
- Результаты исследований предоставляют осуществимое техническое решения для создания интеллектуальной системы ухода за пожилыми людьми, которая имеет широкое социальное значение и рыночные перспективы, и может помочь решить проблемы здравоохранения в стареющем обществе.

Основные положения диссертации, выносимые на защиту:

- Разработанная и усовершенствованный алгоритм обнаружения целей YOLOv8.

- Улучшенный алгоритм YOLOv8 и модели LSTM - YOLOv8-LSTM.
- Разработанная интеллектуальная система мониторинга для обнаружения падений пожилых людей на основе модели YOLOv8-LSTM.

Личный вклад соискателя:

Все научно-технические результаты работы получены диссертантом под руководством научного руководителя.

В опубликованных работах постановка задач и общий подход исследований принадлежат научному руководителю, а оптимизация и улучшение алгоритмических моделей, а также разработка программного обеспечения на их основе выполнены диссертантом.

Апробация результатов диссертации.

Результаты диссертационной работы докладывались на следующих международных симпозиумах, республиканских, межвузовских конференциях:

1. 3-я Международная конференция по компьютерной графике, искусственному интеллекту и обработке данных (ICCAID 2023), Циндао, Китай, 2023 г.
2. 2024 Международная конференция IEEE по обработке изображений (ICIP), Абу-Даби, Объединенные Арабские Эмираты, 2024 г.
3. 4-я Международная конференция по компьютерным технологиям, информационной инженерии и электронным материалам (CTIEEM 2024), Чжэньчжоу, Китай, 2024 г.

Внедрение результатов диссертационных исследований.

Результаты диссертации нашли применение в следующих исследовательских проектах:

1. Проект по улучшению исследовательского потенциала ключевых дисциплин провинции Гуандун по теме "Исследование применения искусственного интеллекта на основе медицинских Больших данных в медицинской визуализации", номер проекта: 2022ZDJS152, Китай.
2. Проект исследования в фонде естественных наук провинции Гуандун - исследование методов распознавания и понимания последовательности действий на основе мультимодального представления реляционного встраивания признаков, январь 2023 - декабрь 2025, Китай.

Полнота отражения результатов диссертации в публикациях.

Основные научные результаты диссертации опубликованы в научных периодических изданиях США («Proceedings of SPIE - The International Society for Optical Engineering», «IEEE International Conference on Image Processing», «Academic Journal of Science and Technology», «Frontiers in Computing and Intelligent Systems»), Нидерландов («iScience»), Алжира («Journal of Electrical Systems»).

Структура и объем диссертации.

Диссертация состоит из введения, четырех глав, приложений и списка литературы. Диссертация состоит из 180 страниц машинописного текста,

включая 29 рисунков и 6 таблиц. Список литературы содержит названия 111 печатных и 12 электронных источников информации.

В первой главе диссертации рассматриваются технологии автоматизации и их применение для обнаружения падений, анализируются ограничения традиционных сенсорных методов и потенциальные преимущества методов искусственного интеллекта.

Во второй главе диссертации представлена теория обнаружения целей и анализа временных рядов на основе глубокого обучения с акцентом на роль YOLOv8 и LSTM в повышении точности обнаружения, робастности и захвата признаков временного ряда.

В третьей главе диссертации предлагается усовершенствованный метод обнаружения падений YOLOv8-LSTM на основе предлагаемой теории, а также проверяется его улучшенный вариант с точки зрения точности, стабильности и производительности в реальном времени с помощью экспериментов.

В четвёртой главе диссертации построен прототип интеллектуальной системы обнаружения падений на основе YOLOv8-LSTM, которая использует пользовательский интерфейс с модуляцией параметров для достижения гибкого и эффективного мониторинга в реальном времени.

Диссертант выражает искреннюю признательность научному руководителю профессору Батырканову Ж.И. за оказанную помощь при выполнении работы.

ГЛАВА 1. ОБЗОР КОМПЬЮТЕРНЫХ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ В СИСТЕМАХ АВТОМАТИЗАЦИИ

1.1. Компьютерные автоматизированные системы

1.1.1. Определение и развитие автоматизированных систем

Система автоматизации - это система, которая позволяет оборудованию или технологическим процессам автономно выполнять определенные задачи в соответствии с заранее установленными правилами и процедурами при меньшем вмешательстве человека или вообще без использования человеческого труда с помощью передовых технологических средств [1,2,3]. Основной целью автоматизации является повышение производительности труда повышении качества продукции при одновременном снижении человеческих ошибок и эксплуатационных расходов [4,5,6,7].

От первоначального простого ручного управления до современной интеллектуальной системы управления, эволюция технологии автоматизации заключается не только в замене ручного управления, но и в том, чтобы сделать систему способной к автономному восприятию, принятию решения и исполнению с помощью современных технических средств и компьютерных технологии. Современные системы автоматизации расширились от выполнения единичных повторяющихся операций до выполнения сложных задач с высокой степенью самооптимизация и решения интеллектуальных задач и широко

используются в различных областях, таких как промышленное производство, сельское хозяйства, в медицине и других [8,9,10].

1.1.2. Основные технологии для современных автоматизированных систем

Реализация систем автоматизации опирается на ряд основных технологий, работающих согласованно. С быстрым развитием компьютерных технологий, технические базы, искусственного интеллекта современные системы автоматизации уже не полагаются исключительно на аппаратные средства, а все больше интегрируют в себя возможности интеллектуальных систем [11,12,13,14]. Ниже перечислены основные подсистемы автоматизации:

Подсистема датчиков и сенсоров: Сенсоры - это "Органы чувств" системы автоматизации, отвечающие за сбор всех видов данных об окружающей среде. Точность и скорость реакции датчиков имеют решающее значение для работы системы автоматизации, что напрямую влияет на возможность реагирования в режиме реального времени и точность системы. К распространенным датчикам относятся датчики температуры, давления, положения, расхода и другие типы, которые широко используются в различных видах автоматизированного управления для обеспечения системы данными в реальном времени, чтобы система могла своевременно вносить коррективы в соответствии с изменениями окружающей среды [15,16].

Подсистема регулирования и управления: Система управления - это мозг системы автоматизации, который обрабатывает данные, поступающие от

датчиков, и генерирует команды управления в соответствии с поставленными задачами управления и скорость реакции системы [17,18,19].

Подсистема приводов: Приводы - это "Исполнители" в системе автоматизации, выполняющие физические действия в соответствии с инструкциями системы управления. К распространенным исполнительным механизмам относятся электродвигатели, пневматические цилиндры, гидроцилиндры и т. д., которые широко используются для механических перемещений, точных операций и других задач. Точность, время отклика и стабильность работы исполнительного механизма имеют решающее значение для общей производительности системы автоматизации и напрямую влияют на эффективность и качество выполнения задачи [20,21,22].

Современные системы автоматизации должны обладать интеллектуальными свойствами как при сборе информации, так и при принятии управленческих решений.

Коммуникационное подсистемы: системы автоматизации обычно состоят из множества подсистем, для обмена информацией между которыми требуются эффективные коммуникационные технологии. Современные системы автоматизации используют такие технологии, как беспроводная связь, оптоволоконная связь и промышленный Ethernet для поддержки более сложных сетевых архитектур и дистанционного управления. Быстрое развитие этих коммуникационных технологий делает интеграцию и расширение системы более

удобными, а также обеспечивает базовую поддержку для приложений "Умного производства" и "Интернета вещей" (IoT) [23,24,25,26].

Подсистемы программного обеспечения: Программное обеспечение – это основной момент с применением компьютерных технологии [27,28,29].

1.1.3. Уровни и эволюция систем автоматизации

Системы автоматизации прошли несколько этапов технологической эволюции - от первоначальных механизированных и электронных до современных интеллектуальных - путь, полный инноваций и перемен [30,31,32,33]. Иерархическое деление систем автоматизации может быть произведено с точки зрения сложности управления и интеграции технологий и включает следующие этапы:

Базовая автоматизация: системы автоматизации на этом этапе нацелены на отдельные машины или простые процессы и обычно используются для замены ручного труда при выполнении повторяющихся, простых рабочих задач. Примерами могут служить начальные операции на станках, розлив жидкостей и т. д.

Автоматизация процессов: Автоматизация процессов в основном используется в отраслях, требующих непрерывного управления производством, таких как химическая, нефтяная и электроэнергетическая. Этот тип систем содержит более сложное управление процессом, направленное на повышение стабильности, эффективности и безопасности производства, оптимизацию производственного процесса и улучшение качества продукции.

Автоматизация производства: на производстве системы автоматизации охватывают не только работу производственных линий, но и обработку материалов, сборку и контроль. Интеллектуальное производство и гибкие производственные системы (FMS) становятся основными на этом этапе, повышая гибкость и точность производственного процесса за счет более высокого уровня интеграции и взаимодействия.

Интеллектуальная автоматизация: интеллектуальные системы автоматизации объединяют передовые технологии, такие как искусственный интеллект, анализ больших данных и интернет вещей, и обладают способностью к самообучению, предиктивному обслуживанию и оптимизации системы. Благодаря автономному принятию решений и адаптивному управлению интеллектуальная автоматизация способна корректировать стратегии работы в реальном времени в зависимости от изменений окружающей среды, что значительно повышает адаптивность и автономность системы.

Технологическая эволюция систем автоматизации прошла путь от простых механических операций до интеллектуального управления, причем каждый этап закладывает основу для следующего этапа технологического прогресса и способствует эффективному развитию всех сфер жизни общества.

1.2. Исследование применения автоматизации в области обнаружения падений пожилых людей

С быстрым развитием технологий автоматизации, особенно с постоянным совершенствованием компьютерных интеллектуальных систем, растет и их

применение в области мониторинга здоровья и личной безопасности. Особенно для групп повышенного риска, таких как пожилые люди, эффективное применение технологии мониторинга здоровья может значительно улучшить качество их жизни и безопасность [34,35,36,37].

Среди пожилых людей механическое падения являются одним из наиболее распространенных и серьезных рисков для их безопасности. Падения могут привести не только к серьезным физическим травмам, но и к другим проблемам со здоровьем и даже к опасным для жизни ситуациям. Поэтому разработка эффективных систем обнаружения падений, позволяющих своевременно выявлять и предупреждать о них, стала одним из приоритетных направлений исследования [38,39,40,41].

На первых порах, обнаружение падений в значительной степени опиралось на сенсорные технологии. Эти технологии обычно включали в себя носимые датчики (например, акселерометры, гироскопы, датчики давления и т. д.) и датчики окружающей среды (например, инфракрасные, акустические датчики и т. д.), которые определяли, произошло ли падение, путем сбора физических параметров [42,43]. В последние годы, с быстрым развитием технологий искусственного интеллекта, особенно глубокого обучения и технологий компьютерного зрения, методы обнаружения падения на основе зрения постепенно становятся основным направлением исследований [44,45].

1.2.1. области применения и ограничения традиционных сенсорных технологий

Сенсорные методы обнаружения падений начали применяться довольно рано, и тот факт, что они не зависят от высокой вычислительной мощности или сложного аппаратного обеспечения, сделал эти технологии широко используемыми уже в первые годы. Сенсорные технологии можно разделить на две категории: датчики окружающей среды и носимые датчики.

Датчики окружающей среды: в этом методе датчики устанавливаются в фиксированной среде, например, на потолке, стене или на полу, чтобы отслеживать изменения в окружающей среде. Когда происходит падение, датчики ощущают необычные физические изменения, такие как вибрация пола, изменения ускорения тела или сигнатуры акустических волн [46]. Основная проблема датчиков окружающей среды заключается в том, что они негибкие, обычно работают только в фиксированных условиях и имеют низкую точность из-за окружающего шума или ложных тревог.

Носимые датчики: этот метод отслеживает движение человеческого тела в режиме реального времени с помощью носимых датчиков. Акселерометры и гироскопы - распространенные типы датчиков, которые отслеживают ускорение и вращательные движения человеческого тела [47]. Хотя носимые датчики не ограничены пространством, их недостатки заключаются в том, что их нужно носить в течение длительного времени.

Хотя сенсорные методы обеспечивают лучшую производительность в реальном времени и более низкую стоимость, они страдают от нескольких недостатков: во-первых, они подвержены помехам от внешних факторов (например, фоновый шум, падающие предметы и т. д.), что приводит к ложным или пропущенным тревогам; во-вторых, многие датчики необходимо носить на теле, что связано с проблемой низкого комфорта; и, в-третьих, датчики окружающей среды не обладают достаточной гибкостью для обеспечения широкого охвата. В результате этих проблем исследования в области сенсорных методов постепенно переходят к интеграции интеллектуальных компьютерных алгоритмов.

1.2.2 Методы обнаружения, основанные на алгоритмах искусственного интеллекта

В отличие от традиционных методов обнаружения с помощью датчиков, алгоритмы искусственного интеллекта способны точно определять поведение при падении с помощью визуальной информации и анализа временных данных, а также демонстрируют высокую устойчивость, особенно в сложных условиях.

Компьютерное зрение стало одним из наиболее широко используемых методов обнаружения падений путем анализа поведения человека на основе изображений или видеоданных. Используя камеру или другое устройство технического зрения, алгоритмы могут извлекать особенности человека из видео и идентифицировать события, связанные с падением. К числу распространенных

методов относятся алгоритмы обнаружения целей на основе конволюционной нейронной сети (CNN) и методы распознавания поз человека.

Например, некоторые исследователи используют конволюционные нейронные сети (CNN) в сочетании с моделями глубокого обучения для оценки позы человека и распознавания действий для точного обнаружения падений. Эти методы могут обрабатывать несколько кадров изображений и данные о различных точках обзора, что, в свою очередь, обеспечивает более точные результаты обнаружения [48,49,50,51]. Есть также некоторые ученые, которые могут точно извлекать скелетную структуру человеческого тела через обнаружение ключевых точек человека (например, OpenPose, PoseNet и т. д.) и определять, произошло ли падение, анализируя движение ключевых точек [52]. Метод обнаружения ключевых точек позволяет эффективно удалять фоновый шум и повышать надежность обнаружения. По сравнению с традиционными методами обнаружения целей, методы обнаружения ключевых точек могут не только уменьшить зависимость от окружающего освещения и фона, но и более точно улавливать ключевые состояния движения человеческого тела. Например, алгоритм OpenPose может точно анализировать и классифицировать движения человека, обнаруживая ключевые точки человеческого тела (например, плечи, колени и т. д.) и комбинируя их с методами глубокого обучения [53]. Такие методы широко используются в области обнаружения падений, особенно в динамических средах, с более высокой точностью и надежностью, чем

традиционные методы, основанные на различии фона или обнаружении движения [54,55].

1.2.3 Сильные стороны и проблемы подхода, основанного на искусственном интеллекте

Метод обнаружения падения на основе искусственного интеллекта имеет следующие преимущества перед традиционными сенсорными методами:

Бесконтактность и высокая точность: алгоритмы искусственного интеллекта обеспечивают бесконтактный мониторинг с помощью видеоустройств без необходимости надевать оборудование, что позволяет избежать неудобств, связанных с ношением или повреждением оборудования. Кроме того, алгоритмы глубокого обучения и компьютерного зрения обеспечивают высокую точность обнаружения и устойчивость, особенно в сложных условиях с высокой адаптивностью.

Обработка данных в реальном времени и интеллектуальное принятие решений: алгоритмы искусственного интеллекта способны обрабатывать данные в реальном времени и автоматически корректировать решения в зависимости от изменений окружающей среды, что делает систему адаптивной в динамичной среде. Благодаря постоянному обучению и оптимизации способность системы к обнаружению будет улучшаться по мере использования.

Объединение многомерных данных: технологии ИИ способны обрабатывать множество источников данных, таких как видеоданные, данные об окружающей

среде и данные датчиков, и объединять эту информацию для многоуровневого анализа, чтобы повысить точность обнаружения падений.

Несмотря на то, что подходы ИИ продемонстрировали значительные преимущества в обнаружении падений, все еще существуют некоторые проблемы, особенно в следующих областях:

Точность и эффективность обнаружения: хотя методы обнаружения на основе ИИ значительно превосходят традиционные методы, точность и эффективность обнаружения целей в некоторых сложных сценах (например, в многопользовательских сценах, в условиях недостаточной освещенности и т. д.) все еще сталкиваются с определенными проблемами [56,57].

Требования к вычислительным ресурсам: алгоритмы глубокого обучения и компьютерного зрения обычно требуют высокой вычислительной мощности, особенно в приложениях реального времени. Оптимизация алгоритмов для эффективной работы на маломощных устройствах - одно из направлений будущих исследований [58,59].

ВЫВОДЫ

Проанализировав автоматизированную систему и ее применение для обнаружения падений, можно сделать следующие выводы:

Развитие технологии автоматизации: технология автоматизации прошла путь от простого механизированного управления до высокоинтеллектуальных систем, которые широко используются в различных отраслях промышленности и значительно повышают эффективность и точность производства.

Ограничения сенсорных подходов: традиционное обнаружение падений опирается на сенсорные технологии (например, датчики окружающей среды и носимые датчики), которые, несмотря на лучшую производительность в реальном времени, страдают от недостаточной гибкости и высокой подверженности помехам.

На сегодняшний день искусственный интеллект с применением компьютерного зрения и глубокого обучения, стал основным методом обнаружения падений с высокой точностью и надежностью. Однако он по-прежнему сталкивается с проблемами точности и реального времени и нуждается в дальнейшей оптимизации.

Обнаружение падений - это, по сути, проблема обнаружения цели. В диссертации рассматривается теория обнаружения цели и анализ временных рядов на основе глубокого обучения, обеспечить точность и эффективность обнаружения падений человека.

ГЛАВА 2. ТЕОРИЯ ОБНАРУЖЕНИЯ ЦЕЛЕЙ И АНАЛИЗА ВРЕМЕННЫХ РЯДОВ НА ОСНОВЕ ГЛУБОКОГО ОБУЧЕНИЯ

2.1. Обзор модели методологии обнаружения целей

Обнаружение целей - это основной и ключевой вопрос в области компьютерного зрения, направленная на идентификацию и определение местоположения целевых объектов на изображениях или видео. Ее основные задачи включают два аспекта: первый - определение категории целевого объекта, а второй - определение положения целевого объекта на изображении, которое обычно представлено ограничивающей рамкой. С быстрым развитием технологии глубокого обучения методы обнаружения целей эволюционировали и получили широкое применение во многих областях, таких как охранное наблюдение, автоматическое вождение, медицинская визуализация, "Умный дом" и т.д. [60,61,62].

Методы обнаружения целей можно разделить на традиционные методы, основанные на извлечении признаков, и современные методы, основанные на глубоком обучении. Несмотря на то, что традиционные методы были успешны в определенных сценариях применения, из-за их зависимости от вручную созданных признаков и плохой адаптации к сложным сценариям, все больше исследований переходит к методам глубокого обучения, которые демонстрируют значительные преимущества, особенно при работе с динамически изменяющимися и многомасштабными целями. В этом разделе мы представим

обзор традиционных методов обнаружения целей и методов на основе глубокого обучения, уделяя особое внимание ключевым технологиям и истории развития методов глубокого обучения.

2.1.1. Традиционные методы обнаружения целей

До широкого применения методов глубокого обучения исследования в области обнаружения целей были сосредоточены на традиционных методах компьютерного зрения. Большинство этих методов опирались на алгоритмы ручного извлечения признаков для обнаружения целей путем сопоставления признаков в сочетании с классификаторами.

1. Метод обнаружения на основе признаков Хаара (Haar)

Метод признаков Хаара - один из самых ранних методов обнаружения целей на основе признаков, а наиболее известное его применение - система обнаружения лиц, предложенная Виолой и Джонсом [61]. Признаки Хаара извлекаются путем вычисления разности яркости в локальной области изображения, и эти признаки классифицируются каскадными классификаторами. Несмотря на то что этот метод позволил добиться значительных результатов в раннем обнаружении лиц, он имеет серьезные ограничения при работе со сложными ситуациями, такими как изменение освещения, различные позы и окклюзии.

2. Метод обнаружения на основе признаков (HOG)

HOG-функции - еще один распространенный традиционный метод обнаружения целей, который описывает характеристики краев и формы цели

путем подсчета направления градиента в локальной области. HOG-методы широко используются в задачах обнаружения пешеходов, особенно при обнаружении пешеходов в сочетании с машинами опорных векторов (SVM) для достижения хороших результатов. Однако метод HOG менее эффективен при работе с крупномасштабными вариациями и сложными фонами, а вычисление признаков занимает много времени, что затрудняет его использование в задачах обнаружения в реальном времени [62].

Хотя традиционные методы успешно справляются с некоторыми специфическими задачами, они в основном опираются на разработанные вручную признаки и хуже адаптируются к меняющимся целям и сложным сценам. С развитием глубокого обучения методы обнаружения целей, основанные на глубоком обучении, постепенно становятся основным направлением исследований и демонстрируют более высокую точность и устойчивость в практических приложениях.

2.1.2. Подходы на основе глубокого обучения

Исходя из проблем традиционных методов обнаружения, методы обнаружения целей на основе глубокого обучения достигли значительного прогресса в последние годы с появлением моделей глубокого обучения, таких как конволюционные нейронные сети (CNN). Методы глубокого обучения могут автоматически изучать особенности изображения на основе большого количества данных, что позволяет избежать утомительной ручной разработки особенностей, а также лучше адаптироваться к сложным сценам и изменчивым

целям. Благодаря постоянному развитию технологии глубокого обучения классификаторы, основанные на методах глубокого обучения, постепенно демонстрируют высокую производительность. Как показано на рисунке 2.1., с 2012 года AlexNet [63]и далее методы, основанные на глубоком обучении, начали появляться в больших масштабах, однако AlexNet, Overfeat[64],R-CNN 错误!未找到引用源。 и R-CNN [65] все еще придерживаются подхода конволюционной нейронной сети к извлечению признаков, который, по сути, является стратегией скользящего окна, что все еще влияет на вычисления и производительность алгоритмов в реальном времени. До появления Fast-RCNN [66], который стал первым методом, заменившим традиционный метод скользящего окна на метод ящика-кандидата области, а также ознаменовал официальное появление двухэтапного алгоритма обнаружения целей. Сразу после появления YOLO [67,68,69] и других серий алгоритмов, эта серия алгоритмов использует одноэтапный подход прямой регрессии для замены оригинальной стратегии извлечения кадров-кандидатов регионов, что знаменует собой создание одноэтапных алгоритмов обнаружения.

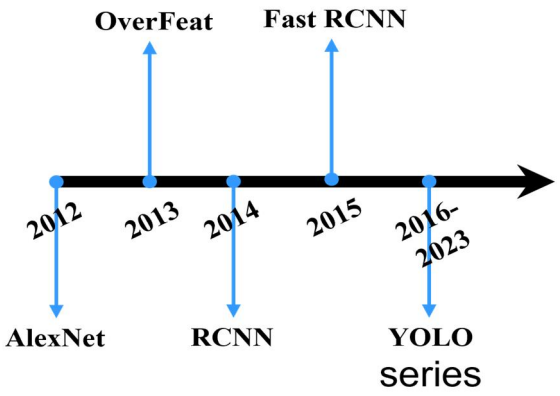


Рисунок.2.1 Разработка алгаритмов обнаружения целей на основе глубокого обучения

Двухэтапный метод обнаружения

Первый этап двухэтапного метода обнаружения цели заключается в точном определении положения цели и поиске лучших кадров-кандидатов на основе обеспечения высокой запоминаемости и точности; второй этап классифицирует кадры-кандидаты, полученные на первом этапе, для более точной локализации цели. Двухфазный подход, естественно, требует большого количества вычислений, что приводит к низкой скорости обнаружения, но жертвуя скоростью, можно значительно повысить точность обнаружения. Основными репрезентативными алгоритмами являются R-CNN, Fast-RCNN. Faster-RCNN [70]. Структура показана на рисунке 2.2., сначала входное изображение проходит операцию изменения размера для получения$H \times W$ Размер; подается в общую опорную сеть конволюционного слоя для извлечения признаков; используется сеть RPN (Region Proposal Networks) для генерации боксов-кандидатов, и после получения матрицы признаков путем сопоставления исходного изображения сгенерированного бокса-кандидата с изображением признаков, подается в softmax и bbxreg (bounding box regression) соответственно для получения более точного бокса-кандидата. Слой объединения ROI (Region of Interst) понижает выборку матрицы признаков в карту признаков 7×7, объединяет контекстную информацию для извлечения боксов-кандидатов и карт признаков, а затем отправляет их обратно в полностью связанный слой для классификации; наконец,

он вычисляет вероятность классифицированных боксов в полностью связанном слое, а затем получает более точную информацию о местоположении и результаты классификации с помощью регрессии ограничивающих боксов.

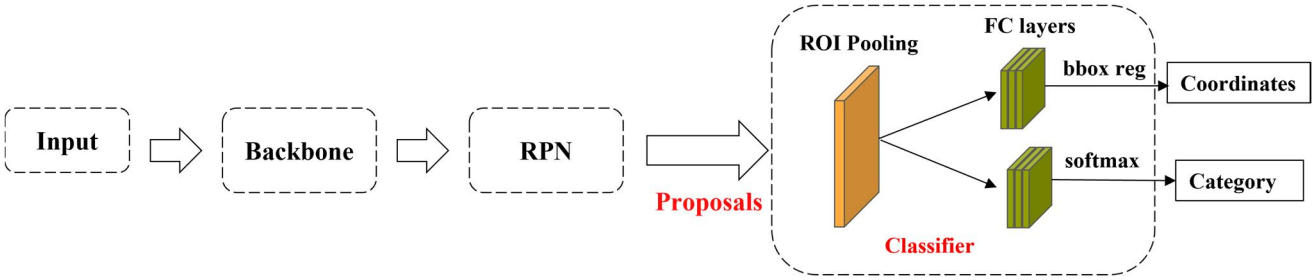


Рисунок 2.2. Структура Faster-RCNN

Одноэтапные методы обнаружения

Одноэтапные методы отбрасывают этап поиска лучшего ящика-кандидата, одноразовая прямая регрессия местоположения и класса целевого объекта, через обнаружение можно напрямую вернуться к результатам, такие методы улучшают скорость обнаружения, жертвуя точностью обнаружения, подходят для широкого спектра промышленных развертываний приложений, и поэтому их ищут многие исследователи, основные представительные алгоритмы для YOLO 错误!未找到引用源。 серия алгоритмов [71] .

Рисунок 2.3. Одноэтапная сеть обнаружения целей

Одноэтапная сеть обнаружения целей, которая отказывается от процесса извлечения кадров-кандидатов в двухэтапном подходе и напрямую регрессирует позиции категории и кадров-кандидатов сразу, жертвует точностью обнаружения, но имеет более высокую скорость обнаружения, чем в литературе [72] Она

улучшается почти в десять раз и открывает новую эру сквозного обнаружения целей в реальном времени. Поток обнаружения этой сети показан на рисунке 2.3 [67] Сначала на вход поступает изображение, затем запускается одна CNN для извлечения признаков, и, наконец, выводится уровень доверия с помощью порогового значения, чтобы отбросить результаты с высокой ошибкой. Алгоритм использует глобальный вывод для повышения эффективности вычислений путем объединения изображений в фиксированную сетку.

2.2. Разработка и принципы работы семейства алгоритмов YOLO

2.2.1. Основные идеи и принципы YOLO

Основная идея YOLO заключается в том, чтобы рассматривать задачу обнаружения целей как задачу регрессии, т. е. регрессировать классы и местоположение целей непосредственно по входному изображению. В частности, алгоритм YOLO делит изображение на сетки, и каждая сетка отвечает за предсказание содержащихся в ней целей. Если сетка содержит цель, YOLO не только предсказывает ее местоположение (по координатам ограничительного поля), но и выводит категорию, к которой относится цель, и балл доверия.

Основной процесс YOLO заключается в следующем:

Разделение сетки: сначала YOLO делит входное изображение на сетки S × S. Каждая сетка отвечает за предсказание целей, которые она содержит. Сетка отвечает за предсказание цели, если ее центральная точка находится в пределах этой сетки.

Регрессия граничных ящиков: для каждой сетки предсказывается фиксированное количество граничных ящиков (обычно 2 или более). Каждая граница определяется четырьмя параметрами: координатами центра границы (x, the$y$), шириной (w) и высотой (h).

Предсказание категорий: каждая сетка предсказывает распределение вероятности категорий для каждого ограничительного поля, т.е. категории объектов, которые могут содержаться в этом ограничительном поле (например, пешеходы, автомобили, кошки и т.д.).

Предсказание уверенности: для каждого ограничительного поля также предсказывается значение уверенности, указывающее на вероятность того, что поле действительно содержит объект, и измеряется точность локализации этого ограничительного поля.

Таким образом, YOLO может выдавать все предсказания непосредственно в процессе прямого вывода, что значительно повышает скорость обнаружения.

2.2.2. Эволюция и сравнение моделей семейства YOLO

Серия YOLO (You Only Look Once) является одним из наиболее широко используемых алгоритмов в области обнаружения целей в последние годы. Ее основная концепция заключается в преобразовании задачи обнаружения цели в задачу регрессии, которая может одновременно классифицировать и определять местоположение цели посредством одного прямого распространения. С тех пор как была впервые предложена версия YOLOv1, серия алгоритмов YOLO продолжает развиваться и эволюционировать, постепенно решая различные

проблемы, возникшие в предыдущих версиях, такие как точность обнаружения мелких объектов, скорость обнаружения и т. д. В таблице приведены алгоритмы серии YOLO. В таблице 2.1. приведена информация о развитии моделей серии YOLO от YOLOv1 до YOLOv8 и их основных характеристиках.

Таблица 2.1. - Таблица истории YOLOv1 - YOLOv8

выпускает	Год выпуска	сетевая инфраструктура	Особенности и инновации	Точность и скорость	Основные проблемы/затруднения
YOLOv1	2016	На основе GoogLeNet	Впервые обнаружение цели предлагается как регрессионная задача с одним прямым распространением для предсказания категорий и	Менее точный, но очень быстрый.	Плохое обнаружение мелких объектов и недостаточная точность регрессии ограничительных рамок.

			ограничительных рамок.		
YOLO v2	2017	Даркнет-19	Были введены якорные боксы, улучшающие предсказание границ и использующие более глубокие сетевые структуры (Darknet-19).	Повышенная точность и относительно высокая скорость.	Мелкие объекты по-прежнему плохо обнаруживаются, а для очень плотных сцен с мишенями производительность посредственная.
YOLO v3	2018	Даркнет-53	Представлено многомасштабное предсказани	Повышение точности и скорости балансировки.	При стрельбе по слишком маленьким мишеням все еще

			е, которое использует несколько конволюционных слоев для вывода результатов обнаружения на разных масштабах.		наблюдается некоторая потеря точности.
YOLO v4	2020	CSPDarknet53	Внедрение дополнительных методов оптимизации (например, функции активации Миша, CSPDarknet53) и	Сочетание высокой точности и высокой скорости делает его пригодным для крупномасштабных применений.	Процесс обучения и вывода является сложным, а модель - большой.

			методов улучшения данных для поддержки аппаратного ускорения.		
YOLO v5	2020	Индивидуальная архитектура сети (Darknet)	Предлагается более гибкий архитектурный дизайн с возможностью выбора моделей разных размеров (YOLOv5s, YOLOv5m, YOLOv5l).	Быстрее и подходит для развертывания на различных платформах.	Не будучи опубликованными авторами YOLO, некоторые исследователи поставили под сомнение оригинальность их алгоритмов.
YOLO v6	2022	Индивидуальная сетевая архитектура	Оптимизирована для промышлен	Отличная производительность при	Адаптация к устройствам с низкими

			ных приложений и потокового видео в реальном времени, чтобы еще больше увеличить скорость вывода и эффективность вычислений, а также улучшить обнаружение мелких объектов.	обнаружении в режиме реального времени.	ресурсами иногда может быть ограничена.

YOLO v7	2022	Индивидуальная сетевая архитектура	Внедрение технологии iCNN и динамической функции потерь повышает скорость вывода и кросс-платформенную применимость модели.	Больше подходит для приложений с низкой задержкой и повышенной точностью.	В некоторых высокоточных приложениях он может немного уступать YOLOv4.
YOLO v8	2023	Усовершенствованная архитектура адаптивной оптимизации	Интеграция адаптивной оптимизации сети с обучением с подкреплением еще больше	Точность повышается еще больше, а скорость увеличивается.	При работе со сложными сценами еще есть возможности для улучшения.

			улучшает баланс между точностью, скоростью и сжатием модели.		

Как показано в таблице 2.1., семейство моделей YOLO претерпело значительное развитие с версии 1 по версию 8. В YOLOv1 впервые предложена одноступенчатая регрессионная структура, которая улучшает скорость обнаружения, но точность и обнаружение мелких объектов остаются слабыми.YOLOv2 вводит якорные блоки и более глубокую сеть (Darknet-19), что улучшает точность, особенно при обнаружении средних и крупных объектов.YOLOv3 еще больше улучшает эффективность обнаружения мелких объектов за счет многомасштабного прогнозирования, но при этом сохраняется потеря точности в YOLOv4 повышает точность и скорость и особенно подходит для промышленных приложений, но модель велика и сложна для обучения.YOLOv5 оптимизирует архитектуру и обеспечивает лучший баланс скорости и точности для широкого спектра устройств.YOLOv6 и YOLOv7 дополнительно оптимизируют скорость и точность вывода, и особенно подходят для приложений реального времени и низких задержек.YOLOv8 обеспечивает дальнейшие улучшения в различных аспектах. YOLOv8 обеспечивает

дальнейшие улучшения в различных аспектах для дальнейшего повышения точности обнаружения. В целом, семейство моделей YOLO эволюционировало в плане точности, скорости и сценариев применения, причем каждое поколение оптимизировало предыдущую версию, что способствовало быстрому развитию технологии обнаружения целей. Однако, несмотря на то, что модели серии YOLO хорошо работают в режиме реального времени и в большинстве распространенных сценариев, при работе со сложными сценариями и обнаружении мелких объектов все еще есть возможности для улучшения [74,75,76,78].

2.3. Алгоритм обнаружения целей YOLOv8

YOLOv8 - модель компьютерного зрения, выпущенная в январе 2023 года компанией Ultralytics, разработавшей YOLOv5. Она превосходит предыдущие версии, такие как YOLOv5 и YOLOv7 [58], по точности и гибкости. YOLOv8 официально классифицируется на пять масштабных версий, YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l и YOLOv8x, в соответствии с параметрами коэффициента глубины и коэффициента ширины модели.YOLOv8 поддерживает широкий спектр визуальных YOLOv8 поддерживает широкий спектр визуальных задач, таких как обнаружение объектов, сегментация, оценка положения, отслеживание и классификация [79,80].

2.3.1. Структура сети

YOLOv8 состоит из трех основных ядер, а именно Backbone, Neck и Head. Основные улучшения включают в себя настройку структуры сети, модуль C2f,

метод обнаружения кадров без якорей, оптимизацию функции потерь и т. д. Схема структуры сети YOLOv8 показана на рисунке 2.4.

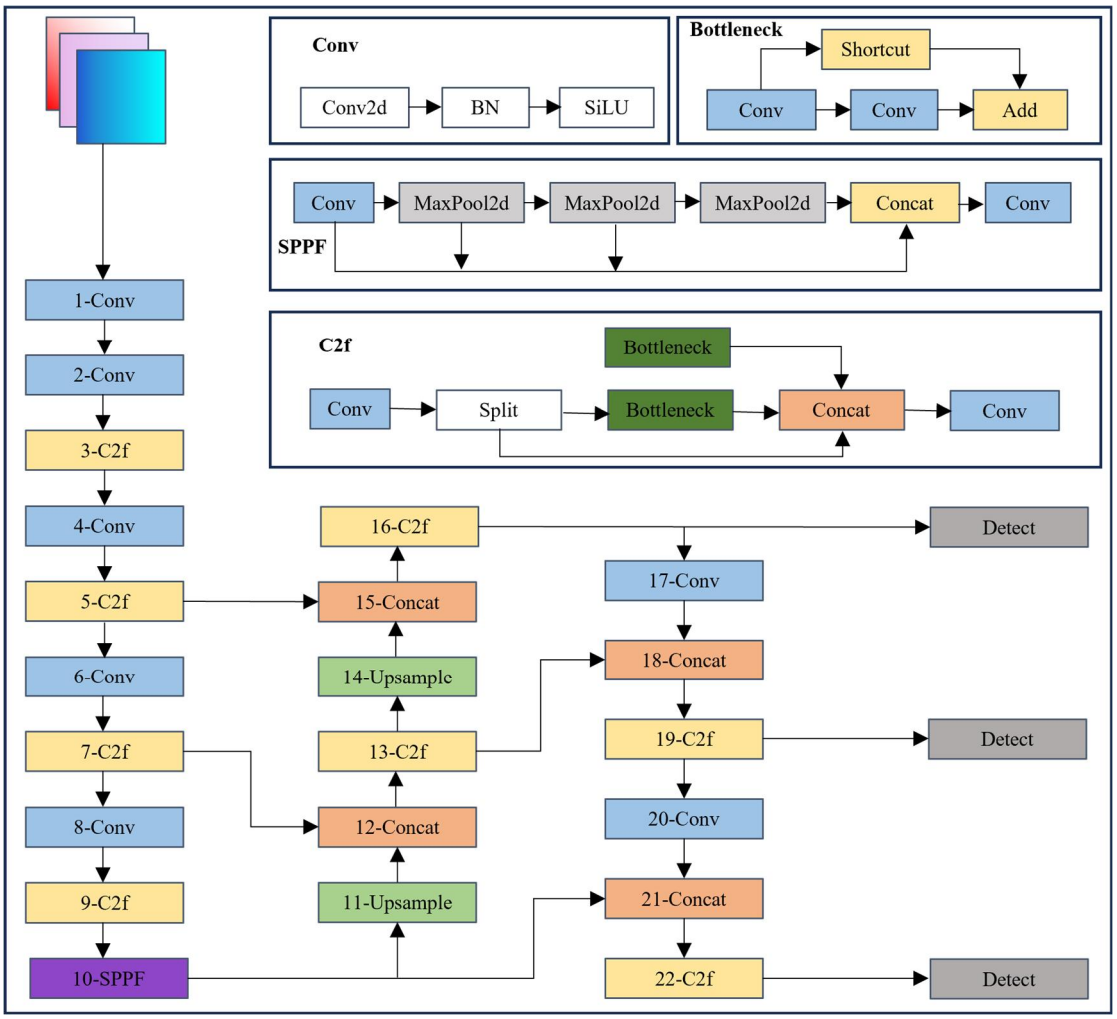


Рисунок 2.4. Структурная схема сети Yolov8

YOLOv8, как последний представитель семейства алгоритмов обнаружения целей YOLO, унаследовал и оптимизировал особенности предыдущих версий. YOLOv8 имеет схожую структуру магистральной сети с YOLOv5. На входе YOLOv8, как и YOLOv5, использует мозаичное улучшение данных для объединения четырех изображений для получения богатой информации для обучения модели. Что касается структуры опорной сети, то, как видно из файла конфигурации исходного кода, YOLOv8 и YOLOv5 имеют примерно одинаковую

архитектуру без учета Head, но с некоторыми улучшениями. Первый сверточный слой 6×6 заменен сверточным слоем 3×3, а все модули C3 в сети YOLOv5 заменены модулями C2f, которые являются узкими модулями, содержащими два сверточных слоя, и имеют больше связей между слоями, чем остаточные связи традиционной сети, что не только оптимизирует процесс обучения и улучшает градиентный поток, но и обогащает признаки, что может повысить точность обнаружения. После извлечения признаков из входного изображения в опорной сети можно получить три эффективных слоя признаков с различными размерами масштаба, которые будут переданы в сеть Neck для дальнейшей обработки. В части Neck YOLOv8 продолжает структуру FPN (Feature Pyramid Networks, FPN) [59] и PAN YOLOv5. Однако YOLOv8 исключает два конволюционных слоя и вводит модуль C2f в шейную часть, что еще больше повышает способность сети к объединению признаков. Три эффективных слоя признаков, полученных из опорной сети, проходят через структуры FPN и PAN, что позволяет добиться не только слияния признаков с понижением дискретизации, но и слияния признаков с повышением дискретизации - процесса, эффективно объединяющего информацию о признаках в разных масштабах. Три эффективных слоя признаков, полученных от опорной сети, доступны для выполнения функции Head после достижения межслойного слияния признаков на этапе Neck. Подход с использованием предопределенных предварительных кадров может снизить скорость обучения модели на индивидуальном наборе данных. YOLOv5 использует три выходных Головы для работы с объектами разных размеров, а

YOLOv8 использует архитектуру с одной выходной Головой, которая отказывается от традиционных малых, средних и больших опорных кадров и использует неанкерную модель с разделенными Головами для независимого решения задач обнаружения объектов, классификации и регрессии. Такая конструкция позволяет каждой ветви сосредоточиться на своей задаче, а также напрямую предсказывать центр объекта, а не предсказывать смещение известной опорной рамки, что позволяет сократить количество предсказаний ограничительной рамки и, таким образом, ускорить этап подавления не максимальных значений [60].

2.3.2. Динамический механизм обнаружения без якоря

YOLOv8 предлагает динамический механизм обнаружения без якорей, основанный на традиционном подходе, основанном на якорях. Традиционные модели серии YOLO (например, YOLOv3, YOLOv4, YOLOv5) обычно используют фиксированные якоря для предсказания положения цели, которые устанавливаются вручную и не всегда позволяют достичь наилучших результатов при столкновении с целями различных размеров и форм. Механизм YOLOv8 без якорей, напротив, позволяет модели более гибко адаптироваться к различным масштабам, формам и позиционным вариациям цели, динамически генерируя позиции целевых блоков.

В YOLOv8 механизм обнаружения напрямую предсказывает положение центра и размер ограничительного поля путем регрессии вероятности существования цели и соответствующего положения каждой точки пикселя.

Такой подход позволяет избежать зависимости от множества якорных ящиков в традиционных методах, снизить вычислительную сложность модели и может быть использован в различных приложениях для электронных устройств. Между тем, при работе со сценариями с несколькими целями механизм YOLOv8 без привязки может значительно повысить точность обнаружения целей и снизить процент ложных обнаружений.

2.3.3. Функция потерь

Что касается функции потерь, YOLOv8 использует бинарную перекрестную энтропию в качестве потери классификации, так как в YOLOv8 нет потери объекта, он может напрямую выводить уровень доверия для каждой категории, а затем найти максимальное значение в качестве уровня доверия коробки, бинарная функция потерь перекрестной энтропии показана в (2.1).

$$BCE = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\widehat{y_i}) + (1 - y_i)\log(1 - \widehat{y_i})] \quad (2.1)$$

где N размер выборки, а y_i число образцов в i истинная метка первого образца, и $\widehat{y_i}$ вероятность того, что модель предсказывает первую i вероятность того, что первая выборка является положительным классом, и log натуральный логарифм. YOLOv8 справляется с потерями в граничном поле в основном с помощью функций $CIoU$ и DFL (Distribution Focal Loss) функции потерь. $CIoU$ В отличие от IoU Функция потерь учитывает соотношение между горизонтальными и вертикальными коэффициентами боксов, чтобы лучше подогнать их к цели, где $CIoU$ Формула показана в (2.2) ниже.

$$CIoU = IoU - \frac{\rho^2(b,b^{gt})}{c^2} - \alpha v \quad (2.2)$$

гдеIoU обозначает отношение пересечения и совпадения предсказанного кадра с истинным кадром, и$\rho 2(b, bgt)$ обозначает коэффициент пересечения предсказанного кадра b евклидово расстояние между центром кадра предсказания и реальным кадромbgt евклидово расстояние между центрами кадра предсказания и истинного кадра, и c длина диагонали, содержащей наименьшую замкнутую область кадра предсказания и истинного кадра, иα параметр компромисса, связанный с IoU ассоциированный параметр компромисса, формула которого приведена в (2.3).

$$\alpha = \frac{v}{1 - IoU + v} \quad (2.3)$$

Где. v обозначает согласованность соотношения сторон между предсказанными и реальными кадрами, как показано в выражении (2.4).wgt иhgt обозначают ширину и высоту реального кадра, соответственно.w иh обозначают ширину и высоту предсказанного кадра.

$$v = \frac{4}{\pi^2}\left(\arctan\ \frac{w^{gt}}{h^{gt}} - \arctan\ \frac{w}{h}\right)^2 \quad (2.4)$$

$CIoU$ является усовершенствованием традиционного IoU Улучшение по сравнению с традиционным, которое учитывает не только области перекрытия между ограничительными рамками, но и расстояние между центрами

ограничительных рамок и соотношение сторон ограничительных рамок, обеспечивая тем самым более полную оценку сходства.

2.4. Анализ временных рядов с помощью LSTM-моделей

2.4.1. Свойства временных рядов при обнаружении падения

Характеристики данных временных рядов играют решающую роль в задачах обнаружения падений. Событие падения обычно представляет собой внезапное изменение в течение короткого периода времени, сопровождающееся значительными изменениями ускорения, резкими колебаниями углов ориентации и быстрыми переходами из одного состояния в другое. Эти особенности делают поведение при падении явно динамичным во времени. Хотя традиционные данные датчиков (например, акселерометров, гироскопов) обычно используются для сбора временных характеристик, в данном исследовании система обнаружения падений анализируется на основе видеоданных. Поэтому, анализируя последовательные кадры видеоданных, особенно информацию об изменении позы человека и траектории движения на видео, можно эффективно фиксировать характеристики временного ряда события падения [81,82,83].

Основными свойствами временных рядов при обнаружении падения являются следующие:

Временная последовательность: падение обычно состоит из серии последовательных движений или постуральных изменений, которые четко определены во времени. Падения имеют уникальные временные характеристики по сравнению с другими обычными действиями, такими как стояние или ходьба.

Например, падения обычно начинаются из положения стоя, сопровождаются резким изменением позы и заканчиваются падением. Анализируя последовательные кадры действий в видеозаписи, мы можем зафиксировать эту временную последовательность изменений, чтобы точно определить, произошло ли падение.

Переходные изменения: падения часто сопровождаются быстрыми изменениями ускорения и внезапными изменениями позы тела. Например, человек на видео за очень короткий промежуток времени совершает резкий наклон или падение, и это резкое изменение проявляется в виде быстрого перехода состояния движения во временном ряду. Поскольку падение - это очень динамичное событие, захват этих переходных изменений имеет решающее значение для повышения точности и эффективности обнаружения в реальном времени.

Периодичность и ненормальность: поведение при падении обычно является непериодическим, ненормальным событием, и его представление во временном ряду обычно значительно отличается от нормального поведения (например, стояния, ходьбы). При нормальном поведении модели движения человека обычно демонстрируют определенные периодические изменения, в то время как поведение при падении является внезапным и аномальным, и эти аномальные модели обычно ярко выражены во временных рядах. Поэтому, сравнивая особенности временных рядов нормальных действий и аномальных падений на

видео, мы можем еще больше повысить способность системы распознавания падений.

С постоянным развитием технологий видеонаблюдения и алгоритмов компьютерного зрения методы, основанные на анализе временных рядов, постепенно стали применяться для обнаружения падений, особенно сети с долговременной кратковременной памятью (LSTM) [84] Внедрение моделей глубокого обучения, таких как LSTM, дает новые идеи для обработки данных временных рядов. Сети LSTM способны улавливать долгосрочные зависимости в данных временных рядов и особенно подходят для изменения позы и внезапных динамических изменений при обнаружении падений [85,86].

2.4.2. Основные принципы и структура модели LSTM

Long Short-Term Memory (LSTM) - это усовершенствованная модель на основе рекуррентных нейронных сетей (РНС), которая призвана решить проблемы исчезновения градиента и взрыва градиента, к которым склонны традиционные РНС при работе с длинными последовательными данными. LSTM способна эффективно управлять передачей и запоминанием информации, лучше улавливать зависимости на больших временных интервалах и, таким образом, хорошо работает во многих задачах анализа временных рядов данных.

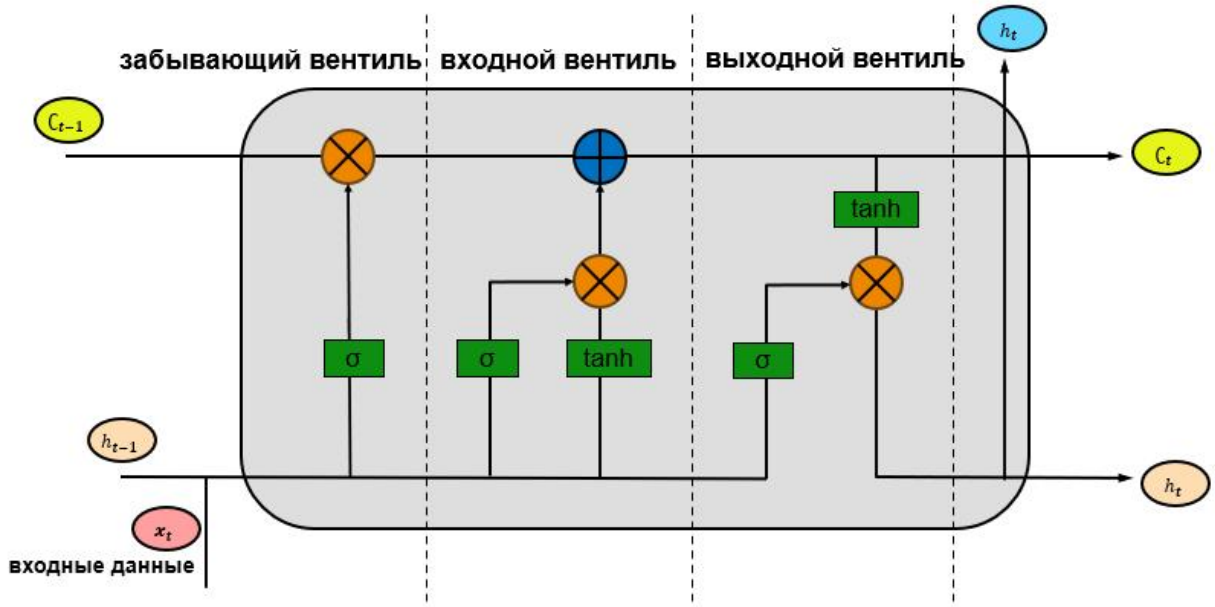


Рисунок. 2.5. Структурная схема сети LSTM

Как показано на рисунке 2.5., LSTM в основном реализует функцию защиты и контроля информации через три блока управления воротами: вход, забывание и выход. Первая задача LSTM - определить, какая информация должна быть отброшена, т.е. это достигается через ворота забывания на рисунке 2.5, которые являются входом для $t-1$ Выход кратковременной памяти мгновенного нейрона h_{t-1} и t вход моментного нейрона x_t Используя сигмоидальную функцию для управления выходными значениями между$[0,1]$ между.0 означает запрет на ввод любой информации, и 1 означает разрешение на ввод всей информации, конкретная формула имеет вид (2.5), гдеσ сигмоидальная функция активации, иW_f вес единицы, иb_f единичное значение активации:

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f \tag{2.5}$$

После ворот забывания необходимо определить, сколько новых данных должно попасть в состояние ячейки, для этого используются две функции активации: сигмоидный слой решает, какая информация должна быть обновлена, а слой tanh создает новое значение вектора-кандидата.$\tilde{C}_t$ и затем, наконец, два типа функций активации объединяются для достижения обновления состояния ячейки памяти, конкретная формула приведена в (2.6), где i_t обозначает информацию, которая должна быть обновлена решением входного шлюза, и$\tilde{C}_t$ обозначает новое значение вектора-кандидата W_C вес ячейки, и b_C значение активации ячейки:

$$\begin{aligned} i_t &= \sigma(W_i.[h_{t-1}, x_t] + b_i) \\ \tilde{C}_t &= tanh(W_C.[h_{t-1}, x_t] + b_C) \end{aligned} \tag{2.6}$$

Затем старое состояние ячейки необходимо обновить, т.е.C_{t-1} обновить до Ct, просто связав старое состояние клеткиC_{t-1} с информацией, которую было решено забытьf_t умноженной на новое значение кандидата$i_t{}^* \tilde{C}_t$ с конкретной формулой, как в (2.7):

$$C_t = f_t{}^* C_{t-1} + i_t{}^* \tilde{C}_t \tag{2.7}$$

Наконец, необходимо определить содержание выхода на основе текущего состояния клетки Ct. Сначала используется сигмоидальный слой для определения части выхода Ct, затем используется слой tanh для ограничения значения между [-1,1], и, наконец, выход tanh умножается на выход,

вычисленный по сигмоидальным воротам, чтобы получить окончательный выход, который задается формулой в (2.8):

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$
$$h_t = o_t * tanh(C_t) \quad (2.8)$$

Благодаря уникальным свойствам памяти и блокам управления воротами в сетях LSTM, их структура способна лучше улавливать долгосрочные зависимости в данных временных рядов, что особенно подходит для задач со значительными временными интервалами и сложными динамическими характеристиками, таких как обнаружение падений. Анализируя последовательные кадры видеоданных, LSTM способна выявить временные закономерности до и после наступления события, тем самым точно определяя, произошло ли падение.

2.4.3. применение LSTM в поведенческом распознавании

Применение LSTM (Long Short-Term Memory Network) в распознавании поведения достигло значительных результатов в исследованиях, особенно при обработке видеоданных или данных датчиков, временные возможности моделирования LSTM показывают сильное преимущество. В задаче распознавания падений уникальным преимуществом LSTM является ее способность улавливать динамические изменения до и после события, отличая нормальное поведение от поведения при падении. События падения часто происходят очень быстро и сопровождаются резкими изменениями позы, которые могут быть эффективно идентифицированы LSTM, что повышает

точность и устойчивость системы обнаружения падений" [87,88,89,90,91,92,93,94].

В частности, LSTM может сыграть свою роль в обнаружении падений несколькими способами. Во-первых, LSTM способна улавливать динамические изменения, особенно во время падения, когда поза тела человека обычно претерпевает внезапные изменения, такие как быстрое падение из положения стоя или ходьбы. LSTM может эффективно улавливать эти особенности временных рядов путем обработки последовательных кадров данных из видео. Запоминая и обучаясь изменению этих временных рядов, модель способна распознать поведение при падении на ранней стадии и отличить его от обычных действий, таких как ходьба и стояние. Используя эту способность, LSTM может быстро реагировать на падение, тем самым обеспечивая своевременное оповещение и вмешательство.

Во-вторых, LSTM также хорошо справляется с распознаванием аномального поведения. Падение - это очевидное аномальное поведение, которое значительно отличается от повседневной деятельности (например, ходьбы, бега и т. д.) LSTM способна различать различные виды деятельности на видео, изучая поведенческие паттерны в разные периоды времени. Его способность обрабатывать временные ряды позволяет ему определять, является ли текущий кадр поведением падения или нет, основываясь на динамике исторических кадров. Такая чувствительность к временным рядам позволяет LSTM эффективно повышать точность обнаружения падений в практических

приложениях, избегая ошибочной идентификации из-за изменения сцены или индивидуальных различий.

Наконец, возможности временного моделирования LSTM могут быть объединены с алгоритмами обнаружения целей (например, YOLO) для дальнейшего повышения производительности и точности системы обнаружения падений в реальном времени. С помощью алгоритма YOLO система способна быстро находить и обнаруживать цели (например, человеческие тела) на видео, а LSTM отвечает за анализ временных изменений этих целей и определение того, произошло ли падение. Комбинируя эти два метода, система обнаружения падений не только повышает точность обнаружения, но и обеспечивает работу в реальном времени, предоставляя более безопасные и надежные средства мониторинга для пожилых людей или других групп, нуждающихся в наблюдении в практических приложениях.

Применение модели LSTM для обнаружения падений имеет большое значение. Моделируя временные ряды данных с помощью глубокого обучения, LSTM может сыграть важную роль в улавливании динамических изменений, выделении аномального поведения, повышении производительности и точности в реальном времени, а также значительно повысить производительность и практическую ценность системы обнаружения падений [95,96].

ВЫВОДЫ

В этой главе рассматриваются теоретические основы обнаружения целей и анализа временных рядов на основе глубокого обучения. Сначала представлена

история развития обнаружения целей, указано, что традиционные методы основаны на ручном извлечении признаков и имеют ограниченную производительность в сложных сценариях, в то время как методы глубокого обучения, особенно сверточные нейронные сети (CNN), значительно повышают точность и устойчивость обнаружения. Серия алгоритмов YOLO достигает сквозного эффективного обнаружения целей путем регрессии на проблемную структуру, и, в частности, в YOLOv8, использование механизма безъякорной рамки и других стратегий оптимизации для дальнейшего повышения точности и скорости обнаружения.

В этой главе, также, рассматриваются свойства временных рядов при обнаружении падений, подчеркиваются преимущества LSTM в улавливании внезапных динамических изменений и длительных временных зависимостей. LSTM способна эффективно различать нормальное и падающее поведение, что повышает точность и эффективность обнаружения в реальном времени.

Данная глава закладывает теоретическую основу для последующих исследований по применению YOLOv8 в сочетании с LSTM для обнаружения падений и способствует реализации высокоточной системы обнаружения падений в реальном времени.

ГЛАВА 3. АЛГОРИТМ ОБНАРУЖЕНИЯ ПАДЕНИЙ НА ОСНОВЕ УЛУЧШЕННОГО YOLOV8 С LSTM

3.1. Анализ и ограничения оригинального алгоритма YOLOv8

Серия YOLO (You Only Look Once), как классический алгоритм обнаружения целей, в последние годы совершает прорывы в скорости и точности. YOLOv8, как последняя версия серии YOLO, наследует высокоэффективные функции YOLO и оптимизирует вычислительную производительность и точность обнаружения модели [97,98]. Однако, несмотря на то, что YOLOv8 демонстрирует хорошие результаты в ряде задач обнаружения целей, особенно в обнаружении падений, все же существует ряд очевидных ограничений [99,100].

- **Динамическая сцена и обнаружение мелких целей**

События падения обычно происходят в течение короткого промежутка времени, а поза персонажа быстро меняется. Хотя YOLOv8 способен обнаруживать обычные цели и выполнять точную локализацию ограничительных рамок, ему не хватает возможностей для обнаружения мелких целей в сложных динамических сценах. Например, когда персонаж падает, в поле зрения камеры может попасть только часть его тела, а фон является сложным, что ставит под сомнение точность локализации цели в YOLOv8. Кроме того, момент падения может сопровождаться резким изменением позы, что делает YOLOv8 восприимчивым к пропуску и неправильному обнаружению в быстро меняющихся динамичных сценах.

- **Изменение позы и захват мелких деталей**

Хотя YOLOv8 хорошо справляется с извлечением общих признаков, он относительно слаб в захвате тонких признаков. Обнаружение падений часто требует точной оценки небольших изменений в позе человека, особенно при переходе от стояния к падению. YOLOv8 может испытывать трудности с точным захватом этих ключевых характеристик при работе со сложными изменениями в позе, особенно при быстрых изменениях в движении, и склонен упускать некоторую подробную информацию.

- **недостаточные возможности моделирования временных рядов**

Хотя YOLOv8 отлично справляется с обнаружением целей на однокадровых изображениях, он не приспособлен для работы с временными сериями. В задачах обнаружения падений характеристики данных временных рядов имеют решающее значение. Падения обычно сопровождаются серией быстрых изменений движения, а традиционные модели YOLO не предназначены для работы с данными временных рядов. Хотя YOLOv8 способна обнаруживать цели в отдельных кадрах и выдавать результаты обнаружения, ей не хватает возможности моделировать взаимосвязи между последовательными кадрами. В результате YOLOv8 не может эффективно отражать временные характеристики событий падения в задачах обнаружения падения, что влияет на ее производительность в таких задачах [101,102].

3.2. Разработка улучшенной программы

Для дальнейшего улучшения работы сети YOLOv8 для обнаружения падений пожилых людей в сложных сценариях, в этой главе мы улучшаем сеть

на основе оригинального алгоритма и повышаем производительность модели. Во-первых, за тремя последними модулями C2f опорной сети вводится трехслойный механизм облегченного внимания ECA[103] , который практически не увеличивает параметры сети и позволяет модели фокусироваться на цели во время обучения, что повышает точность обнаружения. Сеть слияния признаков также оптимизирована с помощью облегченного механизма свертки GSConv[104], который снижает сложность модели по сравнению со стандартными сверточными методами и повышает производительность модели при значительно меньшем количестве вычислительных параметров. По сравнению со сверткой с разделением по глубине, взаимодействие между каналами может быть увеличено, что улучшает производительность модели.

3.2.1.Эффективное канальное внимание (ECA)

Полное название механизма внимания ECA - Efficient Channel Attention, что означает эффективное канальное внимание. Большинство механизмов внимания к каналам имеют тенденцию к разработке более сложных модулей для улучшения производительности, что увеличивает сложность и вычислительные затраты модели. Чтобы решить эту проблему, ECA разрабатывает практически беспараметрический, вычислительно эффективный механизм канального внимания, который снижает вычислительную сложность, сохраняя при этом хорошую производительность. ECA больше похож на улучшенную версию SE внимания. Авторы [62] обнаружили, что операция сокращения размерности в SE не способствует обучению канальному вниманию, поэтому они предложили

локальную стратегию кросс-канального взаимодействия без сокращения размерности. Сетевая структура механизма внимания ECA показана на рисунке 3.1..

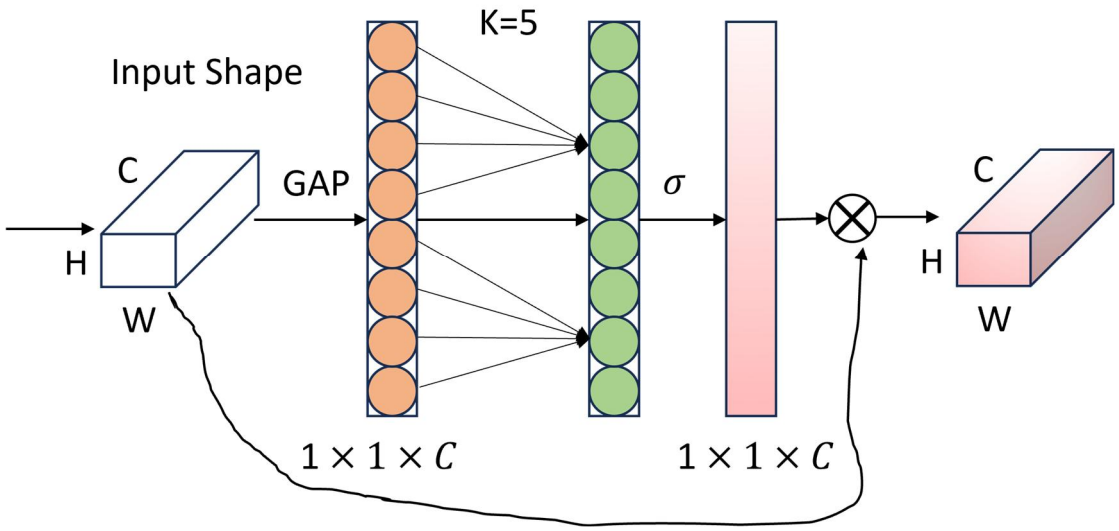


Рисунок 3.1. Схема структуры сети ECA

Пусть форма входной карты признаков имеет вид$H \times W \times C$ гдеH высота, и W ширина, и C количество каналов. Сначала входная карта признаков подвергается глобальному усреднению (GAP) для получения глобального среднего по каждому каналу, после чего форма становится $1 \times 1 \times C$ ECA рассчитывает размер одномерного конволюционного ядра. k Размер одномерного конволюционного ядра, где k размер может быть адаптивно определен, т.е. вес каждого канала связан с соседними k соседними собственными значениями. Вдохновляясь групповой сверткой, весk выбирается пропорциональным размеру каналаC пропорционально размеру канала, который вычисляется, как показано в (3.1), т.е. k Существует отображение между C Существует отображение между φ .

$$C = \varphi(k) \tag{3.1}$$

Одно из простейших линейных отображений можно представить в виде (3.2). гдеφ отображение, иk размер ядра свертки, аγ и b - коэффициенты линейной функции.

$$\varphi(k) = \gamma \cdot k - b \tag{3.2}$$

Однако линейные функции имеют множество ограничений и не могут соответствовать сложным ситуациям, поэтому линейная функция подвергается нелинейному преобразованию, а число характеристических каналов часто является мощностью 2. После математического преобразования окончательнаяk Формула представлена в уравнении (3.3).

$$k = \varphi(C) = \left|\frac{\log_2(C)}{\gamma} + \frac{b}{\gamma}\right|_{odd} \tag{3.3}$$

где φ отображение, и C количество каналов, и γ и b коэффициенты отображения, а odd нечетное число с ближайшим значением. Веса между различными каналами получаются путем одномерной свертки, затем вектор веса превращается в число от 0 до 1 с помощью сигмоидальной функции активации, и, наконец, сгенерированные веса умножаются на каналы исходной карты

входных признаков для создания взвешенной карты признаков, а окончательная формула механизма внимания ECA представлена в уравнении (3.4).

$$w = \sigma(C1D_k y) \tag{3.4}$$

где y вектор признаков после глобального усреднения, а $C1D_k$ размер ядра свертки k одномерной свертки, и σ сигмоидная функция активации, и w вес канала. Одномерная свертка позволяет избежать операции уменьшения размерности, что уменьшает количество вычислительных параметров и обеспечивает кросс-канальное взаимодействие, повышая эффективность обучения кросс-канального внимания и производительность сети.

3.2.2. GSConv

В нейронных сетях балансировка между количеством параметров модели и ее производительностью является очень важной задачей. Введение новых модулей может улучшить выразительные способности модели, но часто увеличивает количество параметров модели, что приводит к большому вычислительному объему и низкой эффективности обучения и вывода. Конволюционная нейронная сеть передает пространственную информацию в канал шаг за шагом в процессе свертки, что приводит к потере части семантической информации при каждом сжатии карты признаков и расширении канала. Стандартные сверточные вычисления максимально сохраняют скрытые связи между каждым каналом, в то время как глубокая сепарабельная свертка

полностью обрывает эти связи. Пусть ширина выходной карты признаков равнаW а высотаH и$K1K2$ размер ядра свертки, и$C1$ количество каналов на ядро свертки, и$C2$ количество каналов выходной карты признаков, а уравнение (3.5) представляет собой временную сложность стандартного процесса свертки.

$$Time_{SC} = O(W \bullet H \bullet K_1 \bullet K_2 \bullet C_1 \bullet C_2) \tag{3.5}$$

В стандартной свертке ядро свертки необходимо вычислять для каждого входного канала, что увеличивает вычислительную сложность и количество параметров. Свертка с разделением по глубине - это попытка решить эту проблему путем разделения операции свертки на две части: свертку по глубине и свертку по точкам. В процессе глубокой свертки каждый канал отдельно свертывается с ядром свертки, а затем выходные карты признаков свертываются по точкам - эта операция позволяет значительно сократить количество параметров в модели. Временная сложность операции глубокой сепарабельной свертки показана в (3.6).

$$Time_{DSC} = O(W \bullet H \bullet K_1 \bullet K_2 \bullet 1 \bullet C_2) \tag{3.6}$$

По сравнению со стандартной сверткой, вычислительная сложность значительно уменьшается, однако этот метод может привести к снижению точности модели, так как информация о канале разделяется в процессе

вычислений. GSConv, с другой стороны, предлагает новое решение, которое сочетает стандартную свертку и глубокую свертку, что позволяет снизить вычислительную сложность и гарантировать точность вычислений. Временная сложность процесса свертки GSConv показана в уравнении (3.7).

$$Time_{GSConv} = O\left[W \bullet H \bullet K_1 \bullet K_2 \bullet \frac{C_2}{2}(C_1 + 1)\right] \tag{3.7}$$

Когда выходная карта признаков, входные каналы и ядро свертки имеют одинаковый размер, GSConv может сократить часть параметров по сравнению со стандартной сверткой, что снижает сложность модели. Хотя свертка с относительным разделением по глубине требует больших вычислительных затрат, в процессе свертки гарантируется взаимодействие между каналами, что позволяет повысить производительность модели. На рисунке 3.2. показана сетевая структура GSConv, которая содержит как стандартную, так и глубинную свертку в процессе вычисления свертки.

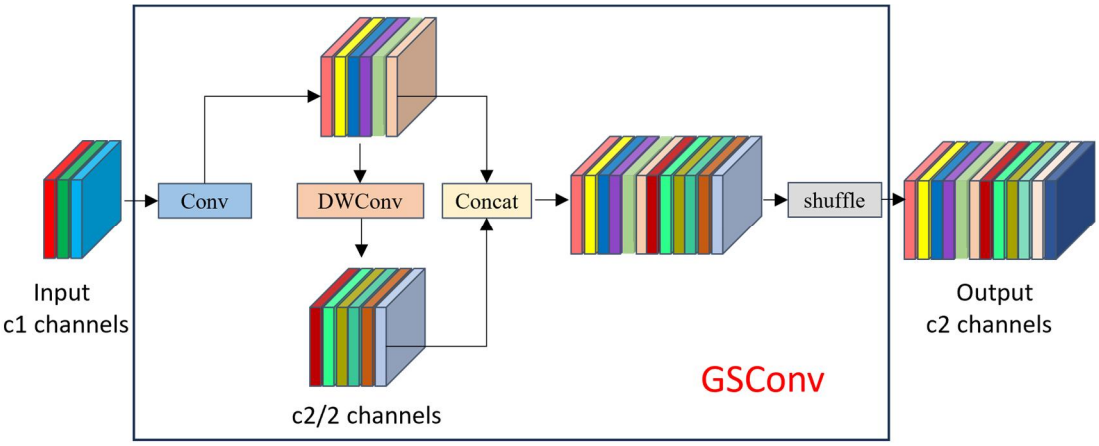


Рисунок 3.2. Схема структуры сети GSConv

Процесс работы GSConv выглядит следующим образом: сначала входное изображение подвергается операции понижающей дискретизации со стандартной сверткой, что уменьшает размер карты признаков. Затем уменьшенная карта признаков свертывается с помощью глубокой свертки (DWConv) для создания новой карты признаков. Затем карта признаков после стандартной свертки и карта признаков после глубокой свертки сращиваются по размеру канала, чтобы объединить их информацию. Наконец, выполняется операция перетасовки, чтобы равномерно смешать количество каналов, соответствующих двум сверткам, что обеспечивает лучшее взаимодействие и слияние информации между каналами. Благодаря вышеописанным шагам GSConv успешно достигает эффективного сокращения числа параметров и вычислительной сложности модели, сохраняя при этом высокую точность, что особенно подходит для использования в сценариях с ограниченными вычислительными ресурсами. По сравнению с глубокой сепарабельной сверткой, GSConv может повысить точность модели, а по сравнению со стандартной сверткой - снизить сложность модели и количество параметров, и производительность GSConv сравнима или даже лучше, чем у стандартной свертки, с очевидными преимуществами в размере модели.

3.2.3. Улучшенный алгоритм обнаружения целей YOLOv8

Усовершенствованная структура сети YOLOv8 показана на рисунке 3.3. ниже.

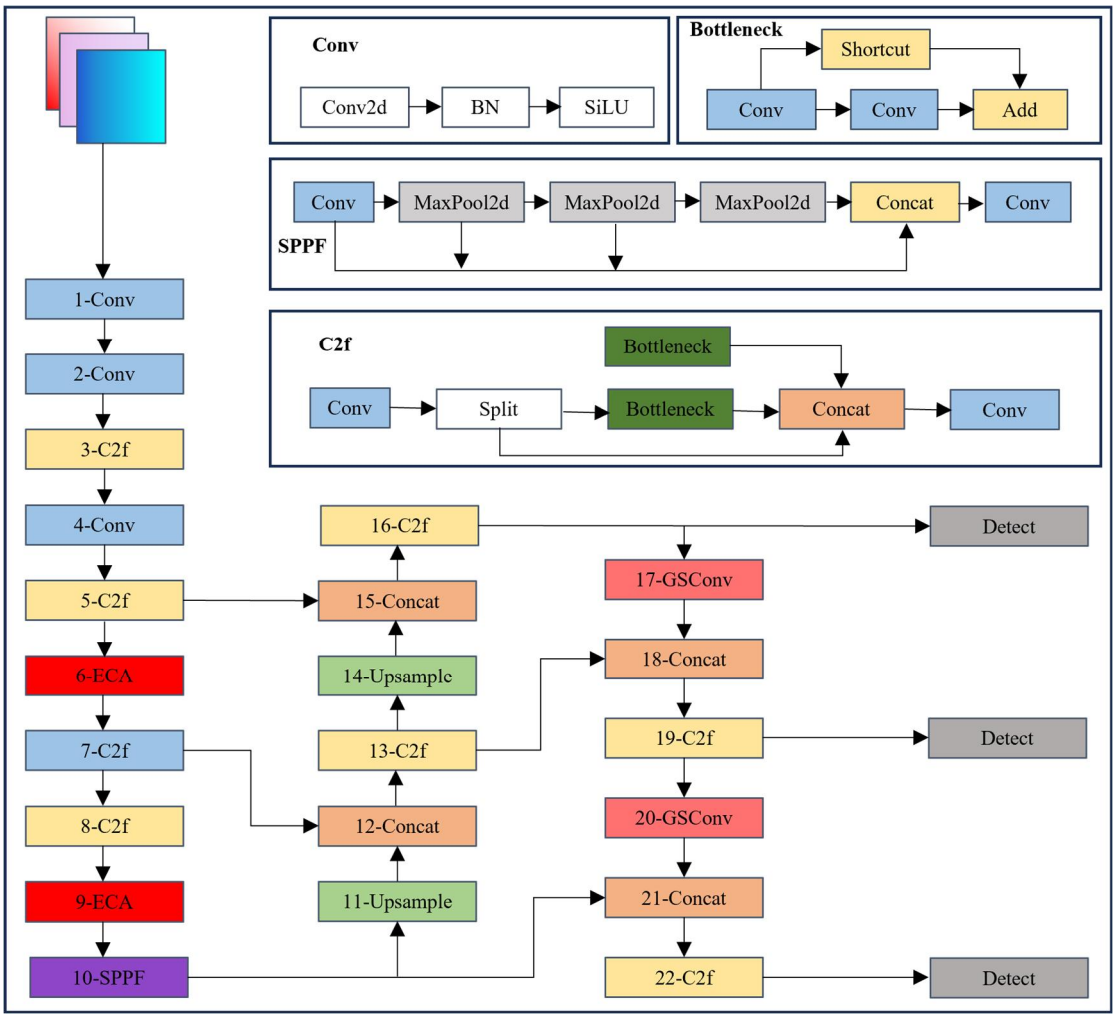


Рисунок 3.3. Улучшенная схема структуры сети YOLOv8

Добавив в магистральную сеть механизм внимания ECA, можно эффективно улучшить возможности магистральной сети по извлечению признаков. Причина использования GSConv только на этапе Neck заключается в небольших вычислительных затратах, на этапе Neck карта признаков становится достаточно тонкой, то есть ширина и высота карты признаков достигают минимума, а глубина карты признаков достигает максимума, поэтому целесообразно использовать GSConv для обработки карты признаков на этом этапе, в ней меньше избыточной повторяющейся информации, не требуется дополнительное сжатие, и модуль внимания работает лучше. Если использовать GSConv на всех

этапах работы сети, это значительно углубит слои сети, что приведет к замедлению вывода модели.

3.2.4. Улучшенный YOLOv8 с сетью LSTM

После вышеуказанных усовершенствований модели YOLOv8, улучшенная модель объединяется с моделью LSTM, названной YOLOv8-LSTM. Модель разработана для использования возможностей эффективного обнаружения целей YOLOv8 и преимуществ LSTM в моделировании временных рядов данных с целью повышения точности и устойчивости системы обнаружения падений в сложных динамических сценах.

Рисунок 3.4. Блок-схема модели YOLOv8-LSTM

Как показано на рисунке 3.4., в модели YOLOv8-LSTM YOLOv8 сначала извлекает пространственные признаки в каждом кадре, а затем вводит эти последовательности признаков в LSTM для обработки. LSTM изучает входные временные признаки через свой специальный блок памяти, таким образом определяя динамические изменения до и после события падения, что еще больше повышает способность модели обнаруживать поведение при падении. В частности, LSTM способна идентифицировать переходные и динамические признаки события падения, изучая эволюцию позы человека на видео, тем самым различая нормальное и ненормальное поведение при падении.

Основное преимущество этой модели заключается в том, что YOLOv8 отвечает за быстрое и эффективное обнаружение целей для каждого кадра изображения, а LSTM выполняет точное временное моделирование и классификацию событий падения на основе временной зависимости между последовательными кадрами. Благодаря такому сочетанию модель YOLOv8-LSTM не только обнаруживает цели в реальном времени, но и динамически анализирует их во временном измерении, что значительно повышает точность и время отклика при обнаружении падений.

3.3. Экспериментальные результаты и анализ

3.3.1. Введение в наборы данных

Обнаружение падений является типичной задачей распознавания поведения, которая сталкивается с множеством вызовов. В частности, сложность данной задачи обусловлена разнообразием и сложностью человеческого поведения, что приводит к высокой гетерогенности типов падений и их проявлений. Таким образом, разработка алгоритмической модели, способной эффективно идентифицировать падения, требует не только обработки различных типов падений, но и учета различий в проявлении одного и того же типа падения в разных условиях и сценариях.

Поскольку падения могут существенно различаться по форме, контексту возникновения и деталям движений, проектирование и реализация алгоритмов обнаружения падений представляют собой сложную задачу. Для преодоления данной проблемы в данном исследовании применяется стратегия отбора наборов

данных, направленная на повышение способности модели к распознаванию падений за счет разнообразия и полноты данных.

При выборе наборов данных все виды падений были объединены в единую категорию — «Fall down» (падение), что означает, что независимо от конкретных обстоятельств происшествия, все падения рассматриваются как один общий класс. Такая упрощенная обработка направлена на снижение сложности модели, а также на уменьшение внутриклассового сходства, что помогает избежать чрезмерного усложнения процесса обучения модели при увеличении числа классов. Кроме того, такая упрощенная модель позволяет сосредоточиться на главной задаче — выявлении факта падения, а не на различении множества его возможных типов.

Для обеспечения широты охвата и репрезентативности данных в данном исследовании используются публичные наборы данных, а также собственный набор данных, который служит дополнительным источником информации, обеспечивая баланс между разнообразием и достаточностью обучающего материала. В качестве публичных наборов данных были выбраны UR Fall Detection Dataset (URFD)[113],Multiple Cameras Fall Dataset (MCFD) [114] и Le2i Fall Detection Dataset (LFD)[115].

Как показано на рисунке 3.5., URFD представляет собой средний по размеру публичный набор данных, содержащий различные сценарии падений и широкий спектр образцов падений. Этот набор данных включает 30 различных последовательностей падений, записанных в разных условиях, причем каждая

последовательность снята с двух разных ракурсов, что гарантирует разнообразие и богатство данных. В URFD поведение при падении детально описано, включая подготовительные движения перед падением и динамику самого процесса падения. Таким образом, данный набор данных помогает модели уловить ключевые характеристики падений, что не только повышает чувствительность модели к выявлению падений, но и способствует ее адаптации к различным вариациям падений при изменении угла обзора.

Примеры из URFD демонстрируют типичные последовательности падений, снятые с разных ракурсов, что позволяет модели лучше изучить паттерны движений при падении и их детальные характеристики. Такой многокамерный подход делает URFD идеальным выбором для обучения сложных моделей, особенно тех, которые предназначены для повышения точности распознавания падений и адаптации к различным сценариям с помощью методов глубокого обучения.

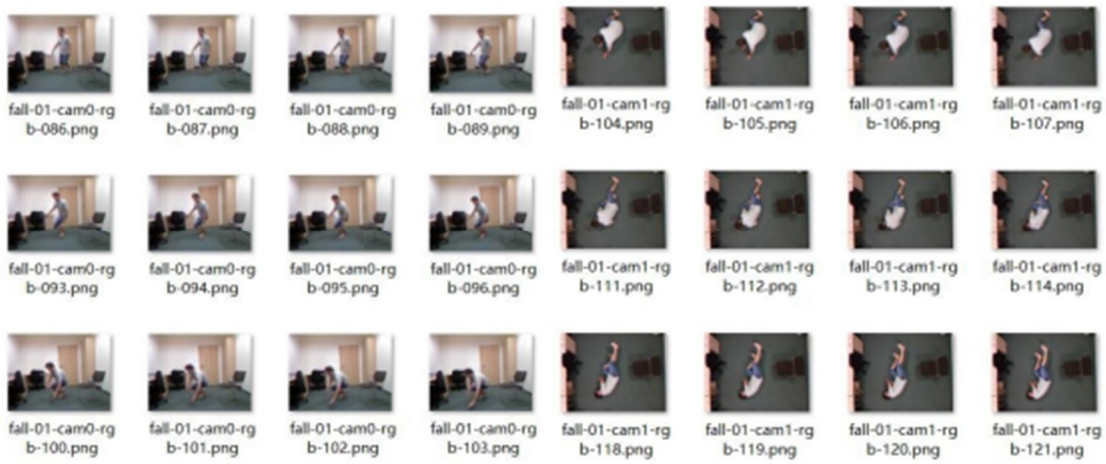


Рисунок 3.5. Набор данных URFD

Набор данных MCFD, показанный на рисунке 3.6., представляет собой видеоданные, содержащие множество точек зрения и предназначенные для изучения обнаружения падений. Набор данных содержит 24 категории видео, каждая из которых охватывает 8 различных положений камеры, что в общей сложности составляет 192 видео. Эти видеоданные получены с различных точек зрения, что обеспечивает разнообразный и полный набор данных, которые помогут модели эффективно идентифицировать поведение при падении в широком диапазоне условий и перспектив. Каждое видео содержит весь процесс поведения при падении, начиная с предвещающих действий перед падением и заканчивая динамическими изменениями во время падения, а также некоторыми возможными последующими действиями после падения. Такой комплексный подход к записи помогает модели изучить все детали поведения при падении и продемонстрировать высокую устойчивость в практических приложениях.

Рисунок 3.6 Набор данных MCFD

Набор данных LFD, показанный на рисунке 3.7., состоит из двух частей, карты глубины и трихроматической карты RGB, которая содержит более богатую визуальную информацию и обеспечивает мультимодальную поддержку для исследований по обнаружению падений. Набор данных содержит в общей сложности 30804 изображения, что представляет собой огромный объем данных, охватывающий большое количество образцов как нормального поведения, так и поведения при падении. Отличительной особенностью набора данных LFD является то, что он содержит файл меток (label.csv), в котором подробно записана аннотационная информация каждого изображения, включая информацию о том, является ли это поведение падением или нет, период времени, когда происходит такое поведение, и специфические особенности поведения. Эта информация о метках служит надежной основой для обучения и оценки алгоритма обнаружения падений, что позволяет повысить эффективность обучения и точность модели.

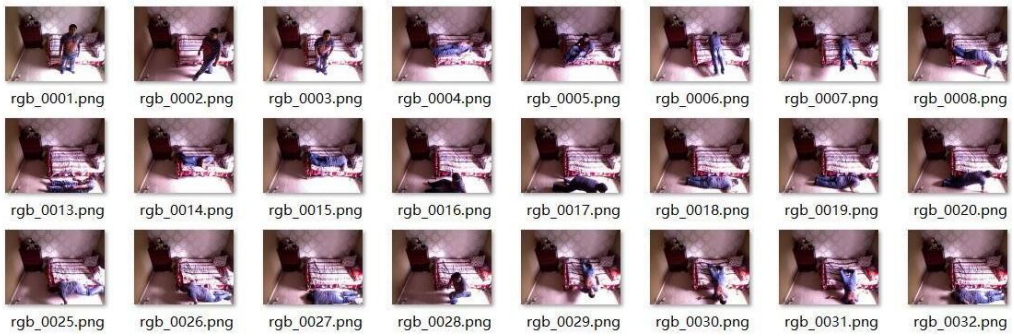


Рисунок 3.7. Набор данных LFD

Как показано на рисунке 3.8. и 3.9., набор данных, созданный самостоятельно, включает в себя 14 видеороликов падений, 3 видеоролика нормального поведения и 3280 фотографий падений. Для того чтобы обеспечить разнообразие и репрезентативность данных, в этом наборе данных представлены модели поведения при падении и обычные модели поведения при различных сценах, ракурсах и условиях освещения. Видеоданные позволяют получить непрерывную информацию о временных рядах, чтобы помочь модели зафиксировать динамические изменения в поведении при падении, в то время как статические изображения обеспечивают четкие визуальные идентификаторы, которые помогут модели определить событие падения в определенный момент. Благодаря таким разнообразным источникам данных модель можно адаптировать к различным условиям применения.

Рисунок 3.8.

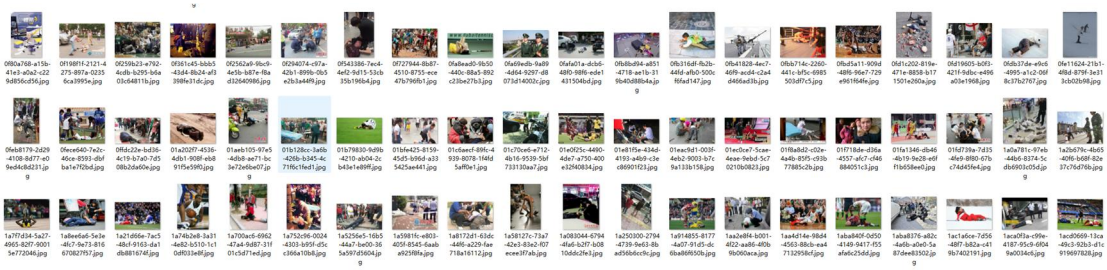


Рисунок 3.9. Самостоятельно сконструированный набор данных

После ручной очистки и предобработки как публичных, так и самостоятельно созданных наборов данных был сформирован окончательный экспериментальный набор данных, который включает в себя 7680 изображений и 200 видеозаписей. В реальных условиях сложность поведения при падении делает точное определение различных типов падений затруднительным. Поэтому в данном эксперименте все типы падений были объединены в единую категорию "Fall down", а остальные виды поведения отнесены к категории "Normal". Набор данных был организован в соответствии с форматом VOC2007 и разметка данных проводилась с использованием LabelImg. Экспериментальный набор данных был разделен на тренировочный и тестовый в соотношении 2:8. Подробное распределение данных представлено в таблице 3.1.

Таблица 3.1. Разделение набора данных

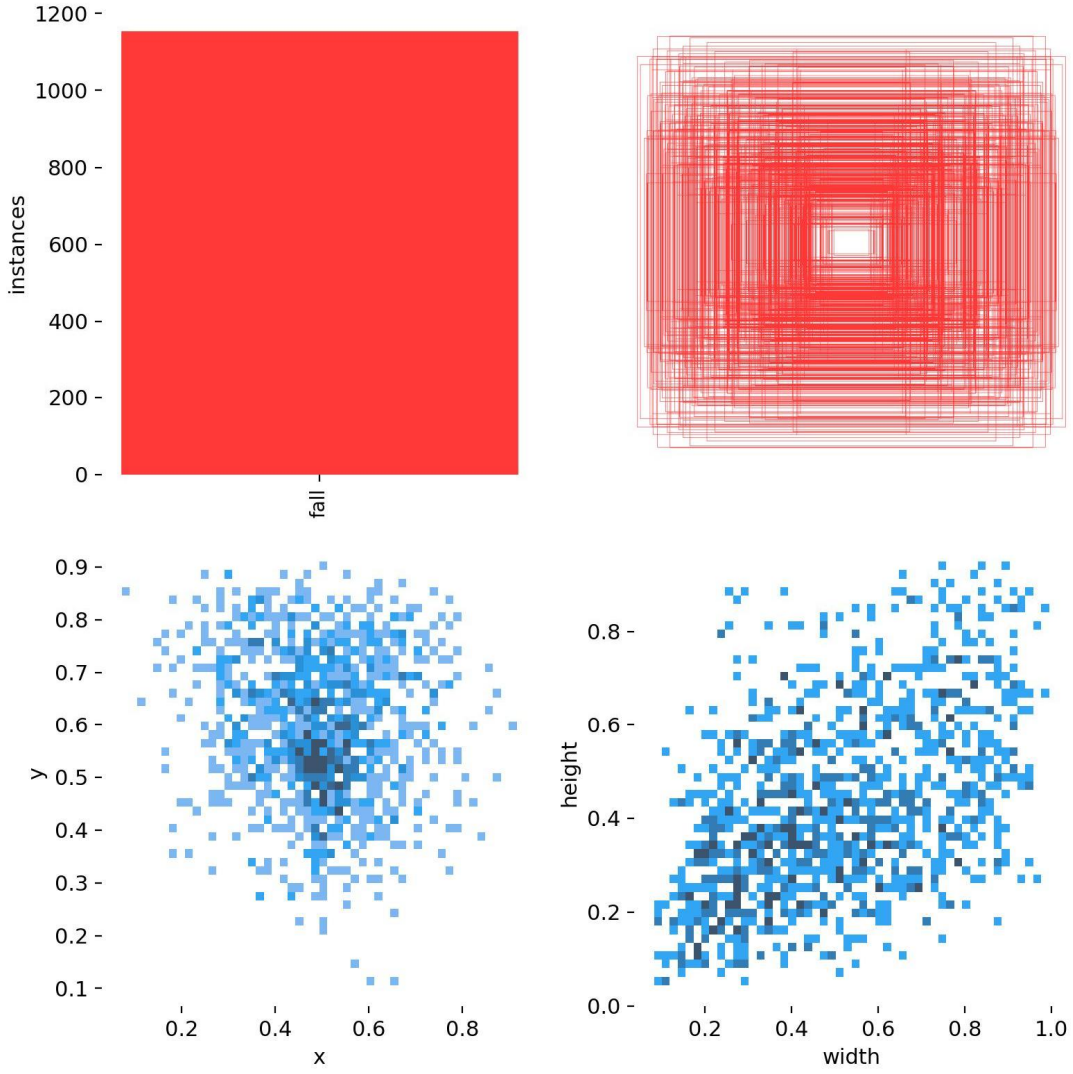
параметры	Количество фотографий	Количество видеороликов
Общая категория наборов данных	7680	200
обучающий набор	6144	160
тестовый набор	1536	40

Перед обучением модели набор данных проходит ряд предварительных обработок, чтобы обеспечить согласованность и адекватность исходных данных. Изображения приводятся к единому размеру, а значения пикселей нормализуются, что ускоряет обучение модели и улучшает качество сходимости.

Рисунок 3.10. Диаграмма маркировки данных

Из карты распределения аннотаций изображения видно, что события, связанные с падением пешеходов, в основном сосредоточены в центральной области изображения, что может быть связано с тем, что в реальных сценариях наблюдения люди обычно более активны в центральной области экрана. Между

тем, карта распределения размеров аннотационных кадров показывает, что аннотационные размеры падений пешеходов более концентрированы в наборе данных, что помогает нам нацелить и оптимизировать экстрактор признаков при разработке модели. Тем не менее, есть и некоторые отклонения в аннотациях, что говорит о необходимости более тщательного анализа этих отклонений для поддержания качества данных.



3.11. Выборочные диаграммы анализа данных

При анализе некоторых данных, как показано на рисунке 3.11., широкий охват наложений маркированных ящиков на полном изображении говорит о том,

что, хотя падения обычно происходят в центре, вся площадь изображения должна быть эффективно охвачена моделью. Чтобы предотвратить смещение модели, нам может потребоваться сбалансировать представление краевых областей с помощью дополнения данных. Наш набор данных демонстрирует полноту маркировки путем проведения тщательной работы по аннотированию на предмет согласованности меток и категорий. Поведение при падении маркируется на каждом изображении, чтобы модель могла обучаться на всем спектре данных, что помогает избежать переобучения модели на некоторых конкретных типах падений.

3.3.2. экспериментальная среда и критерии оценки алгоритмов

- **Конфигурация экспериментальной среды**

Для обеспечения эффективности и стабильности процесса обучения модели в эксперименте используется AutoDL, облачная серверная платформа, предоставляющая мощные вычислительные ресурсы и конфигурации среды для поддержки обучения глубокому обучению. Что касается аппаратного обеспечения, то в эксперименте используется видеокарта NVIDIA Tesla V100 (память 32 ГБ), которая может обеспечить отличную вычислительную производительность в задачах глубокого обучения, особенно при работе с крупными наборами данных и обучении сложных моделей, что значительно повышает скорость обучения и сокращает время обучения. Между тем, оснащенный 6 vCPU процессор Intel(R) Xeon(R) Gold 6130 CPU @ 2.10GHz, хотя видеокарта отвечает за основные вычислительные задачи, CPU играет

вспомогательную роль в предварительной обработке данных и управлении моделями для обеспечения общей бесперебойной работы системы. Программное окружение включает операционную систему Ubuntu 22.04, которая является широко распространенным дистрибутивом Linux для обучения глубокому обучению, а ее отличная совместимость позволяет удовлетворить потребности большинства фреймворков и инструментов глубокого обучения. Кроме того, CUDA версии 11.8 обеспечивает эффективное использование вычислительной производительности видеокарт NVIDIA. В этом эксперименте используется PyTorch, основной фреймворк глубокого обучения, гибкость и производительность которого играют важную роль в обучении и выводе моделей, что хорошо подходит для улучшения требований к обучению YOLOv8 в этом проекте.

Конфигурации аппаратного и программного обеспечения приведены в таблице:

Таблица 3.2. Таблица конфигурации для экспериментальной среды

форма	Подробная конфигурация
терраса	Облачная серверная платформа AutoDL
Графические процессоры	NVIDIA Tesla V100, 32 ГБ графической памяти

ПРОЦЕССОР	6 vCPU Intel(R) Xeon(R) Gold 6130 @ 2,10 ГГц
операционная система	Ubuntu 22.04
Версия CUDA	CUDA 11.8
Фреймворки для глубокого обучения	PyTorch

- **Настройки параметров обучения**

В разделе "Обучение модели" для модели YOLOv8-LSTM задаются следующие ключевые параметры обучения, чтобы обеспечить эффективность обучения модели и сбалансировать производительность и эффективность. Выбор и настройка этих параметров оказывают важное влияние на скорость сходимости и конечную производительность модели. Конкретные параметры и их описание приведены в таблице 3.3.:

Таблица 3.3. Таблица настройки параметров

гиперпараметризация	устанавливать	инструкции
Скорость обучения (lr0)	0.01	определяет размер шага подстройки весов модели, что способствует

		быстрой сходимости на ранних этапах обучения.
Затухание скорости обучения (lrf)	0.01	Управление скоростью снижения скорости обучения в процессе тренировки помогает точно настроить модель на более поздних этапах обучения.
Momentum	0.937	Ускоряет обучение модели в нужном направлении и уменьшает колебания для ускорения сходимости.
вес_распада	0.0005	Предотвратите перебор и уменьшите сложность модели, добавив регулярный

		член в функцию потерь.
Циклы разминочных тренировок (warmup_epochs)	3.0	Обучение начиналось с низкой скорости обучения в течение первых нескольких циклов и постепенно увеличивалось до заданной скорости обучения.
Размер партии (партия)	32	Количество образцов, вводимых в модель в каждой итерации обучения, влияет на использование памяти GPU и производительность модели.
Размер входного изображения (imgsz)	640	Размер входного изображения,

		принимаемого моделью, влияет на способность распознавания и вычислительную нагрузку модели.

Настройки указанных параметров основаны на экспериментальных требованиях и характеристиках модели, что позволяет оптимизировать использование вычислительных ресурсов и повысить эффективность обучения при условии обеспечения производительности модели. При разумных настройках гиперпараметров модель YOLOv8-LSTM демонстрирует высокую точность и устойчивость в задаче обнаружения падений.

- **Критерии оценки алгоритмов**

Основными метриками оценки модели в этом эксперименте являются точность ($Precision$), отзыв ($Recall$), F1-меру (F1-Score)**,** средняя точность (AP) и среднее значение точности (mAP); из них mAP делятся на $mAP@0.5$ и $mAP@0.5{:}\,0.95$ два показателя, первый указывает на то, что коэффициент пересечения и слияния (IoU) порог 0,5, а второй указывает, что коэффициент пересечения и слияния (IoU) порог от 0,5-0,95 в среднем по mAP .

Формулы расчета $Precision$, $Recall$ и $F1-Score$ приведены в уравнениях (3.8) и (3.9).

$$\text{Precision} = \frac{TP}{TP+FP} \quad \text{Recall} = \frac{TP}{TP+FN} \tag{3.8}$$

$$\text{F1} - \text{Score} = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3.9}$$

В данной формуле:

TP (истинно положительные, True Positives) – количество случаев, когда модель правильно определила падение человека как "падение";

TN (истинно отрицательные, True Negatives) – количество случаев, когда модель правильно определила, что человек не падал, как "стояние";

FP (ложно положительные, False Positives) – количество случаев, когда модель ошибочно классифицировала непадающего человека как "падение";

FN (ложно отрицательные, False Negatives) – количество случаев, когда модель ошибочно классифицировала падающего человека как "стояние".

Точность (Precision) – оценивает, насколько точны предсказания модели среди всех положительно классифицированных образцов. Это доля истинно положительных случаев среди всех, предсказанных как положительные.

Полнота (Recall) – оценивает способность модели предсказывать положительные случаи. Это доля истинно положительных случаев среди всех фактических положительных примеров.

F1-мера (F1-Score) – гармоническое среднее Precision и Recall, обеспечивающее комплексную оценку производительности модели.

AP Взаимосвязь между mAP Расчеты приведены в уравнениях (3.10) и (3.11):

$$\mathrm{AP} = \frac{1}{n}\sum \mathrm{Max}\left(p\left(r(k)\right)\right) \times \left(r(k) - r(k-1)\right) \quad r \in \{0, r(0), r(1), \ldots, r(k), 1\} \tag{3.10}$$

$$mAP = \frac{1}{m}\sum AP(i),\ i \in [0, m] \tag{3.11}$$

в уравнении (3.10): $r(k)$ обозначает r наибольший из k наибольший $Recall$; $\mathrm{Max}\left(p\left(r(k)\right)\right)$ обозначает максимальное значение точности в точке r; в уравнении (3.11):m обозначает количество категорий;$AP(i)$ обозначает количество категорий в первомi средняя точность категории. Кроме того, в уравнение также включены частота кадров/FPS для оценки частоты кадров алгоритма в реальном времени на одном устройстве; размер веса модели/MB и продолжительность обучения/h для оценки вычислений алгоритма, а также размера модели.

3.3.3. Анализ экспериментальных результатов и оценка модели

В этом разделе мы проводим глубокий анализ производительности улучшенной модели YOLOv8 (YOLOv8-LSTM) в задаче обнаружения падений и оцениваем ее в сравнении с оригинальной YOLOv8, а также с другими основными моделями обнаружения целей (YOLOv5, YOLOv6, YOLOv7). Цель сравнительной оценки - подчеркнуть преимущества улучшенной модели, особенно в таких ключевых показателях, как точность($Precision$) обнаружения цели,отзыв ($Recall$), F1-меру (F1-Score) и скорость вывода.

Сравнительный анализ YOLOv8 и YOLOv8-LSTM

Сначала мы сравним процесс обучения и результаты оригинального YOLOv8 и улучшенного YOLOv8. Ниже представлен график результатов обучения YOLOv8 и YOLOv8-LSTM:

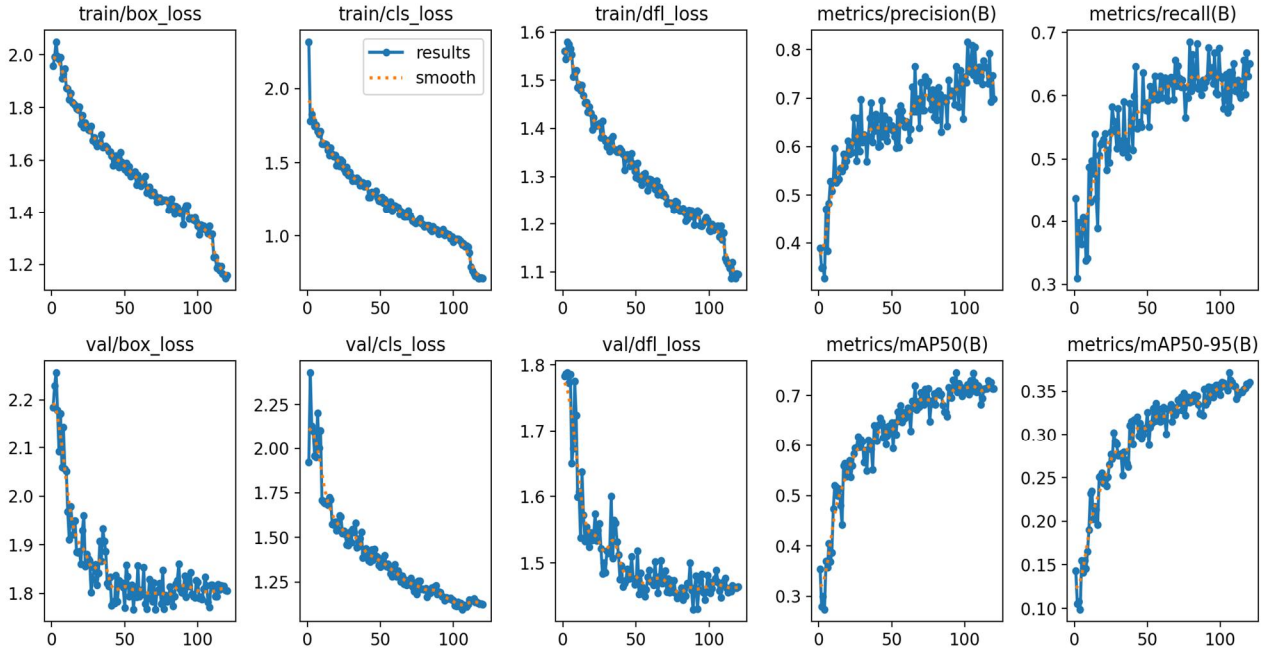


Рисунок 3.12. График результатов YOLOv8

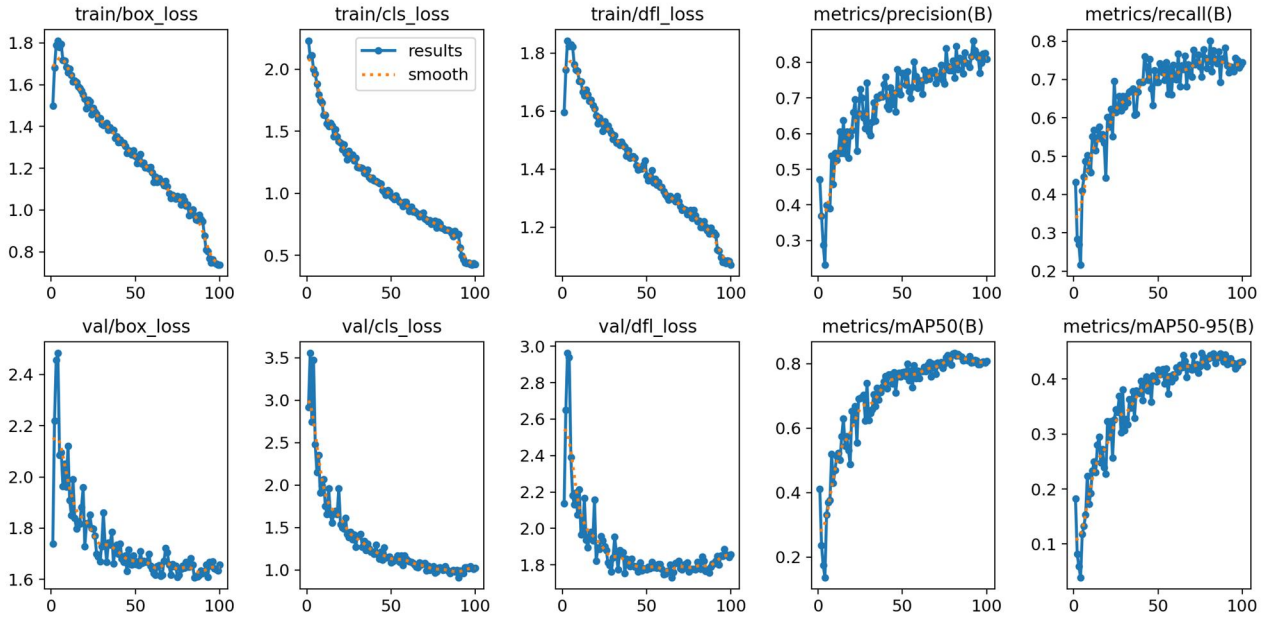


Рисунок 3.13. График результатов YOLOv8-LSTM

Первое, что можно заметить по результатам изображения, это то, что потери при обучении (train/box_loss, train/cls_loss, train/obj_loss) и потери при проверке (val/box_loss, val/cls_loss, val/obj_loss) показывают общую тенденцию к уменьшению по мере увеличения периода обучения. На начальном этапе кривые потерь демонстрируют большие колебания, что является отражением первоначальной подгонки модели под данные в процессе обучения, а по мере обучения модель постепенно стабилизируется и лучше понимает распределение данных. По сравнению с YOLOv8, улучшенная модель YOLOv8-LSTM демонстрирует более плавную кривую убывания потерь в процессе обучения, что свидетельствует о более быстрой сходимости и значительном улучшении нескольких показателей потерь. В частности, потери в границах (train/box_loss) улучшенной модели значительно уменьшаются, показывая ее прогресс в определении границ поведения при падении; потери в категориях (train/cls_loss) постепенно уменьшаются, указывая на то, что модель отлично справляется с определением категорий падения; и потери в целях (train/obj_loss) также постепенно уменьшается по мере обучения, что свидетельствует о повышении способности модели точно различать фон и цели падения.

На этапе проверки кривая потерь стабилизируется после снижения до определенного значения, что свидетельствует о хорошей способности модели к обобщению на невидимых данных и отсутствии явления переподгонки. По сравнению с YOLOv8, улучшенная модель показывает меньшие потери при проверке на этапе проверки, особенно в случае потерь в граничном поле, что

подтверждает ее лучшую обобщающую способность при работе с неизвестными данными. Этот феномен говорит о том, что YOLOv8-LSTM более стабильна и точна в обнаружении и распознавании падающих целей. Таким образом, YOLOv8-LSTM имеет более плавное уменьшение потерь во время обучения и проверки, а также более высокие способности к обучению и обобщению, особенно в задаче обнаружения падений.

Как показано на рисунках 3.14. и 3.15., мы можем дополнительно проанализировать разницу в производительности моделей на основе графиков F1-кривых двух моделей:.

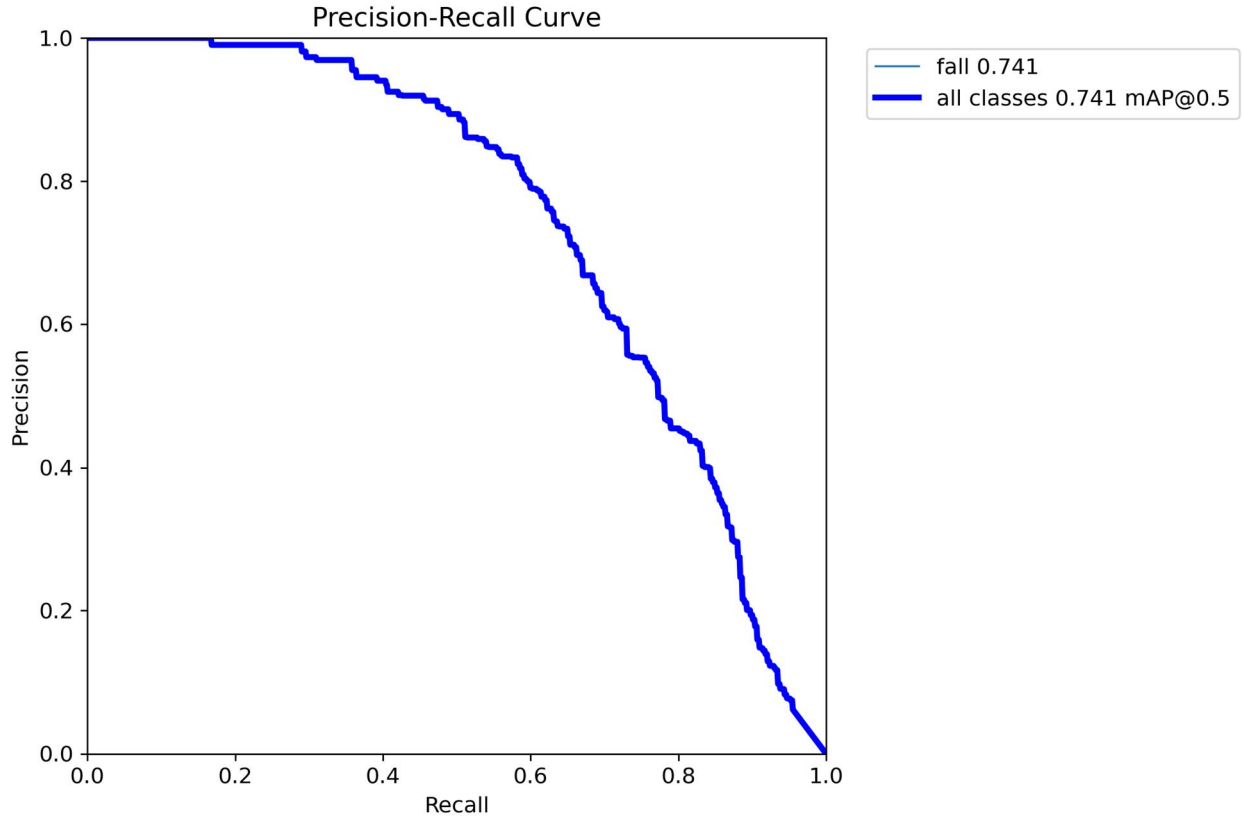


Рисунок 3.14. График PR-кривой для YOLOv8

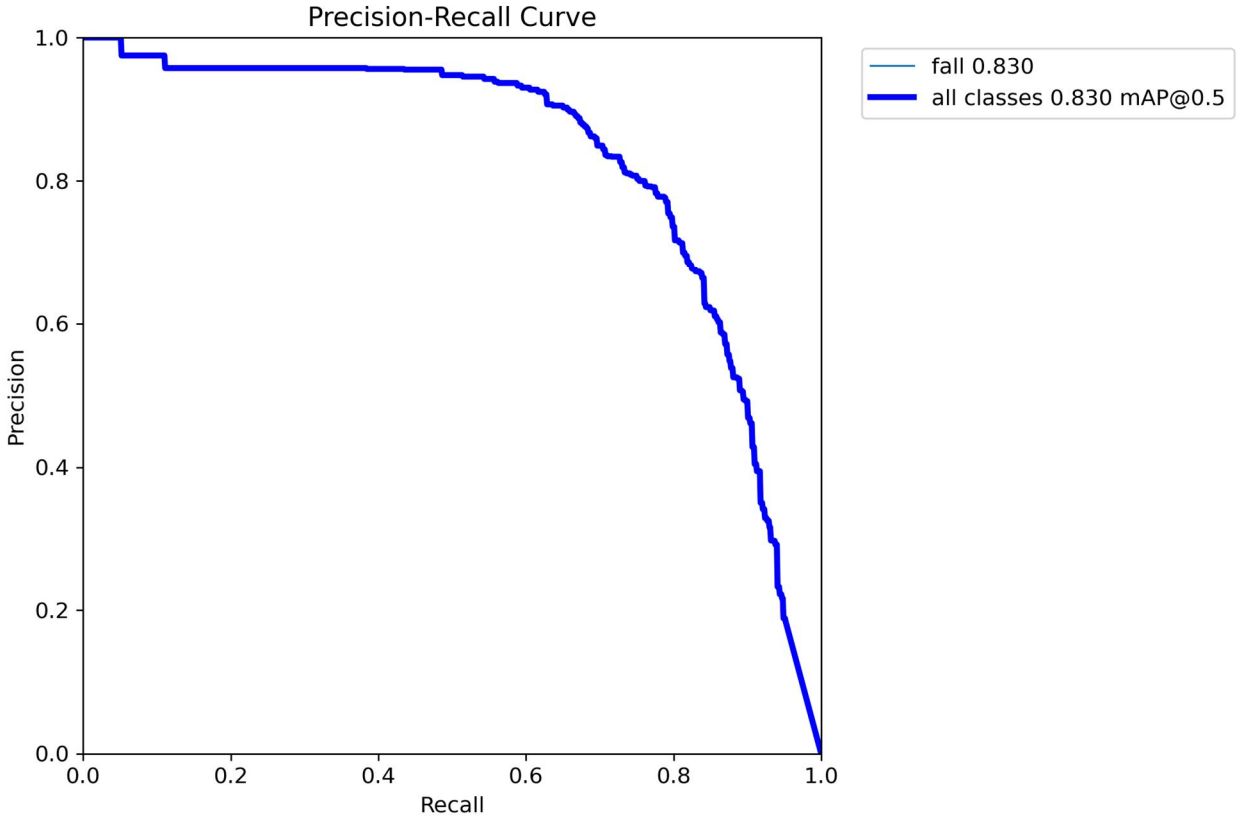


Рисунок 3.15. График PR-кривой YOLOv8-LSTM

На графике **Precision-Recall Curve** наблюдается значительная разница в производительности между YOLOv8 и YOLOv8-LSTM по всем категориям. На графике YOLOv8 значение **mAP@0.5** составляет 0.741, что указывает на средние показатели модели в плане баланса между точностью и отзывом. Хотя модель способна сохранять относительно высокий уровень точности, её отзыв остаётся недостаточно высоким, что говорит о пропусках при обнаружении некоторых целей. В целом, Precision-Recall Curve для YOLOv8 показывает, что при низком уровне отзыва модель всё ещё сохраняет высокую точность, однако при увеличении отзыва точность начинает значительно снижаться. Это свидетельствует о слабом контроле модели над ложными срабатываниями при высоком уровне отзыва.

Напротив, **Precision-Recall Curve** модели YOLOv8-LSTM демонстрирует улучшенные показатели, и **mAP@0.5** достигает 0.830, что значительно превышает результаты оригинальной модели. Это говорит о том, что улучшенная YOLOv8-LSTM способна поддерживать более высокий уровень отзыва, сохраняя при этом хорошую точность, что обеспечивает её превосходную общую производительность. Precision-Recall Curve модели YOLOv8-LSTM показывает, что даже при высоком уровне отзыва модель эффективно контролирует ложные срабатывания, а точность остаётся относительно стабильной. Это свидетельствует о лучшем балансе между точностью и способностью обнаруживать больше целей.

Разница в производительности может быть связана с оптимизацией архитектуры и процесса обучения модели YOLOv8-LSTM. В частности, внедрение модуля LSTM и механизма внимания позволило модели более эффективно распознавать и классифицировать падения. Использование LSTM улучшает способность модели захватывать временные зависимости, а механизм внимания усиливает фокусировку на значимых признаках. Эти усовершенствования повышают надёжность и точность обнаружения падений, что позволяет модели достигать лучшего баланса между точностью и отзывом, уменьшая как пропуски, так и ложные срабатывания.

Сравнительный анализ YOLOv8-LSTM с другими оригинальными версиями yolov8

В этом эксперименте мы провели сравнительный анализ четырех версий моделей YOLO: YOLOv8-LSTM, YOLOv8, YOLOv7 и YOLOv6, чтобы оценить их производительность в задаче обнаружения падений пожилых людей (Табл.3.3). Сравнивая различия в mAP, F1-Score, скорости вывода, количестве параметров и FLOPs, мы смогли более полно понять сильные и слабые стороны различных версий моделей YOLO, и в следующей таблице приведена сравнительная информация моделей.

Таблица 3.3. Сравнение производительности YOLOv8- LSTM с другими версиями YOLO

название (вещи)	**mAP**	**F1-Score**	**Размер изображения (пиксели)**	**mAPval 50-95**	**Скорость процессора ONNX (мс)**	**V100 TensorRT Скорость (мс)**	**Количество параметров (млн)**	**FLOPs (миллиарды)**
YOLOv8-lstm	0.830	0.78	640	34.3	73.6	1.06	3.2	8.7
YOLOv8	0.741	0.63	640	37.3	80.4	0.99	2.6	7.7
YOLOv7	0.730	0.625	640	37.5	90.6	1.56	4.7	11.4
YOLOv6	0.714	0.61	640	37.4	98.3	1.78	6.01	13.1

Сравнительный анализ mAP и F1-Score

Согласно результатам, приведенным в таблице 3.3., YOLOv8-LSTM демонстрирует хорошие результаты как по mAP (0,830), так и по F1-Score (0,78), что значительно лучше, чем у других версий модели YOLO. Напротив, YOLOv8, имеющая mAP 0,741 и F1-Score 0,63, работает немного хуже, чем YOLOv8-lstm, особенно в F1-Score. YOLOv7 и YOLOv6 имеют mAP 0,730 и 0,714, соответственно, а F1-Score 0,625 и 0,61, соответственно, показывая, что эти

модели не так хороши, как YOLOv8-LSTM, с точки зрения баланса точности обнаружения и запоминания.

Эти различия в производительности указывают на то, что YOLOv8- LSTM способен лучше понять и уловить динамические особенности при обнаружении падений и повысить точность и стабильность обнаружения с точки зрения оптимизированной структуры сети, извлечения признаков и стратегии обучения, особенно с включением модуля LSTM.

Сравнительный анализ скорости вывода и вычислительной эффективности

При скорости процессорного ONNX 73,6 мс и скорости A100 TensorRT 1,06 мс YOLOv8- LSTM имеет относительно быстрый вывод, особенно при ускорении с помощью TensorRT.

Выводы YOLOv8 выполняются немного медленнее: скорость процессорного ONNX составляет 80,4 мс по сравнению со скоростью A100 TensorRT, равной 0,99 мс. И хотя выводы на TensorRT выполняются немного быстрее, YOLOv8-lstm превосходит их на CPU.

В отличие от них, скорость вычислений YOLOv7 и YOLOv6 подробно не описана, но по количеству параметров и FLOP можно предположить, что они более требовательны к вычислениям, что может привести к более низкой скорости вычислений, особенно на CPU.

Сравнительный анализ сложности моделей и вычислительных усилий. Судя по количеству параметров и FLOPs в таблице, YOLOv8-LSTM имеет 3,2 миллиона параметров и 8,7 миллиарда FLOPs. По сравнению с YOLOv8 (2,6

миллиона параметров и 7,7 миллиарда FLOPs), YOLOv8-lstm имеет немного больше параметров и немного увеличил вычислительную сложность, но это увеличение произошло после оптимизации. оптимизирована и по-прежнему сохраняет более эффективную производительность.

YOLOv7 имеет 4,7 миллиона параметров и 11,4 миллиарда FLOP, в то время как YOLOv6 имеет 6,01 миллиона параметров и 13,1 миллиарда FLOP, что говорит о том, что большая вычислительная сложность и количество параметров в этих двух версиях может привести к снижению скорости вывода.

Хотя YOLOv7 и YOLOv6 имеют большее количество параметров и FLOP, они относительно неэффективны с вычислительной точки зрения, что может объяснять их более медленный вывод и худшие показатели F1-Score.

В результате проведенного анализа YOLOv8-LSTM превосходит другие версии модели YOLO по нескольким ключевым показателям, особенно по mAP, F1-Score и скорости вывода. Ее mAP достигает 0,830, а F1-Score - 0,78, что свидетельствует о ее превосходстве в точности и запоминании. Что касается скорости вывода, то хотя YOLOv8-lstm и YOLOv8 работают одинаково, YOLOv8-LSTM немного превосходит по производительности CPU, а скорость работы TensorRT является выдающейся. В отличие от них, YOLOv7 и YOLOv6 медленнее по количеству параметров и вычислительной сложности, и их производительность немного ниже, чем у YOLOv8-LSTM по mAP и F1-Score, поэтому улучшенный YOLOv8-lstm показывает лучшую производительность с точки зрения точности, эффективности и реального времени по сравнению с

оригинальными YOLOv8, YOLOv7 и YOLOv6. и YOLOv6, особенно в задаче обнаружения падения.

ВЫВОДЫ

В этой главе предлагается алгоритм обнаружения падений (YOLOv8-LSTM), основанный на сочетании улучшенного YOLOv8 и LSTM, а также подробно анализируется оригинальный алгоритм YOLOv8 с обсуждением его ограничений. Хотя YOLOv8 хорошо справляется с обнаружением целей, он все еще страдает от недостатков в динамических сценах, обнаружении мелких целей, захвате изменений жестов и моделировании временных рядов, особенно при работе с быстрыми, переходными и динамическими событиями, такими как поведение при падении. Для решения этих проблем в данной главе предлагаются три ключевых усовершенствования: внедрение механизма внимания ECA для улучшения извлечения признаков, использование GSConv для оптимизации сети слияния признаков и включение LSTM в структуру YOLOv8 для фиксации динамических изменений во временных рядах.

Благодаря этим улучшениям YOLOv8-LSTM не только повышает точность обнаружения падений, но и усиливает способность модели анализировать поведение при падении во временном ряду. YOLOv8-LSTM демонстрирует более быструю сходимость, более плавный спуск потерь и более высокую точность, чем оригинальная YOLOv8 в экспериментах. По сравнению с другими моделями обнаружения целей, такими как YOLOv5, YOLOv6 и YOLOv7, YOLOv8-LSTM демонстрирует хорошие результаты по нескольким аспектам, таким как mAP, F1-

Score, скорость вывода и вычислительная эффективность, и показывает значительное преимущество, особенно при решении задач обнаружения падений.

В целом, исследование, проведенное в этой главе, показывает, что, сочетая эффективную способность YOLOv8 к обнаружению целей с преимуществами LSTM в моделировании данных временных рядов, YOLOv8-LSTM может лучше справиться с проблемами обнаружения падений и обеспечивает эффективное решение для систем мониторинга падений с высокой точностью и требованиями реального времени.

ГЛАВА 4. ИНТЕЛЛЕКТУАЛЬНАЯ СИСТЕМА МОНИТОРИНГА ДЛЯ ОБНАРУЖЕНИЯ ПАДЕНИЙ НА ОСНОВЕ УЛУЧШЕННОГО YOLOV8-LSTM

4.1. Системный анализ

4.1.1. Анализ осуществимости

Основная задача этой системы основана на улучшенной модели YOLOv8 (YOLOv8- LSTM), применяемой для мониторинга и обнаружения событий падения в режиме реального времени. После проведения достаточного количества исследований и предварительных экспериментов была подтверждена целесообразность создания системы в следующих аспектах:

ТЕХНИЧЕСКАЯ ВОЗМОЖНОСТЬ. YOLOv8, как самый передовой одноэтапный алгоритм обнаружения целей, обладает эффективной скоростью вывода и высокой точностью. По сравнению с традиционными алгоритмами обнаружения целей, скорость обнаружения YOLOv8 значительно улучшена и может соответствовать требованиям реального времени. Благодаря использованию модели LSTM система способна анализировать информацию временных рядов в видеоданных и эффективно обнаруживать динамические изменения до и после падения. Кроме того, усовершенствования YOLOv8-LSTM, особенно внедрение механизма внимания ECA и модуля GSConv, еще больше повышают точность и устойчивость модели. Все эти методы были проверены в предварительных экспериментах.

ВОЗМОЖНОСТЬ ПОЛУЧЕНИЯ ДАННЫХ. В отличие от традиционных методов обнаружения падений, эта система опирается на видеоданные, а не на данные датчиков. Это обеспечивает богатые пространственные и временные характеристики для модели, позволяющие лучше определить динамику событий, связанных с падением. Видеоданные, используемые в системе, поступают из широкого спектра источников, включая как общедоступные наборы данных по обнаружению падений, так и самостоятельно собранные видеоданные из различных сценариев. Разнообразие и высокое качество этих наборов данных обеспечивают эффективность процесса обучения, повышают обобщающую способность модели и обеспечивают стабильную работу в различных условиях.

Аппаратная реализуемость: работа системы более гибкая с точки зрения требований к оборудованию, и она может адаптироваться к вычислительным средам различного масштаба. Хотя модель YOLOv8-LSTM более сложная, она может обеспечить достаточную поддержку при обработке видео в реальном времени при соответствующей конфигурации оборудования. Оптимизируя скорость вывода и вычислительную эффективность модели, система способна обеспечить обнаружение и обратную связь с низкой задержкой, что подходит для сценариев наблюдения в реальном времени и аварийного оповещения, позволяя системе эффективно работать на распространенных вычислительных платформах с хорошей аппаратной адаптацией.

Экономическая целесообразность: учитывая, что система использует фреймворк YOLOv8 и LSTM с открытым исходным кодом, стоимость разработки

относительно невысока. Благодаря популярности устройств видеонаблюдения система имеет низкую стоимость развертывания по сравнению с традиционными методами сбора и обработки сенсорных данных. Система не требует дополнительных аппаратных средств и может быть использована только со стандартными устройствами видеозахвата и вычислительными ресурсами, поэтому имеет хорошую экономическую целесообразность.

В совокупности система обладает хорошей реализуемостью с точки зрения технических, информационных, аппаратных и экономических аспектов и может быть успешно развернута и эксплуатироваться в реальных условиях.

4.1.2. Функциональный анализ

Эта система обнаружения падений представляет собой интуитивно понятную и эффективную платформу для обнаружения целей и распознавания поведения, объединяющую улучшенную модель YOLOv8-LSTM с графическим пользовательским интерфейсом, созданным Streamlit. Основные функции системы можно разделить на следующие части:

- **Загрузка и предварительная обработка моделей**

Система поддерживает загрузку улучшенной модели YOLOv8 (YOLOv8-LSTM) для обнаружения целей и предварительную обработку входного изображения или видеокадра. С помощью класса YOLOv8v5Detector система может загружать предварительно обученные модели, нормализовать и изменять размер изображений в соответствии с требованиями модели YOLOv8-LSTM. Модуль также способен адаптироваться к различным источникам данных

(например, к неподвижным изображениям, видеофайлам и сигналам с камер в реальном времени), что обеспечивает широкое применение системы в различных сценариях обнаружения.

- **Управление параметрами конфигурации**

В системе пользователь может динамически настраивать несколько параметров обнаружения с помощью класса Detection_UI. С помощью графического интерфейса пользователи могут настраивать такие параметры, как порог уверенности (conf_threshold), порог IOU (iou_threshold) и т. д. Настройка этих параметров может напрямую влиять на точность и производительность результатов обнаружения, позволяя системе адаптироваться к потребностям различных сред и повышать точность и эффективность обнаружения.

- **Переключение между несколькими моделями и их сравнение**

Помимо использования улучшенной модели YOLOv8-LSTM, система также позволяет пользователю переключаться между различными версиями моделей обнаружения целей YOLO в интерфейсе, такими как оригинальная модель YOLOv8, YOLOv7, YOLOv6, YOLOv5 и так далее. Пользователи могут выбрать наиболее подходящую модель для обнаружения в соответствии с потребностями различных сценариев. Эта функция обеспечивает большую гибкость системы при сравнении нескольких алгоритмов обнаружения целей и выборе оптимальной модели обнаружения, что позволяет еще больше повысить эффективность обнаружения.

- **Обработка изображений и видео**

Класс Detection_UI отвечает за обработку различных входных данных, включая видеопотоки в реальном времени с камеры, неподвижные изображения и видеофайлы, загруженные из файлов. Пользователь может выбрать необходимый метод ввода, и система автоматически обработает и отобразит соответствующие результаты обнаружения. Благодаря методу process_camera_or_file система может эффективно обрабатывать данные из различных источников и отображать результаты обнаружения в реальном времени для дальнейшего анализа.

- **Тесты и презентация результатов**

После обработки кадра изображения или видео вызывается метод frame_process для выполнения задачи предсказания модели YOLOv8-LSTM. Результаты обнаружения, выданные моделью, включая такую информацию, как ограничительные рамки, метки категорий и уровни доверия к целям, далее обрабатываются методом postprocess. Эти результаты передаются в графический интерфейс пользователя для представления, где пользователь может визуализировать результаты обнаружения для каждой цели и просмотреть соответствующую информацию по мере необходимости.

- **Ведение журнала и управление данными**

Классы LogTable и ResultLogger отвечают за сохранение подробной информации о каждом обнаружении, такой как время обнаружения, местоположение, уровень достоверности и т. д., так что пользователь может в

любой момент проверить историю записей об обнаружении. Кроме того, ResultLogger поддерживает объединение и форматирование результатов для дальнейшего анализа.

- **пользовательский интерфейс**

Интерфейс взаимодействия с пользователем, созданный Streamlit, делает использование системы более интуитивным и простым. Методы setup_sidebar и setupMainWindow отвечают за инициализацию схемы интерфейса, включая настройку модели, выбор источника входного сигнала, настройку режима отображения и т. д. Пользователь может быстро запустить процесс инспекции, просмотреть результаты и настроить параметры. Пользователи могут быстро запустить процесс проверки, просмотреть результаты и выполнить настройку параметров с помощью лаконичного интерфейса, что повышает удобство использования системы и улучшает впечатления от работы с ней.

Таким образом, данная система полностью отвечает потребностям задач обнаружения падений в практических приложениях, интегрируя модель YOLOv8-LSTM и поддерживая переключение между несколькими моделями, динамическую настройку параметров, обработку изображений и видео, обнаружение в реальном времени и ведение журнала. Система разработана как эффективная, гибкая и интеллектуальная, чтобы предоставить пользователям высокоточную платформу для обнаружения целей и распознавания поведения с низкой задержкой, способную справиться со всеми видами задач обнаружения в сложных условиях.

4.2. Общий дизайн архитектуры системы

Общая архитектура системы обнаружения падений показана на рисунке 4.1.. Система обеспечивает высокую эффективность, точность и работоспособность обнаружения падений благодаря совместной работе трех функциональных модулей: пользовательского интерфейса (UI), модели обнаружения, а также отображения и записи результатов. Функции каждого модуля независимы друг от друга и органично сочетаются, представляя собой законченное решение для обнаружения падений.

Рисунок 4.1. Схема построения архитектуры системы

4.2.1. Дизайн пользовательского интерфейса (UI)

Модуль пользовательского интерфейса - это уровень взаимодействия человека и компьютера в системе, цель которого - предоставить пользователям интуитивно понятный опыт работы. Созданный на основе фреймворка Streamlit, интерфейс системы прост и понятен, и пользователям легко быстро освоиться. В основном он включает в себя следующие подмодули:

Переключение режимов отображения: Обеспечение различных режимов отображения результатов (например, результаты тестирования в реальном времени, статичные результаты тестирования и т.д.) для удовлетворения потребностей пользователей в различных сценариях.

Веб-сборка: благодаря поддержке Streamlit веб-интерфейса пользователи могут получить доступ к системе непосредственно через браузер без необходимости установки сложного программного обеспечения.

Панель конфигурации: позволяет пользователям динамически настраивать параметры обнаружения (например, пороги уверенности, пороги IOU и т. д.) для оптимизации точности и производительности обнаружения.

Выбор источника ввода: поддерживает выбор пользователем различных методов ввода данных, включая потоковое видео в реальном времени (камера) и загрузку фотографий или видеофайлов.

Модуль пользовательского интерфейса обеспечивает высоконастраиваемое взаимодействие с пользователями, делая систему более гибкой и эффективной в различных сценариях.

4.2.2. Разработка модели обнаружения

Модель обнаружения - это основной модуль системы, который обеспечивает точное обнаружение целей с помощью алгоритмов глубокого обучения. Система использует улучшенную модель YOLOv8-LSTM и поддерживает переключение между несколькими моделями, включая оригинальную модель YOLOv8, YOLOv7, YOLOv6, YOLOv5 и так далее, чтобы адаптироваться к различным сценариям задач и требованиям к производительности.

Основные функции модуля Detection Models включают в себя:

Загрузка модели: система поддерживает загрузку улучшенной модели YOLOv8-LSTM, и в то же время может переключаться на другие модели обнаружения целей для удовлетворения потребностей сравнительных экспериментов и конкретных приложений.

Обучение и оценка модели: поддерживает обучение модели на наборе данных и предоставляет функцию оценки модели для проверки эффекта от улучшения модели.

Предварительная обработка изображений: система стандартизирует и изменяет размеры входных данных перед тем, как они попадают в модель, обеспечивая совместимость и оптимизируя обнаружение.

Вывод модели и постобработка: использование модели обнаружения для рассуждений о входном изображении или видеокадре, генерирование результатов обнаружения цели и оптимизация представления результатов с помощью постобработки.

Модуль модели обнаружения обеспечивает основную поддержку задачи обнаружения целей в системе за счет эффективного вывода модели и гибкого управления моделью.

4.2.3. Представление и регистрация результатов

Модуль "Представление и запись результатов" отвечает за представление результатов тестирования пользователю в интуитивно понятной и удобной для анализа форме, а также обеспечивает функции записи и экспорта для последующего использования. Основные функции включают:

Отображение изображений в реальном времени: отображение результатов обнаружения в реальном времени на интерфейсе системы, включая такую информацию, как ограничительная рамка цели, метки категорий и уровень доверия.

Фильтрация результатов: поддержка пользователей для фильтрации результатов тестирования в зависимости от времени обнаружения, категории цели и других условий, чтобы облегчить быстрое позиционирование ключевых данных.

Ведение журнала: записывает подробную информацию о каждом обнаружении, включая время обнаружения, категорию цели, информацию о

местоположении и т. д., предоставляя пользователям отслеживаемые исторические данные.

Экспорт результатов: поддерживает экспорт результатов тестирования в файлы структурированных данных (например, в формате CSV) для удобства последующего анализа и архивирования.

Благодаря модулю отображения и записи результатов пользователи могут интуитивно понять текущую ситуацию с инспекцией, а также просмотреть и проанализировать исторические данные, что повышает прикладную ценность системы.

4.3. Выбор среды разработки программного обеспечения и инструментов

Чтобы обеспечить бесперебойную разработку, эксплуатацию и развертывание системы обнаружения падений, данная система придерживается принципа эффективности и совместимости при выборе среды разработки, которая основана на управляющей Python-среде Anaconda и оснащена различными инструментами и библиотеками с открытым исходным кодом. Далее будет подробно рассказано о построении общей среды разработки, выборе ключевых программных инструментов и их использовании.

4.3.1. Сборка среды разработки

Среда разработки системы использует Anaconda в качестве инструмента управления средой Python. Anaconda - это платформа для разработки и исполнения Python/R data science с открытым исходным кодом и хорошими

функциями управления пакетами и средами, которая позволяет легко создавать и управлять несколькими виртуальными средами, что позволяет избежать проблемы конфликтов зависимостей между различными проектами. В данной системе Anaconda используется для создания автономной виртуальной среды Python для обучения и вывода модели YOLOv8-LSTM. В виртуальной среде установлены язык Python и соответствующие фреймворки глубокого обучения, инструменты обработки данных и визуализации, а информация о версиях и их использовании приведена в таблице 4.1.

Таблица 4.1. Программные библиотеки и использование среды разработки

библиотека программного обеспечения	требования к версионированию	использовать
факел	>= 1.8.0	Для построения и обучения сетей глубокого обучения и проведения выводов для обнаружения целей и анализа временных рядов.
torchvision	>= 0.9.0	Обеспечивает загрузку наборов данных изображений

		и предварительное обучение моделей для решения задач обнаружения целей.
TensorboardX	>= 2.5.1	Используется для визуализации структуры сети и изменения параметров в процессе обучения.
Streamlit	>= 1.26.0	Построить интерфейс взаимодействия с пользователем для реализации функции конфигурирования и отображения результатов системы в реальном времени.
Matplotlib	>= 3.2.2	Используется для создания графиков визуализации данных, включая

		статистический анализ и визуальное представление результатов тестирования.
numpy	>= 1.18.5	Обеспечение эффективных операций с массивами и библиотек математических функций для поддержки обработки данных и расчета моделей.
opencv-python	>= 4.1.2	Для обработки изображений и видеоданных, включая предварительную обработку исходных данных и маркировку

		результатов.
подушка	== 8.4.0	Используется для выполнения основных операций с изображениями, таких как загрузка, сохранение и преобразование.
pyyyaml	>= 5.3.1	Используется для чтения и разбора файлов YAML для удобного управления параметрами конфигурации системы.
запросы	>= 2.23.0	Используется для отправки HTTP-запросов для получения дополнительных данных или моделей из Интернета.

scipy	>= 1.4.1	Для передовых научных вычислений, включая обработку сигналов и реализацию оптимизационных алгоритмов.
tqdm	>= 4.41.0	Обеспечьте чистый и простой в использовании инструмент индикатора прогресса, чтобы повысить удобство работы с ним.
панды	>= 1.1.4	Предоставляет эффективные средства манипулирования данными для управления результатами тестов и

		анализа наборов данных.
сиборн	>= 0.11.0	Используется для создания расширенных статистических графиков, помогающих проводить углубленный анализ и представлять результаты тестов.

4.3.2. Выбор программных средств

- **Фреймворки для глубокого обучения**

Система использует PyTorch в качестве фреймворка глубокого обучения для поддержки построения, обучения и вывода моделей с помощью библиотек torch и torchvision. PyTorch известен своими динамическими вычислительными графами и эффективными тензорными операциями, которые особенно подходят для сложных задач глубокого обучения. В данной системе PyTorch используется для реализации улучшенной модели YOLOv8-LSTM с обнаружением целей и временным анализом видеоданных.

- **Средства разработки интерфейсов**

Пользовательский интерфейс системы разработан на основе Streamlit - легковесной библиотеки Python, предназначенной для работы с данными и интерактивными интерфейсами, которая позволяет быстро создавать динамические веб-интерфейсы. Streamlit позволяет пользователям динамически настраивать параметры инспекции, выбирать источники входных данных и просматривать результаты инспекции в режиме реального времени, что значительно повышает удобство использования системы.

- **Инструменты для обработки и визуализации данных**

numpy и pandas: для эффективного управления данными и их анализа numpy предоставляет базовые функции работы с массивами, а pandas облегчает табулирование и статистический анализ результатов тестов.

Matplotlib и seaborn: используются для создания визуальных графиков результатов тестирования. matplotlib поддерживает создание широкого спектра 2D- и 3D-графиков, а seaborn с его расширенными возможностями статистической графики способен визуализировать распределение и тенденции результатов тестирования системы.

- **Инструменты для обработки изображений и видео**

Система использует opencv-python и pillow для обработки изображений и видеоданных. opencv-python используется для чтения и обработки видеопотоков и выполнения задач обнаружения на уровне кадра, а pillow обеспечивает базовые

функции манипулирования изображениями для загрузки и сохранения обнаруженных изображений.

- **Средства визуализации обучения**

Во время обучения модели TensorboardX используется для записи и визуализации структуры сети, изменения параметров и показателей производительности в процессе обучения. Внедрение этого инструмента облегчает разработчикам отладку и оптимизацию модели.

- **помощь**

tqdm: предоставляет пользователям функцию лаконичного отображения прогресс-бара для повышения удобства работы.

Запросы: поддержка в получении необходимых данных или моделей из внешних источников данных или онлайн-ресурсов.

pyyaml: помогает системе загружать и разбирать конфигурационные файлы в формате YAML для удобства управления параметрами конфигурации.

4.4. Реализация функциональности системы

В этом разделе подробно описан процесс реализации каждого функционального модуля интеллектуальной системы мониторинга для обнаружения падений на основе улучшенной YOLOv8 в соответствии с блок-схемой системы. Функциональные модули системы включают инициализацию, настройку параметров системы, выбор источника входного сигнала, обработку данных, предсказание модели, представление результатов, запись результатов

обнаружения и взаимодействие с пользователем. Детали реализации каждого модуля описаны ниже.

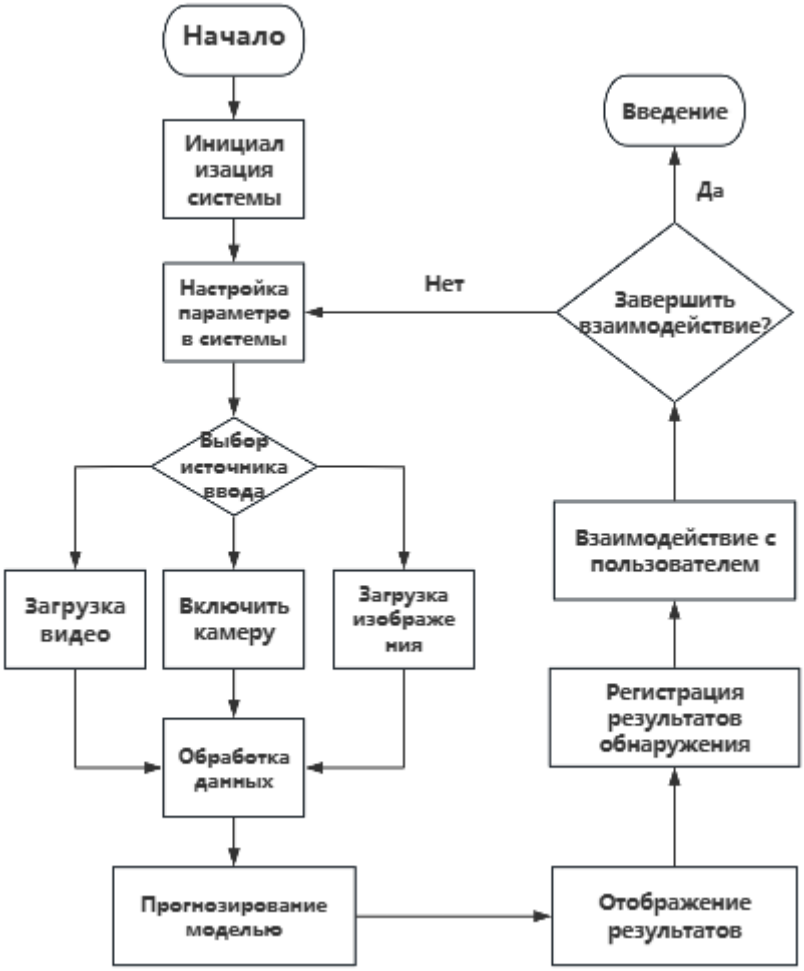


Рисунок 4.2. Блок-схема системы

4.4.1. Инициализация и конфигурация системы

Во-первых, при запуске система инициализируется, чтобы обеспечить правильную загрузку различных модулей и аппаратных устройств, а интерфейс конфигурации позволяет пользователю задать параметры. Как показано на рисунке 4.2., здесь пользователь может выбирать различные режимы работы и настраивать параметры системы (например, выбирать источники входного сигнала, устанавливать пороговые значения тревоги и т. д.).

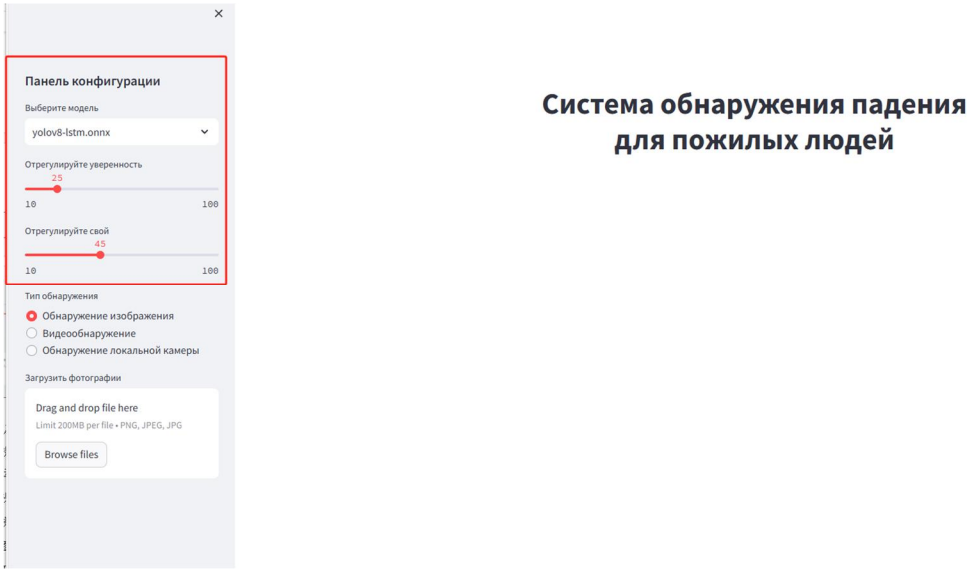


Рисунок 4.3. Схема панели конфигурации

4.4.2. Выбор источника входного сигнала

Система предоставляет три метода выбора источника ввода, как показано рисунке 4.3., так что пользователь может выбрать подходящий метод ввода для обработки данных в зависимости от потребностей. После выбора поток данных переходит в следующий модуль.

Загрузить видео: пользователь выбирает загрузку видеофайла, записанного в истории.

Включите камеру: система будет захватывать видеопоток в реальном времени через камеру.

Загрузка изображения: пользователь загружает одно изображение для обработки.

4.4.3. Прогнозы модели

Далее система передает обработанные данные в улучшенную модель YOLOv8 для прогнозирования, идентификации людей на видео и определения факта падения.

Обнаружение по фотографии: система поддерживает загрузку фотографий для обнаружения падений. Как показано на рисунке 4.4, пользователь может выбрать фотографию с поведением пожилого человека с локального устройства и загрузить ее на веб-страницу, система автоматически проанализирует фотографию и отметит обнаруженное поведение при падении. Эта функция подходит для анализа конкретных фотографий и проверки способности модели к обнаружению.

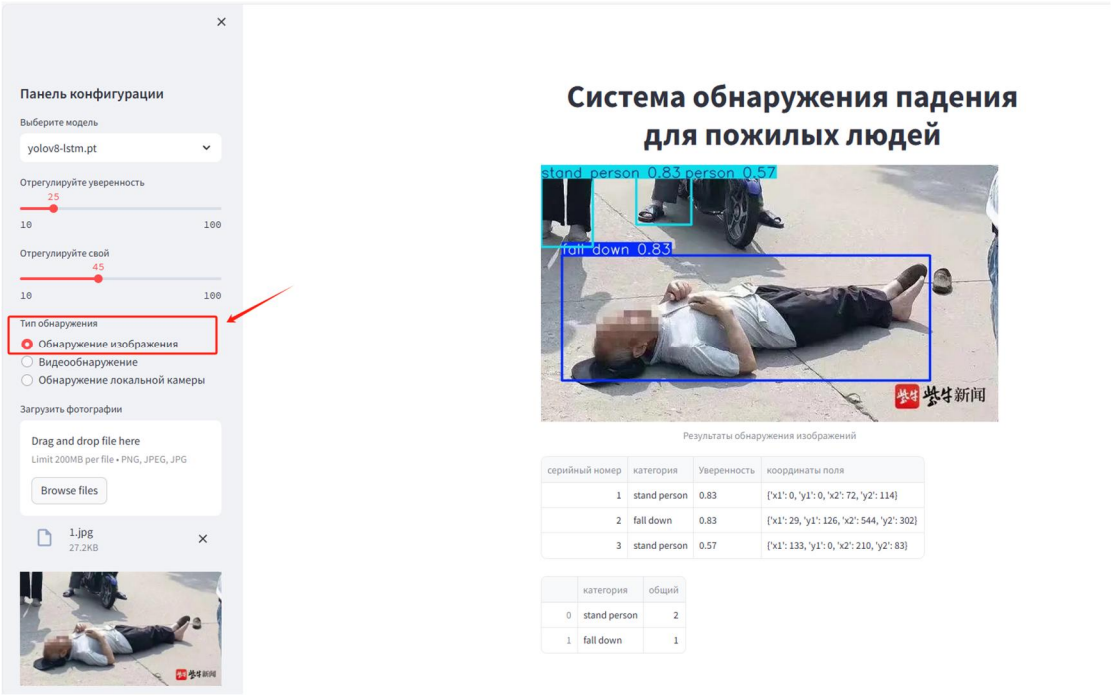


Рисунок 4.4. Обнаружение на изображении

Выбор видеофайла для обнаружения: Помимо мониторинга камер в режиме реального времени и обнаружения изображений, эта система позволяет

пользователям загружать видеофайлы для обнаружения падений. Пользователи могут выбрать видеофайл с поведением пешеходов, и система покадрово проанализирует видеоматериал, чтобы выявить и отметить события, связанные с падением. Эта функция помогает проанализировать ситуацию с безопасностью пешеходов за определенный период времени и полезна для обратного анализа исторических данных.

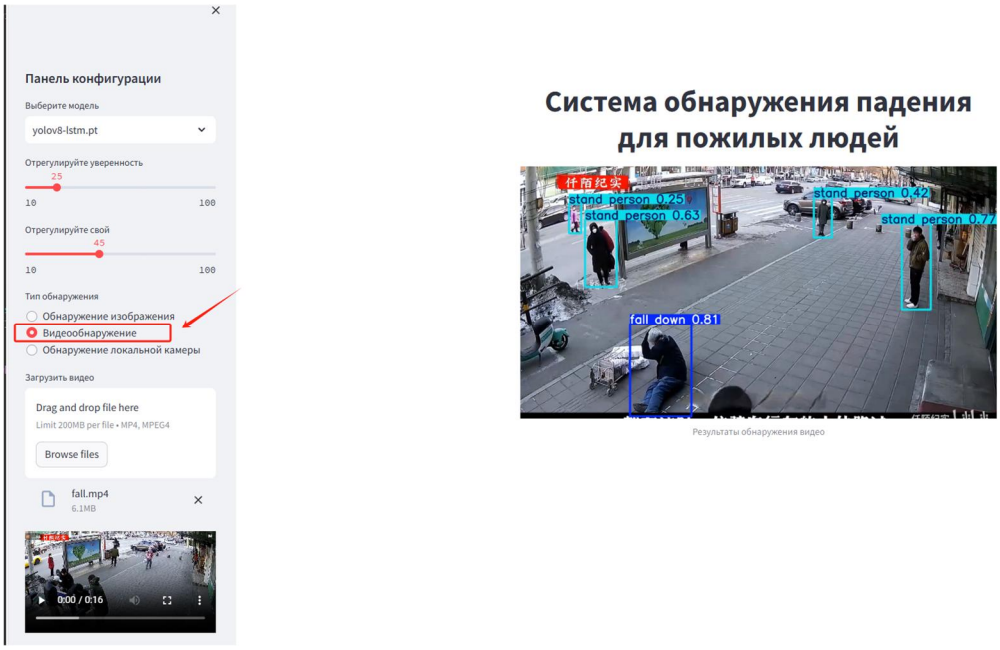


Рисунок 4.5. Обнаружение на видео

Включение камеры для обнаружения в режиме реального времени: эта система обеспечивает функцию включения камеры одним щелчком мыши для обнаружения падения пешехода на экране в режиме реального времени. Пользователю достаточно нажать соответствующую кнопку на веб-странице, чтобы активировать камеру и немедленно начать наблюдать за поведением пешехода при падении на экране. Эта функция особенно важна в ситуациях, требующих мониторинга в реальном времени (например, в местах проживания

пожилых людей, общественных местах и т. д.), чтобы своевременно обнаружить падение и поддержать оперативное реагирование.

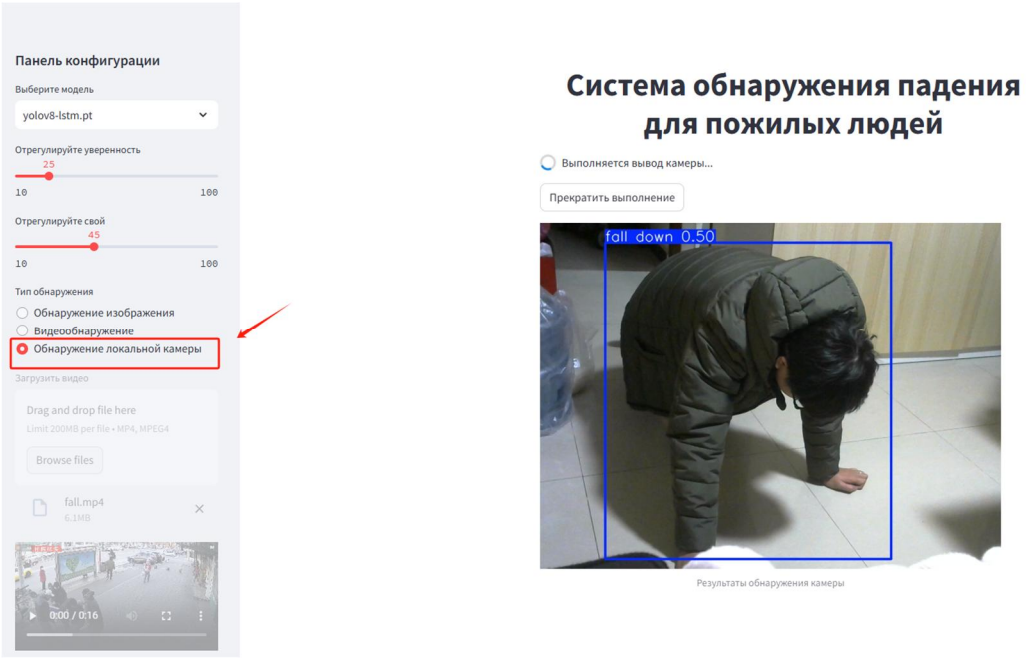


Рисунок 4.6. Обнаружение с камеры

4.4.4. Представление результатов

После предсказания модели система отображает результаты обнаружения, включая информацию о том, было ли обнаружено падение, и подсказывает пользователю с помощью интерфейса.

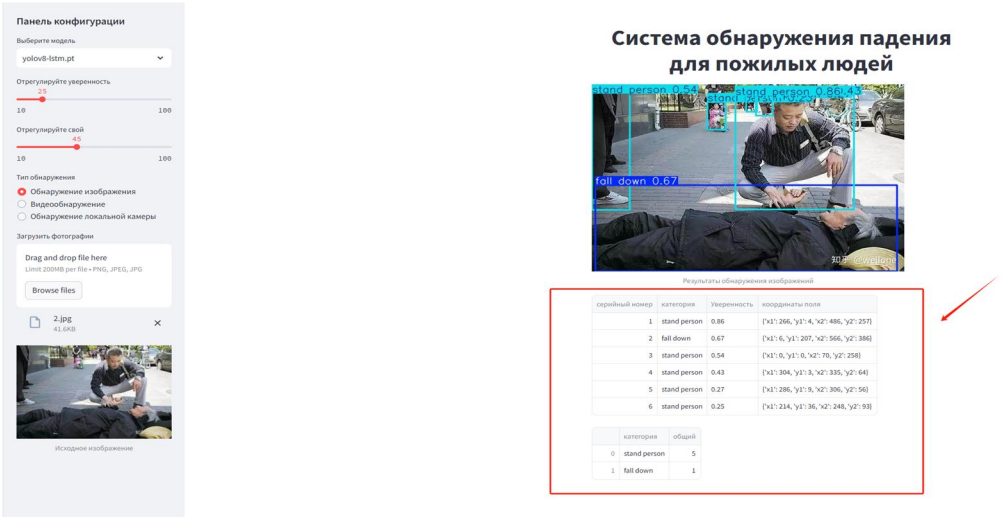


Рисунок 4.7. Результаты обнаружения

4.4.5. Регистрация результатов испытаний

Чтобы облегчить последующий анализ и запись, система сохраняет обнаруженные события падения. Записанная информация включает время события, результат обнаружения и тип источника входного сигнала.

	A	B	C	D	E
1	путь к файлу	Результат	Расположение	Уверенность	время
2	Camera: 0	падать	[137, 183, 293, 317]	0.8390421271324158	0.0169997215270996
3	Camera: 0	падать	[300, 120, 552, 313]	0.8722308874130249	0.0169997215270996
4	Camera: 0	падать	[137, 183, 293, 317]	0.8390796184539795	0.017536640167236328
5	Camera: 0	падать	[300, 120, 552, 313]	0.8721989393234253	0.017536640167236328
6	Camera: 0	падать	[137, 183, 293, 317]	0.8390421271324158	0.021044254302978516
7	Camera: 0	падать	[300, 120, 552, 313]	0.8722308874130249	0.021044254302978516
8	Camera: 0	падать	[137, 183, 293, 317]	0.8390796184539795	0.017000675201416016
9	Camera: 0	падать	[300, 120, 552, 313]	0.8721989393234253	0.017000675201416016
10	Camera: 0	падать	[137, 183, 293, 317]	0.8390528559684753	0.018999338150024414
11	Camera: 0	падать	[300, 120, 552, 313]	0.872211754322052	0.018999338150024414
12	Camera: 0	падать	[137, 183, 293, 317]	0.839103639125824	0.016999244689941406
13	Camera: 0	падать	[300, 120, 552, 313]	0.8722010254859924	0.016999244689941406
14	Camera: 0	падать	[137, 183, 293, 317]	0.8390816450119019	0.01700282096862793
15	Camera: 0	падать	[300, 120, 552, 313]	0.8721992373466492	0.01700282096862793
16	Camera: 0	падать	[137, 183, 293, 317]	0.8390882611274719	0.019999265670776367
17	Camera: 0	падать	[300, 120, 552, 313]	0.8721900582313538	0.019999265670776367
18	Camera: 0	падать	[137, 183, 293, 317]	0.8392011523246765	0.016000032424926758
19	Camera: 0	падать	[300, 120, 552, 313]	0.8722130656242371	0.016000032424926758
20	Camera: 0	падать	[137, 183, 293, 316]	0.8373792171478271	0.015997648239135742
21	Camera: 0	падать	[300, 120, 552, 313]	0.8720425367355347	0.015997648239135742
22	Camera: 0	падать	[329, 363, 638, 500]	0.799804151058197	0.0189971923828125
23	Camera: 0	падать	[329, 363, 638, 500]	0.7997873425483704	0.021999120712280273
24	Camera: 0	падать	[329, 363, 638, 500]	0.7998053431510925	0.016995668411254883
25	Camera: 0	падать	[329, 363, 638, 500]	0.7997495532035828	0.01900172233581543

Рисунок 4.8. Запись результатов обнаружения

ВЫВОДЫ

В этой главе описывается разработка и реализация интеллектуальной системы мониторинга для обнаружения падений на основе улучшенного YOLOv8-LSTM. Система сочетает в себе эффективное обнаружение целей YOLOv8 и возможности анализа временных рядов LSTM, а также обеспечивает функции мониторинга видеопотока, загрузки изображений и обнаружения в реальном времени с помощью пользовательского интерфейса, созданного Streamlit. Система поддерживает переключение между несколькими моделями, динамическую настройку параметров и обработку данных в реальном времени, обладая высокой гибкостью и адаптивностью. После анализа целесообразности система имеет хорошие условия для реализации с точки зрения технологии, данных, аппаратного обеспечения и экономики, и может удовлетворить потребности в мониторинге в реальном времени и обнаружении падений. Благодаря модульной конструкции система обеспечивает эффективное и точное обнаружение падения с высокой стабильностью и расширяемостью, что подходит для различных потребностей в практических сценариях применения.

ЗАКЛЮЧЕНИЕ

В диссертации исследуется применение методов компьютерного интеллекта в автоматизированных системах в области обнаружения падений, предлагается метод обнаружения падений, основанный на улучшенном YOLOv8 и LSTM, и реализуется интеллектуальная система мониторинга. Основные материалы изложены вопросах:

1.Рассматривается применение автоматизации и искусственного интеллекта в мониторинге здоровья, анализируются преимущества и недостатки традиционных методов и технологий искусственного интеллекта.

2.Рассматриваются вопросы глубокого обучения и анализа временных рядов, описываются принципы работы моделей YOLOv8 и LSTM и их преимущества при обнаружении падений.

3.Для повышения точности обнаружения и временной устойчивости модели предлагается усовершенствованный алгоритм YOLOv8-LSTM с использованием механизма внимания ECA и GSConv.

4.Для интеграции улучшенной модели обнаружения и проверки ее эффективности в практическом применении была разработана автоматизированная компьютерная система интеллектуального мониторинга.

Будущие исследования могут быть направлены на дальнейшую оптимизацию эффективности и обобщающей способности модели, а также на изучение мультимодального слияния данных для повышения полезности системы обнаружения.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Intelligent Control Systems for Industrial Automation and Robotics [Текст] / V. Biradar, A. Al-Jiboory, G. Sahu и др. // 2023 10th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON). – 2023. – P.1238–1243. – DOI: https://doi.org/10.1109/UPCON59197.2023.10434927.

2. Chen, J. Integrations between autonomous systems and modern computing techniques: a mini review [Текст] / J. Chen, M. Abbod, J.S. Shieh // Sensors. – 2019. – Vol.19, №18. – P.3897.

3. Aste, N. Building Automation and Control Systems and performance optimization: A framework for analysis [Текст] / N. Aste, M. Manfren, G. Marenzi // Renewable and Sustainable Energy Reviews. – 2017. – Vol.75. – P.313–330.

4. The role of software automation in improving industrial operations and efficiency [Текст] / D. Ajiga, P.A. Okeleke, S.O. Folorunsho, C. Ezeigweneme // International Journal of Engineering Research Updates. – 2024. – Vol.7, №1. – P.22–35.

5. Mourtzis, D. A Literature Review of the Challenges and Opportunities of the Transition from Industry 4.0 to Society 5.0 [Текст] / D. Mourtzis, J. Angelopoulos, N. Panopoulos // Energies. – 2022. – Vol.15, №17. – P.6276.

6. Automated Machine Learning (AutoML) in industry 4.0, 5.0, and society 5.0: Applications, opportunities, challenges, and future directions [Текст] / J. Rane, S.K. Mallick, O. Kaya, N.L. Rane // Future Research Opportunities for Artificial

Intelligence in Industry 4.0 and. – 2024. – Vol.5. – P.2.

7. Vetrivel, S.C. Industry 5.0 From Automation to Autonomy: Engineering the Shift [Текст] / S.C. Vetrivel, T. Mohanasundaram // In: Innovations in Engineering and Food Science. – IGI Global, 2024. – P.88–118.

8. Evolution of software in automated production systems: Challenges and research directions [Текст] / B. Vogel-Heuser, A. Fay, I. Schaefer, M. Tichy // Journal of Systems and Software. – 2015. – Vol.110. – P.54–84.

9. Parasuraman, R. Humans: Still vital after all these years of automation [Текст] / R. Parasuraman, C.D. Wickens // In: Decision Making in Aviation. – Routledge, 2017. – P.251–260.

10. A review on wire arc additive manufacturing: Monitoring, control and a framework of automated system [Текст] / C. Xia, Z. Pan, J. Polden и др. // Journal of manufacturing systems. – 2020. – Vol.57. – P.31–45.

11. 王焱. 智能制造的基础, 组成及发展途径 [Текст] / 王焱, 王湘念 // 航空制造技术. – 2015. – №58(13). – P.32–37.

12. 刘强. 智能制造理论体系架构研究 [Текст] / 刘强 // 中国机械工程. – 2020. – Vol.31(1). – P.24–36.

13. Djuric, A.M. A framework for collaborative robot (CoBot) integration in advanced manufacturing systems [Текст] / A.M. Djuric, R.J. Urbanic, J.L. Rickli // SAE International Journal of Materials and Manufacturing. – 2016. – Vol.9, №2. – P.457–464.

14. Lu, Y. Smart manufacturing process and system automation–a critical

review of the standards and envisioned scenarios [Текст] / Y. Lu, X. Xu, L. Wang // Journal of Manufacturing Systems. – 2020. – Vol.56. – P.312–325.

15. Artificial perception built on memristive system: Visual, auditory, and tactile sensations [Текст] / X. Ji, X. Zhao, M.C. Tan, R. Zhao // Advanced Intelligent Systems. – 2020. – Vol.2, №3. – P.1900118.

16. Albustanji, R.N. Robotics: five senses plus one—an overview [Текст] / R.N. Albustanji, S. Elmanaseer, A.A. Alkhatib // Robotics. – 2023. – Vol.12, №3. – P.68.

17. Brain-computer interface-based humanoid control: A review [Текст] / V. Chamola, A. Vineet, A. Nayyar, E. Hossain // Sensors. – 2020. – Vol.20, №13. – P.3620.

18. Architecture of a wheelchair control system for disabled people: Towards multifunctional robotic solution with neurobiological interfaces [Текст] / V.E. Karpov, D.G. Malakhov, A.D. Moscowsky и др. // Современные технологии в медицине. – 2019. – Vol.11, №1(eng). – P.90–100.

19. Callejas-Cuervo, M. Control systems and electronic instrumentation applied to autonomy in wheelchair mobility: The state of the art [Текст] / M. Callejas-Cuervo, A.X. González-Cely, T. Bastos-Filho // Sensors. – 2020. – Vol.20, №21. – P.6326.

20. Gatouillat, A. Smart and safe self-adaption of connected devices based on discrete controllers [Текст] / A. Gatouillat, Y. Badr, B. Massot // IET Software. – 2019. – Vol.13, №1. – P.49–59.

21. Semantics-based platform for context-aware and personalized robot

interaction in the internet of robotic things [Текст] / C. Mahieu, F. Ongenae, F. De Backere и др. // Journal of Systems and Software. – 2019. – Vol.149. – P.138–157.

22. Smarter grid in the 5G Era: A framework integrating power internet of things with a cyber physical system [Текст] / Y. Liu, X. Yang, W. Wen, M. Xia // Frontiers in Communications and Networks. – 2021. – Vol.2. – P.689590.

23. 何涛. 自动化仪表控制系统技术研究 [Текст] / 何涛 // 机械与电子控制工程. – 2024. – Vol.6(18). – P.104–106.

24. Aleksic, S. A survey on optical technologies for IoT, smart industry, and smart infrastructures [Текст] / S. Aleksic // Journal of Sensor and Actuator Networks. – 2019. – Vol.8, №3. – P.47.

25. A review of modern communication technologies for digital manufacturing processes in industry 4.0 [Текст] / T.R. Kurfess, C. Saldana, K. Saleeby, M.P. Dezfouli // Journal of Manufacturing Science and Engineering. – 2020. – Vol.142, №11. – P.110815.

26. Bansal, R. Communication protocols used for industrial automation [Текст] / R. Bansal, A.K. Dubey // In: Computational Intelligence in the Industry 4.0. – CRC Press, 2024. – P.73–94.

27. 吴龙飞. 智能技术在计算机信息系统中的应用 [Текст] / 吴龙飞 // 电子通信与计算机科学. – 2024. – Vol.6(10). – P.73–75.

28. 面向人本智造的新一代操作工: 参考架构, 使能技术与典型场景 [Текст] / 黄思翰, 王柏村, 张美迪 и др. // 机械工程学报. – 2022. – Vol.58(18). – P.251–264.

29. Engineering platforms for synthetic biology research [Текст] / C.U.I. Jinming, Z.H.A.N.G. Bingzhao, M.A. Yingfei и др. // Bulletin of Chinese Academy of Sciences (Chinese Version). – 2018. – Vol.33(11). – P.1249–1257.

30. 樊春良. 新中国科技现代化之路探析 [Текст] / 樊春良 // 中国科学院院刊. – 2024. – Vol.39(10). – P.1678–1698.

31. 何萍. 何谓新质生产力 [Текст] / 何萍 // 武汉大学学报 (哲学社会科学版). – 2024. – №5.

32. Borghesan, F. Unmanned and autonomous systems: Future of automation in process and energy industries [Текст] / F. Borghesan, M. Zagorowska, M. Mercangöz // IFAC-PapersOnLine. – 2022. – Vol.55, №7. – P.875–882.

33. Boppana, V.R. Industry 4.0: Revolutionizing the Future of Manufacturing and Automation [Текст] / V.R. Boppana // Innovative Computer Sciences Journal. – 2024. – Vol.10, №1.

34. Introduction to the special section: convergence of automation technology, biomedical engineering, and health informatics toward the healthcare 4.0 [Текст] / Z. Pang, G. Yang, R. Khedri, Y.T. Zhang // IEEE Reviews in Biomedical Engineering. – 2018. – Vol.11. – P.249–259.

35. Artificial intelligence (AI) and internet of medical things (IoMT) assisted biomedical systems for intelligent healthcare [Текст] / P. Manickam, S.A. Mariappan, S.M. Murugesan и др. // Biosensors. – 2022. – Vol.12, №8. – P.562.

36. A survey on health monitoring systems for health smart homes [Текст] / H. Mshali, T. Lemlouma, M. Moloney, D. Magoni // International Journal of Industrial

Ergonomics. – 2018. – Vol.66. – P.26–56.

37. Al-Turjman, F. Intelligence in the Internet of Medical Things era: A systematic review of current and future trends [Текст] / F. Al-Turjman, M.H. Nawaz, U.D. Ulusar // Computer Communications. – 2020. – Vol.150. – P.644–660.

38. Ren, L. Research of fall detection and fall prevention technologies: A systematic review [Текст] / L. Ren, Y. Peng // IEEE Access. – 2019. – Vol.7. – P.77702–77722.

39. Rajagopalan, R. Fall prediction and prevention systems: recent trends, challenges, and future research directions [Текст] / R. Rajagopalan, I. Litvan, T.P. Jung // Sensors. – 2017. – Vol.17, №11. – P.2509.

40. Vallabh, P. Fall detection monitoring systems: a comprehensive review [Текст] / P. Vallabh, R. Malekian // Journal of Ambient Intelligence and Humanized Computing. – 2018. – Vol.9, №6. – P.1809–1833.

41. Sensor technologies for fall detection systems: A review [Текст] / A. Singh, S.U. Rehman, S. Yongchareon, P.H.J. Chong // IEEE Sensors Journal. – 2020. – Vol.20, №13. – P.6889–6919.

42. 尤政. 智能传感器技术的研究进展及应用展望 [Текст] / 尤政 // 科技导报. – 2016. – Vol.34(17). – P.72–78.

43. SmartFall: A smartwatch-based fall detection system using deep learning [Текст] / T.R. Mauldin, M.E. Canby, V. Metsis и др. // Sensors. – 2018. – Vol.18, №10. – P.3363.

44. Gharghan, S.K. A comprehensive review of elderly fall detection using

wireless communication and artificial intelligence techniques [Текст] / S.K. Gharghan, H.A. Hashim // Measurement. – 2024. – Article 114186.

45. Tao, S. Privacy-preserved behavior analysis and fall detection by an infrared ceiling sensor network [Текст] / S. Tao, M. Kudo, H. Nonaka // Sensors. – 2012. – Vol.12, №12. – P.16920–16936.

46. Hegde, N. A comparative review of footwear-based wearable systems [Текст] / N. Hegde, M. Bries, E. Sazonov // Electronics. – 2016. – Vol.5, №3. – P.48.

47. 基于卷积神经网络和多判别特征的跌倒检测算法 [Текст] / 王鑫, 郑晓岩, 高焕兵 и др. // 计算机辅助设计与图形学学报. – 2023. – Vol.35(3). – P.452–462.

48. 钱慧芳. 基于深度学习的人体动作识别综述 [Текст] / 钱慧芳, 易剑平, 付云虎 // Journal of Frontiers of Computer Science & Technology. – 2021. – Vol.15(3).

49. Fall detection and activity recognition using human skeleton features [Текст] / H. Ramirez, S.A. Velastin, I. Meza и др. // IEEE Access. – 2021. – Vol.9. – P.33532–33542.

50. Salimi, M. Using deep neural networks for human fall detection based on pose estimation [Текст] / M. Salimi, J.J. Machado, J.M.R. Tavares // Sensors. – 2022. – Vol.22, №12. – P.4544.

51. 融合注意力机制的 OpenPose 人体跌倒检测算法 [Текст] / 孟彩霞, 薛洪秋, 石磊 и др. // 计算机辅助设计与图形学学报.

52. Fall detection based on key points of human-skeleton using openpose

[Текст] / W. Chen, Z. Jiang, H. Guo, X. Ni // Symmetry. – 2020. – Vol.12, №5. – P.744.

53. 基于递归神经网络的跌倒检测系统 [Текст] / 牛德姣, 刘亚文, 蔡涛 и др. // 智能系统学报. – 2018. – Vol.13(3). – P.380–387.

54. 宋凯. 深度学习方法在运动健康领域的应用 [Текст] / 宋凯 // Advances in Applied Mathematics. – 2023. – Vol.12. – P.3327.

55. 肖雨晴. 目标检测算法在交通场景中应用综述 [Текст] / 肖雨晴, 杨慧敏 // Journal of Computer Engineering & Applications. – 2021. – Vol.57(6).

56. 赵朗月. 基于机器视觉的表面缺陷检测方法研究进展 [Текст] / 赵朗月, 吴一全 // 仪器仪表学报. – 2023. – Vol.43(1). – P.198–219.

57. NETWORK, B.O.C.N. 基于卷积神经网络的机械故障诊断方法综述 [Текст] / B.O.C.N. NETWORK // Journal of Mechanical Strength. – 2020. – Vol.42(5). – P.1024–1032.

58. 多尺度目标检测的深度学习研究综述 [Текст] / 陈科圻, 朱志亮, 邓小明 и др. // 软件学报. – 2020. – Vol.32(4). – P.1201–1227.

59. 基于深度学习的通用目标检测研究综述 [Текст] / 程旭, 宋晨, 史金钢 и др. // 电子学报. – 2021. – Vol.49(7). – P.1428–1438.

60. 张顺. 深度卷积神经网络的发展及其在计算机视觉领域的应用 [Текст] / 张顺, 龚怡宏, 王进军 // 计算机学报. – 2019. – Vol.42(3). – P.453–482.

61. 深度学习目标检测方法及其主流框架综述 [Текст] / 段仲静, 李少波, 胡建军 и др. // Laser & Optoelectronics Progress. – 2020. – Vol.57(12). – P.120005.

62. Choudhury, S. Vehicle detection and counting using haar feature-based

classifier [Текст] / S. Choudhury, S.P. Chattopadhyay, T.K. Hazra // 2017 8th annual industrial automation and electromechanical engineering conference (IEMECON). – 2017. – P.106–109.

63. Zheng, C. Fatigue driving detection based on Haar feature and extreme learning machine [Текст] / C. Zheng, B. Xiaojuan, W. Yu // The Journal of China Universities of Posts and Telecommunications. – 2016. – Vol.23(4). – P.91–100.

64. Histogram of oriented gradients for human detection in video [Текст] / T. Surasak, I. Takahiro, C.H. Cheng и др. // 2018 5th International conference on business and industrial research (ICBIR). – 2018. – P.172–176.

65. Histogram of oriented gradients feature extraction from raw bayer pattern images [Текст] / W. Zhou, S. Gao, L. Zhang, X. Lou // IEEE Transactions on Circuits and Systems II: Express Briefs. – 2020. – Vol.67, №5. – P.946–950.

66. Krizhevsky, A. ImageNet Classification with Deep Convolutional Neural Networks [Текст] / A. Krizhevsky, I. Sutskever, G.E. Hinton // Advances in Neural Information Processing Systems (NeurIPS). – 2012. – Vol.25.

67. OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks [Текст] / P. Sermanet, D. Eigen, X. Zhang и др. // International Conference on Learning Representations (ICLR). – 2014.

68. Rich feature hierarchies for accurate object detection and semantic segmentation [Текст] / R. Girshick, J. Donahue, T. Darrell, J. Malik // IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – 2014.

69. Girshick, R. Fast R-CNN [Текст] / R. Girshick // IEEE International

Conference on Computer Vision (ICCV). – 2015.

70. You Only Look Once: Unified, Real-Time Object Detection [Текст] / J. Redmon, S. Divvala, R. Girshick, A. Farhadi // CVPR. – 2016.

71. Redmon, J. YOLO9000: Better, Faster, Stronger [Текст] / J. Redmon, A. Farhadi // CVPR. – 2017.

72. Redmon, J. YOLOv3: An Incremental Improvement [Текст] / J. Redmon, A. Farhadi // arXiv:1804.02767. – 2018.

73. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks [Текст] / S. Shaoqing Ren, K. He, R. Girshick, J. Sun // NIPS (NeurIPS). – 2015.

74. SSD: Single Shot MultiBox Detector [Текст] / W. Liu, D. Anguelov, D. Erhan и др. // European Conference on Computer Vision (ECCV). – 2016.

75. Accelerometer-based fall detection using optimized ZigBee data streaming [Текст] / B. Marco и др. // Microelectronics Journal. – 2010. – Vol.41(11). – P.703–710.

76. Khekan, A.R. The impact of YOLO Algorithms within fall detection application: A review [Текст] / A.R. Khekan, H.S. Aghdasi, P. Salehpour // IEEE Access. – 2024.

77. A Review of Yolo algorithm developments [Текст] / P. Jiang, D. Ergu, F. Liu и др. // Procedia computer science. – 2022. – Vol.199. – P.1066–1073.

78. A review of YOLO object detection algorithms based on deep learning [Текст] / X. Cong, S. Li, F. Chen и др. // Frontiers in Computing and Intelligent

Systems. – 2023. – Vol.4(2). – P.17–20.

79. Diwan, T. Object detection using YOLO: Challenges, architectural successors, datasets and applications [Текст] / T. Diwan, G. Anirudh, J.V. Tembhurne // Multimedia Tools and Applications. – 2023. – Vol.82(6). – P.9243–9275.

80. Terven, J. A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas [Текст] / J. Terven, D.M. Córdova-Esparza, J.A. Romero-González // Machine Learning and Knowledge Extraction. – 2023. – Vol.5(4). – P.1680–1716.

81. A comprehensive systematic review of yolo for medical object detection (2018 to 2023) [Текст] / R. Qureshi, M.G. RAGAB, S.J. ABDULKADER и др. // Authorea Preprints. – 2023.

82. DC-YOLOv8: small-size object detection algorithm based on camera sensor [Текст] / H. Lou, X. Duan, J. Guo и др. // Electronics. – 2023. – Vol.12(10). – P.2323.

83. Small object detection algorithm based on improved YOLOv8 for remote sensing [Текст] / H. Yi, B. Liu, B. Zhao, E. Liu // IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing. – 2023.

84. 基于深度学习的人体行为检测方法研究综述 [Текст] / 陆卫忠, 宋正伟, 吴宏杰 и др. // Computer Engineering & Science. – 2021. – Vol.43(12).

85. 朱红蕾. 人体行为识别数据集研究进展 [Текст] / 朱红蕾, 朱昶胜, 徐志刚 // 自动化学报. – 2018. – Vol.44(6). – P.978–1004.

86. Overview of human behavior detection methods based on deep learning

[Текст] / W.Z. LU, Z.W. SONG, H.J. WU и др. // Computer Engineering & Science. – 2021. – Vol.43(12). – P.2206.

87. Hochreiter, S. Long Short-Term Memory [Текст] / S. Hochreiter, J. Schmidhuber // Neural Computation. – 1997. – Vol.9(8). – P.1735–1780. – DOI: https://doi.org/10.1162/neco.1997.9.8.1735

88. A review of recurrent neural networks: LSTM cells and network architectures [Текст] / Y. Yu, X. Si, C. Hu, J. Zhang // Neural computation. – 2019. – Vol.31(7). – P.1235–1270.

89. Bi-LSTM network for multimodal continuous human activity recognition and fall detection [Текст] / H. Li, A. Shrestha, H. Heidari и др. // IEEE Sensors Journal. – 2019. – Vol.20(3). – P.1191–1201.

90. Fall detection for shipboard seafarers based on optimized BlazePose and LSTM [Текст] / W. Liu, X. Liu, Y. Hu и др. // Sensors. – 2022. – Vol.22(14). – P.5449.

91. Time series generative adversarial network for muscle force prognostication using statistical outlier detection [Текст] / H. Bansal, B. Chinagundi, P.S. Rana, N. Kumar // Expert Systems. – e13653.

92. Chang, C.W. A hybrid CNN and LSTM-based deep learning model for abnormal behavior detection [Текст] / C.W. Chang, C.Y. Chang, Y.Y. Lin // Multimedia Tools and Applications. – 2022. – Vol.81(9). – P.11825–11843.

93. Long-short-term-memory-based deep stacked sequence-to-sequence autoencoder for health prediction of industrial workers in closed environments based on wearable devices [Текст] / W. Xu, J. He, W. Li и др. // Sensors. – 2023. –

Vol.23(18). – P.7874.

94. Modeling long-and short-term temporal patterns with deep neural networks [Текст] / G. Lai, W.C. Chang, Y. Yang, H. Liu // The 41st international ACM SIGIR conference on research & development in information retrieval. – 2018. – P.95–104.

95. Skeleton-based fall detection with multiple inertial sensors using spatial-temporal graph convolutional networks [Текст] / J. Yan, X. Wang, J. Shi, S. Hu // Sensors. – 2023. – Vol.23(4). – P.2153.

96. A comparative study of time frequency representation techniques for freeze of gait detection and prediction [Текст] / T. Ashfaque Mostafa, S. Soltaninejad, T.L. McIsaac, I. Cheng // Sensors. – 2021. – Vol.21(19). – P.6446.

97. A hybrid spiking neurons embedded lstm network for multivariate time series learning under concept-drift environment [Текст] / W. Zheng, P. Zhao, G. Chen и др. // IEEE Transactions on Knowledge and Data Engineering. – 2022. – Vol.35(7). – P.6561–6574.

98. UAV-YOLOv8: A small-object-detection model based on improved YOLOv8 for UAV aerial photography scenarios [Текст] / G. Wang, Y. Chen, P. An и др. // Sensors. – 2023. – Vol.23(16). – P.7190.

99. A review on yolov8 and its advancements [Текст] / M. Sohan, T. Sai Ram, R. Reddy, C. Venkata // In: International Conference on Data Intelligence and Cognitive Informatics. – Springer, Singapore, 2024. – P.529–545.

100. BL-YOLOv8: An improved road defect detection model based on YOLOv8 [Текст] / X. Wang, H. Gao, Z. Jia, Z. Li // Sensors. – 2023. – Vol.23(20). – P.8361.

101. Talaat, F.M. An improved fire detection approach based on YOLO-v8 for smart cities [Текст] / F.M. Talaat, H. ZainEldin // Neural Computing and Applications. – 2023. – Vol.35(28). – P.20939–20954.

102. Development of an early detection and automatic targeting system for cotton weeds using an improved lightweight YOLOv8 architecture on an edge device [Текст] / M.J. Karim, M. Nahiduzzaman, M. Ahsan, J. Haider // Knowledge-Based Systems. – 2024. – Vol.300. – P.112204.

103. Wang Q, Wu B, Zhu P, et al. ECA-Net: Efficient channel attention for deep convolutional neural networks[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020: 11534-11542.

104. Li H, Li J, Wei H, et al. Slim-neck by GSConv: a lightweight-design for real-time detector architectures[J]. Journal of Real-Time Image Processing, 2024, 21(3): 62.

105. https://yolov8.com/ [Электронный ресурс]. – Режим доступа: https://yolov8.com/

106. https://github.com/ultralytics/ultralytics [Электронный ресурс]. – Режим доступа: https://github.com/ultralytics/ultralytics

107. https://docs.ultralytics.com/ [Электронный ресурс]. – Режим доступа: https://docs.ultralytics.com/

108. https://pytorch.org/ [Электронный ресурс]. – Режим доступа: https://pytorch.org/

109. https://www.tensorflow.org/ [Электронный ресурс]. – Режим доступа:

https://www.tensorflow.org/

110. http://colah.github.io/posts/2015-08-Understanding-LSTMs/ [Электронный ресурс]. – Режим доступа: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

111. https://docs.opencv.org/ [Электронный ресурс]. – Режим доступа: https://docs.opencv.org/

112. https://arxiv.org/ [Электронный ресурс]. – Режим доступа: https://arxiv.org/

113. http://fenix.ur.edu.pl/~mkepski/ds/uf.html[Электронный ресурс]. – Режим доступа: http://fenix.ur.edu.pl/~mkepski/ds/uf.html

114. https://www.iro.umontreal.ca/~labimage/Dataset/[Электронный ресурс]. – Режим доступа: https://www.iro.umontreal.ca/~labimage/Dataset/

115. https://www.kaggle.com/datasets/tuyenldvn/falldataset-imvia/[Электронный ресурс]. – Режим доступа: https://www.kaggle.com/datasets/tuyenldvn/falldataset-imvia/

116. https://paperswithcode.com/ [Электронный ресурс]. – Режим доступа: https://paperswithcode.com/

117. Liang, Z. Baseball Action Classification Based on OpenPose [Текст] / Z. Liang, B.J. Isakunovich // Academic Journal of Science and Technology. – 2023. – Т. 8, № 2. – С. 62–64.

118. Liang, Z. Volleyball Action Recognition Based on Skeleton Data [Текст] / Z. Liang, B.J. Isakunovich // Frontiers in Computing and Intelligent Systems. –

2023. – Т. 5, № 3. – С. 143–145.

119. Comparative study of deep learning models for action recognition based on skeleton data [Текст] : [доклад] / [Z. Liang, K.K. Kudayberdievna, B.J. Isakunovich и др.] // International Conference on Computer Graphics, Artificial Intelligence, and Data Processing (ICCAID 2023). – SPIE, 2024. – Vol. 13105. – P. 673–678.

120. Ancient Building Crack Detection Based on YOLOv8 Algorithm [Текст] / [W. Xiong, W. Meng, Z. Liang и др.] // Journal of Electrical Systems. – 2024. – Т. 20, № 10s. – С. 1238–1243.

121. Compliant control of an upper limb rehabilitation robot based on admittance control [Текст] / [I.V. Merkuryev, G Wu, T. B. Duishenaliev и др.] // Journal of Electrical Systems. – 2024. – Т. 20, № 3. – С. 4605–4612.

122. Graph spiking neural network for advanced urban flood risk assessment [Текст] / [Z. Liang, X. Fang, Z. Liang и др.] // iScience. – 2024. – Т. 27, № 11.

123. MSGAT: Multi-Stage Graph Attention Network For Human Motion Prediction [Текст] : [доклад] / [Z. Zheng, Z. Ren, Z. Liang и др.] // 2024 IEEE International Conference on Image Processing (ICIP). – IEEE, 2024. – P. 2306–2312.

ПРИЛОЖЕНИЕ

ПРИЛОЖЕНИЕ 2

Математические доказательства, формулы и вычисления

1. **Формулы для оценки точности детекции**

Использованная метрика IoU (Intersection over Union):

$$IoU = \frac{Area(B_{pred} \cap B_{true})}{Area(B_{pred} \cup B_{true})} \quad (П.1)$$

где B_{pred} — предсказанная рамка, B_{true} — истинная рамка.

Потеря по метрике IoU:

$$\text{IoU-Loss} = 1 - IoU \quad (П.2)$$

Расширенная метрика потерь с учётом центра и пропорций:

$$Loss = \alpha \cdot \text{IoU} + \beta \cdot \text{CenterLoss} + \gamma \cdot \text{AspectRatioLoss} \quad (П.3)$$

2. **Уравнение динамического anchor-free механизма в YOLOv8**

Для оценки центров объектов:

$$C_{pred} = \sigma(x_{pred}) + \text{GridOffset} \quad (П.4)$$

где σ — сигмоидная функция активации.

Для регрессии размеров:

$$W_{pred}, H_{pred} = e^{x_{scale}} \quad (П.5)$$

3. **Формула функции внимания ECA**

Сигнал активации внимания:

$$A_i = \text{sigmoid}\left(Conv1D\left(GAP(F)\right)\right) \quad \text{(П.6)}$$

где $GAP(F)$ — глобальный средний пуллинг по входному признаку F, $Conv1D$ — одномерная свёртка.

4. **Ошибка модели**

Среднеквадратичная ошибка (MSE):

$$MSE = \frac{1}{N}\sum_{i=1}^{N}\left(y_{true}^{(i)} - y_{pred}^{(i)}\right)^2 \quad \text{(П.7)}$$

- Относительная ошибка:

$$\text{Error}\% = \frac{|y_{true} - y_{pred}|}{y_{true}} \times 100\% \quad \text{(П.8)}$$

ПРИЛОЖЕНИЕ 3

Оценка погрешностей измерений

1. **Погрешность для детекции объектов (YOLOv8):**

Средняя точность (AP):

$$AP = \int_0^1 Precision(Recall)\ dRecall \quad (П.9)$$

Расчёт точности для категорий:

$$mAP = \frac{\sum_{i=1}^{k} AP_i}{k} \quad (П.10)$$

2. **Погрешность временных последовательностей (LSTM):**

Для временного шага t предсказания $\widehat{y_t}$:

$$E_t = y_t - \widehat{y_t} \quad (П.11)$$

Накопленная ошибка:

$$E_{cumulative} = \sum_{t=1}^{T} |E_t| \quad (П.12)$$

3. **Сравнение с эталоном:**

Коэффициент детекции падений (FDR):

$$FDR = \frac{TP}{TP+FN} \quad (П.13)$$

где TP — верно распознанные падения, FN — пропущенные падения.

Коэффициент ложных срабатываний (FPR):

$$FPR = \frac{FP}{FP+TN} \quad (П.14)$$

где FP — ложные срабатывания, TN — верно нераспознанные падения.

ПРИЛОЖЕНИЕ 3

Листинг программы системы управления

Ключевой код YOLOv8:

```
#Код основного модуля block.py.

# Ultralytics YOLOv8 , AGPL-3.0 license
"""Block modules."""

import torch
import torch.nn as nn
import torch.nn.functional as F

from .conv import Conv, DWConv, GhostConv, LightConv, RepConv
from .transformer import TransformerBlock

__all__ = (
    "DFL",
    "HGBlock",
    "HGStem",
    "SPP",
    "SPPF",
    "C1",
```

```
        "C2",
        "C3",
        "C2f",
        "C3x",
        "C3TR",
        "C3Ghost",
        "GhostBottleneck",
        "Bottleneck",
        "BottleneckCSP",
        "Proto",
        "RepC3",
        "ResNetLayer",
)

class DFL(nn.Module):
        """
        Integral module of Distribution Focal Loss (DFL).

        Proposed in Generalized Focal Loss
https://ieeexplore.ieee.org/document/9792391
        """
```

```
    def __init__(self, c1=16):
        """Initialize a convolutional layer with a given number of input
channels."""
        super().__init__()
        self.conv = nn.Conv2d(c1, 1, 1, bias=False).requires_grad_(False)
        x = torch.arange(c1, dtype=torch.float)
        self.conv.weight.data[:] = nn.Parameter(x.view(1, c1, 1, 1))
        self.c1 = c1

    def forward(self, x):
        """Applies a transformer layer on input tensor 'x' and returns a tensor."""
        b, c, a = x.shape    # batch, channels, anchors
        return self.conv(x.view(b, 4, self.c1, a).transpose(2, 1).softmax(1)).view(b,
4, a)
        # return self.conv(x.view(b, self.c1, 4, a).softmax(1)).view(b, 4, a)

class Proto(nn.Module):
    """YOLOv8 mask Proto module for segmentation models."""

    def __init__(self, c1, c_=256, c2=32):
        """
        Initializes the YOLOv8 mask Proto module with specified number of
```

```
protos and masks.

        Input arguments are ch_in, number of protos, number of masks.
        """
        super().__init__()
        self.cv1 = Conv(c1, c_, k=3)
        self.upsample = nn.ConvTranspose2d(c_, c_, 2, 2, 0, bias=True)  #
nn.Upsample(scale_factor=2, mode='nearest')
        self.cv2 = Conv(c_, c_, k=3)
        self.cv3 = Conv(c_, c2)

    def forward(self, x):
        """Performs a forward pass through layers using an upsampled input
image."""
        return self.cv3(self.cv2(self.upsample(self.cv1(x))))

class HGStem(nn.Module):

    def __init__(self, c1, cm, c2):
        """Initialize the SPP layer with input/output channels and specified kernel
sizes for max pooling."""
        super().__init__()
```

```
            self.stem1 = Conv(c1, cm, 3, 2, act=nn.ReLU())
            self.stem2a = Conv(cm, cm // 2, 2, 1, 0, act=nn.ReLU())
            self.stem2b = Conv(cm // 2, cm, 2, 1, 0, act=nn.ReLU())
            self.stem3 = Conv(cm * 2, cm, 3, 2, act=nn.ReLU())
            self.stem4 = Conv(cm, c2, 1, 1, act=nn.ReLU())
            self.pool = nn.MaxPool2d(kernel_size=2, stride=1, padding=0,
ceil_mode=True)

        def forward(self, x):
            """Forward pass of a PPHGNetV2 backbone layer."""
            x = self.stem1(x)
            x = F.pad(x, [0, 1, 0, 1])
            x2 = self.stem2a(x)
            x2 = F.pad(x2, [0, 1, 0, 1])
            x2 = self.stem2b(x2)
            x1 = self.pool(x)
            x = torch.cat([x1, x2], dim=1)
            x = self.stem3(x)
            x = self.stem4(x)
            return x
```

```
class HGBlock(nn.Module):


    def __init__(self, c1, cm, c2, k=3, n=6, lightconv=False, shortcut=False,
act=nn.ReLU()):
        """Initializes a CSP Bottleneck with 1 convolution using specified input
and output channels."""
        super().__init__()
        block = LightConv if lightconv else Conv
        self.m = nn.ModuleList(block(c1 if i == 0 else cm, cm, k=k, act=act) for i
in range(n))
        self.sc = Conv(c1 + n * cm, c2 // 2, 1, 1, act=act)  # squeeze conv
        self.ec = Conv(c2 // 2, c2, 1, 1, act=act)  # excitation conv
        self.add = shortcut and c1 == c2


    def forward(self, x):
        """Forward pass of a PPHGNetV2 backbone layer."""
        y = [x]
        y.extend(m(y[-1]) for m in self.m)
        y = self.ec(self.sc(torch.cat(y, 1)))
        return y + x if self.add else y
```

```
class SPP(nn.Module):
    """Spatial Pyramid Pooling (SPP) layer https://arxiv.org/abs/1406.4729."""

    def __init__(self, c1, c2, k=(5, 9, 13)):
        """Initialize the SPP layer with input/output channels and pooling kernel sizes."""
        super().__init__()
        c_ = c1 // 2  # hidden channels
        self.cv1 = Conv(c1, c_, 1, 1)
        self.cv2 = Conv(c_ * (len(k) + 1), c2, 1, 1)
        self.m = nn.ModuleList([nn.MaxPool2d(kernel_size=x, stride=1, padding=x // 2) for x in k])

    def forward(self, x):
        """Forward pass of the SPP layer, performing spatial pyramid pooling."""
        x = self.cv1(x)
        return self.cv2(torch.cat([x] + [m(x) for m in self.m], 1))


class SPPF(nn.Module):
    """Spatial Pyramid Pooling - Fast (SPPF) layer for YOLOv5 by Glenn
```

```
Jocher."""

    def __init__(self, c1, c2, k=5):
        """
        Initializes the SPPF layer with given input/output channels and kernel size.

        This module is equivalent to SPP(k=(5, 9, 13)).
        """
        super().__init__()
        c_ = c1 // 2  # hidden channels
        self.cv1 = Conv(c1, c_, 1, 1)
        self.cv2 = Conv(c_ * 4, c2, 1, 1)
        self.m = nn.MaxPool2d(kernel_size=k, stride=1, padding=k // 2)

    def forward(self, x):
        """Forward pass through Ghost Convolution block."""
        x = self.cv1(x)
        y1 = self.m(x)
        y2 = self.m(y1)
        return self.cv2(torch.cat((x, y1, y2, self.m(y2)), 1))
```

```
class C1(nn.Module):
    """CSP Bottleneck with 1 convolution."""

    def __init__(self, c1, c2, n=1):
        """Initializes the CSP Bottleneck with configurations for 1 convolution with arguments ch_in, ch_out, number."""
        super().__init__()
        self.cv1 = Conv(c1, c2, 1, 1)
        self.m = nn.Sequential(*(Conv(c2, c2, 3) for _ in range(n)))

    def forward(self, x):
        """Applies cross-convolutions to input in the C3 module."""
        y = self.cv1(x)
        return self.m(y) + y


class C2(nn.Module):
    """CSP Bottleneck with 2 convolutions."""

    def __init__(self, c1, c2, n=1, shortcut=True, g=1, e=0.5):
        """Initializes the CSP Bottleneck with 2 convolutions module with arguments ch_in, ch_out, number, shortcut,
```

```
            groups, expansion.
            """
            super().__init__()
            self.c = int(c2 * e)  # hidden channels
            self.cv1 = Conv(c1, 2 * self.c, 1, 1)
            self.cv2 = Conv(2 * self.c, c2, 1)  # optional act=FReLU(c2)
            # self.attention = ChannelAttention(2 * self.c)  # or SpatialAttention()
            self.m = nn.Sequential(*(Bottleneck(self.c, self.c, shortcut, g, k=((3, 3), (3, 3)), e=1.0) for _ in range(n)))

        def forward(self, x):
            """Forward pass through the CSP bottleneck with 2 convolutions."""
            a, b = self.cv1(x).chunk(2, 1)
            return self.cv2(torch.cat((self.m(a), b), 1))


class C2f(nn.Module):
    """Faster Implementation of CSP Bottleneck with 2 convolutions."""

    def __init__(self, c1, c2, n=1, shortcut=False, g=1, e=0.5):
        """Initialize CSP bottleneck layer with two convolutions with arguments ch_in, ch_out, number, shortcut, groups,
```

```
        expansion.
        """
        super().__init__()
        self.c = int(c2 * e)  # hidden channels
        self.cv1 = Conv(c1, 2 * self.c, 1, 1)
        self.cv2 = Conv((2 + n) * self.c, c2, 1)  # optional act=FReLU(c2)
        self.m = nn.ModuleList(Bottleneck(self.c, self.c, shortcut, g, k=((3, 3), (3,
3)), e=1.0) for _ in range(n))

    def forward(self, x):
        """Forward pass through C2f layer."""
        y = list(self.cv1(x).chunk(2, 1))
        y.extend(m(y[-1]) for m in self.m)
        return self.cv2(torch.cat(y, 1))

    def forward_split(self, x):
        """Forward pass using split() instead of chunk()."""
        y = list(self.cv1(x).split((self.c, self.c), 1))
        y.extend(m(y[-1]) for m in self.m)
        return self.cv2(torch.cat(y, 1))
```

```
class C3(nn.Module):
    """CSP Bottleneck with 3 convolutions."""

    def __init__(self, c1, c2, n=1, shortcut=True, g=1, e=0.5):
        """Initialize the CSP Bottleneck with given channels, number, shortcut, groups, and expansion values."""
        super().__init__()
        c_ = int(c2 * e)  # hidden channels
        self.cv1 = Conv(c1, c_, 1, 1)
        self.cv2 = Conv(c1, c_, 1, 1)
        self.cv3 = Conv(2 * c_, c2, 1)  # optional act=FReLU(c2)
        self.m = nn.Sequential(*(Bottleneck(c_, c_, shortcut, g, k=((1, 1), (3, 3)), e=1.0) for _ in range(n)))

    def forward(self, x):
        """Forward pass through the CSP bottleneck with 2 convolutions."""
        return self.cv3(torch.cat((self.m(self.cv1(x)), self.cv2(x)), 1))


class C3x(C3):
    """C3 module with cross-convolutions."""
```

```
    def __init__(self, c1, c2, n=1, shortcut=True, g=1, e=0.5):
        """Initialize C3TR instance and set default parameters."""
        super().__init__(c1, c2, n, shortcut, g, e)
        self.c_ = int(c2 * e)
        self.m = nn.Sequential(*(Bottleneck(self.c_, self.c_, shortcut, g, k=((1, 3), (3, 1)), e=1) for _ in range(n)))


class RepC3(nn.Module):
    """Rep C3."""

    def __init__(self, c1, c2, n=3, e=1.0):
        """Initialize CSP Bottleneck with a single convolution using input channels, output channels, and number."""
        super().__init__()
        c_ = int(c2 * e)  # hidden channels
        self.cv1 = Conv(c1, c2, 1, 1)
        self.cv2 = Conv(c1, c2, 1, 1)
        self.m = nn.Sequential(*[RepConv(c_, c_) for _ in range(n)])
        self.cv3 = Conv(c_, c2, 1, 1) if c_ != c2 else nn.Identity()

    def forward(self, x):
```

```
        """Forward pass of RT-DETR neck layer."""
        return self.cv3(self.m(self.cv1(x)) + self.cv2(x))


class C3TR(C3):
    """C3 module with TransformerBlock()."""

    def __init__(self, c1, c2, n=1, shortcut=True, g=1, e=0.5):
        """Initialize C3Ghost module with GhostBottleneck()."""
        super().__init__(c1, c2, n, shortcut, g, e)
        c_ = int(c2 * e)
        self.m = TransformerBlock(c_, c_, 4, n)


class C3Ghost(C3):
    """C3 module with GhostBottleneck()."""

    def __init__(self, c1, c2, n=1, shortcut=True, g=1, e=0.5):
        """Initialize 'SPP' module with various pooling sizes for spatial pyramid pooling."""
        super().__init__(c1, c2, n, shortcut, g, e)
        c_ = int(c2 * e)  # hidden channels
```

```
        self.m = nn.Sequential(*(GhostBottleneck(c_, c_) for _ in range(n)))


class GhostBottleneck(nn.Module):


    def __init__(self, c1, c2, k=3, s=1):
        """Initializes GhostBottleneck module with arguments ch_in, ch_out,
kernel, stride."""
        super().__init__()
        c_ = c2 // 2
        self.conv = nn.Sequential(
            GhostConv(c1, c_, 1, 1),    # pw
            DWConv(c_, c_, k, s, act=False) if s == 2 else nn.Identity(),    # dw
            GhostConv(c_, c2, 1, 1, act=False),    # pw-linear
        )
        self.shortcut = (
            nn.Sequential(DWConv(c1, c1, k, s, act=False), Conv(c1, c2, 1, 1,
act=False)) if s == 2 else nn.Identity()
        )


    def forward(self, x):
        """Applies skip connection and concatenation to input tensor."""
```

```
        return self.conv(x) + self.shortcut(x)


class Bottleneck(nn.Module):
    """Standard bottleneck."""

    def __init__(self, c1, c2, shortcut=True, g=1, k=(3, 3), e=0.5):
        """Initializes a bottleneck module with given input/output channels,
shortcut option, group, kernels, and
        expansion.
        """
        super().__init__()
        c_ = int(c2 * e)  # hidden channels
        self.cv1 = Conv(c1, c_, k[0], 1)
        self.cv2 = Conv(c_, c2, k[1], 1, g=g)
        self.add = shortcut and c1 == c2

    def forward(self, x):
        """forward()' applies the YOLO FPN to input data."""
        return x + self.cv2(self.cv1(x)) if self.add else self.cv2(self.cv1(x))
```

```
class BottleneckCSP(nn.Module):


    def __init__(self, c1, c2, n=1, shortcut=True, g=1, e=0.5):
        """Initializes the CSP Bottleneck given arguments for ch_in, ch_out,
number, shortcut, groups, expansion."""
        super().__init__()
        c_ = int(c2 * e)  # hidden channels
        self.cv1 = Conv(c1, c_, 1, 1)
        self.cv2 = nn.Conv2d(c1, c_, 1, 1, bias=False)
        self.cv3 = nn.Conv2d(c_, c_, 1, 1, bias=False)
        self.cv4 = Conv(2 * c_, c2, 1, 1)
        self.bn = nn.BatchNorm2d(2 * c_)  # applied to cat(cv2, cv3)
        self.act = nn.SiLU()
        self.m = nn.Sequential(*(Bottleneck(c_, c_, shortcut, g, e=1.0) for _ in
range(n)))


    def forward(self, x):
        """Applies a CSP bottleneck with 3 convolutions."""
        y1 = self.cv3(self.m(self.cv1(x)))
        y2 = self.cv2(x)
        return self.cv4(self.act(self.bn(torch.cat((y1, y2), 1))))
```

```
class ResNetBlock(nn.Module):
    """ResNet block with standard convolution layers."""

    def __init__(self, c1, c2, s=1, e=4):
        """Initialize convolution with given parameters."""
        super().__init__()
        c3 = e * c2
        self.cv1 = Conv(c1, c2, k=1, s=1, act=True)
        self.cv2 = Conv(c2, c2, k=3, s=s, p=1, act=True)
        self.cv3 = Conv(c2, c3, k=1, act=False)
        self.shortcut = nn.Sequential(Conv(c1, c3, k=1, s=s, act=False)) if s != 1 or
c1 != c3 else nn.Identity()

    def forward(self, x):
        """Forward pass through the ResNet block."""
        return F.relu(self.cv3(self.cv2(self.cv1(x))) + self.shortcut(x))

class ResNetLayer(nn.Module):
    """ResNet layer with multiple ResNet blocks."""

    def __init__(self, c1, c2, s=1, is_first=False, n=1, e=4):
```

```
        """Initializes the ResNetLayer given arguments."""
        super().__init__()
        self.is_first = is_first

        if self.is_first:
            self.layer = nn.Sequential(
                Conv(c1, c2, k=7, s=2, p=3, act=True),
nn.MaxPool2d(kernel_size=3, stride=2, padding=1)
            )
        else:
            blocks = [ResNetBlock(c1, c2, s, e=e)]
            blocks.extend([ResNetBlock(e * c2, c2, 1, e=e) for _ in range(n - 1)])
            self.layer = nn.Sequential(*blocks)

    def forward(self, x):
        """Forward pass through the ResNet layer."""
        return self.layer(x)
```

Улучшенный код:

```
import torch
import torch.nn as nn
```

```
import torch.nn.functional as F


# ECA Attention Mechanism
class ECAAttention(nn.Module):
    def __init__(self, channel, b=1):
        super(ECAAttention, self).__init__()
        self.channel = channel
        self.b = b
        # Calculate the size of the convolution kernel
        kernel_size = self._calculate_kernel_size(channel)
        self.conv1d = nn.Conv1d(in_channels=channel, out_channels=channel,
kernel_size=kernel_size, padding=(kernel_size - 1) // 2, groups=channel, bias=False)


    def _calculate_kernel_size(self, channel):
        t = int(abs((math.log(channel) / math.log(2)) + self.b) / 2) * 2 + 1
        return t


    def forward(self, x):
        # Global average pooling
        avg_pool = F.adaptive_avg_pool2d(x, (1, 1))
        avg_pool = avg_pool.view(avg_pool.size(0), avg_pool.size(1), -1)
        avg_pool = avg_pool.squeeze(-1)
```

```
            # ECA channel attention mechanism
            attention = self.conv1d(avg_pool)
            attention = attention.unsqueeze(-1).unsqueeze(-1)
            return x * torch.sigmoid(attention)

# GSConv (Group-wise Separable Convolution) module
class GSConv(nn.Module):
    def __init__(self, in_channels, out_channels, kernel_size=3, stride=1,
padding=1):
            super(GSConv, self).__init__()
            self.depthwise = nn.Conv2d(in_channels, in_channels,
kernel_size=kernel_size, stride=stride, padding=padding, groups=in_channels,
bias=False)
            self.pointwise = nn.Conv2d(in_channels, out_channels, kernel_size=1,
stride=1, padding=0, bias=False)

    def forward(self, x):
            x = self.depthwise(x)
            x = self.pointwise(x)
            return x

# Modifications to YOLOv8
```

```
class YOLOv8Modified(nn.Module):
    def __init__(self, original_yolov8_model):
        super(YOLOv8Modified, self).__init__()
        self.yolov8_model = original_yolov8_model

        # Add ECA modules after the backbone
        self.eca1 = ECAAttention(256)
        self.eca2 = ECAAttention(512)
        self.eca3 = ECAAttention(1024)

        # Use GSConv for feature fusion
        self.gsconv = GSConv(1024, 512)

    def forward(self, x):
        # Use the original YOLOv8 model for feature extraction
        x = self.yolov8_model.backbone(x)

        # Add ECA after the C2f module
        x = self.eca1(x)
        x = self.eca2(x)
        x = self.eca3(x)
```

```
            # Perform feature fusion using GSConv
            x = self.gsconv(x)

            # Return the final output
            return self.yolov8_model.head(x)

# Integrate with LSTM
import torch
import torch.nn as nn
import torch.nn.functional as F
from models.yolov8 import YOLOv8    # Assuming YOLOv8 model is defined
from torch.utils.data import DataLoader
from datasets import VideoDataset    # Custom video dataset class

# LSTM module for processing temporal information
class LSTMForFallDetection(nn.Module):
    def __init__(self, input_size, hidden_size, num_layers, num_classes):
        super(LSTMForFallDetection, self).__init__()

        # Multi-layer LSTM
        self.lstm = nn.LSTM(input_size, hidden_size, num_layers,
batch_first=True, dropout=0.5)    # Added dropout to avoid overfitting
```

```
        # Classification layer
        self.fc = nn.Linear(hidden_size, num_classes)

    def forward(self, x):
        # Process temporal data with LSTM
        lstm_out, (hn, cn) = self.lstm(x)

        # Take the output from the last time step
        output = self.fc(hn[-1])

        return output

# Combined YOLOv8 + LSTM model
class YOLOv8LSTMModel(nn.Module):
    def __init__(self, num_classes=80, lstm_hidden_size=256, lstm_num_layers=4,
fall_classes=2):
        super(YOLOv8LSTMModel, self).__init__()

        # Use YOLOv8 for feature extraction
        self.yolov8 = YOLOv8(num_classes=num_classes)
```

```
        # LSTM module
        self.lstm_model = LSTMForFallDetection(input_size=4,    # Assume
YOLOv8 outputs 4 features (e.g., [class, x, y, confidence])

hidden_size=lstm_hidden_size,

num_layers=lstm_num_layers,    # Increase the number of LSTM layers
                                                    num_classes=fall_classes)
# 2 classes: fall/no fall

    def forward(self, x):
        # Input shape: (batch_size, seq_len, 3, H, W), where seq_len represents the
number of video frames

        # Extract YOLOv8 detection results for all frames
        batch_size, seq_len, C, H, W = x.size()
        lstm_input = []

        for i in range(seq_len):
            frame = x[:, i, :, :, :]    # Get the i-th frame

            # Extract features from each frame using YOLOv8 (e.g., bounding box
```

```
coordinates)
                yolo_output = self.yolov8(frame)    # Shape: (batch_size, num_boxes,
4+num_classes)

                # Select a subset of YOLO output features, such as bounding box
coordinates and confidence
                # Assume we use the first detected box per frame
                frame_features = yolo_output[:, 0, :4]    # Take x, y, w, h of the first
detected box
                lstm_input.append(frame_features)

        # Combine features of all frames into a sequence for LSTM processing
        lstm_input = torch.stack(lstm_input, dim=1)    # Shape: (batch_size,
seq_len, 4)

        # Process temporal information with LSTM
        output = self.lstm_model(lstm_input)

        return output

# Assuming a predefined video dataset
video_dataset = VideoDataset('/path/to/video/dataset')
```

```
data_loader = DataLoader(video_dataset, batch_size=2, shuffle=True)

# Define the model
model = YOLOv8LSTMModel(num_classes=80, lstm_hidden_size=256,
lstm_num_layers=4, fall_classes=2)

# Assume a training process
criterion = torch.nn.CrossEntropyLoss()
optimizer = torch.optim.Adam(model.parameters(), lr=1e-4)

# Training loop
for epoch in range(10):
    model.train()
    for video_frames, labels in data_loader:
        optimizer.zero_grad()

        # Pass video frames through the model
        outputs = model(video_frames)

        # Compute loss
        loss = criterion(outputs, labels)
        loss.backward()
```

```
        optimizer.step()

print(f'Epoch {epoch}, Loss: {loss.item()}')
```

Код для развертывания:

```
import random
import tempfile
import time

import cv2
import numpy as np
import streamlit as st
from QtFusion.path import abs_path
from QtFusion.utils import drawRectBox

from LoggerRes import ResultLogger, LogTable
from YOLOv8v5Model import YOLOv8v5Detector
from datasets.PedFall.label_name import Label_list
from style_css import def_css_hitml
from utils_web import save_uploaded_file, concat_results, load_default_image,
```

```
get_camera_names
    class Detection_UI:
        """
        Detection system class.

        Attributes:
            model_type (str): Model type.
            conf_threshold (float): Confidence threshold.
            iou_threshold (float): IOU threshold.
            selected_camera (str): Selected camera.
            file_type (str): File type.
            uploaded_file (FileUploader): Uploaded file.
            detection_result (str): Detection result.
            detection_location (str): Detection location.
            detection_confidence (str): Detection confidence.
            detection_time (str): Detection time.
        """

        def __init__(self):
            """
            Initialize parameters for the pedestrian fall detection system.
            """
```

```
# Initialize class label list and assign random colors to each class
self.cls_name = Label_list
self.colors = [[random.randint(0, 255) for _ in range(3)] for _ in
                    range(len(self.cls_name))]

# Set the page title
self.title = "YOLOv8/v5-based Pedestrian Fall Detection System"
self.setup_page()    # Initialize page layout
def_css_hitml()    # Apply CSS styles

# Initialize configuration parameters related to detection
self.model_type = None
self.conf_threshold = 0.25    # Default confidence threshold
self.iou_threshold = 0.5    # Default IOU threshold

# Initialize variables related to cameras and files
self.selected_camera = None
self.file_type = None
self.uploaded_file = None
self.uploaded_video = None
self.custom_model_file = None    # Custom model file
```

```
        # Initialize variables related to detection results
        self.detection_result = None
        self.detection_location = None
        self.detection_confidence = None
        self.detection_time = None

        # Initialize variables related to UI display
        self.display_mode = None  # Set display mode
        self.close_flag = None  # Flag to control image display termination
        self.close_placeholder = None  # Placeholder for the close button
        self.image_placeholder = None  # Placeholder for displaying images
        self.image_placeholder_res = None  # Placeholder for displaying
images
        self.table_placeholder = None  # Placeholder for displaying tables
        self.log_table_placeholder = None  # Placeholder for full results
table
        self.selectbox_placeholder = None  # Placeholder for the dropdown
menu
        self.selectbox_target = None  # Dropdown menu selection
        self.progress_bar = None  # Progress bar for display

        # Initialize the log data save path
```

```
        self.saved_log_data = abs_path("tempDir/log_table_data.csv",
path_type="current")

        # If 'logTable' is not in session state, create a new LogTable instance
        if 'logTable' not in st.session_state:
            st.session_state['logTable'] = LogTable(self.saved_log_data)

        # Get or update the list of available cameras
        if 'available_cameras' not in st.session_state:
            st.session_state['available_cameras'] = get_camera_names()
        self.available_cameras = st.session_state['available_cameras']

        # Initialize or get the table for recognition results
        self.logTable = st.session_state['logTable']

        # Load or create a model instance
        if 'model' not in st.session_state:
            st.session_state['model'] = YOLOv8v5Detector()    # Create
YOLOv8/v5Detector model instance

        self.model = st.session_state['model']
        # Load trained model weights
```

```
            self.model.load_model(model_path=abs_path("weights/best-
yolov8n.pt", path_type="current"))
            # Reassign colors for categories in the model
            self.colors = [[random.randint(0, 255) for _ in range(3)] for _ in
                                range(len(self.model.names))]
            self.setup_sidebar()    # Initialize sidebar layout

        def setup_page(self):
            # Set page layout
            st.set_page_config(
                page_title=self.title,
                page_icon=" ",
                initial_sidebar_state="expanded"
            )

        def setup_sidebar(self):
            """
            Set up the Streamlit sidebar.

            Configure model settings, camera selection, and recognition project
settings in the sidebar.
            """
```

```
                # Configure the model settings section in the sidebar
                st.sidebar.header("Model Settings")
                # Dropdown menu to select model type
                self.model_type = st.sidebar.selectbox("Select Model Type",
["YOLOv8/v5", "Other Models"])

                # Radio buttons to select model file type, either default or custom
                model_file_option = st.sidebar.radio("Model File", ["Default",
"Custom"])
                if model_file_option == "Custom":
                    # If custom model file is selected, provide a file uploader
                    model_file = st.sidebar.file_uploader("Select .pt File", type="pt")

                    # If a model file is uploaded, save and load it
                    if model_file is not None:
                        self.custom_model_file = save_uploaded_file(model_file)

self.model.load_model(model_path=self.custom_model_file)
                        self.colors = [[random.randint(0, 255) for _ in range(3)] for
_ in
                                       range(len(self.model.names))]
                elif model_file_option == "Default":
```

```
                    self.model.load_model(model_path=abs_path("weights/best-
yolov8n.pt", path_type="current"))
                    # Reassign colors for categories in the model
                    self.colors = [[random.randint(0, 255) for _ in range(3)] for _ in
                                   range(len(self.model.names))]

            # Slider for confidence threshold
            self.conf_threshold = float(st.sidebar.slider("Confidence Threshold",
min_value=0.0, max_value=1.0, value=0.25))
            # Slider for IOU threshold
            self.iou_threshold = float(st.sidebar.slider("IOU Threshold",
min_value=0.0, max_value=1.0, value=0.5))

            # Configure the camera settings section in the sidebar
            st.sidebar.header("Camera Configuration")
            # Dropdown menu to select camera
            self.selected_camera = st.sidebar.selectbox("Select Camera",
self.available_cameras)

            # Configure the recognition project settings section in the sidebar
            st.sidebar.header("Recognition Project Settings")
            # Dropdown menu to select file type
```

```
            self.file_type = st.sidebar.selectbox("Select File Type", ["Image File",
"Video File"])
            # Provide corresponding file uploader based on selected file type
            if self.file_type == "Image File":
                self.uploaded_file = st.sidebar.file_uploader("Upload Image",
type=["jpg", "png", "jpeg"])
            elif self.file_type == "Video File":
                self.uploaded_video = st.sidebar.file_uploader("Upload Video
File", type=["mp4"])

            # Provide relevant instructions based on the selected camera and file
type
            if self.selected_camera == "Camera Not Enabled":
                if self.file_type == "Image File":
                    st.sidebar.write("Select an image and click the 'Run' button
to start image detection!")
                if self.file_type == "Video File":
                    st.sidebar.write("Select a video and click the 'Run' button to
start video detection!")
            else:
                st.sidebar.write("Click the 'Run' button to start camera
detection!")
```

ПРИЛОЖЕНИЕ 4

1. Пат. ZL202310335615.9 Китай. Способ распознавания последовательности действий и определения намерений, устройство, оборудование и носитель для хранения данных [Текст] / [Ren Ziliang, Luo Li, Liang Zhanhao и др.]; правообладатели: Дунгуанский технологический университет, Гуандунская компания «Руиэнь Технолоджи» (ООО).– Заявл. 30.03.2023; опубл. 30.04.2024, CN 116434335 B.

证书号第6951406号

发明专利证书

发 明 名 称：动作序列识别和意图推断方法、装置、设备及存储介质

发 明 人：任子良;罗里;梁展豪;秦勇;吴成悦;陈嘉振;李银凤

专 利 号：ZL 2023 1 0335615.9

专利申请日：2023年03月30日

专 利 权 人：东莞理工学院;广东瑞恩科技有限公司

地 址：523000 广东省东莞市松山湖科技产业园大学路一号

授权公告日：2024年04月30日　　授权公告号：CN 116434335 B

国家知识产权局依照中华人民共和国专利法进行审查，决定授予专利权，颁发发明专利证书并在专利登记簿上予以登记。专利权自授权公告之日起生效。专利权期限为二十年，自申请日起算。

专利证书记载专利权登记时的法律状况。专利权的转移、质押、无效、终止、恢复和专利权人的姓名或名称、国籍、地址变更等事项记载在专利登记簿上。

局长
申长雨

国家知识产权局

2024年04月30日

第1页(共2页)

其他事项参见续页

2. Пат. ZL202210675830.9 Китай. Способ распознавания движений человека и интерпретации намерений, конечное устройство и носитель для хранения данных [Текст] / [Ren Ziliang, Wei Wenhong, Liang Zhanhao и др.]; правообладатели: Дунгуаньский технологический университет, Дунгуаньский городской институт. – Заявл. 15.06.2022; опубл. 05.04.2024, CN 115100740 B.

证书号第6868632号

发明专利证书

发 明 名 称：一种人体动作识别和意图理解方法、终端设备及存储介质

发 明 人：任子良;魏文红;梁展豪;李清霞;舒希南

专 利 号：ZL 2022 1 0675830.9

专利申请日：2022年06月15日

专 利 权 人：东莞理工学院;东莞城市学院

地 址：523808 广东省东莞市松山湖科技产业园区大学路1号

授权公告日：2024年04月05日　　授权公告号：CN 115100740 B

国家知识产权局依照中华人民共和国专利法进行审查，决定授予专利权。颁发发明专利证书并在专利登记簿上予以登记。专利权自授权公告之日起生效。专利权期限为二十年，自申请日起算。

专利证书记载专利权登记时的法律状况。专利权的转移、质押、无效、终止、恢复和专利权人的姓名或名称、国籍、地址变更等事项记载在专利登记簿上。

局长
申长雨

国家知识产权局
2024年04月05日

第1页(共2页)

其他事项参见续页

ПРИЛОЖЕНИЕ 5

Свидетельство о награде

Награда:
Проект награды: Новый метод обнаружения формальдегида (метаналя) и устройство для калибровки порошков.

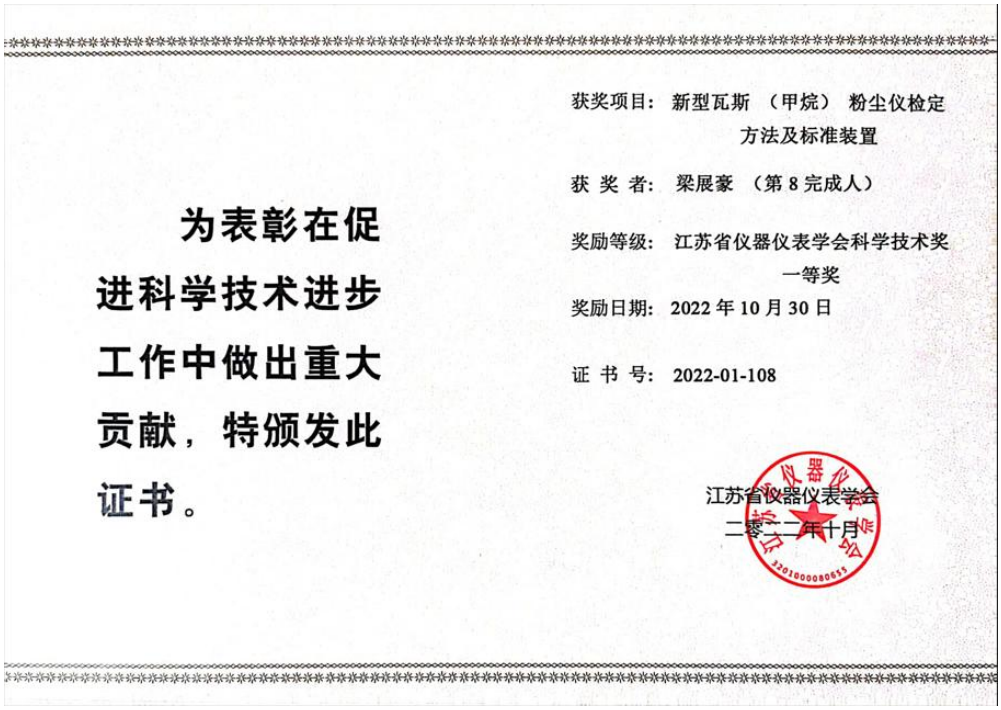
Лауреат:
Лян Чжаньхао (восьмой участник завершения).

Категория награды:
Премия первого уровня в области науки и технологий Ассоциации приборов и измерений провинции Цзянсу, Китай.

Дата награждения:
30 октября 2022 года.

Номер свидетельства:
2022-01-108.

Текст:
В знак признания значительного вклада в продвижение научно-технического прогресса, настоящим выдается данное свидетельство.


为表彰在促进科学技术进步工作中做出重大贡献，特颁发此证书。

获奖项目：新型瓦斯（甲烷）粉尘仪检定方法及标准装置

获 奖 者：梁展豪（第 8 完成人）

奖励等级：江苏省仪器仪表学会科学技术奖一等奖

奖励日期：2022 年 10 月 30 日

证 书 号：2022-01-108

江苏省仪器仪表学会

二零二二年十月

Премия за научно-технический прогресс города Тяньцзинь, Китай.

Свидетельство

В знак признания достижения лауреатов Премии за научно-технический прогресс города Тяньцзинь, настоящим выдается данное свидетельство.

Проект: «Робот-регенератор, основанный на пластичности мозга с применением глубокого усиленного обучения»

Уровень награды: Второй уровень

Лауреат: Лян Чжаньхао

Дата награждения: 10 апреля 2024 года

Номер награды: 2023JB-2-076-R8

Печать: Народное правительство города Тяньцзинь