

**Проект
« RS-VisionAI »
R-Style Softlab**

Руководство по эксплуатации

Оглавление

1 ВВЕДЕНИЕ	4
1.1 ТЕРМИНЫ И СОКРАЩЕНИЯ	4
2 РАБОТА С СИСТЕМОЙ ДИАГНОЗ.....	5
2.1 НАЧАЛО РАБОТЫ С СИСТЕМОЙ.....	5
2.2 ОПИСАНИЕ ПУНКТОВ МЕНЮ	5
2.3 АНАЛИЗ.....	5
2.3.1 Выполнение диагностики	6
2.3.2 Просмотр и интерпретация результатов.....	7
2.4 ИСТОРИЯ.....	7
2.4.1 Просмотр истории	7
2.4.2 Экспорт результатов	8
2.5 АНАЛИТИКА	8
2.6 ТИПИЧНЫЕ ОШИБКИ И РЕКОМЕНДАЦИИ	9
2.6.1 Ошибки пользователя	9
2.6.2 Рекомендации по качеству изображений.....	9
2.7 ЗАВЕРШЕНИЕ РАБОТЫ.....	9
3 РАБОТА С СИСТЕМОЙ RS-LABELMARK	10
3.1 БЫСТРЫЙ СТАРТ	10
3.2 ВХОД И РЕГИСТРАЦИЯ В СИСТЕМЕ.....	10
3.3 ДОБАВЛЕНИЕ ПОЛЬЗОВАТЕЛЕЙ	10
3.4 ЛИЧНЫЙ КАБИНЕТ	11
3.4.1 Личная информация	12
3.4.2 Горячие клавиши	12
3.5 СОЗДАНИЕ И НАСТРОЙКА ПРОЕКТОВ	13
3.5.1 Процесс настройки проекта.....	14
3.5.2 Создание проекта.....	14
3.5.3 Настройка параметров аннотирования	16
3.6 НАСТРОЙКА ИНТЕРФЕЙСА РАЗМЕТКИ	17
3.6.1 Настройка интерфейса разметки для проекта.....	17
3.6.2 Настройка шаблона для разметки.....	18
3.7 УПРАВЛЕНИЕ ПРОЕКТАМИ	19
3.7.1 Настройки проекта	19
3.7.2 Менеджер данных	25
3.8 ИМПОРТ ДАННЫХ	28
3.8.1 Общие рекомендации по импорту данных.....	28
3.8.2 Типы данных, которые можно импортировать	28
3.8.3 Как получить данные	30
3.8.4 Как отформатировать данные для импорта	32
3.8.5 Импорт данных из локального каталога.....	35
3.8.6 Импортируйте данные из пользовательского интерфейса.....	36
3.8.7 Импорт предварительно аннотированных данных.....	37
3.9 РАЗМЕТКА ДАННЫХ.....	51
3.9.1 Начало разметки.....	51
3.9.2 Маркировка с участием соавторов	54
3.9.3 Автоматическая разметка.....	54
3.9.4 Настройте интерфейс для маркировки	55
3.9.5 Расширенная разметка изображений	55
3.9.6 Аннотация к многостраничному документу.....	56
3.9.7 Выделение.....	59
3.9.8 Сохранение результатов в аннотациях	61

3.10 Экспорт аннотаций и данных	61
3.10.1 Сохранение результатов в аннотациях	61
3.10.2 Форматы экспорта.....	62
3.10.3 Формат JSON аннотированных задач	65
3.11 ЗАВЕРШЕНИЕ РАБОТЫ.....	69
4 РАБОТА С СИСТЕМОЙ RS-STREAMML	70
4.1 НАЧАЛО РАБОТЫ С СИСТЕМОЙ.....	70
4.2 ОТСЛЕЖИВАНИЕ ЭКСПЕРИМЕНТОВ	70
4.2.1 Отслеживание запусков	71
4.2.2 Отслеживание моделей	71
4.2.3 Отслеживание датасетов	72
4.3 РАБОТА С СЕРВЕРОМ ОТСЛЕЖИВАНИЯ	73
4.3.1 Запуск <i>Tracking Server</i>	73
4.3.2 Логирование в <i>Tracking Server</i>	73
4.3.3 Настройка сервера.....	74
4.3.4 <i>Tracking API</i>	76
4.4 МОДЕЛИ	79
4.4.1 Реестр моделей	79
4.4.2 Работа с зарегистрированными моделями.....	81
4.4.3 Использование реестра моделей	84
4.5 ПРОМПТЫ	85
4.5.1 Начало работы	85
4.5.2 Объект <i>Prompt</i>	86
4.5.3 Конфигурация модели	88
4.5.4 Кэширование промптов	89
4.6 НАСТРОЙКИ	90
4.6.1 Раздел «Общие».....	91
4.6.2 Раздел «Подключения LLM»	91
4.6.3 Раздел «Вебхуки»	92
4.7 ЗАВЕРШЕНИЕ РАБОТЫ.....	93

1 Введение

Платформа RS-VisionAI обеспечивает комплексную обработку разнородных источников данных (включая документы, видео, аудио, потоковые данные, изображения и пр.) для автоматизированного выявления проблем и задач, анализа их стадий, формирования рекомендаций, а также полного или частичного автоматизированного разрешения выявленных проблем.

Данный документ описывает работу с функциональными подсистемами ДИАГНОЗ, RS-LabelMark и RS-StreamML платформы RS-VisionAI.

Документ разделен на три части:

- ◆ В первой части рассматриваются возможности работы с подсистемой Диагноз (веб-интерфейсом).
- ◆ Во второй части приведено описание подсистемы RS-LabelMark и разметки данных.
- ◆ Третья часть посвящена описанию работы с подсистемой RS-StreamML.

1.1 Термины и сокращения

Диагностика – процесс анализа изображения с целью выявления проблем.

ИИ – искусственный интеллект.

Метаданные – дополнительные данные, описывающие изображение или объект.

Реестр моделей – хранилище версий моделей и их метаданных.

LLM (Large Language Model) – большая языковая модель.

MLflow – платформа для мониторинга экспериментов, логирования метрик и хранения артефактов.

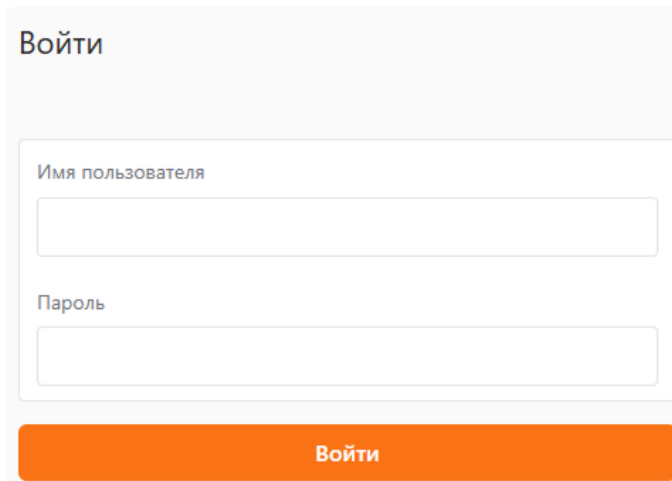
RAG (Retrieval Augmented Generation) – метод в области искусственного интеллекта, при котором генеративная языковая модель (LLM) обеспечивается доступом к внешним источникам информации для улучшения точности и достоверности ответов.

2 Работа с системой Диагноз

ДИАГНОЗ – это прикладной контур (веб-интерфейс), который отвечает за прием запросов пользователя, управление сценарием диагностики, сбор ответа, отображение результата, историю анализов.

2.1 Начало работы с системой

Для того чтобы начать работу с системой ДИАГНОЗ, откройте браузер и в строке адреса введите ссылку на приложение. После этого откроется страница входа в систему.



Войти

Имя пользователя

Пароль

Войти

Вход в систему

Для того чтобы войти в ДИАГНОЗ, заполните следующие поля:

- В поле «Имя пользователя» введите идентификатор вашей учетной записи.
- В поле «Пароль» укажите пароль доступа к вашей учетной записи.

После того как вы заполнили все поля, нажмите на кнопку **Войти**.

После успешного входа вы перейдете в пункт меню **Анализ**.

2.2 Описание пунктов меню

Главное меню, с помощью которого вы будете работать с программой, расположено в верхней левой части страницы.

Пункт меню	Описание
Анализ	Пункт меню Анализ предназначен для загрузки изображений, запуска диагностики, отображения результатов анализа и просмотра рекомендаций.
История	Пункт меню История предназначен для просмотра истории выполненных анализов.
Аналитика	Пункт меню Аналитика позволяет анализировать данные по накопленным диагностикам.

2.3 Анализ

В пункте меню **Анализ** вы можете загрузить изображения, выполнить диагностику, просмотреть результаты анализа и рекомендации.

Пункт меню **Анализ**

2.3.1 Выполнение диагностики

В данном разделе описаны пошаговые действия для выполнения анализа данных.

2.3.1.1 Шаг 1. Загрузка изображения

В блоке **Загрузить изображение** вы можете загрузить изображение для анализа следующими способами:

- Перетащите файл с рабочего стола или из папки в данную область.
- Нажмите на область «Нажмите для загрузки» и в открывшемся окне выберите интересующий файл на вашем компьютере.

Поддерживаются форматы: JPEG, PNG

В блоке **Модель распознавания** из выпадающего списка выберите модель для распознавания данных.

2.3.1.2 Шаг 2. Запуск диагностики

После того как все поля заполнены, нажмите на кнопку **Определить**.

В процессе анализа выполняются следующие операции:

- предобработка изображения;
- запуск модели распознавания;
- формирование одного или нескольких вариантов диагноза;
- расчет уровня уверенности;
- получение текстовых рекомендаций из подсистемы LLM.

2.3.1.3 Шаг 3. Просмотр результатов

Результаты анализа отображаются в правой части страницы:

- В блоке **Топ-3 результата** отображаются три наиболее вероятных заключения с уровнем уверенности диагностики (в процентах). Первый в списке – самый вероятный диагноз.
- В блоке **Культура** показано название распознанной культуры.
- В блоке **Все рекомендации** отображается полный перечень мер и предложений по улучшению состояния.
- На вкладке **LLM Рекомендации** показаны рекомендации, полученные от текущей модели.
- На вкладке **RAG Рекомендации** показаны улучшенные рекомендации, основанные на внешней базе знаний.

2.3.2 Просмотр и интерпретация результатов

2.3.2.1 Уровень уверенности

Уровень уверенности отражает степень достоверности результата и используется для оценки надежности диагноза.

Рекомендуется:

- при низком уровне уверенности повторить диагностику с другим изображением;
- при сомнениях обратиться к эксперту.

2.4 История

Система ДИАГНОЗ сохраняет результаты всех выполненных анализов.

В пункте меню **История** вы можете работать с историей выполненных анализов данных:

- просматривать ранее выполненные анализы;
- фильтровать записи;
- выполнять поиск по параметрам.

2.4.1 Просмотр истории

Для того чтобы найти интересующие вас записи, заполните параметры фильтра в левой части страницы и нажмите на кнопку **Применить фильтры**. В результате на экране отобразятся записи, соответствующие указанным критериям.

Для каждой записи в истории отображаются следующие сведения:

- номер записи,
- дата анализа,
- наименование загруженного файла,
- сведения об объекте анализа,
- статус,
- уровень достоверности результатов,
- результат анализа,
- используемая модель для анализа данных.

Вы можете отсортировать записи в списке по возрастанию или убыванию, а также отфильтровать их по интересующим вас критериям.

2.4.2 Экспорт результатов

При необходимости вы можете выполнить следующие действия:

- Если вы хотите скачать проанализированное изображение, нажмите на кнопку **Скачать фото**.
- Если вы хотите выгрузить результаты анализа в файл в формате .pdf, нажмите на кнопку **Сохранить детали записи в PDF**.
- Если вы хотите использовать данные для отчетности и аналитики, выгрузите историю анализов, нажав на кнопку **Сохранить таблицу в CSV**.

2.5 Аналитика

В пункте меню **Аналитика** вы можете работать с накопленными диагностиками.

The screenshot displays the 'Аналитика' (Analytics) section of a software interface. It features a date range selector at the top left with four radio buttons: 'Последние 7 дней' (selected), 'Последние 30 дней', 'Последние 90 дней', and 'Все время'. To the right is an orange button labeled 'Обновить аналитику'. Below the date selector is a box titled 'Количество анализов по периодам' containing a list of dates and their corresponding analysis counts. At the bottom, there are two side-by-side lists: 'Распределение по культурам' (Distribution by crop type) and 'Частота выявленных заболеваний' (Frequency of detected diseases).

Дата	Количество анализов
2026-05-08:	4
2026-05-15:	6
2026-05-18:	7
2026-05-19:	8
2026-05-27:	1
2026-05-29:	14
2026-06-02:	24

Культура	Процент
Трава / Газон:	1.56%
Кукуруза (Zea mays):	1.56%
Подсолнечник (Sunflower):	4.69%
Рис (Oryza sativa):	1.56%
Неизвестно (вероятно, подсолнечник или соя на ранней стадии):	1.56%
Соя:	1.56%
Unknown:	14.06%
Картофель:	7.81%
Ячмень:	1.56%

Заболевание	Частота
пероноспороз:	9 раз
Здоровый лист:	9 раз
Листовые рубцы:	9 раз
Серая гниль:	6 раз
Фузариоз колоса (Fusarium Head Blight):	2 раз
Гнили зерна (фузариоз/аспергиллез):	2 раз
Смаджица кукурузы (Ustilago maydis):	2 раз
Бурая ржавчина пшеницы (Puccinia triticina):	2 раз
Серая гниль (Botrytis cinerea):	2 раз
Бактериальная пятнистость:	2 раз

Пункт меню **Аналитика**

В данном пункте меню вы можете выполнить следующие действия:

- Просматривать статистику по всем выполненным диагностикам за выбранный период.
- Анализировать, какие данные чаще всего проверялись.
- Выявлять наиболее распространенные проблемы.
- Отслеживать динамику количества анализов во времени.

Для того чтобы построить отчет, в блоке **Диапазон дат** выберите интересующий временной интервал:

- Последние 7 дней
- Последние 30 дней
- Последние 90 дней
- Все время

После выбора диапазона все нижние блоки автоматически обновляются.

В блоке **Количество анализов по периодам** отображается число выполненных диагностик на конкретные даты.

В блоке **Распределение по культурам** показано процентное соотношение всех распознанных культур за выбранный период.

В блоке **Частота выявленных заболеваний** отображается список диагнозов, отсортированных по убыванию частоты встречаемости. Каждая запись содержит название диагноза и количество случаев.

2.6 Типичные ошибки и рекомендации

2.6.1 Ошибки пользователя

Возможные ошибки:

- загрузка неподдерживаемого формата файла;
- использование изображений низкого качества;
- отсутствие основного объекта в кадре.

2.6.2 Рекомендации по качеству изображений

Для повышения точности диагностики рекомендуется:

- использовать четкие изображения без размытия;
- фотографировать при хорошем освещении;
- фокусировать кадр на пораженном участке;
- избегать сильных теней и бликов.

2.7 Завершение работы

Для того чтобы завершить работу с программой ДИАГНОЗ, закройте окно браузера.

3 Работа с системой RS-LabelMark

RS-LabelMark – программный продукт для разметки данных, созданный на основе открытого проекта Label Studio (лицензия Apache 2.0, оригинальный автор – Heartex, Inc). RS-LabelMark позволяет размечать типы данных и поддерживает работу с несколькими проектами, пользователями и типами данных на одной платформе.

Он позволяет делать следующее:

- Выполнять различные виды разметки для множества форматов данных: аудио, текст, изображения, видео и временные ряды.
- Интегрировать RS-LabelMark с моделями машинного обучения для прогнозирования меток (предварительной разметки) или непрерывного активного обучения.

3.1 Быстрый старт

1. Откройте RS-LabelMark.
 2. Зарегистрируйтесь, указав адрес электронной почты и пароль.
 3. Нажмите **Создать**, чтобы создать проект и начать маркировку данных.
 4. Укажите название проекта, при желании добавьте описание и выберите цвет.
 5. Нажмите **Импорт данных** и загрузите файлы с данными, которые вы хотите использовать.
 6. Нажмите **Настройка маркировки**, выберите шаблон и настройте названия меток в соответствии с вашими задачами.
 7. Нажмите **Сохранить**, чтобы сохранить проект.
- Далее перейдите к [маркировке и аннотированию данных](#).

3.2 Вход и регистрация в системе

При первом запуске RS-LabelMark вы увидите экран регистрации.

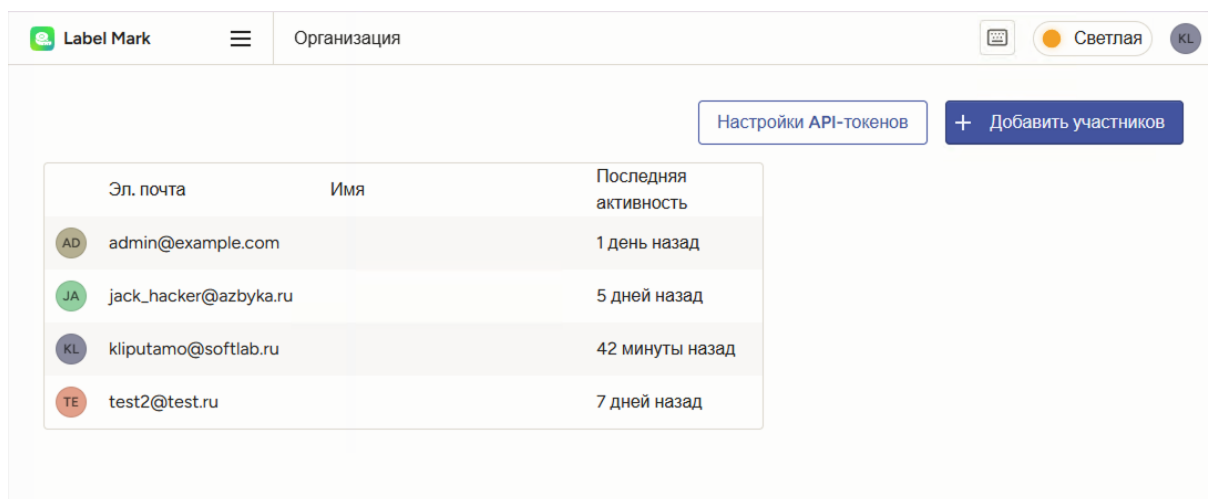
1. Создайте учетную запись, указав свой адрес электронной почты и пароль.
2. Войдите в RS-LabelMark.

Созданные вами учетные записи хранятся локально на сервере RS-LabelMark и позволяют нескольким разметчикам совместно работать над конкретным проектом по разметке данных.

3.3 Добавление пользователей

Зарегистрируйтесь и создайте учетную запись в RS-LabelMark, чтобы начать маркировку данных и настройку проектов.

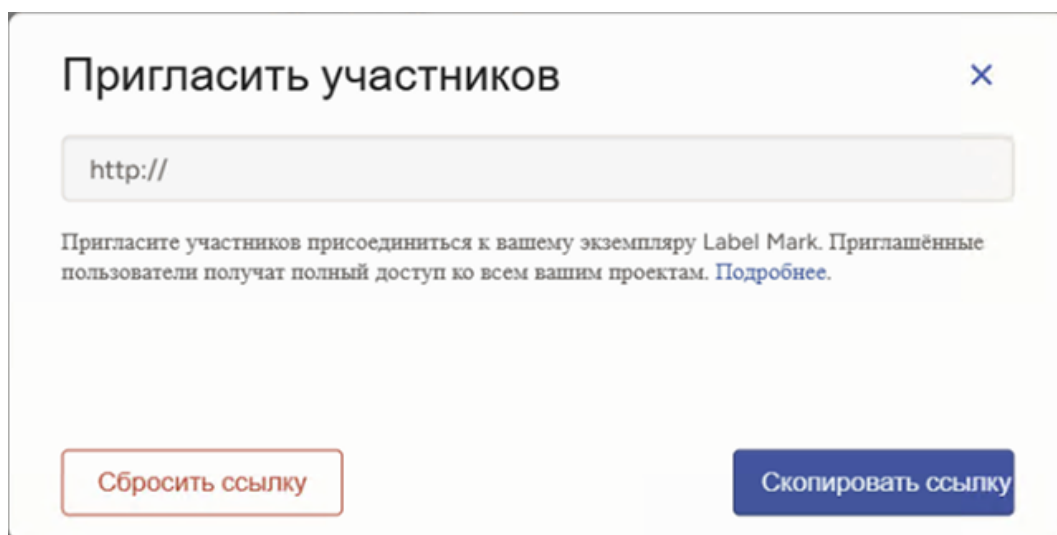
После создания первоначальной учетной записи организации вы можете начать приглашать пользователей:



Страница «Организация»

1. Откройте RS-LabelMark и нажмите на меню в левом верхнем углу.
2. Выберите **Организация**.
3. На странице «Организация» нажмите **Добавить участников**.

Откроется окно, в котором вы можете скопировать уникальную для вашей организации ссылку. Далее вы можете отправить пользователям ссылку для завершения процесса регистрации.

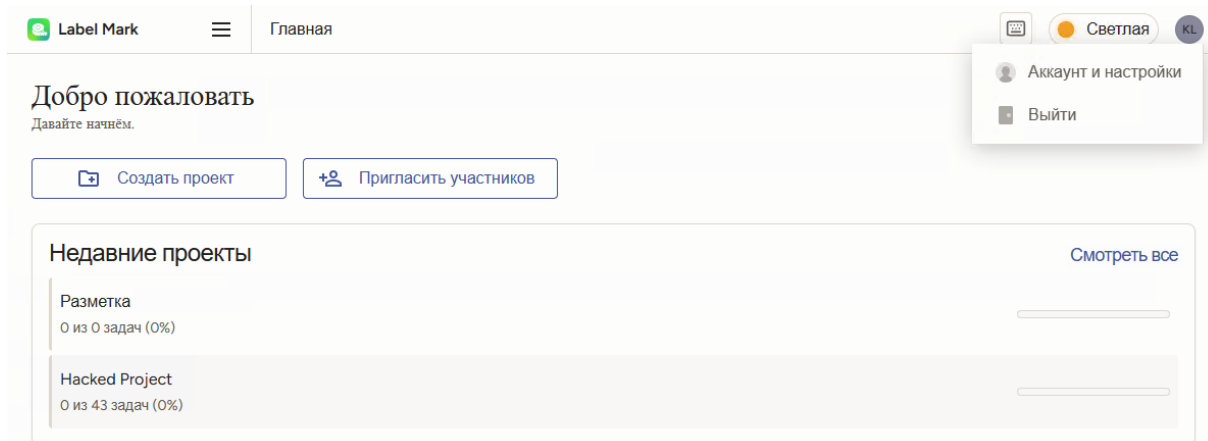


Приглашение участников

Нажмите **Сбросить ссылку**, чтобы сбросить ее. После сброса все, у кого есть старая ссылка, не смогут зарегистрировать новую учетную запись.

3.4 Личный кабинет

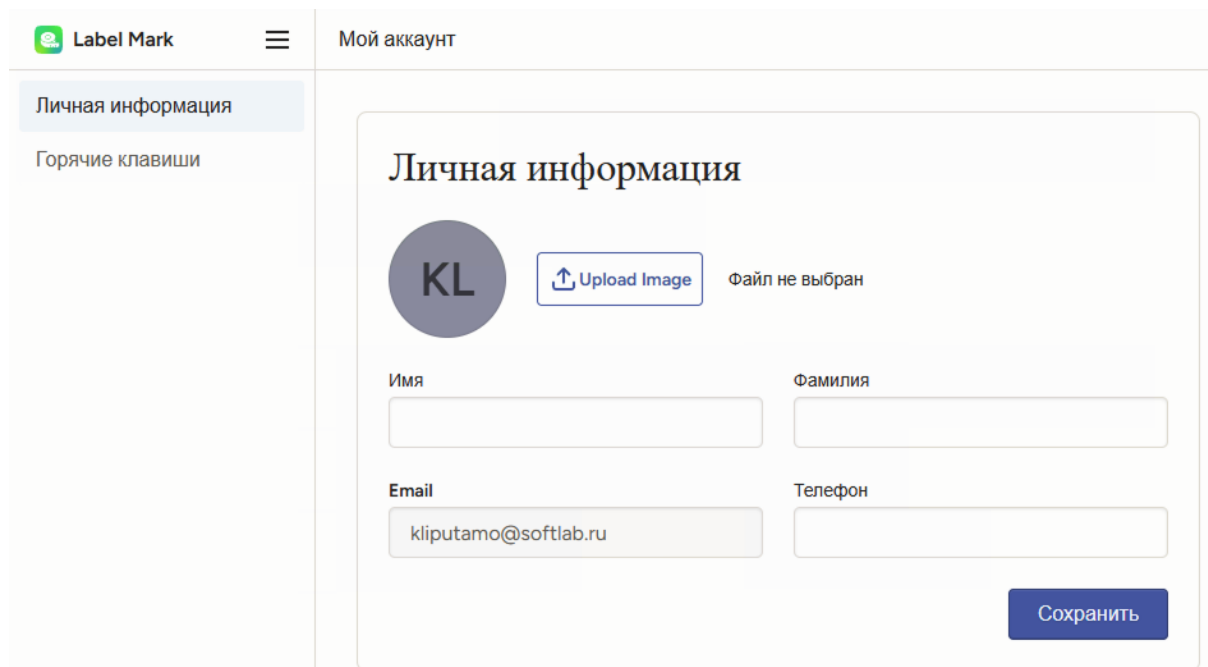
Чтобы получить доступ к своей учетной записи и настройкам, нажмите на значок своего профиля в правом верхнем углу и выберите **Аккаунт и настройки**.



Главная страница. Переход в аккаунт и настройки

3.4.1 Личная информация

В аккаунте на вкладке **Личная информация** вы можете изменить следующие данные:



Личная информация

- Имя
- Фамилия
- Номер телефона
- Изображение профиля (используйте изображение размером не более 1200 x 1200 пикселей или 1024 КБ)

Поскольку ваш адрес электронной почты является вашим логином, вы не можете его изменить. Если вам нужно изменить адрес электронной почты, создайте нового пользователя с новым адресом электронной почты.

3.4.2 Горячие клавиши

В аккаунте на вкладке **Горячие клавиши** вы можете настроить собственные горячие клавиши для своей учетной записи.

Label Mark

Мой аккаунт

Личная информация

Горячие клавиши

Горячие клавиши

ИмпортЭкспортСбросить по умолчанию

Настройте сочетания клавиш, чтобы ускорить работу. Нажмите на любую горячую клавишу ниже, чтобы назначить новое сочетание, которое вам удобнее.

Действия аннотации

Сочетания клавиш для типовых действий: отправить, пропустить, отменить и повторить

Отправить аннотацию

Отправить текущую аннотацию

CTRLENTER

Пропустить задачу

Пропустить текущую задачу

CTRLSPACE

Отменить

Отменить последнее действие

CTRLZ

Повторить

Повторить ранее отменённое действие

CTRLSHIFTZ

Сохранить

Горячие клавиши

Действия:

Действие	Описание
Экспорт горячих клавиш	Создайте файл в формате JSON, которым можно поделиться с другими пользователями RS-LabelMark.
Импорт горячих клавиш	Скопируйте и вставьте пользовательское сопоставление горячих клавиш в формате JSON.
Возврат к настройкам по умолчанию	Верните все горячие клавиши к настройкам по умолчанию.

Для настройки каждой горячей клавиши нажмите на горячие клавиши по умолчанию, указанные рядом с каждым действием. Затем нажмите клавиши, которые вы хотите использовать вместо них.

Каждая горячая клавиша может представлять собой одну буквенную клавишу (цифру, букву, символ) или комбинацию буквенной клавиши и нескольких управляющих клавиш (например, **Shift**, **Control Option**, и т. д.).

3.5 Создание и настройка проектов

Все действия по маркировке в RS-LabelMark выполняются в контексте проекта.

3.5.1 Процесс настройки проекта

После того как вы запустили RS-LabelMark и создали учетную запись, создайте проект, для того чтобы приступить к разметке данных.

1. [Создайте проект](#)
2. [Импортируйте данные в RS-LabelMark](#)

Для небольших проектов, тестирования или проверки концепции вы можете импортировать локальные файлы прямо в RS-LabelMark через пользовательский интерфейс. Это можно сделать во время создания проекта.

Для более крупных проектов рекомендуем настроить облачное хранилище.

3. [Настройте интерфейс для маркировки](#)

Рекомендуется начать с предварительно настроенного шаблона и внести в него изменения, но вы также можете создать собственный шаблон с помощью нашей библиотеки тегов.

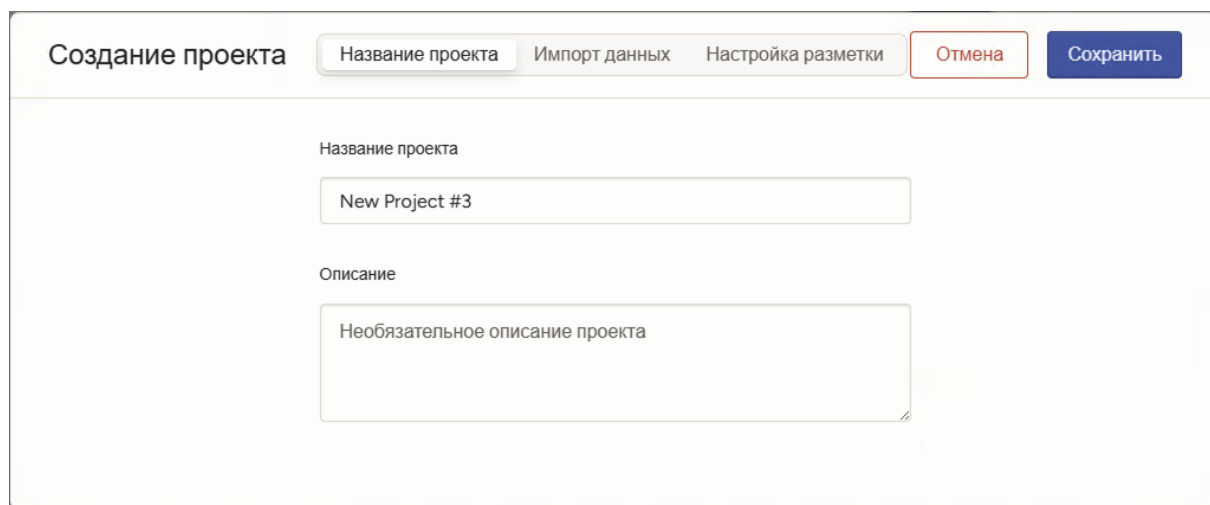
4. [Настройте параметры проекта](#)

Наконец, вы можете настроить дополнительные параметры, такие как выборка задач и инструкции по аннотированию.

3.5.2 Создание проекта

В RS-LabelMark в пункте меню **Проекты** нажмите **Создать проект** в правом верхнем углу.

В результате откроется окно с тремя вкладками:



Создание проекта

Название проекта Импорт данных Настройка разметки

Отмена Сохранить

Название проекта

New Project #3

Описание

Необязательное описание проекта

Название проекта

Это единственный обязательный раздел.

В данном разделе выберите свое рабочее пространство, введите название проекта и описание проекта.

После этого нажмите **Сохранить**, чтобы создать проект, или перейдите на другие вкладки.

Создание проекта

Название проекта

Импорт данных

Настройка разметки

Отмена

Сохранить

URL набора данных

Добавить URL

или

Загрузить файлы



Перетащите файлы сюда
или нажмите, чтобы выбрать

Изображения	bmp, gif, jpg, jpeg, png, svg, webp
Аудио	wav, mp3, flac, m4a, ogg
Видео 	mp4, webm
HTML	html, htm, xml
Текст	txt
Структурированные данные	csv, tsv, json
PDF	pdf

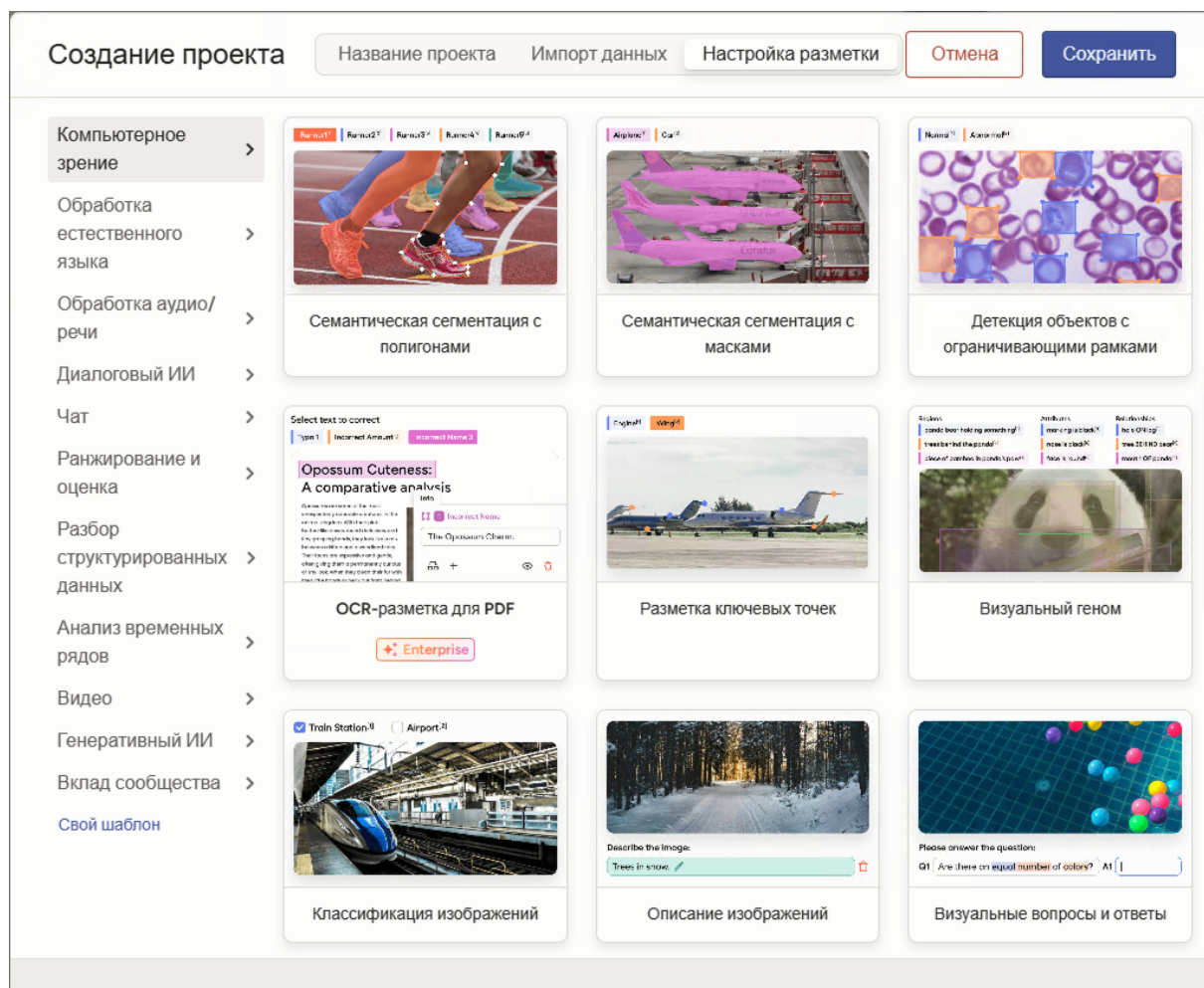
Важно:

- Рекомендуем [Cloud Storage](#) вместо прямой загрузки из-за ограничений загрузки.
- Для PDF используйте [разметку многостраничных документов](#). Файлы JSONL или Parquet (только Enterprise) требуют облачное хранилище.
- Ознакомьтесь с документацией, чтобы импортировать преданнотированные данные.

Импорт данных

В данном разделе вы можете загружать файлы в RS-LabelMark. Это можно сделать сразу или после создания проекта.

Для более крупных проектов рекомендуем настроить облачное хранилище или использовать другой способ импорта.



Настройка разметки

В данном разделе вы можете выбрать шаблон для настройки разметки. Для упрощения настройки выберите шаблон. Позже вы сможете настроить шаблон в соответствии со своими потребностями.

Когда закончите, нажмите **Сохранить**.

3.5.3 Настройка параметров аннотирования

Перед тем как пользователи начнут аннотировать данные, вы можете настроить несколько параметров:

- Инструкции для аннотаторов (разметчиков данных)

Вы можете выбрать, будут ли эти инструкции отображаться во всплывающем окне, когда пользователь переходит в режим аннотирования. Режим аннотирования активируется, когда пользователь нажимает **Аннотировать все задачи**.

- Выборка задач

Выборка задач определяет порядок, в котором задачи отображаются пользователям в режиме аннотирования. По умолчанию используется последовательная выборка, то есть пользователи видят задачи в том порядке, в котором они отсортированы в **Менеджере данных**. Вы можете изменить этот параметр, чтобы задачи отображались в случайном порядке.

В **Менеджере данных** нажмите **Настройки** в правом верхнем углу. Вы также можете открыть настройки в дополнительном меню для каждого проекта.

3.6 Настройка интерфейса разметки

Все действия по разметке в RS-LabelMark выполняются в рамках проекта. После того как вы создали проект и импортировали данные, настройте интерфейс и конфигурацию разметки для своего проекта.

3.6.1 Настройка интерфейса разметки для проекта

Настройте метки и тип задачи для аннотаторов с помощью шаблонов, входящих в состав RS-LabelMark, или создайте собственную комбинацию тегов, чтобы настроить интерфейс разметки для своего проекта.

1. Выберите шаблон из доступных шаблонов или создайте свой.
2. RS-LabelMark автоматически выбирает поле для разметки на основе ваших данных. При необходимости измените выбранное поле.
3. Добавляйте названия меток с новой строки.
4. Выберите новые цвета для меток, нажав на название метки и выбрав новый цвет с помощью селектора.
5. Настройте дополнительные параметры, связанные с функционалом интерфейса разметки. Например, при разметке текста у вас может быть возможность выделять текст по словам.
6. Нажмите на кнопку **Сохранить**.

3.6.1.1 Изменение интерфейса разметки

Вы можете внести изменения в интерфейс и конфигурацию разметки в настройках проекта.

ПРИМЕЧАНИЕ: если вы вносите изменения в проект, над которым ведется работа, обратите внимание на следующее:

- Вы не сможете удалить метки или изменить тип выполняемой разметки, пока не удалите все существующие аннотации, в которых используются эти метки.
- При внесении изменений в конфигурацию разметки все вкладки, которые вы могли создать в **Диспетчере данных**, удаляются.

1. В RS-LabelMark откройте проект, который хотите изменить.
2. Нажмите **Настройки**.
3. Нажмите **Интерфейс** для создания меток.
4. Просмотрите шаблоны, обновите доступные метки или воспользуйтесь опцией **Code** для дополнительной настройки интерфейса с помощью тегов.

3.6.1.2 Автозаполнение в режиме просмотра кода

В режиме просмотра кода есть помощник по автозаполнению, который появляется по мере ввода.

Автозаполнение включает подсказки как для тегов, так и для параметров, доступных для выбранного тега.

Предложения по тегам появляются после ввода открывающей угловой скобки <. Предложения по параметрам появляются после добавления пробела внутри тега.

Чтобы принять предложение, нажмите на него или нажмите клавишу Tab, когда предложение будет выделено.

3.6.2 Настройка шаблона для разметки

Вы можете настроить шаблон конфигурации для разметки или использовать собственную конфигурацию, созданную с нуля с помощью тегов.

Конфигурация разметки для проекта представляет собой XML-файл, содержащий три типа тегов, характерных для RS-LabelMark.

Тип тега	Когда использовать
Объект	Укажите тип данных и источники входных данных из вашего набора данных.
Управление	Настройте тип аннотирования и способ отображения результатов аннотирования.
Визуализация	Определите, как пользовательский интерфейс будет отображать разметку.

Вы можете комбинировать эти теги, чтобы создать собственную конфигурацию разметки для своего набора данных.

3.6.2.1 Пример конфигурации разметки

Например, чтобы классифицировать изображения, на которые в ваших данных есть ссылки в виде URL-адресов (**\$image_url**), по двум категориям – «Кошка» или «Собака», используйте следующую конфигурацию разметки:

```
<View>
  <Image name="image_object" value="$image_url"/>
  <Choices name="image_classes" toName="image_object">
    <Choice value="Cat"/>
    <Choice value="Dog"/>
  </Choices>
</View>
```

Эта конфигурация разметки ссылается на ресурс изображения в теге объекта **Image** и указывает доступные метки для выбора в теге управления **Choices**.

Если вы хотите изменить этот пример, например разрешить разметчикам выбирать метки «Кошка» и «Собака» для одного изображения, измените параметры, используемые в теге управления **Choices**:

```
<View>
  <Image name="image_object" value="$image_url"/>
  <Choices name="image_classes" toName="image_object" choice="multiple">
    <Choice value="Cat"/>
    <Choice value="Dog"/>
  </Choices>
</View>
```

3.6.2.2 Добавление конфигурации разметки из командной строки

Вы можете задать конфигурацию разметки в файле **config.xml** и инициализировать с ее помощью конкретный проект в RS-LabelMark.

```
bashlabel-RS-LabelMark my_new_project start --label-config config.xml
```

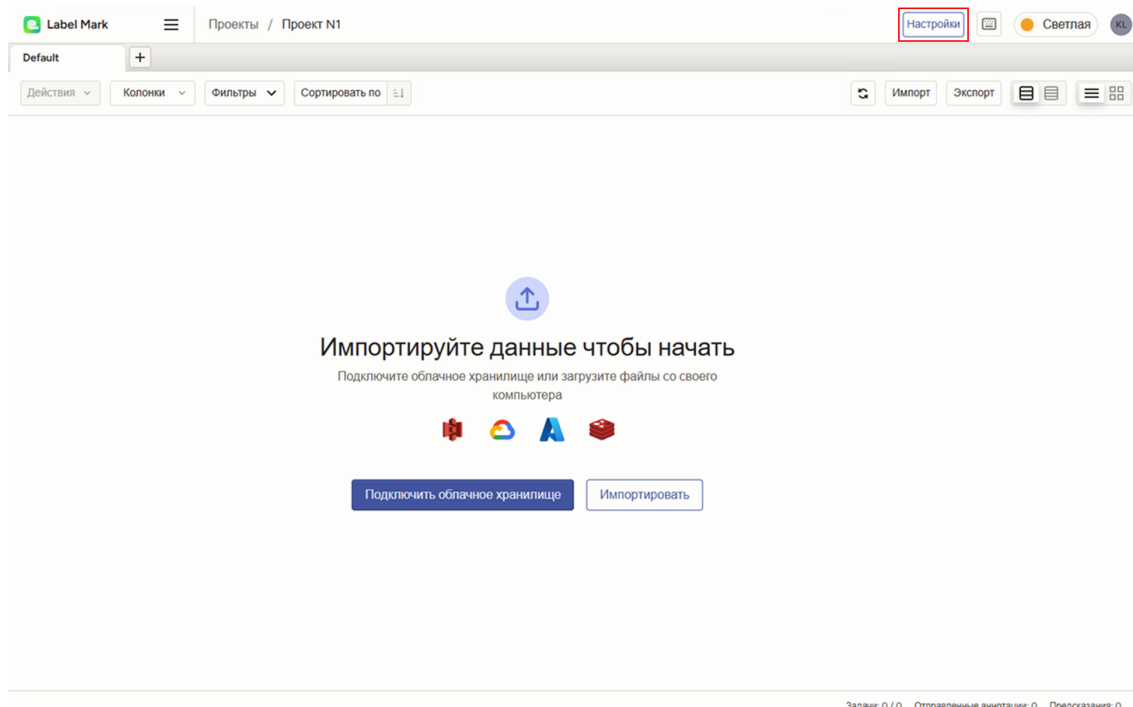
3.7 Управление проектами

В данном разделе описаны следующие возможности управления проектами:

- Настройка параметров проекта
- Использование менеджера данных для управления задачами по разметке

3.7.1 Настройки проекта

Для того чтобы перейти к настройкам проекта, выберите проект и в правом верхнем углу нажмите **Настройки**. В результате вы перейдете на страницу настроек проекта.



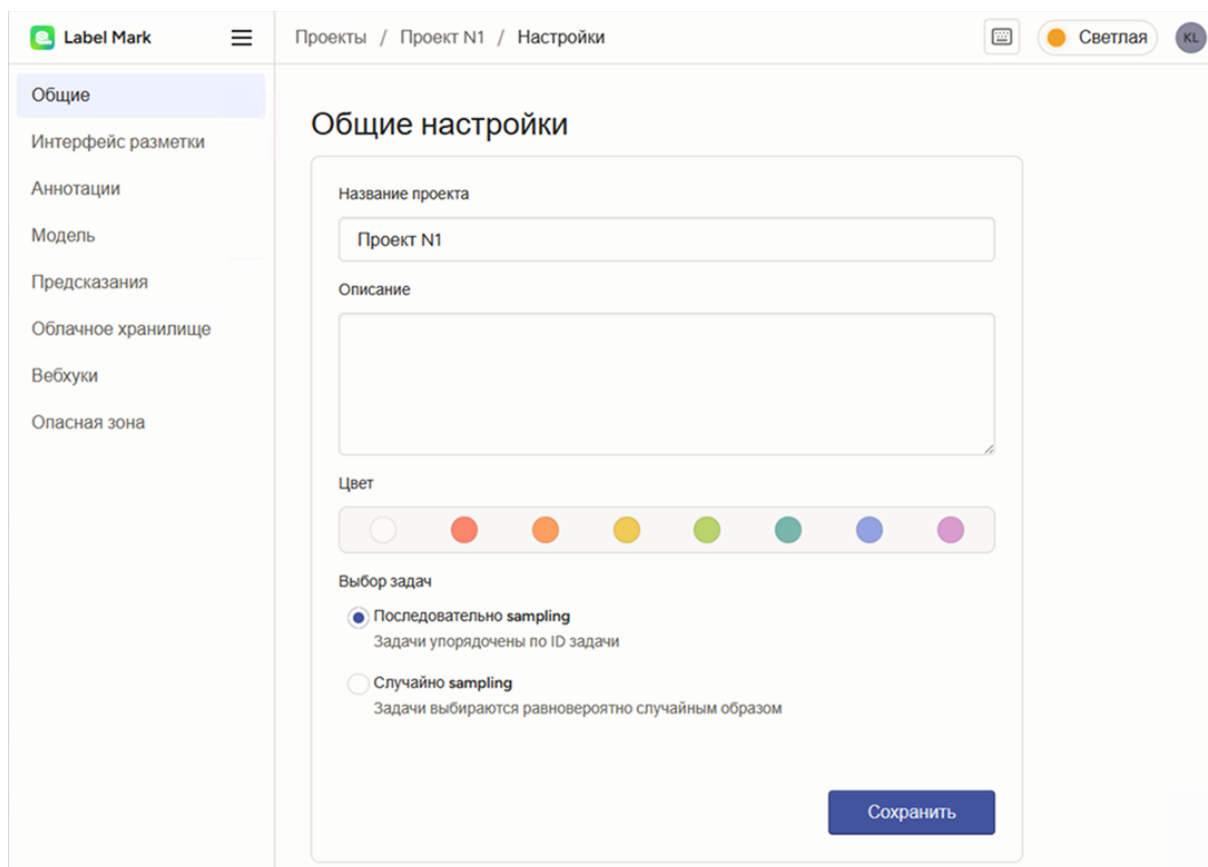
Страница проекта. Переход к настройкам

3.7.1.1 Общие настройки

На вкладке **Общие** вы можете указать основную информацию о проекте.

Правила заполнения формы:

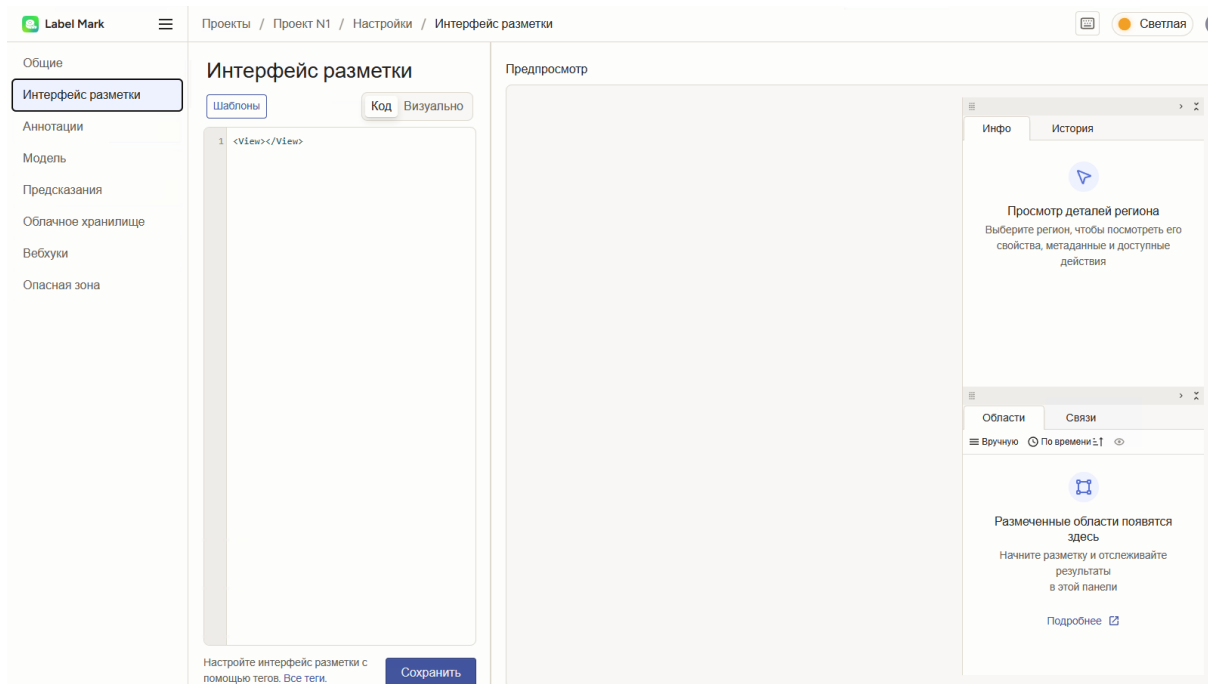
Поле	Описание
Название проекта	Введите название проекта.
Описание	Введите описание проекта.
Цвет	Вы можете выбрать цвет проекта. Проект будет выделен этим цветом при просмотре страницы «Проекты».
Выборка задач	• Последовательная выборка – задачи отображаются для аннотаторов в том же порядке, в котором они отображаются в Менеджере данных • Случайная выборка – задачи отображаются в случайном порядке.



Вкладка Общие

3.7.1.2 Интерфейс разметки

Интерфейс разметки – это центральная точка настройки проектов. Он определяет, как задачи будут представлены разметчикам.



Вкладка Интерфейс разметки

Информацию о настройке интерфейса разметки см. в разделе [Настройка интерфейса разметки](#).

3.7.1.3 Аннотации

На вкладке **Аннотации** вы можете выполнить следующие действия:

The screenshot shows the 'Label Mark' application interface. On the left is a sidebar with a menu: 'Общие', 'Интерфейс разметки', 'Аннотации' (highlighted), 'Модель', 'Предсказания', 'Облачное хранилище', 'Вебхуки', and 'Опасная зона'. The main area is titled 'Настройки аннотаций'. It contains two sections: 'Инструкции по разметке' and 'Предразметка'. The 'Инструкции по разметке' section has a text area for instructions, a toggle switch for 'Показывать перед разметкой' (currently off), and a large empty text box. The 'Предразметка' section has a toggle switch for 'Использовать предсказания для предразметки задач' (currently on), a text box for selecting a model or prediction set (currently showing 'Нет доступных моделей или предсказаний'), and a 'Сохранить' button.

Вкладка Аннотации

Инструкции по разметке

Укажите инструкции, которые будут отображаться перед пользователями во время аннотирования задачи. Это поле поддерживает форматирование в формате HTML.

Включите параметр **Показывать перед разметкой**, чтобы при входе в поток разметки пользователям отображалось всплывающее сообщение.

Если этот параметр отключен, пользователям нужно будет нажать кнопку **Показать инструкции** в нижней части интерфейса разметки.

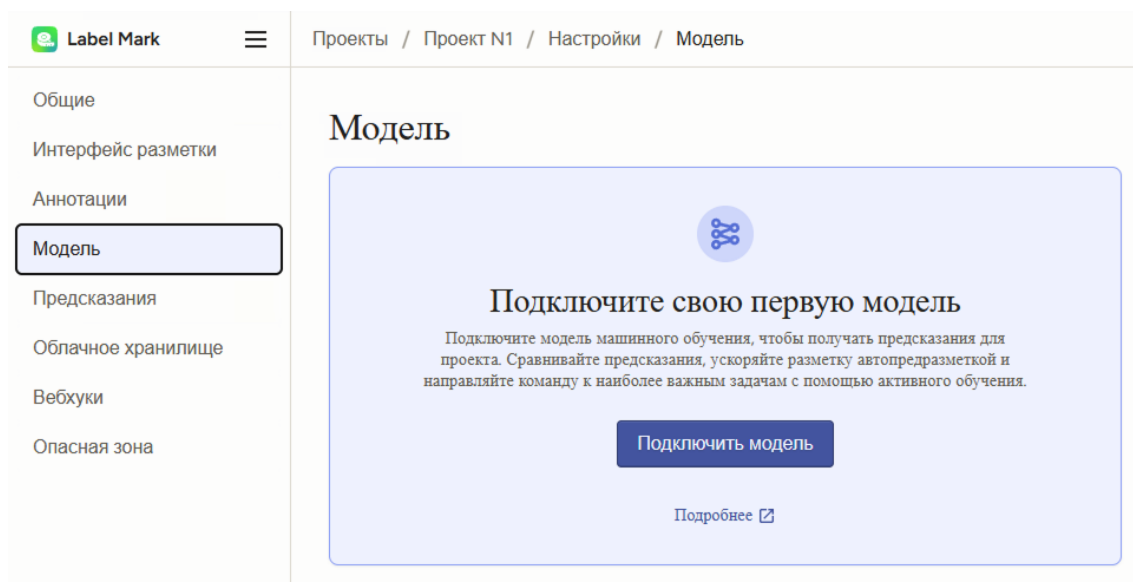
Предразметка

Если у вас есть серверная часть или модель машинного обучения, вы можете использовать этот параметр, чтобы определить, нужно ли предварительно помечать задачи с помощью прогнозов модели.

В раскрывающемся меню выберите источник прогнозов. Например, вы можете выбрать подключенную модель или набор прогнозов.

3.7.1.4 Модель

На вкладке **Модель** вы можете выполнить следующие действия:



Вкладка Модель

Нажмите **Подключить модель**, чтобы подключить серверную часть машинного обучения (ML) к своему проекту.

Есть следующие варианты настройки:

Поле	Описание
Начать обучение модели при отправке аннотаций	Запускает процесс обучения подключенного сервера машинного обучения при создании или обновлении аннотации. Это часть цикла активного обучения, в рамках которого модель может постоянно совершенствоваться по мере добавления новых аннотаций в набор данных. Если этот параметр включен, вызывается метод fit() сервера машинного обучения, что позволяет модели учиться на основе последних аннотаций и потенциально улучшать свои прогнозы для последующих задач.
Интерактивные предварительные аннотации	(доступно при создании или редактировании подключения к модели) Включите эту опцию, чтобы модель помогала в процессе разметки, предоставляя прогнозы или предложения в режиме реального времени, пока разметчики работают над задачами. Когда вы взаимодействуете с данными (например, рисуете область на изображении, выделяете текст или задаете вопрос большой языковой модели), серверная часть машинного обучения получает эти данные и на их основе выдает прогнозы.

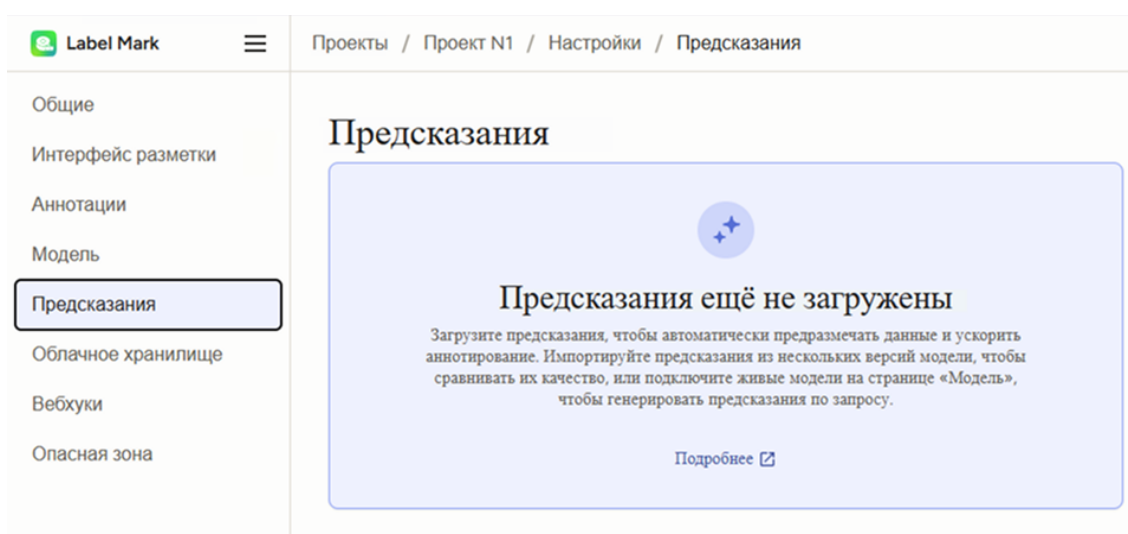
В выпадающем меню рядом с подключенной моделью доступны следующие действия:

Действие	Описание
Начать обучение	Запустить обучение вручную. Используйте это действие, если хотите контролировать процесс обучения модели, например после сбора определенного количества аннотаций или через определенные промежутки времени.

Действие	Описание
Отправить тестовый запрос	(доступно в выпадающем меню рядом с подключенной моделью) Используйте эту функцию для устранения неполадок и отправки тестового запроса в подключенную модель.
Редактировать	Измените название модели, URL-адрес и параметры.
Удалить	Отмените подключение к модели.

3.7.1.5 Предсказания

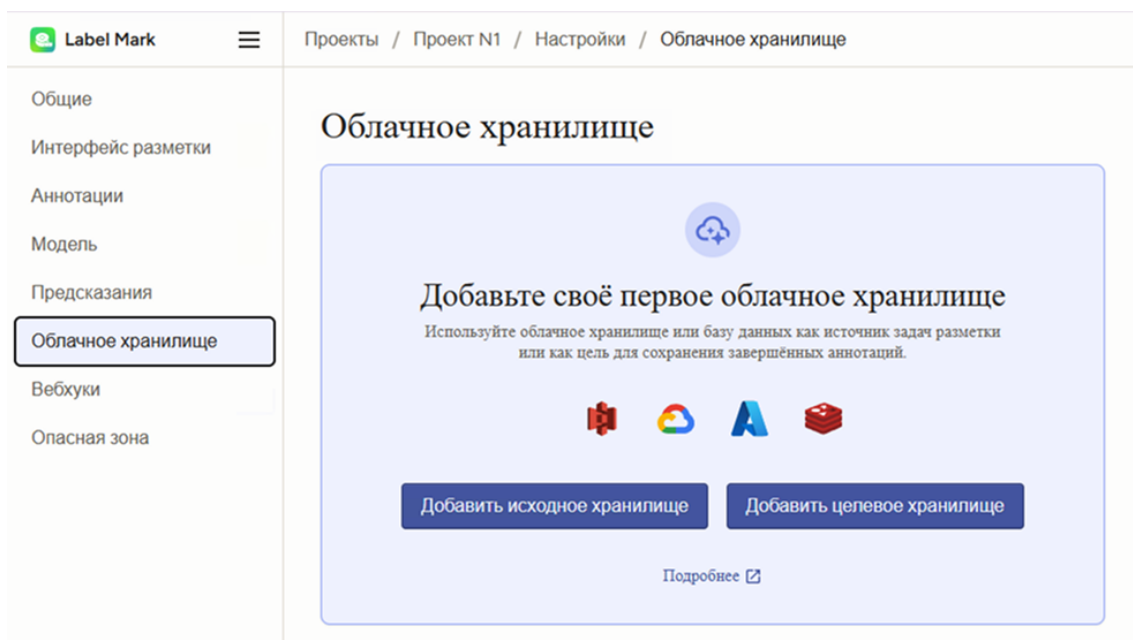
Здесь вы можете просмотреть прогнозы, импортированные или сгенерированные при выполнении действия **Получить прогнозы** в Менеджере данных. Дополнительную информацию см. в разделе [Импорт предварительно аннотированных данных](#).



Вкладка Предсказания

3.7.1.6 Облачное хранилище

На вкладке **Облачное хранилище** вы можете подключить RS-LabelMark к провайдеру облачного хранилища:

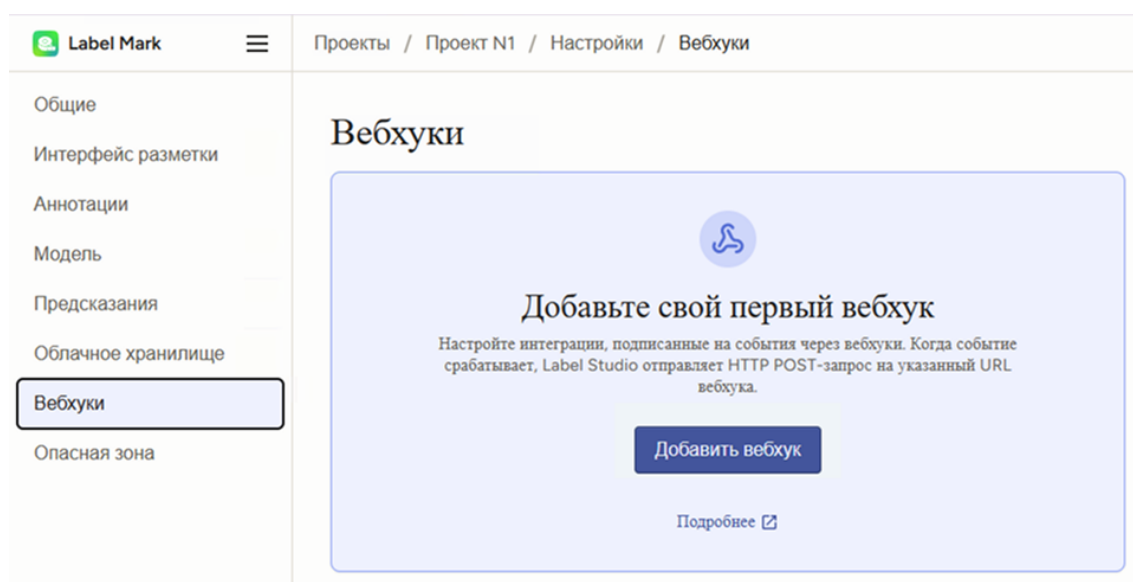


Вкладка Облачное хранилище

- **Исходное облачное хранилище** – здесь сохраняются исходные данные для вашего проекта. При синхронизации исходного хранилища RS-LabelMark извлекает данные для аннотирования.
- **Целевое облачное хранилище** – здесь сохраняются ваши аннотации. При синхронизации целевого хранилища аннотации отправляются из RS-LabelMark в целевое хранилище.

3.7.1.7 Вебхуки

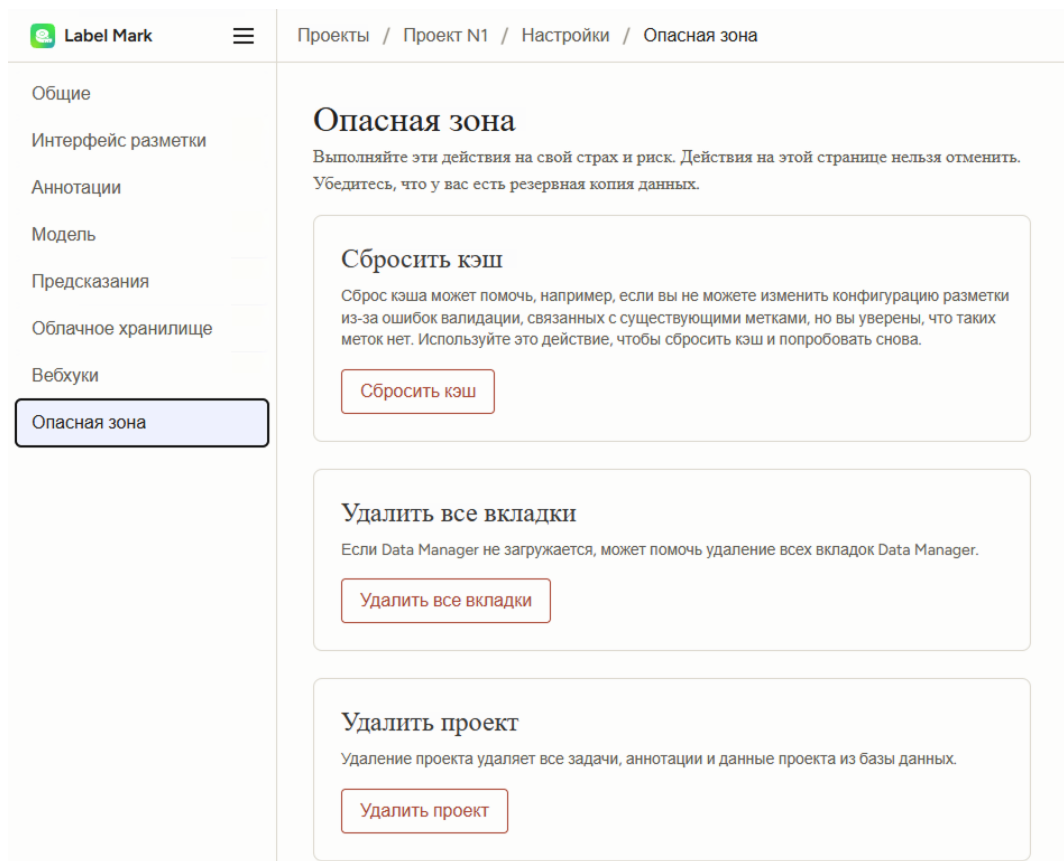
С помощью вебхуков можно интегрировать сторонние приложения.



Вкладка Вебхуки

3.7.1.8 Опасная зона

Здесь вы можете получить доступ к действиям, которые приводят к потере данных и должны выполняться с осторожностью.



Вкладка Опасная зона

- **Сбросить кэш**

Сброс кэша может помочь, например, если вы не можете изменить конфигурацию разметки из-за ошибок валидации, связанных с существующими метками, но вы уверены, что таких меток нет. Используйте это действие, чтобы сбросить кэш и попробовать снова.

- **Удалить все вкладки**

Если диспетчер данных не загружается, попробуйте закрыть все вкладки диспетчера данных.

- **Удалить проект**

При удалении проекта из RS-LabelMark безвозвратно удаляются все задачи, аннотации и данные проекта.

3.7.2 Менеджер данных

На странице Менеджера данных вы можете просматривать все задачи по маркировке, сортировать и фильтровать данные, импортировать и экспортировать их, а также выполнять различные действия, связанные с задачами.

The screenshot shows the 'Label Mark' application interface. At the top, there's a header with the application name, a menu icon, the project name 'Проекты / Hacked Project', and user controls like 'Настройки', 'Светлая', and a profile icon. Below the header is a toolbar with buttons for 'Действия', 'Колонки', 'Фильтры', 'Сортировать по', and 'Разметить все задачи'. The main area is a table with columns: ID, Completed, Annotated by, url, image, and data. The table contains 11 rows of data, each representing a task. The 'Completed' column has checkboxes, and the 'Annotated by' column has status icons. The 'url' column contains various HTTP addresses, and the 'data' column contains JSON-like structures or text snippets.

ID	Completed	Annotated by	url	image	data
2	☐	0 0 0			owned by jack {}
3	☐	0 0 0			owned by jack {}
4	☐	0 0 0			<?xml version="1.0" encoding="UTF-8"?><DOCTYPE foo [...] {}
5	☐	0 0 0			/data/uploads/4/95d75042-test.org {}
6	☐	0 0 0	http://127.0.0.1:1111/		{}
7	☐	0 0 0	http://127.0.0.1:22/		{}
8	☐	0 0 0	http://127.0.0.1:80/		{}
9	☐	0 0 0	http://127.0.0.1:443/		{}
10	☐	0 0 0	http://127.0.0.1:1111/		{}
11	☐	0 0 0	http://127.0.0.1:2048/		{}

Менеджер данных

Каждая строка в менеджере данных представляет собой задачу по разметке в вашем наборе данных.

3.7.2.1 Фильтрация или сортировка данных проекта

С помощью фильтров и вкладок вы можете разделить данные на разделы, чтобы их разметкой занимались разные аннотаторы, или упорядочить данные для выполнения разметки в соответствии с оценкой прогнозирования или другим значением в ваших данных.

При фильтрации или сортировке данных перед разметкой вы меняете набор задач и порядок их отображения при разметке. В то время как выборка задач влияет на порядок выполнения задач в рамках всего проекта и не может быть изменена, фильтрацию и сортировку задач можно менять в любое время.

Пример: сначала аннотируйте новые данные

Отсортируйте данные в проекте по дате, чтобы в первую очередь аннотировать самые новые.

1. В проекте измените **порядок** данных со значения по умолчанию на **по дате создания**.
2. Измените порядок элементов на возрастающий, чтобы самые новые элементы отображались первыми.
3. Выберите **Пометить задачи как отображаемые**, чтобы начать пометить задачи в порядке от самых новых к самым старым.

Пример: сортировка по оценке прогноза

Вы можете отсортировать данные в своем проекте по оценке вероятности, если загрузите предварительно размеченные данные с оценками вероятности или если ваш сервер машинного обучения выдает оценки вероятности в качестве выходных данных модели.

1. В проекте измените **порядок** сортировки данных со стандартного на сортировку по полю **Оценка вероятности**.
2. Измените порядок элементов на возрастающий или убывающий, чтобы выполнять разметку на основе прогнозов с более высокой или низкой степенью достоверности.

3. Выберите **Разметить задачи в порядке отображения**, чтобы начать разметку задач в порядке оценки вероятности. Вы также можете использовать выборку задач для упорядочивания оценок прогнозирования.

Пример: разделение набора данных с помощью вкладок и фильтров

Если вам нужно разметить большой набор данных, вы можете использовать вкладки и фильтры, чтобы разделить его на более мелкие разделы и назначить разным аннотаторам разные вкладки. В RS-LabelMark нельзя назначать аннотаторов для выполнения конкретных задач, но можно переименовывать вкладки в именах аннотаторов, чтобы распределять задачи с помощью вкладок.

Например, можно разделить набор данных из 300 изображений на 3 вкладки и поручить разным аннотаторам работу с каждой из них:

1. В проекте создайте фильтр, в котором поле **ID находится в диапазоне** значений от «1» до «100». Чтобы просмотреть отфильтрованные элементы на вкладке, нажмите за пределами фильтра.

2. Нажмите на вертикальное многоточие на вкладке и выберите **Переименовать**. Назовите ее в честь конкретного аннотатора, которому вы хотите показать элементы на этой вкладке.

3. Нажмите на значок +, чтобы создать новую вкладку. Нажмите на вертикальное многоточие рядом с новой вкладкой и выберите **Переименовать**, чтобы назвать ее в честь второго аннотатора.

4. На новой вкладке создайте фильтр, в котором поле **ID находится в диапазоне** значений от «101» до «200». Нажмите за пределами фильтра, чтобы просмотреть отфильтрованные элементы на вкладке.

5. Нажмите на значок + для создания новой вкладки. Нажмите на вертикальное многоточие рядом с новой вкладкой и выберите **Переименовать**, чтобы назвать ее в честь третьего аннотатора.

6. На новой вкладке создайте фильтр, в котором поле **ID находится в диапазоне** значений от «201» до «300». Чтобы просмотреть отфильтрованные элементы на вкладке, нажмите за пределами фильтра.

7. Любой аннотатор может войти в систему, перейти на вкладку, соответствующую его работе, и нажать кнопку **Разметить**, чтобы начать разметку подмножества задач на своей вкладке.

3.7.2.2 Сохранение фильтров в виде вкладок

Вы можете создавать вкладки в менеджере данных, чтобы разделить набор данных для разметки, разделить задачи по статусу (аннотированные, предсказанные, неразмеченные) или по другим причинам.

Создаваемые вами вкладки зависят от настройки конфигурации разметки, поскольку конфигурация разметки определяет поля данных, доступные для фильтрации и сортировки.

Если вы вносите изменения в конфигурацию разметки, все вкладки в менеджере данных удаляются. Обязательно завершите настройку проекта, прежде чем создавать сложные вкладки в менеджере данных.

3.7.2.3 Удаление задач или аннотаций

Если у вас есть повторяющиеся задачи или вы хотите удалить аннотации, вы можете удалить задачи и аннотации в RS-LabelMark.

1. В пользовательском интерфейсе RS-LabelMark откройте проект, который нужно обновить.

2. Отфильтруйте страницу менеджера данных, чтобы отображались только те данные, которые вы хотите удалить. Например, определенные аннотации или задачи, аннотированные конкретным аннотатором.
3. Установите флажки напротив задач или аннотаций, которые хотите удалить.
4. В раскрывающемся списке выберите количество задач и нажмите **Удалить задачи** или **Удалить аннотации**.
5. Нажмите **ОК**, чтобы подтвердить действие.

Если вы хотите внести изменения в интерфейс разметки или выполнить разметку данных другого типа, сначала выберите все аннотации для своего набора данных и удалите их.

3.8 Импорт данных

Вы можете перенести данные в RS-LabelMark, импортировав файлы, указав URL-адреса или синхронизировав данные с облачным хранилищем или базой данных.

3.8.1 Общие рекомендации по импорту данных

- Для оптимальной производительности лучше всего хранить около 100 000 задач / 100 000 аннотаций на проект.
- Избегайте частого импорта, так как каждый новый импорт требует длительных фоновых операций. Один импорт в 30 секунд не приведет к перегрузке.

Обратите внимание:

При работе над масштабными проектами или проектами, критически важными для бизнеса, не загружайте медиафайлы через интерфейс RS-LabelMark. Это особенно актуально для таких файлов, как изображения, аудио, видео, таймсерии и т. д.

RS-LabelMark не предназначена для масштабного хостинга и не имеет резервных копий для импортируемых медиаресурсов.

Риски при загрузке через пользовательский интерфейс:

- Импорт задач с прогнозами
- Экспорт данных
- Перенос данных в другой экземпляр RS-LabelMark
- Повторное развертывание RS-LabelMark

Рекомендуется настроить исходное хранилище.

3.8.2 Типы данных, которые можно импортировать

Вы можете импортировать множество типов данных, включая текст, временные ряды, аудио и изображения. Поддерживаемые типы файлов зависят от типа данных.

Тип данных	Поддерживаемые типы файлов
Аудио	.flac, .m4a, .mp3, .ogg, .wav
Гипертекст (HTML)	.html, .htm, .xml
Изображения	.bmp, .gif, .jpg, .png, .svg, .webp

Тип данных	Поддерживаемые типы файлов
Параграфы (диалоги)	.json
Структурированные данные	.csv, .tsv
Текст	.txt, .json
Временные ряды	.csv, .tsv, .json
Задачи с несколькими типами данных	.csv, .tsv, .json
Видео	.mp4, .webm

3.8.2.1 Как импортировать данные

Самый безопасный и надежный способ импортировать данные – хранить их вне RS-LabelMark и импортировать ссылки на них с помощью URL-адресов. Вы можете импортировать список URL-адресов в файле TXT, CSV или TSV или использовать ссылки на URL-адреса в формате задачи JSON.

Если вы импортируете аудио-, графические или видеоданные, для ссылки на эти типы данных необходимо использовать URL-адреса.

Если вы импортируете HTML-, текстовые, диалоговые данные или данные временных рядов с помощью тегов **<HyperText>**, **<Text>**, **<Paragraphs>** или **<TimeSeries>** в конфигурации разметки, вы можете загрузить данные напрямую или с URL-адреса.

- Чтобы загрузить данные с URL-адреса, укажите **valueType="url"** в конфигурации разметки.
- Чтобы загрузить данные непосредственно в базу данных RS-LabelMark, укажите **valueType="text"** для данных **HyperText** или **Text**, **valueType="json"** для данных **Paragraph** или **TimeSeries**.

ПРИМЕЧАНИЕ: если вы загружаете данные с URL-адреса, они не сохраняются в RS-LabelMark. Если вы хотите, чтобы при экспорте задачи с аннотациями были включены аннотированные данные, вам нужно импортировать их в базу данных RS-LabelMark без использования URL-ссылок или объединить данные с аннотациями после экспорта.

Пример с valueType="text"

Конфигурация с метками:

```
<View>
  <Text name="text1" value="text" valueType="text">
</View>
```

Файл JSON для импорта:

```
{
  "text": "My awesome opossum"
}
```

CSV-файл для импорта:

```
csvtext
My awesome opossum
```

Пример с valueType="url"

Конфигурация маркировки:

```
<View>
  <Text name="text1" value="text" valueType="url">
</View>
```

Импорт файла JSON:

```
{
  "text": "http://example.com/text.txt"
}
```

Импортировать CSV-файл:

```
csvtext
http://example.com/text.txt
```

3.8.3 Как получить данные

Для получения данных, отображаемых в теге **Object**, необходимо выполнить несколько шагов. Получение данных также используется в динамических вариантах и метках. Используйте следующие параметры в теге **Object**:

value (обязательно)

Параметр **value** представляет собой источник данных. Это может быть обычный текст или этап сложной системы получения данных. Его можно обозначить с помощью следующих форм: **value** (обязательно)

Переменные

В большинстве случаев тег **Object** содержит значение с одной переменной (с префиксом \$).

Например, `<Audio value="$audio" ... />` ищет поле «audio» в импортированном объекте JSON:

```
{
  "data": {
    "audio": "https://host.name/myaudio.wav"
  }
}
```

Обычный текст

Параметр **value** может быть строкой. Это удобно для **Header** и **Text**.

Кроме того, в качестве значения можно использовать содержимое тега. Это удобно для описательных текстовых тегов и применяется для **Label** и **Choice**.

Например:

```
<Header>RS-LabelMark value audio:</Header>
<Header value="Label only fully visible cars" />
<Text name="instruction" value="Label only fully visible cars" />
<Label>cat</Label>

<Choice>other</Choice>
```

Другие случаи

1. Параметр **value** может представлять собой текст, содержащий переменные с префиксом \$.

Например:

```
<Header value="url: $image"/>
```

2. Параметр **value** также может ссылаться на вложенные данные в массивах и словарях (**\$texts[2]** и **\$audio.url**).

Например:

```
xml<Image name="image" value="$images[0]" />
```

valueType (необязательно)

Параметр **valueType** определяет, как обрабатывать данные, полученные на предыдущих этапах. Есть два варианта: «url» и необработанные данные. В настоящее время необработанные данные могут быть в формате «text» или «json». Формат «text» используется для тегов **HyperText** и **Text**, а «json» – для тега **TimeSeries**.

Например:

- При использовании параметра «url»: `<Text name="text1" value="$text" valueType="url"/>` отображается текст, загруженный по URL.
- При использовании параметра «text»: `<Text name="text" value="$text" valueType="text"/>` отображается URL без загрузки текста.

resolver (необязательно)

Используйте этот параметр для получения данных из многоколоночного CSV-файла на S3 или в другом облачном хранилище. RS-LabelMark может получать данные только во время выполнения, что обеспечивает их безопасность.

Если вы импортируете файл со списком задач, то каждая задача в этом списке будет являться ссылкой на другой файл в хранилище. В этом случае вы можете использовать параметр **resolver** для получения содержимого этих файлов из хранилища.

Пример использования

Есть список задач, в котором поле «удаленная» для каждой задачи является ссылкой на CSV-файл в хранилище. В каждом CSV-файле есть столбец «текст», в котором нужно расставить метки. В каждом CSV-файле есть столбец «текст», в котором нужно расставить метки. Например:

Задачи:

```
[
  { "remote": "s3://bucket/text1.csv" },
  { "remote": "s3://bucket/text2.csv" }
]
```

CSV-файл:

```
csvid;text
12;The most flexible data annotation tool. Quickly installable. Build custom
UIs or use pre-built labeling templates.
```

Решение

Чтобы получить файл, используйте следующие параметры:

1. **value="\$remote"**: URL-адрес CSV-файла на S3 находится в поле «remote» данных задачи. Если вы используете параметр **resolver**, **value** всегда будет восприниматься как URL-адрес, поэтому вам не нужно указывать **valueType**.

2. **resolver="csv|separator=;|column=text"**: загрузите этот файл во время выполнения, проанализируйте его как CSV и извлеките столбец «text» из первой строки.

3. Отобразите результат.

Синтаксис

Синтаксис параметра **resolver** состоит из списка опций, разделенных символом |.

Первая опция – это тип файла.

ПРИМЕЧАНИЕ: в настоящее время поддерживаются только файлы CSV.

Остальные параметры – это параметры указанного типа файла с необязательными значениями. Параметры для CSV-файлов:

- **headless**: в CSV-файле нет заголовков (этот параметр является логическим и не может иметь значения).
- **separator=;**: разделитель в CSV-файле, обычно определяется автоматически.
- **column=1**: в режиме **headless** используется индекс с отсчетом от нуля, в противном случае используется название столбца.

Например, `resolver="csv|headless|separator=;|column=1"`

3.8.4 Как отформатировать данные для импорта

RS-LabelMark по-разному обрабатывает файлы разных типов.

Если вы хотите одновременно импортировать несколько типов данных для разметки, например изображения с подписями или аудиозаписи с расшифровками, вам необходимо использовать базовый формат JSON для RS-LabelMark.

Вы также можете использовать CSV-файл или список задач в формате JSON, чтобы указать URL-адреса с данными, а не импортировать данные напрямую, если вам нужно импортировать тысячи файлов. В RS-LabelMark можно импортировать файлы, содержащие до 250 000 задач или занимающие до 50 МБ.

3.8.4.1 Базовый формат JSON

Лучший способ импортировать данные в RS-LabelMark – использовать список задач в формате JSON. Ключ **data** в файле JSON указывает на каждую задачу как на запись в словаре JSON. Если ключ **data** отсутствует, RS-LabelMark воспринимает весь файл JSON как одну задачу.

В **data** словаре JSON используйте пары «ключ-значение», соответствующие исходному ключу, ожидаемому тегу объекта в конфигурации меток, которую вы настроили для своего проекта.

В зависимости от типа тега объекта RS-LabelMark интерпретирует значения полей по-разному:

- **<Text value="\$key">**: **value** интерпретируется как обычный текст.
- **<HyperText value="\$key">**: **value** интерпретируется как HTML-разметка.
- **<HyperText value="\$key" encoding="base64">**: **value** интерпретируется как HTML-разметка в кодировке base64.
- **<Audio value="\$key">**: **value** интерпретируется как действительный URL-адрес аудиофайла с включенной на стороне сервера политикой CORS.
- **<Image value="\$key">**: **value** интерпретируется как действительный URL-адрес файла изображения.
- **<TimeSeries value="\$key">**: **value** интерпретируется как действительный URL-адрес файла CSV/TSV, если **valueType="url"**, в противном случае интерпретируется как словарь JSON с массивами столбцов: **"value": {"first_column": [...], ...}** если **valueType="json"**.

В файл JSON можно добавить другие необязательные ключи.

Ключ JSON	Описание
аннотации	Необязательно. Список аннотаций, экспортированных из RS-LabelMark. Формат аннотаций RS-LabelMark позволяет импортировать результаты аннотирования для использования в последующих задачах по разметке.
прогнозы	Необязательно. Список результатов прогнозирования модели, каждый из которых сохранен в формате прогнозирования RS-LabelMark. Импортируйте прогнозы для автоматической предварительной разметки и активного обучения.

3.8.4.2 Пример формата JSON

Для примера проекта по классификации текста можно настроить конфигурацию меток следующим образом:

```
<View>
  <Text name="message" value="$my_text"/>
  <Choices name="sentiment_class" toName="message">
    <Choice value="Positive"/>
    <Choice value="Neutral"/>
    <Choice value="Negative"/>
  </Choices>
</View>
```

Затем вы можете импортировать текстовые задачи для маркировки, соответствующие следующему формату JSON:

```
{
  # "data" must contain the "my_text" field defined in the text labeling
  # config as the value and can optionally include other fields
  "data": {
    "my_text": "Opossums are great",
    "ref_id": 456,
    "meta_info": {
      "timestamp": "2020-03-09 18:15:28.212882",
      "location": "North Pole"
    }
  },

  # annotations are not required and are the list of annotation results
  # matching the labeling config schema
  "annotations": [{
    "result": [{
      "from_name": "sentiment_class",
      "to_name": "message",
      "type": "choices",
      "readonly": false,
      "hidden": false,
      "value": {
        "choices": ["Positive"]
      }
    }]
  }],

  # "predictions" are pretty similar to "annotations"
```

```
# except that they also include some ML-related fields like a prediction
"score"
"predictions": [{
  "result": [{
    "from_name": "sentiment_class",
    "to_name": "message",
    "type": "choices",
    "readonly": false,
    "hidden": false,
    "value": {
      "choices": ["Neutral"]
    }
  }],
# score is used for active learning sampling mode
"score": 0.95
}]
}]
```

Пример JSON с несколькими задачами

Вы можете разместить несколько задач в одном файле JSON, если загружаете его только с помощью диалогового окна импорта RS-LabelMark (Менеджер данных => кнопка **Импорт**) или при импорте из облачного хранилища. При использовании облачного хранилища необходимо убедиться, что все задачи в файле отформатированы одинаково.

Если вы используете облачное хранилище Source, вы также можете разместить несколько задач в файле JSON с разделителями в виде символов новой строки (JSONL/NDJSON).

Пример для размещения нескольких задач в одном JSON-файле

Параметр data должен содержать поле my_text, указанное в конфигурации разметки текста, и при необходимости может включать другие поля.

Параметр id не является обязательным.

```
{
  "id":1,
  "data":{
    "my_text":"Opossums like to be aloft in trees."
  }
},
{
  "id":2,
  "data":{
    "my_text":"Opossums are opportunistic."
  }
},
{
  "id":3,
  "data":{
    "my_text":"Opossums like to forage for food."
  }
}
```

Вы также можете использовать только содержимое поля «data» без вложенности, если у вас нет аннотаций или прогнозов.

```
{
  "my_text":"Opossums like to be aloft in trees."
},
{
```

```
"my_text": "Opossums are opportunistic."
},
{
  "my_text": "Opossums like to forage for food."
}
```

3.8.4.3 Импорт данных в формате CSV или TSV

При импорте текстового файла в формате CSV / TSV RS-LabelMark интерпретирует названия столбцов как ключи данных задачи, соответствующие настроенной вами конфигурации разметки:

```
my_text,optional_field
this is a first task,123
this is a second task,456
note
```

Если в вашей конфигурации разметки есть тег **TimeSeries**, RS-LabelMark при импорте интерпретирует CSV/TSV как данные временных рядов. Этот CSV/TSV размещается в качестве файла ресурсов, и RS-LabelMark автоматически создает задачу со ссылкой на загруженный CSV/TSV.

Обычный текст

Импортируйте данные в виде обычного текста. RS-LabelMark интерпретирует каждую строку в текстовом файле как отдельную задачу по разметке данных.

Вы можете использовать обычный текст для задач по разметке, если у вас есть только один поток входных данных и только один тег объекта, указанный в конфигурации разметки.

```
this is a first task
this is a second task
```

Если вы хотите импортировать целые текстовые файлы без разделения на отдельные задачи по разметке, настройте конфигурацию разметки так, чтобы в теге **Text** указывалось **valueType="url"**.

3.8.4.4 Импорт данных в формате HTML

Вы можете импортировать **HyperText** данные в формате HTML и аннотировать их в RS-LabelMark. При прямом импорте HTML-файлов содержимое сжимается за счет удаления пробелов и других нефункциональных данных в HTML-коде. Создаваемые вами аннотации применяются к минимизированной версии HTML.

Если вы хотите пометить HTML-файлы без минимизации данных, вы можете сделать следующее:

- Импортировать HTML-файлы в виде BLOB-объектов из внешнего облачного хранилища.
- Обновите тег **HyperText** в конфигурации маркировки, указав **valueType="url"**.

3.8.5 Импорт данных из локального каталога

Для того чтобы импортировать данные из локального каталога, есть два варианта:

- Запустите веб-сервер, чтобы сгенерировать URL-адреса для файлов, а затем загрузите файл со ссылками на эти URL-адреса в RS-LabelMark.
- Добавьте каталог с файлами в качестве исходного или целевого локального хранилища подключения в пользовательском интерфейсе RS-LabelMark.

Запустите веб-сервер для создания URL-адресов для локальных файлов

Чтобы запустить веб-сервер для генерации URL-адресов файлов, вы можете воспользоваться этим вспомогательным скриптом в репозитории RS-LabelMark или написать свой собственный скрипт. С помощью этого скрипта выполните следующие действия:

1. На компьютере с каталогом файлов, который вы хотите импортировать в RS-LabelMark, запустите вспомогательный скрипт и укажите шаблон регулярного выражения для поиска файлов, которые вы хотите импортировать. В этом примере скрипт находит файлы с расширением JPG:

```
./script/serve_local_files.sh <directory/with/files> *.jpg
```

Скрипт собирает ссылки на файлы, предоставляемые этим HTTP-сервером, и сохраняет их в **files.txt** файле, по одной ссылке на строку.

2. Импортируйте файл со ссылками в RS-LabelMark через пользовательский интерфейс.

ПРИМЕЧАНИЕ: во время разметки данных веб-сервер должен быть включен, чтобы URL-адреса оставались доступными для RS-LabelMark.

Если ваша конфигурация разметки поддерживает HyperText или несколько типов данных, используйте формат JSON RS-LabelMark для указания локальных расположений файлов вместо **txt** файла.

Если вы предоставляете доступ к данным с HTTP-сервера, созданного следующим образом: **python -m http.server 8081 -d**, вам может потребоваться настроить CORS для этого сервера, чтобы RS-LabelMark могла успешно получать доступ к файлам с данными. При необходимости выполните в командной строке следующую команду:

```
npm install http-server -g
http-server -p 3000 --cors
```

Добавьте каталог файлов в качестве исходного хранилища в пользовательском интерфейсе RS-LabelMark

Если вы используете RS-LabelMark в Docker и хотите добавить локальное хранилище файлов, вам нужно подключить каталог файлов и настроить переменные среды.

3.8.6 Импортируйте данные из пользовательского интерфейса

Обратите внимание:

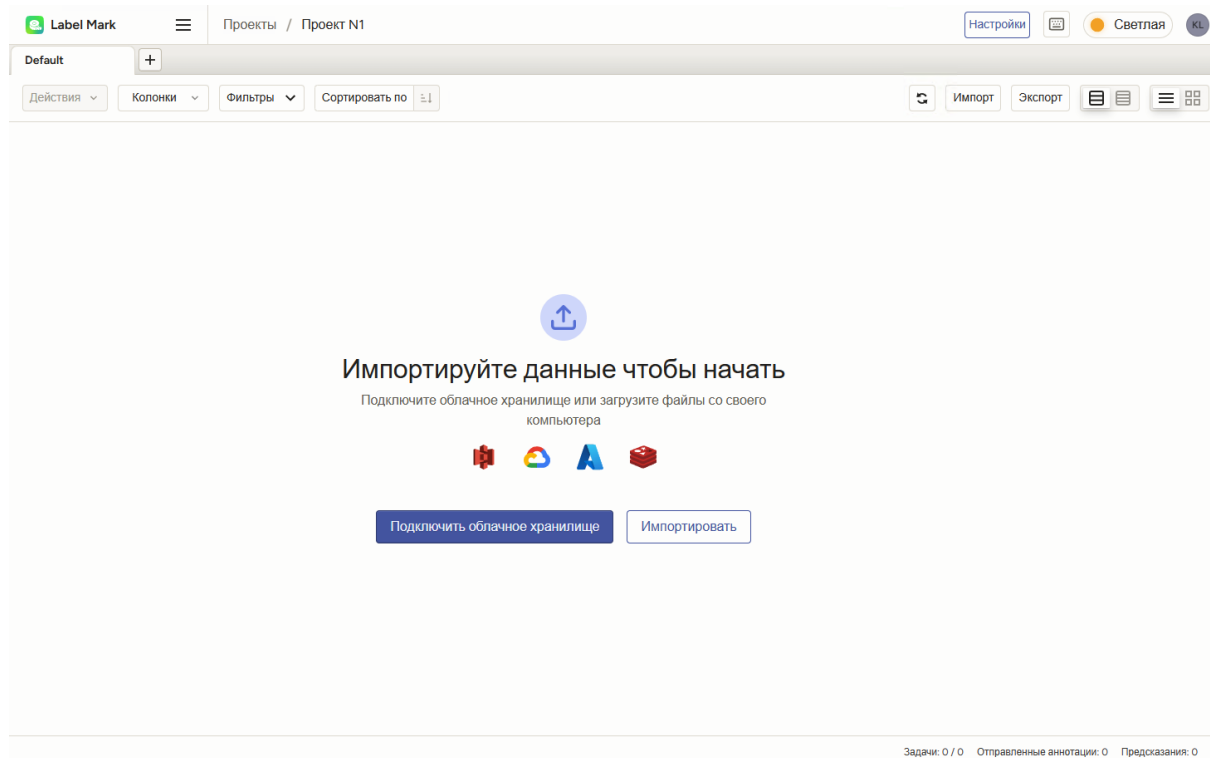
Для масштабных проектов или проектов, критически важных для бизнеса, не загружайте медиафайлы через интерфейс RS-LabelMark. Это особенно актуально для таких файлов, как изображения, аудио, видео, временные ряды и т. д.

RS-LabelMark не предназначена для масштабного хостинга и не обеспечивает резервное копирование импортируемых медиаресурсов.

Риски при загрузке через пользовательский интерфейс:

- Импорт задач с прогнозами
- Экспорт данных
- Перенос данных в другой экземпляр RS-LabelMark
- Повторное развертывание RS-LabelMark

Рекомендуется настроить исходное хранилище.



Импорт данных

Для того чтобы импортировать данные из пользовательского интерфейса RS-LabelMark, выполните следующие действия:

1. В пользовательском интерфейсе RS-LabelMark откройте страницу «Менеджер данных» для конкретного проекта.
2. Нажмите **Импортировать**, чтобы открыть диалоговое окно импорта.
3. Импортируйте данные из файлов или по URL-адресам.

Импортируемые данные зависят от проекта.

3.8.7 Импорт предварительно аннотированных данных

Если у вас есть прогнозы, сгенерированные для вашего набора данных на основе модели либо в виде предварительно аннотированных задач, либо в виде предварительно помеченных задач, вы можете импортировать прогнозы вместе с вашим набором данных в RS-LabelMark для просмотра и исправления. RS-LabelMark автоматически отображает предварительно аннотированные задачи, которые вы импортируете на страницу разметки для каждой задачи.

Вы можете импортировать предварительно аннотированные задачи в RS-LabelMark с помощью пользовательского интерфейса или с помощью API.

3.8.7.1 Подготовьте предварительные аннотации

Чтобы импортировать предсказанные метки в RS-LabelMark, необходимо использовать базовый формат JSON для RS-LabelMark и настроить задачи с помощью ключа **predictions** JSON. Серверная часть RS-LabelMark ML также выводит задачи в этом формате.

3.8.7.1.1 Формат JSON для предварительных аннотаций

Формат JSON для предварительных аннотаций в RS-LabelMark должен содержать два раздела:

- Объект **data**, указывающий на источник данных, к которым применяются предварительные аннотации. Это может быть URL-адрес аудиофайла, предварительно подписанная ссылка на изображение в облачном хранилище, обычный текст, ссылка на CSV-файл, хранящийся в RS-LabelMark.

- Массив **predictions** с результатами предварительной разметки для различных типов разметки.

Формат JSON для предварительных аннотаций должен соответствовать конфигурации разметки, используемой в вашем проекте по разметке данных.

Укажите объект данных

Используйте объект **data** для ссылки на **value** данных, указанных с помощью тега **Object** в конфигурации маркировки. Например, ниже приведен фрагмент конфигурации маркировки временных рядов:

```
...  
<TimeSeries name="ts" value="$csv" valueType="url">  
  <Channel column="first_column"/>  
</TimeSeries>  
...
```

В этом фрагменте указано **value="\$csv"** в теге объекта **TimeSeries**. Таким образом, в объекте данных для предварительных аннотаций в JSON-файле для этой конфигурации разметки необходимо указать «csv» в качестве расположения данных CSV для предварительных аннотаций временных рядов, как в следующем примере:

```
{  
  "data": {  
    "csv": "https://app.heartex.ai/samples/time-series.csv?time=None&values=first_column"  
  },  
  "predictions": [ ]  
}
```

Добавьте результаты в массив прогнозов

Массив **predictions** также зависит от конфигурации разметки. Некоторые поля предварительной разметки актуальны только для определенных типов разметки. В следующей таблице описаны объекты и массивы JSON, существующие для всех предварительных аннотаций:

Ключ JSON	Тип	Описание
predictions	массив	Содержит все предварительные аннотации для конкретной задачи.
predictions.model_version	строка	Указывает версию модели, на основе которой был сделан прогноз.
predictions.result	array	Содержит все предварительно размеченные результаты для конкретной задачи в объекте JSON.
result.value	object	Содержит сведения о конкретной размеченной области.
result.id	string	Произвольная строка, используемая для идентификации размеченной области. Должна

Ключ JSON	Тип	Описание
		совпадать с соответствующими областями, например в предварительных аннотациях для оптического распознавания символов.
result.from_name	string	Строка, используемая для ссылки на конфигурацию разметки from_name для выполняемого типа разметки. Должна совпадать с конфигурацией разметки.
result.to_name	string	Строка, используемая для ссылки на конфигурацию разметки to_name для выполняемого типа разметки. Должна совпадать с конфигурацией разметки.
result.type	string	Укажите тег разметки для выполняемого типа разметки. Например, задача распознавания именованных сущностей имеет тип labels .
result.readonly	bool	Режим только для чтения для определенной области.
result.hidden	bool	Видимость по умолчанию (значок глаза) для определенной области.

3.8.7.1.2 Прогнозы доступны только для чтения

Прогнозы нельзя изменить, они всегда доступны только для чтения.

Если в настройках проекта (раздел «Машинное обучение») включен переключатель «Показывать прогнозы аннотаторам в потоке меток и в режиме быстрого просмотра», прогнозы будут автоматически копироваться во вновь создаваемые аннотации.

В режиме быстрого просмотра аннотации можно создавать на основе прогнозов с помощью специальной кнопки копирования на панели инструментов редактора: 

3.8.7.2 Примеры предварительных аннотаций

В данном разделе приведены примеры форматов предварительных аннотаций.

Для того чтобы отформатировать предварительные аннотации для RS-LabelMark, не представленные в этих примерах, обратитесь к образцу результатов в формате JSON для соответствующих тегов объектов и элементов управления в вашей конфигурации разметки, например к тегу Audio для задач классификации аудио.

Каждый тег должен быть представлен в формате предварительных аннотаций JSON, чтобы прогнозы отображались в пользовательском интерфейсе RS-LabelMark. Не для всех тегов объектов и элементов управления есть образцы результатов в формате JSON.

3.8.7.2.1 Импорт предварительных аннотаций рамок и вариантов для изображений

Например, импортируйте предсказанные **области ограничивающих рамок (прямоугольников)** и **варианты** для задач по определению того, является ли объект на изображении самолетом или автомобилем.

Для предварительной разметки изображений в RS-LabelMark значения x, y, ширины и высоты аннотаций должны быть указаны в процентах от общего размера изображения.

Используйте следующую конфигурацию разметки:

```
<View>
  <Choices name="choice" toName="image" showInLine="true">
    <Choice value="Boeing" background="blue"/>
    <Choice value="Airbus" background="green" />
  </Choices>

  <RectangleLabels name="label" toName="image">
    <RS-LabelMark value="Airplane" background="green"/>
    <RS-LabelMark value="Car" background="blue"/>
  </RectangleLabels>

  <Image name="image" value="$image"/>
</View>
```

После настройки примера проекта создайте примеры задач в следующем формате:

```
[{
  "data": {
    "image": "/static/samples/sample.jpg"
  },
  "predictions": [{
    "model_version": "one",
    "score": 0.5,
    "result": [
      {
        "id": "result1",
        "type": "rectanglelabels",
        "from_name": "label", "to_name": "image",
        "original_width": 600, "original_height": 403,
        "image_rotation": 0,
        "value": {
          "rotation": 0,
          "x": 4.98, "y": 12.82,
          "width": 32.52, "height": 44.91,
          "rectanglelabels": ["Airplane"]
        }
      },
      {
        "id": "result2",
        "type": "rectanglelabels",
        "from_name": "label", "to_name": "image",
        "original_width": 600, "original_height": 403,
        "image_rotation": 0,
        "value": {
          "rotation": 0,
          "x": 75.47, "y": 82.33,
          "width": 5.74, "height": 7.40,
          "rectanglelabels": ["Car"]
        }
      },
      {
        "id": "result3",
        "type": "choices",
        "from_name": "choice", "to_name": "image",
```

```

    "value": {
      "choices": ["Airbus"]
    }
  }
}]
}]
}]

```

В этом примере 3 результата объединены в 1 прогноз, или предварительную аннотацию:

- **result1** – первая ограничивающая рамка.
- **result2** – вторая ограничивающая рамка.
- **result3** – выборка. Оценка прогноза применяется ко всему прогнозу.

3.8.7.2.2 Импорт предварительно размеченных многоугольников, эллипсов и ключевых точек без меток для изображений

Если вы хотите импортировать изображения с предварительно размеченными областями без присвоенных им меток, используйте следующую конфигурацию маркировки:

```
<View>
  <Image name="image" value="$image" zoom="true" zoomControl="true"
rotateControl="false"/>

  <Rectangle name="rect" toName="image" showInline="false"/>
  <Polygon name="polygon" toName="image"/>
  <Ellipse name="ellipse" toName="image"/>
  <KeyPoint name="kp" toName="image"/>

  <Labels name="labels" toName="image" fillOpacity="0.5" strokeWidth="5">
    <RS-LabelMark value="Vehicle" background="green"/>
    <RS-LabelMark value="Building" background="blue"/>
    <RS-LabelMark value="Pavement" background="red"/>
  </Labels>
</View>
```

После настройки примера проекта создайте примеры задач в следующем формате:

```
{
  "data": {
    "image": "/static/samples/sample.jpg"
  },
  "predictions": [
    {
      "model_version": "one",
      "score": 0.5,
      "result": [
        {
          "original_width": 800,
          "original_height": 450,
          "image_rotation": 0,
          "value": {
            "x": 55.46,
            "y": 2.36,
            "width": 35.86,
            "height": 46.9,
            "rotation": 0
          }
        }
      ]
    }
  ]
}
```

```

        "id": "ABC",
        "from_name": "rect",
        "to_name": "image",
        "type": "rectangle"
    },
    {
        "original_width": 800,
        "original_height": 450,
        "image_rotation": 0,
        "value": {
            "points": [
                [
                    20.93,
                    28.90
                ],
                [
                    25.86,
                    64.69
                ],
                [
                    38.40,
                    62.79
                ],
                [
                    34.13,
                    27.48
                ]
            ]
        },
        "id": "GHI",
        "from_name": "polygon",
        "to_name": "image",
        "type": "polygon"
    },
    {
        "original_width": 800,
        "original_height": 450,
        "image_rotation": 0,
        "value": {
            "x": 8.4,
            "y": 20.14,
            "radiusX": 4,
            "radiusY": 7.10,
            "rotation": 0
        },
        "id": "JKL",
        "from_name": "ellipse",
        "to_name": "image",
        "type": "ellipse"
    },
    {
        "original_width": 800,
        "original_height": 450,
        "image_rotation": 0,
        "value": {
            "x": 38.40,
            "y": 34.21,

```

```

        "width": 1.0
      },
      "id": "DEF",
      "from_name": "rect",
      "to_name": "image",
      "type": "keypoint"
    }
  ]
}

```

В этом примере в одном поле результатов для прогнозирования или предварительной разметки есть 3 области:

- область **ABC** – прямоугольная ограничивающая рамка
- область **DEF** – вторая прямоугольная ограничивающая рамка
- область **GHI** – многоугольная сегментация
- область **JKL** – эллипс.

Ни к одной из областей не применены метки.

Для поддержки этого типа прогнозирования в конфигурации разметки должен использоваться тег **Rectangle** вместо тега **RectangleLabels**. Несмотря на то, что в конфигурации разметки для этого примера используется тег **Labels**, в прогнозах не нужно указывать метки для областей.

3.8.7.2.3 Импорт предварительных аннотаций для текста

В этом примере импортируются предварительные аннотации для текста с использованием шаблона распознавания именованных сущностей (NER):

```

<View>
  <Labels name="label" toName="text">
    <RS-LabelMark value="Person"></Label>
    <RS-LabelMark value="Organization"></Label>
    <RS-LabelMark value="Fact"></Label>
    <RS-LabelMark value="Money"></Label>
    <RS-LabelMark value="Date"></Label>
    <RS-LabelMark value="Time"></Label>
    <RS-LabelMark value="Ordinal"></Label>
    <RS-LabelMark value="Percent"></Label>
    <RS-LabelMark value="Product"></Label>
    <RS-LabelMark value="Language"></Label>
    <RS-LabelMark value="Location"></Label>
  </Labels>
  <Text name="text" value="$text"></Text>
</View>

```

Этот пример JSON-файла содержит две задачи, каждая из которых включает два набора предварительных аннотаций от разных моделей. Первая задача также содержит оценки прогнозирования для каждого фрагмента NER:

```

{
  "data": {
    "text": "All that changed when he was 27 and he came to Jerusalem. It was the weekend of both Easter and Passover, and the city was flooded with tourists."
  },
  "predictions": [
    {
      "model_version": "one",
      "score": 0.5,
      "result": [

```

```
{
  "id": "abc",
  "from_name": "label",
  "to_name": "text",
  "type": "labels",
  "value": {
    "start": 29,
    "end": 31,
    "score": 0.70,
    "text": "27",
    "labels": [
      "Date"
    ]
  }
},
{
  "id": "def",
  "from_name": "label",
  "to_name": "text",
  "type": "labels",
  "value": {
    "start": 47,
    "end": 56,
    "score": 0.65,
    "text": "Jerusalem",
    "labels": [
      "Location"
    ]
  }
},
{
  "id": "ghi",
  "from_name": "label",
  "to_name": "text",
  "type": "labels",
  "value": {
    "start": 65,
    "end": 76,
    "score": 0.95,
    "text": "the weekend",
    "labels": [
      "Date"
    ]
  }
},
{
  "id": "jkl",
  "from_name": "label",
  "to_name": "text",
  "type": "labels",
  "value": {
    "start": 85,
    "end": 91,
    "score": 0.50,
    "text": "Easter",
    "labels": [
      "Date"
    ]
  }
}
```

```

        ]
      }
    }
  ]
},
{
  "model_version": "two",
  "score": 0.42,
  "result": [
    {
      "id": "mno",
      "from_name": "label",
      "to_name": "text",
      "type": "labels",
      "value": {
        "start": 29,
        "end": 31,
        "score": 0.55,
        "text": "27",
        "labels": [
          "Date"
        ]
      }
    },
    {
      "id": "pqr",
      "from_name": "label",
      "to_name": "text",
      "type": "labels",
      "value": {
        "start": 47,
        "end": 56,
        "score": 0.40,
        "text": "Jerusalem",
        "labels": [
          "Location"
        ]
      }
    },
    {
      "id": "stu",
      "from_name": "label",
      "to_name": "text",
      "type": "labels",
      "value": {
        "start": 65,
        "end": 76,
        "score": 0.32,
        "text": "the weekend",
        "labels": [
          "Time"
        ]
      }
    },
    {
      "id": "vwx",
      "from_name": "label",

```

```

        "to_name": "text",
        "type": "labels",
        "value": {
            "start": 85,
            "end": 91,
            "score": 0.22,
            "text": "Easter",
            "labels": [
                "Location"
            ]
        }
    },
    {
        "id": "yz1",
        "from_name": "label",
        "to_name": "text",
        "type": "labels",
        "value": {
            "start": 96,
            "end": 104,
            "score": 0.96,
            "text": "Passover",
            "labels": [
                "Date"
            ]
        }
    }
]
}
]
},
{
    "data": {
        "text": " Each journal was several inches thick and bound in leather.
On one page are drawn portraits of Sunny in a flowery, Easter dress and sun
hat. On another page are hundreds of sketches of leaves that Niyati saw in
her yard."
    },
    "predictions": [
        {
            "model_version": "one",
            "result": [
                {
                    "id": "ab2",
                    "from_name": "label",
                    "to_name": "text",
                    "type": "labels",
                    "value": {
                        "start": 18,
                        "end": 32,
                        "text": "several inches",
                        "labels": [
                            "Product"
                        ]
                    }
                }
            ]
        }
    ],
    {
        "id": "cd3",

```

```

        "from_name": "label",
        "to_name": "text",
        "type": "labels",
        "value": {
            "start": 64,
            "end": 67,
            "text": "one",
            "labels": [
                "Percent"
            ]
        }
    },
    {
        "id": "ef4",
        "from_name": "label",
        "to_name": "text",
        "type": "labels",
        "value": {
            "start": 96,
            "end": 101,
            "text": "Sunny",
            "labels": [
                "Person"
            ]
        }
    },
    {
        "id": "gh5",
        "from_name": "label",
        "to_name": "text",
        "type": "labels",
        "value": {
            "start": 162,
            "end": 170,
            "text": "hundreds",
            "labels": [
                "Percent"
            ]
        }
    },
    {
        "id": "ij6",
        "from_name": "label",
        "to_name": "text",
        "type": "labels",
        "value": {
            "start": 198,
            "end": 204,
            "text": "Niyati",
            "labels": [
                "Person"
            ]
        }
    }
]
},
{

```

```
"model_version": "two",
"result": [
  {
    "id": "kl7",
    "from_name": "label",
    "to_name": "text",
    "type": "labels",
    "value": {
      "start": 18,
      "end": 32,
      "text": "several inches",
      "labels": [
        "Fact"
      ]
    }
  },
  {
    "id": "mn8",
    "from_name": "label",
    "to_name": "text",
    "type": "labels",
    "value": {
      "start": 64,
      "end": 67,
      "text": "one",
      "labels": [
        "Percent"
      ]
    }
  },
  {
    "id": "op9",
    "from_name": "label",
    "to_name": "text",
    "type": "labels",
    "value": {
      "start": 96,
      "end": 101,
      "text": "Sunny",
      "labels": [
        "Time"
      ]
    }
  },
  {
    "id": "qr10",
    "from_name": "label",
    "to_name": "text",
    "type": "labels",
    "value": {
      "start": 116,
      "end": 122,
      "text": "Easter",
      "labels": [
        "Location"
      ]
    }
  }
]
```

```

    },
    {
      "id": "stl1",
      "from_name": "label",
      "to_name": "text",
      "type": "labels",
      "value": {
        "start": 162,
        "end": 170,
        "text": "hundreds",
        "labels": [
          "Money"
        ]
      }
    },
    {
      "id": "uv12",
      "from_name": "label",
      "to_name": "text",
      "type": "labels",
      "value": {
        "start": 198,
        "end": 204,
        "text": "Niyati",
        "labels": [
          "Person"
        ]
      }
    }
  ]
}
]
}
}

```

3.8.7.2.4 Импорт предварительных аннотаций для оптического распознавания символов

Импортируйте предварительные аннотации для оптического распознавания символов (OCR).

В этом примере импортируются предварительные аннотации для задач OCR с использованием шаблона OCR:

```

<View>
  <Image name="image" value="$ocr"/>
  <Labels name="label" toName="image">
    <RS-LabelMark value="Text" background="green"/>
    <RS-LabelMark value="Handwriting" background="blue"/>
  </Labels>
  <Rectangle name="bbox" toName="image" strokeWidth="3"/>
  <Polygon name="poly" toName="image" strokeWidth="3"/>
  <TextArea name="transcription" toName="image" editable="true"
perRegion="true" required="true" maxSubmissions="1" rows="5"
placeholder="Recognized Text" displayMode="region-list"/>
</View>

```

Этот пример JSON содержит одну задачу с тремя словарями результатов, по одному для каждого типа тегов в конфигурации разметки: «Прямоугольник», «Метки» и «Текстовая область»:

```
{
  "data": {
    "ocr": "/data/upload/receipt_00523.png"
  },
  "predictions": [
    {
      "model_version": "best_ocr_model_1_final",
      "result": [
        {
          "original_width": 864,
          "original_height": 1296,
          "image_rotation": 0,
          "value": {
            "x": 48.9333,
            "y": 61.3336,
            "width": 9.73333,
            "height": 2.8446,
            "rotation": 0
          },
          "id": "bb1",
          "from_name": "bbox",
          "to_name": "image",
          "type": "rectangle"
        },
        {
          "original_width": 864,
          "original_height": 1296,
          "image_rotation": 0,
          "value": {
            "x": 48.9333,
            "y": 61.3336,
            "width": 9.7333,
            "height": 2.8446,
            "rotation": 0,
            "labels": [
              "Text"
            ]
          },
          "id": "bb1",
          "from_name": "label",
          "to_name": "image",
          "type": "labels"
        },
        {
          "original_width": 864,
          "original_height": 1296,
          "image_rotation": 0,
          "value": {
            "x": 48.9333,
            "y": 61.3336,
            "width": 9.7333,
            "height": 2.8446,
            "rotation": 0,
            "text": [
              "TOTAL"
            ]
          },
          "id": "bb1",
          "from_name": "label",
          "to_name": "image",
          "type": "labels"
        }
      ]
    }
  ]
}
```

```

        "id": "bb1",
        "from_name": "transcription",
        "to_name": "image",
        "type": "textarea"
      }
    ],
    "score": 0.89
  }
]
}

```

В этом примере JSON также указана оценка прогнозирования для задачи. Идентификаторы для каждого результата в виде прямоугольника соответствуют метке, присвоенной области, и расшифровке текстовой области для этой области.

3.9 Разметка данных

В данном разделе описана разметка и аннотирование данных. После того как вы настроили проект, интерфейс разметки и импортировали данные, вы можете приступить к разметке и аннотированию.

1. Откройте проект в RS-LabelMark и при необходимости отфильтруйте или отсортируйте данные.
2. Нажмите **Разметить все задачи**, чтобы начать разметку.
3. Используйте сочетания клавиш или мышь, чтобы разметить данные и отправить свои аннотации.
4. Следуйте инструкциям по разметке и решайте, какие задачи можно пропустить.
5. Нажмите на название проекта, чтобы вернуться в менеджер данных.

3.9.1 Начало разметки

Некоторые задачи по разметке могут быть сложными, например разметка, включающая текстовые, графические и аудиообъекты в рамках одного набора данных и задачи по разметке, или создание связей между аннотациями в рамках задачи по разметке.

В этом разделе приведены рекомендации по выполнению более сложных задач по разметке, таких как разметка с использованием связей, перекрывающихся областей, выбранных задач или изменение меток.

3.9.1.1 Выбор задач для разметки

В проекте нажмите **Разметить все задачи**, чтобы начать разметку всех задач. Чтобы пометить задачи, отфильтрованные и отсортированные в менеджере данных, выберите **Пометить задачи как отображаемые**.

Вы также можете пометить конкретную задачу в **быстром просмотре** или **предварительном просмотре**, нажав на нее в менеджере данных проекта, но после отправки аннотаций вы не увидите автоматически следующую задачу в очереди на пометки.

Вы также можете установить флажки рядом с конкретными задачами, а затем нажать **«Пометить \$n задач»**, чтобы пометить выбранное количество задач. Например, установите флажки рядом с 5 разными задачами, а затем нажмите **«Пометить 5 задач»**, чтобы пометить только эти 5 задач.

ПРИМЕЧАНИЕ: при разметке задач не следует одновременно открывать поток разметки (то есть нажимать «Разметить все задачи») на двух вкладках. Это может

привести к тому, что вы получите одну и ту же задачу дважды, что может обойти настройки проекта, предотвращающие дублирование аннотаторов.

3.9.1.2 Установка метки для области данных

Аннотируйте часть данных, добавив область.

1. Выберите метку, которую хотите применить к области. В некоторых конфигурациях этот шаг можно пропустить.
2. Нажмите на текст, изображение, аудиозапись или другой объект данных, чтобы применить метку к области. Изменения сохраняются автоматически.
3. Нажмите **Отправить**, чтобы отправить выполненную аннотацию и перейти к следующему заданию.

3.9.1.3 Установка метки перекрывающихся областей

При разметке с помощью ограничивающих рамок и других задач сегментации изображений, а также при выделении текста для разметки в области обработки естественного языка и именованных сущностей вам может понадобиться помечать перекрывающиеся области. Чтобы упростить эту задачу, скрывайте помеченные области после их аннотирования.

1. Выберите метку, которую хотите применить к области.
2. Нарисуйте ограничивающую рамку или выделите текст, который хотите пометить.
3. На боковой панели **Области** или **Метки** найдите и выберите область, которую вы пометили, и нажмите на значок глаза, чтобы скрыть область.
4. Или нажмите `cmd` или `ctrl`, чтобы нарисовать поверх существующих **Областей** или **Меток**.

ПРИМЕЧАНИЕ: это не работает с ключевыми точками.

5. Выберите следующую метку, которую хотите применить к перекрывающейся области.
6. Нарисуйте ограничивающую рамку или выделите текст, который нужно подписать.
7. Продолжайте скрывать и подписывать области, пока не закончите аннотирование задачи. При желании нажмите на значок глаза рядом с **Областями**, чтобы скрыть, а затем снова показать все области, подписанные в задаче, чтобы проверить результат.
8. Нажмите **Отправить**, чтобы отправить завершенную аннотацию и перейти к следующей задаче.

3.9.1.4 Выбор нескольких регионов

При маркировке можно выбрать несколько регионов, чтобы вносить изменения сразу в них.

1. После создания нескольких регионов нажмите **ctrl** и щелкните каждый регион, который хотите выбрать. Вы можете выбирать регионы на объекте, который маркируете, или на боковой панели **Регионы**. Выберите несколько регионов на боковой панели **Регионы**, щелкнув первый регион в списке, который хотите выбрать, и удерживая **Shift** при щелчке последнего региона в списке, который хотите выбрать.
2. После выбора регионов вы можете применить метку ко всем выбранным регионам или удалить их.

3. Нажмите на выбранный регион или нажмите **u** для отмены выбора.

3.9.1.5 Дублирование областей

Вы можете продублировать область, чтобы создать множество многоугольников, прямоугольников или эллипсов одинакового размера.

1. Выберите область, которую нужно продублировать.
2. Нажмите **ctrl + d**, чтобы продублировать область, или комбинацию **ctrl + c** и **ctrl + v**, чтобы скопировать и вставить область. Дублированные области отображаются в том же месте, что и дублируемая область. Если вы используете клавиатуру Mac, вместо **cmd** нажимайте **ctrl**.
3. Нажмите на инструмент со стрелкой, затем переместите и измените размер только что созданной области.

3.9.1.6 Изменение метки

Вы можете изменить метку существующего региона.

1. Выберите помеченный регион, например фрагмент, ограничивающую рамку, сегмент изображения, аудиорегион или другой регион, на объекте или с помощью боковой панели **Регионы**.
2. Выберите новую метку. Изменения в метке сохраняются автоматически.
3. Нажмите **Отправить**, чтобы отправить выполненную аннотацию и перейти к следующему заданию.

3.9.1.7 Удаление выделенной области

При необходимости вы можете удалить выделенные области, например ограничивающие рамки или фрагменты текста.

1. Выберите отмеченную область на объекте или на боковой панели **Области**.
2. Нажмите клавишу Backspace или перейдите на панель **Результаты** и удалите выбранную аннотацию.

3.9.1.8 Удаление аннотации

После добавления метки к задаче вы можете удалить ее, нажав на значок корзины. Если вы еще не сохранили метку, нажмите на значок **X**, чтобы сбросить задачу и удалить действие по добавлению метки.

Вы также можете удалить все метки задачи на странице проекта.

Добавьте связи между метками

Вы можете создать связи между двумя результатами с указанием направления и меток. Чтобы добавить метки к маршрутам, необходимо настроить конфигурацию меток с помощью тега `relations`.

1. Выберите регион для аннотации, которую вы хотите связать с другой аннотацией. Если вы создаете связь на основе направления, сначала выберите первую аннотацию.
2. В разделе **Регионы** на боковой панели **Результаты** нажмите кнопку **Создать связь**, которая выглядит как значок гиперссылки.
3. Выберите второй регион для аннотации, чтобы завершить создание связи.

После того как вы свяжете две области с аннотациями, вы сможете изменить связь в разделе **Связи** на боковой панели **Результаты**.

- Чтобы изменить направление связи, нажмите кнопку со стрелкой между двумя связанными областями.
- Чтобы добавить метки к стрелке, указывающей направление связи между двумя аннотациями, нажмите кнопку с вертикальным многоточием рядом с двумя связанными областями, чтобы добавить предустановленные метки. Для этого у вас должна быть конфигурация меток, включающая связи.

3.9.1.9 Выход из потока маркировки

Если аннотатор не закончил работу над аннотацией, но хочет выйти из процесса разметки, например, чтобы приостановить расчет срока выполнения аннотаций, он может нажать на раскрывающееся меню рядом с кнопкой **Отправить**, а затем выбрать **Отправить и выйти** (или **Обновить и выйти**). Это позволит отправить текущую аннотацию и перейти на страницу «Проекты».

Если аннотатор не закончил аннотирование, то можно проверить, сохранен ли черновик на панели истории, а затем перейти на страницу «Проекты», чтобы остановить расчет срока выполнения.

3.9.2 Маркировка с участием соавторов

В RS-LabelMark вы можете маркировать задачи вместе с соавторами. Задачи блокируются, пока кто-то выполняет аннотирование, чтобы вы случайно не перезаписали аннотации другого соавтора. После того как другой соавтор закончит работу над задачей, она может появиться в вашей очереди на маркировку, если для задачи установлено минимальное количество аннотаций более одного. По умолчанию задачи могут быть аннотированы только одним соавтором.

Если вы хотите пометить задачи несколько раз, даже если минимальное количество аннотаций равно одному, сделайте следующее:

Чтобы пометить задачи несколько раз, даже если минимальное количество аннотаций равно одному, сделайте следующее:

1. В менеджере данных проекта нажмите на задачу, чтобы открыть окно быстрого добавления меток.
2. Нажмите на значок + рядом с идентификатором аннотации задачи, чтобы открыть вкладку аннотации.
3. Пометьте задачу.
4. Нажмите **Отправить**, чтобы сохранить аннотацию.
5. Нажмите на следующую задачу в менеджере данных, чтобы открыть окно быстрой маркировки для этой задачи, и повторите шаги 2–4.

3.9.3 Автоматическая разметка

Если у вас есть серверная часть для машинного обучения, настроенная на получение интерактивных предварительных аннотаций, вы можете выбрать, использовать ли эти прогнозы при разметке.

1. После начала разметки вы можете включить **автоматическую разметку**, чтобы использовать интеллектуальную функцию для присвоения метки, рисования фигуры, маски или определения ключевой точки. После использования интеллектуальной функции для рисования на изображении или разметки текста или HTML-элемента серверная часть для машинного обучения возвращает прогнозы.

2. При разметке изображений вы можете выбрать, **автоматически принимать предложения по разметке** после включения функции автоматической разметки. Если вы автоматически принимаете предложения по разметке, области появляются автоматически и сразу же создаются. Если вы не принимаете предложения автоматически, области появляются, но вы можете отклонить или утвердить их вручную, по отдельности или все сразу. Предсказанные текстовые области принимаются автоматически.

3.9.4 Настройте интерфейс для маркировки

Нажмите на значок настроек во время маркировки, чтобы настроить интерфейс в соответствии с вашими задачами.

Например, можно оставить выбранную метку после создания области, отображать метки на ограничивающих прямоугольниках, многоугольниках и других областях во время разметки, а также отображать номера строк при разметке текста.

Вы также можете изменить расположение элементов на экране, скрыть или отобразить прогнозы, аннотации или панель результатов, а также скрыть или отобразить различные элементы управления и кнопки.

3.9.5 Расширенная разметка изображений

Если вы хотите выполнить расширенную разметку изображений, воспользуйтесь примерами и рекомендациями ниже.

3.9.5.1 Добавление нескольких типов областей в аннотации к изображениям

В аннотации к изображениям можно добавить несколько типов областей. Вы можете добавить любую из следующих областей:

- Прямоугольники
- Эллипсы
- Ключевые точки
- Многоугольники
- Маски-кисти

Чтобы добавить различные типы областей к аннотациям на изображении, следуйте примеру ниже.

Создайте собственный шаблон для интерфейса разметки, используя следующий пример:

```
<View>
  <Image name="image" value="$image" />
  <Rectangle name="rect" toName="image" />
  <Ellipse name="ellipse" toName="image" />
  <KeyPoint name="kp" toName="image" />
  <Polygon name="polygon" toName="image" />
  <Brush name="brush" toName="image" />
  <Choices name="choices" toName="image">
    <Choice value="yes"></Choice>
    <Choice value="no"></Choice>
  </Choices>
  <Labels name="labels" toName="image" fillOpacity="0.5" strokeWidth="5">
    <RS-LabelMark value="building" background="green"></Label>
  </Labels>
</View>
```

```
<RS-LabelMark value="vehicle" background="blue"></Label>  
</Labels>  
</View>
```

В этом примере аннотатору доступны прямоугольники, эллипсы, многоугольники, ключевые точки и маски-кисти, а также варианты классификации изображений «да» и «нет» и метки областей «здание» и «транспортное средство».

3.9.5.2 Быстрая разметка изображений

Вы можете добавить прямоугольник или эллипс на изображение всего двумя щелчками мыши или дважды щелкнуть, чтобы создать многоугольник, прямоугольник или эллипс.

Если при создании многоугольника вы случайно выбрали точку на изображении, просто дважды щелкните, чтобы удалить ее и продолжить создание области. Чтобы удалить точку, в многоугольнике должно быть не менее трех точек.

3.9.5.3 Создание областей без меток

При аннотировании изображений можно создавать области без применения меток.

1. Создайте собственный шаблон, используя приведенный выше пример в разделе [Добавление нескольких типов областей в аннотации к изображениям](#).

2. Выберите инструмент, который хотите использовать, на панели инструментов для разметки. Это нужно сделать только для первой задачи.

3. Создайте область, дважды щелкнув по ней или щелкнув и перетаскив курсор, чтобы создать ограничивающую рамку, либо щелкните по точкам, необходимым для построения многоугольника.

4. Выберите созданную область на боковой панели или на изображении.

5. Выберите метку, которую хотите применить к области.

6. Повторите эти действия для всех областей, которые хотите создать.

Это может быть полезно при двухэтапной разметке, когда один разметчик создает области, а другой их размечает.

3.9.5.4 Удаление меток с помощью маски

Если вы допустили ошибку при разметке с помощью маски-кисти, ее можно удалить. Прежде чем удалять какую-либо часть маски-кисти, необходимо выбрать ее на боковой панели.

Если вы хотите полностью удалить область и начать заново, удалите ее, а не стирайте. При стирании область не удаляется.

3.9.6 Аннотация к многостраничному документу

Аннотирование многостраничных документов позволяет добавлять примечания к документам, состоящим из нескольких страниц. Представляя каждую страницу в виде изображения, вы можете импортировать их в одну задачу и аннотировать каждую страницу, сохраняя контекст всего документа.

Если вам нужно классифицировать PDF-документ целиком, вы также можете использовать шаблон классификации PDF.

3.9.6.1 Необходимые условия

Для аннотирования многостраничного документа необходимо предварительно обработать его, разделив на отдельные изображения.

Для повышения производительности предварительно загрузите изображения, скачивая их через XHR и сохраняя в локальном хранилище браузера. Для корректной работы этой функции необходимо настроить политику CORS на CDN/хранилище, где находятся изображения.

3.9.6.2 Ограничения

Из соображений производительности рекомендуется ограничить максимальное количество задач в проекте примерно 100 000.

Если вы создаете задачи с несколькими изображениями, возможно, вам стоит ограничиться меньшим количеством задач в каждом проекте. Например, задача со 100 изображениями примерно эквивалентна 100 задачам (с точки зрения производительности).

3.9.6.3 Навигация по странице

При создании многостраничных аннотаций вы увидите меню навигации:



Навигация

- Для того чтобы перейти на определенную страницу, нажмите на номера страниц в центре и введите нужную страницу.
- Для навигации также можно использовать следующие горячие клавиши: **command + left/command + right** или **ctrl + left/ctrl + right**.

3.9.6.4 Настройка разметки

Включите многостраничную разметку, используя тег **<Image>** и задав параметр **valueList** вместо параметра **value**.

```
<View>
  <RectangleLabels name="rectangles" toName="pdf" showInline="true">
    <RS-LabelMark value="Title" background="red"/>
    <RS-LabelMark value="Summary" background="blue"/>
    <RS-LabelMark value="Author" background="green"/>
    <RS-LabelMark value="Table" background="gold"/>
    <RS-LabelMark value="Citation" background="purple"/>
  </RectangleLabels>
  <Image valueList="$pages" name="pdf"/>
</View>
```

Подобно **value**, **valueList** параметр принимает переменную. Разница в этом случае заключается в том, что вы ссылаетесь на массив в вашем JSON.

Например, следующие ссылки **pages** в примере входных данных ниже.

```
<Image valueList="$pages" name="pdf"/>
```

Классификация

В примере используется маркировка в виде ограничительной рамки. Вы можете изменить шаблон для выполнения задач классификации, используя **Choices** тег. При использовании **Choices** тега обратите внимание на следующее:

- Вы можете выполнять классификацию для каждой страницы с помощью **perItem="true"**.
- Вы также можете оставить **perItem="false"** (значение по умолчанию) для выполнения классификации для каждого документа.
- **perRegion="true"** поддерживается при рисовании областей на страницах.

3.9.6.5 Формат входных данных

В примере показан процесс загрузки 5-страничного PDF-файла, предварительно преобразованного в 5 изображений. Вы можете ссылаться на страницы в своей конфигурации разметки с помощью **valueList="\$pages"**.

```
{
  "data": {
    "pages": [
      "https://htx-
pub.s3.amazonaws.com/demo/images/demo_stock_purchase_agreement/0001.jpg",
      "https://htx-
pub.s3.amazonaws.com/demo/images/demo_stock_purchase_agreement/0002.jpg",
      "https://htx-
pub.s3.amazonaws.com/demo/images/demo_stock_purchase_agreement/0003.jpg"
    ]
  }
}
```

3.9.6.6 Формат экспорта

При экспорте аннотаций свойство **item_index** возвращает индекс изображения, к которому прикреплена область. Индексы начинаются с 0, то есть первое изображение будет иметь индекс **0**, второе – **1** и так далее.

```
{
  "id": 859,
  "annotations": [
    {
      "id": 25569355,
      "result": [
        {
          "id": "tdS8Of63VJ",
          "type": "rectanglelabels",
          "value": {
            "x": 2.48868778280543,
            "y": 4.807692307692308,
            "width": 52.88461538461539,
            "height": 5.128205128205128,
            "rotation": 0,
            "rectanglelabels": [
              "Title"
            ]
          },
          "origin": "manual",
          "to_name": "pdf",
          "from_name": "choices",
          "item_index": 0,

```

```

        "image_rotation": 0,
        "original_width": 2550,
        "original_height": 3300
    },
    {
        "id": "RGGrHm95R7",
        "type": "rectanglelabels",
        "value": {
            "x": 30.693815987933636,
            "y": 12.179487179487179,
            "width": 58.4841628959276,
            "height": 1.2820512820512828,
            "rotation": 0,
            "rectanglelabels": [
                "Citation"
            ]
        },
        "origin": "manual",
        "to_name": "pdf",
        "from_name": "choices",
        "item_index": 1,
        "image_rotation": 0,
        "original_width": 2550,
        "original_height": 3300
    }
]
}
]
}

```

3.9.7 Выделение

Outliner – это замена существующего интерфейса с несколькими панелями. Эта функция улучшает пользовательский опыт, позволяя размещать больше данных в этих двух столбцах (**Outliner** и **Детали**), и предоставляет больше возможностей для работы с регионами. Например, сегментация изображений – это единственный тип разметки, который поддерживает пользовательские элементы управления на панели сведений.

3.9.7.1 Функциональные возможности

Окно аннотирования – это область, в которой отображаются все сведения об аннотациях, регионах и истории разметки. Область разделена на две панели: **Окно аннотирования** и **Подробности**.

3.9.7.1.1 Действия на панели

Есть две панели, которые служат навигатором для списка регионов. Эти панели можно сворачивать, перемещать, менять местами, перетаскивать, сбрасывать и разворачивать, чтобы получить полноэкранный режим. Например, если вы перейдете к потоку меток, в этом режиме у вас будет больше места.

Панели **Навигатор** и **Подробности** можно сворачивать и отсоединять, поэтому вы можете расположить их так, как вам удобно.

3.9.7.1.2 Редактор регионов

Вы можете редактировать области, уделяя особое внимание разметке. Вы можете увеличивать масштаб и точно настраивать количество и размеры каждого атрибута

области. Такие параметры области изображения, как высота, поворот и т. д., можно изменить с помощью элементов управления на панели «Сведения». Вы можете свернуть все панели и получить максимальную рабочую область.

Чтобы добавить область в шаблоны транскрипции оптического распознавания символов, нарисуйте прямоугольник. На панели **Outliner** появится текстовое поле для ввода текста, распознанного с помощью оптического распознавания символов. Функция позволяет выбрать область, после чего изменится панель **Детали**. Вы можете сделать следующее:

- Посмотреть текст, прикрепленный к делу, в котором использовалось оптическое распознавание символов.
- Управлять координатами прямоугольника. Например, можно поворачивать, перемещать и изменять размер координат.
- Заблокировать область, чтобы она не перемещалась.
- Скрытие/блокировка областей.
- Используйте инструмент перемещения, чтобы предотвратить перемещение областей.
- Скройте или отобразите выбранную область.

3.9.7.1.3 Группировка или сортировка регионов

Вы можете сгруппировать или отсортировать регионы с помощью следующей функции из выпадающего списка.

- **Группировать вручную:** ручная группировка областей (перетаскивание). При выборе этого параметра вы можете перетаскивать области друг в друга, создавая древовидную структуру, и группировать области.
- **Группировать по инструменту:** при выборе этого параметра добавляются инструменты (прямоугольники, многоугольники). Вы можете использовать шаблон со всеми этими инструментами, чтобы группировать области по инструменту, с помощью которого они были созданы.
- **Группировать по метке:** когда вы выберете эту опцию и нарисуете область, теперь вы увидите текстовую метку, которую можно развернуть. Текстовая метка отображает только текст, относящийся к этой функции.

*ПРИМЕЧАНИЕ: опции **Группировать по метке** и **Группировать по инструменту** не работают для сценария с несколькими изображениями или сценария с несколькими помеченными типами. Для этого режима доступна только опция **Группировать вручную**.*

Вы также можете упорядочить свои регионы по вариантам "Порядок по баллам" или "Порядок по времени".

Индикация прогнозирования для региона

Эта функция показывает, была ли область создана с помощью модели машинного обучения.

Список связей

Один регион относится к другому. Обычно это делается для обработки естественного языка. Например, название города – это регион 1, а название района – регион 2. Теперь в иерархии списка связей вы поместите регион 2 под регион 1. Списки связей могут быть однонаправленными или двунаправленными. Один регион можно включить в состав другого.

ПРИМЕЧАНИЕ: список связей не поддерживает многоуровневое перечисление регионов. Не рисуйте ограничивающие рамки поверх другой аннотации. При создании аннотаций в виде ограничивающих рамок для изображений нельзя начинать рисовать одну рамку внутри другой, так как это приведет к перетаскиванию первой рамки. Вместо

этого при выборе инструмента «Прямоугольник» будет создана новая ограничивающая рамка перед изображением.

Чтобы создать ограничивающие рамки сверху, можно воспользоваться двумя способами:

1. Нажмите на инструмент «Прямоугольник» (R).
2. Пометьте ограничивающую рамку, а затем внутри нее – еще одну.

Теперь вы можете перетаскивать объекты, даже если инструмент «Прямоугольник» не выбран. Помните, что при выборе инструмента «Прямоугольник» вы можете создавать прямоугольники, даже если они находятся поверх другого прямоугольника.

ИЛИ

1. Нажмите кнопку CTRL и начните рисовать ограничивающую рамку поверх другой.

3.9.8 Сохранение результатов в аннотациях

Каждая аннотация, которую вы создаете при разметке задачи, содержит области и результаты.

- **Регионы** – это выделенная область данных, будь то фрагмент текста, область изображения, аудиосегмент или другой объект.
- **Результаты** – это метки, присвоенные региону.

У каждого региона есть уникальный идентификатор для каждой аннотации, представляющий собой строку из символов **A-Za-z0-9_-**. Идентификатор каждого результата совпадает с идентификатором региона, к которому он относится.

При использовании прогноза для создания аннотации идентификаторы результатов в поле аннотации остаются прежними. Это позволяет отслеживать регионы, созданные с помощью модели машинного обучения, и напрямую сравнивать их с аннотациями, созданными и проверенными людьми.

3.10 Экспорт аннотаций и данных

В любой момент работы над проектом по разметке вы можете экспортировать аннотации из RS-LabelMark.

RS-LabelMark хранит ваши аннотации в необработанном формате JSON в базе данных SQLite, базе данных PostgreSQL или в любом другом облачном хранилище или базе данных, которое вы укажете в качестве целевого хранилища. В облачных хранилищах для каждой помеченной задачи создается один файл с именем **task_id.json**.

В аннотациях к изображениям, экспортируемых в формате JSON, для описания размера и расположения ограничивающих рамок используются проценты от общего размера изображения, а не пиксели.

3.10.1 Сохранение результатов в аннотациях

Каждая аннотация, которую вы создаете при разметке задачи, содержит области и результаты.

- **Области** – это выделенные участки данных, будь то фрагмент текста, область изображения, аудиосегмент или другой объект.
- **Результаты** – это метки, присвоенные области.

У каждого региона есть уникальный идентификатор для каждой аннотации, представляющий собой строку из символов **A-Za-z0-9_-**. Идентификатор каждого результата совпадает с идентификатором региона, к которому он относится.

Когда прогноз используется для создания аннотации, идентификаторы результатов остаются прежними в поле аннотации. Это позволяет отслеживать области, сгенерированные с помощью модели машинного обучения, и напрямую сравнивать их с аннотациями, созданными и проверенными вручную.

3.10.1.1 Экспорт с помощью пользовательского интерфейса

Чтобы экспортировать данные и аннотации из пользовательского интерфейса RS-LabelMark, выполните следующие действия.

1. В проекте нажмите **Экспорт**.
2. Выберите доступный формат экспорта.
3. Нажмите **Экспорт**, чтобы экспортировать данные.

ПРИМЕЧАНИЕ:

1. В экспортируемые данные всегда будут включены задачи с аннотациями, независимо от фильтров, установленных на вкладке.
2. Отмененные задачи с аннотациями также будут включены в экспортируемые данные.
3. Если вы хотите применить фильтры на вкладке к экспортируемым данным, попробуйте создать экспортные снимки с помощью SDK.

3.10.2 Форматы экспорта

RS-LabelMark поддерживает множество распространенных и стандартных форматов для экспорта завершенных задач по разметке.

3.10.2.1 ASR_MANIFEST

Экспорт меток аудиотранскрипции для автоматического распознавания речи в формате манифеста JSON, ожидаемом моделями NVIDIA NeMo. Поддерживает проекты по разметке аудиотранскрипции, в которых используется тег **Audio** с тегом **TextArea**.

```
{"audio_filepath": "/path/to/audio.wav", "text": "the transcription",  
"offset": 301.75, "duration": 0.82, "utt": "utterance_id", "ctm_utt":  
"en_4156", "side": "A"}
```

3.10.2.2 Экспорт меток кисти в NumPy и PNG

Экспортируйте метки маски кисти в виде двумерных массивов NumPy и изображений в формате PNG. Каждая метка выводится в виде отдельного изображения. Поддерживает проекты по разметке изображений с помощью тега **BrushLabels**.

3.10.2.3 COCO

Популярный формат машинного обучения, используемый в наборе данных COCO для задач обнаружения объектов и сегментации изображений. Поддерживает проекты по разметке изображений с помощью ограничивающих рамок и полигонов, в которых используются теги **BrushLabels**, **RectangleLabels**, **KeyPointLabels** или **PolygonLabels**.

Поддержка экспорта KeyPointLabels

Если вы используете **KeyPointLabels**, вам нужно будет добавить в конфигурацию разметки следующее:

- По крайней мере, один вариант **<RectangleLabels>**. Вы будете использовать его в качестве родительского ограничивающего прямоугольника для ключевых точек.
- Добавьте **model_index** к каждому **<Label>** внутри тега **<KeyPointLabels>**.

Например:

```
<View>
  <Image name="image" value="$image"/>
  <KeyPointLabels name="kp" toName="image">
    <RS-LabelMark value="nose" model_index="0"/>
    <RS-LabelMark value="eye" model_index="1"/>
    <RS-LabelMark value="tail" model_index="2"/>
  </KeyPointLabels>
  <RectangleLabels name="bbox" toName="image">
    <RS-LabelMark value="animal"/>
  </RectangleLabels>
</View>
```

После аннотирования необходимо перетащить каждую область ключевых точек под соответствующую прямоугольную область на панели **Области**.

Это создаст иерархию «родитель – потомок» (через parentID), которая необходима для экспорта.

Примеры экспорта

```
{
  "result": [
    {
      "id": "17n06ubOJs",
      "type": "keypointlabels",
      "value": {
        "x": 6.675567423230974,
        "y": 20.597014925373134,
        "width": 0.26702269692923897,
        "keypointlabels": [
          "nose"
        ]
      },
      "origin": "manual",
      "to_name": "image",
      "parentID": "QHG4TBXuNC",
      "from_name": "kp",
      "image_rotation": 0,
      "original_width": 200,
      "original_height": 179
    },
    {
      "id": "QHG4TBXuNC",
      "type": "rectanglelabels",
      "value": {
        "x": 3.871829105473965,
        "y": 4.029850746268656,
        "width": 94.39252336448598,
        "height": 92.08955223880598,
        "rotation": 0,
        "rectanglelabels": [
```

```

        "animal"
    ]
},
"origin": "manual",
"to_name": "image",
"from_name": "bbox",
"image_rotation": 0,
"original_width": 200,
"original_height": 179
}

```

3.10.2.4 CSV

Результаты сохраняются в виде значений, разделенных запятыми, с названиями столбцов, указанными в значениях полей **"from_name"** и **"to_name"** в конфигурации разметки. Поддерживается для всех типов проектов.

3.10.2.5 JSON

Список элементов в исходном формате JSON, хранящийся в одном файле JSON. Используйте этот формат для экспорта данных и аннотаций для набора данных. Поддерживается для всех типов проектов.

3.10.2.6 JSON_MIN

Список элементов, в котором экспортируются только **"from_name"**, **"to_name"** значений из исходного формата JSON. Используйте этот формат для экспорта аннотаций и данных для набора данных без полей, специфичных для RS-LabelMark. Поддерживает все типы проектов.

Например:

```

{
  "image": "https://htx-pub.s3.us-east-1.amazonaws.com/examples/images/nick-owuor-astro-nic-visuals-wDifg5xc9Z4-unsplash.jpg",
  "tag": [{
    "height": 10.458911419423693,
    "rectanglelabels": [
      "Moonwalker"
    ],
    "rotation": 0,
    "width": 12.4,
    "x": 50.8,
    "y": 5.869797225186766
  }]
}

```

3.10.2.7 Pascal VOC XML

Для задач по обнаружению объектов и сегментации изображений используются популярные данные в формате XML. Поддерживает проекты по разметке изображений с помощью ограничивающих рамок, в которых используется тег **RectangleLabels**.

3.10.2.8 TSV

Результаты сохраняются в табличном файле с разделителями табуляцией, где названия столбцов указаны в **"from_name"** и **"to_name"** значениях в конфигурации разметки. Поддерживает все типы проектов.

3.10.3 Формат JSON аннотированных задач

При аннотировании данных RS-LabelMark сохраняет результат в формате JSON. В необработанной структуре JSON каждой выполненной задачи используется следующий пример:

```
{
  "id": 1,
  "created_at": "2021-03-09T21:52:49.513742Z",
  "updated_at": "2021-03-09T22:16:08.746926Z",
  "project": 83,

  "data": {
    "image": "https://example.com/opensource/RS-LabelMark/1.jpg"
  },

  "annotations": [
    {
      "id": "1001",
      "result": [
        {
          "from_name": "tag",
          "id": "Dx_aB91ISN",
          "source": "$image",
          "to_name": "img",
          "type": "rectanglelabels",
          "value": {
            "height": 10.458911419423693,
            "rectanglelabels": [
              "Moonwalker"
            ],
            "rotation": 0,
            "width": 12.4,
            "x": 50.8,
            "y": 5.869797225186766
          }
        }
      ],
      "was_cancelled": false,
      "ground_truth": false,
      "created_at": "2021-03-09T22:16:08.728353Z",
      "updated_at": "2021-03-09T22:16:08.728378Z",
      "lead_time": 4.288,
      "result_count": 0,
      "task": 1,
      "completed_by": 10
    }
  ],

  "predictions": [
    {
      "created_ago": "3 hours",
      "model_version": "model 1",
      "result": [
        {
          "from_name": "tag",
          "id": "t5sp3TyXPo",
          "source": "$image",
```

```

        "to_name": "img",
        "type": "rectanglelabels",
        "value": {
            "height": 11.612284069097889,
            "rectanglelabels": [
                "Moonwalker"
            ],
            "rotation": 0,
            "width": 39.6,
            "x": 13.2,
            "y": 34.702495201535505
        }
    }
}
],
},
{
    "created_ago": "4 hours",
    "model_version": "model 2",
    "result": [
        {
            "from_name": "tag",
            "id": "t5sp3TyXPo",
            "source": "$image",
            "to_name": "img",
            "type": "rectanglelabels",
            "value": {
                "height": 33.612284069097889,
                "rectanglelabels": [
                    "Moonwalker"
                ],
                "rotation": 0,
                "width": 39.6,
                "x": 13.2,
                "y": 54.702495201535505
            }
        }
    ]
}
]
}
}

```

3.10.3.1 Соответствующие описания свойств JSON

Имя свойства JSON	Описание
id	Идентификатор задачи по разметке из набора данных.
data	Данные, скопированные из формата задачи по вводу данных.
project	Идентификатор конкретного проекта в RS-LabelMark.
annotations	Массив, содержащий результаты разметки для задачи.
annotations.id	Идентификатор завершенной задачи.

Имя свойства JSON	Описание
annotations.lead_time	Время, затраченное на разметку задачи, в секундах.
annotations.result	Массив, содержащий результаты задачи по разметке или аннотированию.
annotations.updated_at	Отметка времени создания или изменения аннотации.
annotations.completed_at	Отметка времени создания или отправки аннотации.
annotations.completed_by	Идентификатор пользователя, создавшего аннотацию. Соответствует порядку отображения пользователей на странице «Люди» в пользовательском интерфейсе RS-LabelMark.
annotations.was_cancelled	Boolean. Информация о том, была ли аннотация пропущена или отменена.
result.id	Идентификатор конкретного результата аннотирования для этой задачи. Используйте его для объединения областей из разных контрольных тегов, например <Labels> и <Rectangle>
result.parentID	(необязательно) Ссылка на родительскую область result.id. Позволяет упорядочить области в виде иерархического дерева на панели «Область»
result.from_name	Имя тега, используемого для обозначения области.
result.to_name	Имя тега объекта, на основе которого была обозначена область.
result.type	Тип тега, используемого для аннотирования задачи.
result.value	Значение тега, содержащее подробную информацию о результатах аннотирования задачи. Структура значения зависит от тега метки.
drafts	Массив черновых аннотаций. Имеет тот же формат, что и массив аннотаций. Включается только в задачи, экспортированные в виде снимка из пользовательского интерфейса или с помощью API.
predictions	Массив прогнозов машинного обучения. Имеет тот же формат, что и массив аннотаций, с одним дополнительным параметром.
predictions.score	Общая оценка результата на основе вероятностного вывода, уровня достоверности и других параметров.
task.updated_at	Отметка времени создания, обновления или удаления задачи или любой из ее аннотаций или отзывов.

3.10.3.2 Единицы измерения аннотаций к изображениям

Единицы измерения x , y , ширины и высоты аннотаций к изображениям указаны в процентах от общего размера изображения.

Используйте следующие формулы преобразования для x , y , **width**, **height**:

```
pixel_x = x / 100.0 * original_width
pixel_y = y / 100.0 * original_height
pixel_width = width / 100.0 * original_width
pixel_height = height / 100.0 * original_height
```

Например:

```
task = {
    "annotations": [{
        "result": [
            {
                "...": "...",

                "original_width": 600,
                "original_height": 403,
                "image_rotation": 0,

                "value": {
                    "x": 5.33,
                    "y": 23.57,
                    "width": 29.16,
                    "height": 31.26,
                    "rotation": 0,
                    "rectanglelabels": [
                        "Airplane"
                    ]
                }
            }
        ]
    }]
}

# convert from LS percent units to pixels
def convert_from_ls(result):
    if 'original_width' not in result or 'original_height' not in result:
        return None

    value = result['value']
    w, h = result['original_width'], result['original_height']

    if all([key in value for key in ['x', 'y', 'width', 'height']]):
        return w * value['x'] / 100.0, \
            h * value['y'] / 100.0, \
            w * value['width'] / 100.0, \
            h * value['height'] / 100.0

# convert from pixels to LS percent units
def convert_to_ls(x, y, width, height, original_width, original_height):
    return x / original_width * 100.0, y / original_height * 100.0, \
        width / original_width * 100.0, height / original_height * 100
```

```
# convert from LS
output = convert_from_ls(task['annotations'][0]['result'][0])
if output is None:
    raise Exception('Wrong convert')
pixel_x, pixel_y, pixel_width, pixel_height = output
print(pixel_x, pixel_y, pixel_width, pixel_height)

# convert back to LS
x, y, width, height = convert_to_ls(pixel_x, pixel_y, pixel_width,
pixel_height, 600, 403)
print(x, y, width, height)
```

3.11 Завершение работы

Для того чтобы завершить работу с программой RS-LabelMark, щелкните по иконке профиля в правом верхнем углу и нажмите на кнопку **Выйти**. Для корректного закрытия сессии рекомендуется завершать работу с системой с использованием данной кнопки.

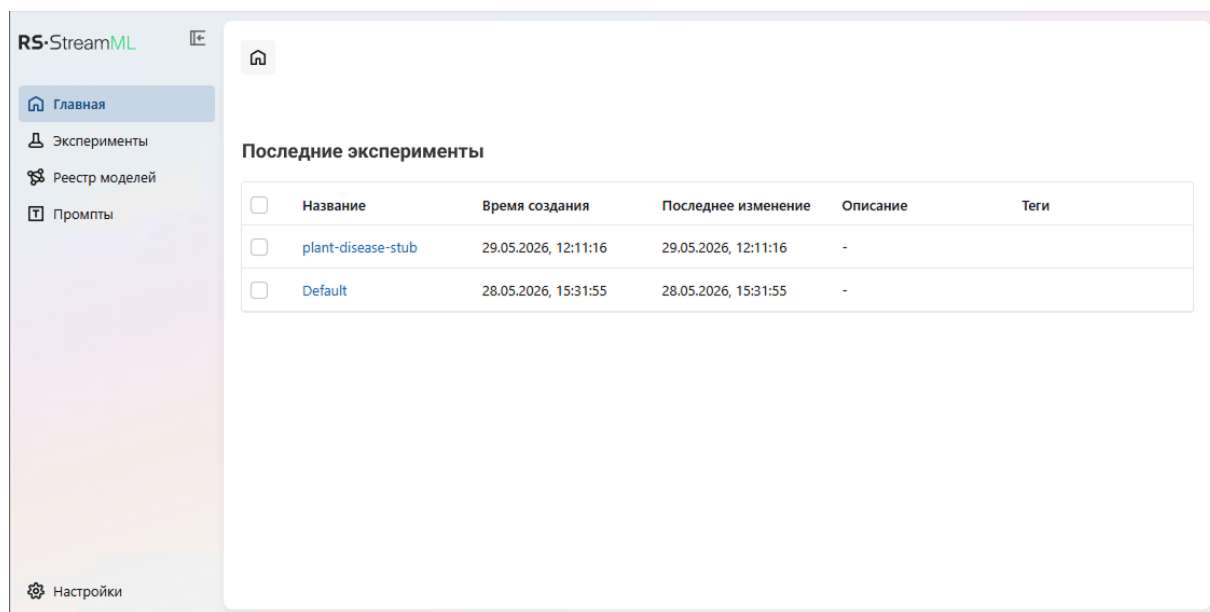
4 Работа с системой RS-StreamML

RS-StreamML – платформа инженерии ИИ для агентов, LLM и моделей машинного обучения, созданная на основе открытого проекта MLflow (лицензия Apache 2.0; оригинальный автор MLflow – Databricks, Inc).

RS-StreamML позволяет командам любого размера отлаживать, оценивать, мониторить и оптимизировать AI-приложения промышленного уровня, одновременно контролируя затраты и управляя доступом к моделям и данным.

4.1 Начало работы с системой

Для того чтобы начать работу с системой RS-StreamML, откройте браузер и в строке адреса введите ссылку на приложение. После этого откроется **Главная** страница RS-StreamML.



Главная страница

4.2 Отслеживание экспериментов

RS-StreamML – это API и пользовательский интерфейс (UI) для логирования параметров, версий кода, метрик и выходных файлов при выполнении кода машинного обучения, а также для последующей визуализации результатов.

Основные понятия

Запуски (Runs)

Запуск (run) представляет собой выполнение некоторого фрагмента кода для анализа данных. Каждый запуск сохраняет метаданные (различную информацию о запуске: метрики, параметры, время начала и окончания) и артефакты (выходные файлы запуска, такие как веса модели, изображения и т. д.).

Модели (Models)

Модели представляют собой обученные артефакты машинного обучения, создаваемые в процессе запусков. Залогированные модели содержат собственные метаданные и артефакты, аналогично запускам.

Эксперименты (Experiments)

Эксперимент группирует запуски и модели для конкретной задачи. Вы можете создать эксперимент с помощью CLI, API или UI. API и интерфейс RS-StreamML также позволяют создавать эксперименты и выполнять их поиск.

4.2.1 Отслеживание запусков

API RS-StreamML предоставляет набор функций для отслеживания запусков. Например, вы можете вызвать `RSStreamML.start_run()` для начала нового запуска, а затем использовать функции логирования, такие как `RSStreamML.log_param()` и `RSStreamML.log_metric()`, для логирования параметров и метрик соответственно.

```
import RSStreamML

with RSStreamML.start_run():
    RSStreamML.log_param("lr", 0.001)
    # ML код
    ...
    RSStreamML.log_metric("val_loss", val_loss)
```

В качестве альтернативы, автоматическое логирование (Auto-logging) предлагает быструю настройку для начала отслеживания в RS-StreamML.

Эта функция позволяет логировать метрики, параметры и модели без необходимости явных операторов логирования – все, что нужно сделать, это вызвать `RSStreamML.autolog()` перед кодом обучения. Автоматическое логирование поддерживает популярные библиотеки.

4.2.1.1 Ключевые возможности

- **SQL-подобная фильтрация:** используйте префиксы `metrics.`, `params.` и `attribute.` для построения сложных запросов.
- **Поиск с учетом датасетов:** фильтруйте метрики на основе конкретных наборов данных для честного сравнения моделей.
- **Гибкая сортировка:** упорядочивайте по нескольким критериям для поиска лучших моделей.
- **Прямая загрузка модели:** используйте новый формат URI `models:/<model_id>` для немедленного доступа к модели.

4.2.2 Отслеживание моделей

RS-StreamML предоставляет расширенные возможности отслеживания моделей, позволяя логировать несколько контрольных точек (checkpoint) модели в рамках одного запуска и отслеживать их производительность на разных наборах данных.

4.2.2.1 Логирование контрольных точек модели

Вы можете логировать контрольные точки модели на разных шагах в процессе обучения, используя параметр `step` в функциях логирования модели. Каждая залогированная модель получает уникальный идентификатор `model ID`, который вы можете использовать для последующей ссылки на нее.

```
import RSStreamML
import RSStreamML.pytorch

with RSStreamML.start_run() as run:
    for epoch in range(100):
        # Обучение модели
        train_model(model, epoch)

        # Логирование контрольной точки модели каждые 10 эпох
        if epoch % 10 == 0:
            model_info = RSStreamML.pytorch.log_model(
```

```

        pytorch_model=model,
        name=f"checkpoint-epoch-{epoch}",
        step=epoch,
        input_example=sample_input,
    )

    # Логирование метрик, связанных с этой конкретной контрольной
точкой модели
    accuracy = evaluate_model(model, validation_data)
    RSStreamML.log_metric(
        key="accuracy",
        value=accuracy,
        step=epoch,
        model_id=model_info.model_id, # Привязка метрики к
конкретной модели
        dataset=validation_dataset,
    )

```

4.2.2.2 Привязка метрик к моделям и датасетам

RS-StreamML позволяет связывать метрики с конкретными контрольными точками моделей и наборами данных, обеспечивая отслеживаемость производительности модели:

```

# Создание ссылки на датасет
train_dataset = RSStreamML.data.from_pandas(train_df, name="training_data")

# Логирование метрики с привязкой к модели и датасету
RSStreamML.log_metric(
    key="f1_score",
    value=0.95,
    step=epoch,
    model_id=model_info.model_id, # Привязка к конкретной контрольной точке
модели
    dataset=train_dataset,        # Привязка к конкретному датасету
)

```

4.2.2.3 Поиск и ранжирование контрольных точек моделей

Используйте `RSStreamML.search_logged_models()` для поиска и ранжирования контрольных точек моделей на основе их метрик производительности:

```

# Поиск всех моделей в запуске, упорядоченных по accuracy
ranked_models = RSStreamML.search_logged_models(
    filter_string=f"source_run_id='{run.info.run_id}'",
    order_by=[{"field_name": "metrics.accuracy", "ascending": False}],
    output_format="list",
)

# Получение лучшей модели
best_model = ranked_models[0]
print(f"Best model: {best_model.name}")
print(f"Accuracy: {best_model.metrics[0].value}")

# Загрузка лучшей модели для инференса
loaded_model = RSStreamML.pyfunc.load_model(f"models/{best_model.model_id}")

```

4.2.3 Отслеживание датасетов

RS-StreamML предоставляет возможность отслеживать датасеты, связанные с событиями обучения модели. Метаданные, связанные с датасетом, могут храниться с использованием API `RSStreamML.log_input()`.

4.2.3.1 Изучение запусков, моделей и результатов

Tracking UI позволяет визуально исследовать эксперименты, запуски и модели.

- Список запусков на основе экспериментов и их сравнение (включая сравнение запусков из нескольких экспериментов).
- Поиск запусков по значению параметра или метрики.
- Визуализация метрик запуска.
- Скачивание результатов запусков (артефактов и метаданных).

4.3 Работа с сервером отслеживания

Tracking server (сервер отслеживания) — это автономный HTTP-сервер, который предоставляет несколько конечных точек REST API для отслеживания запусков и экспериментов.

4.3.1 Запуск Tracking Server

Запуск сервера отслеживания осуществляется следующей командой:

```
mlflow server --host 127.0.0.1 --port 8080
```

После запуска сервера вы увидите следующий вывод:

```
INFO:      Started server process [28550]
INFO:      Waiting for application startup.
INFO:      Application startup complete.
INFO:      Uvicorn running on http://127.0.0.1:8080 (Press CTRL+C to quit)
```

Файловые системы «только для чтения»

При работе в контейнерах с корневыми файловыми системами «только для чтения» настройте удаленное хранилище артефактов с помощью `--artifacts-destination` (обслуживание артефактов включено по умолчанию). Сервер отслеживания не создает локальные директории при запуске, если используется удаленное хранилище артефактов, что делает его совместимым с окружениями «только для чтения».

```
mlflow server \
  --host 0.0.0.0 \
  --backend-store-uri postgresql://user:pass@host/db \
  --artifacts-destination s3://my-bucket
```

4.3.2 Логирование в Tracking Server

После запуска сервера отслеживания подключите локальные клиенты, установив переменную окружения `RSSTREAMML_TRACKING_URI` в URI сервера (включая схему и порт, например `http://10.0.0.1:5000`) или вызвав `mlflow.set_tracking_uri()`.

API RS-StreamML, такие как `mlflow.start_run()`, `mlflow.log_param()`, `mlflow.start_trace()`, отправляют API-запросы к вашему удаленному серверу отслеживания и логируют данные.

```
import mlflow

remote_server_uri = "..." # укажите URI вашего сервера, например
http://127.0.0.1:8080
mlflow.set_tracking_uri(remote_server_uri)
mlflow.set_experiment("/my-experiment")

# Логирование запуска
with mlflow.start_run():
    mlflow.log_param("a", 1)
    mlflow.log_metric("b", 2)

# Логирование трассировки
with mlflow.start_span(name="test_trace") as span:
    span.set_inputs({"x": 1, "y": 2})
    span.set_outputs(3)
import * as mlflow from "@mlflow/core";
```

```

mlflow.init({
  trackingUri: "<your-tracking-server-uri>",
  experimentId: "<your-experiment-id>",
});

const myFunc = mlflow.trace(
  (a: number, b: number) => {
    return a + b;
  },
  { name: 'my-func' }
)
myFunc(1, 2);
library(mlflow)
install_mlflow()
remote_server_uri = "... " # укажите URI вашего сервера
mlflow_set_tracking_uri(remote_server_uri)
mlflow_set_experiment("/my-experiment")
mlflow_log_param("a", "1")
import org.mlflow.tracking.MlflowClient

val remoteServerUri = "... " // укажите URI вашего сервера
val client = new MlflowClient(remoteServerUri)

val experimentId = client.createExperiment("my-experiment")
client.setExperiment(experimentId)

val run = client.createRun(experimentId)
client.logParam(run.getRunId(), "a", "1")

```

4.3.3 Настройка сервера

В этом разделе описывается, как настроить сервер отслеживания для некоторых распространенных случаев использования.

4.3.3.1 Хранилище метаданных (Backend Store)

По умолчанию сервер отслеживания использует базу данных SQLite (sqlite:///mlflow.db) для хранения метаданных запусков. Вы можете настроить другое хранилище метаданных, добавив опцию `--backend-store-uri`:

Примеры

```

# По умолчанию: база данных SQLite (mlflow.db в текущей директории)
mlflow server

# PostgreSQL: подключение к существующей базе данных PostgreSQL
mlflow server --backend-store-uri
postgresql://username:password@host:port/database

# Файловое хранилище (устаревшее): использование локальной файловой системы в
директории ./mlruns
mlflow server --backend-store-uri ./mlruns

```

SQLite используется по умолчанию и обеспечивает хорошую производительность и надежность для большинства случаев использования. Для производственных развертываний с высокой степенью параллелизма рассмотрите возможность использования PostgreSQL или MySQL.

4.3.3.2 Архивация трассировок (Trace Archival)

RS-StreamML может архивировать старые полезные нагрузки спанов трассировок из базы данных отслеживания в настроенное расположение артефактов, сохраняя при этом возможность чтения трассировок в UI и API.

Создайте файл конфигурации YAML и укажите его в

RSSTREAMML_TRACE_ARCHIVAL_CONFIG:

```
trace-archival.yaml
trace_archival:
  enabled: true
  location: s3://mlflow-trace-archive
  retention: 30d
  interval_seconds: 300
  # Опционально: опустите этот ключ, чтобы сделать каждый проход планировщика
  # неограниченным.
  max_traces_per_pass: 1000
  long_retention_allowlist:
    # ID эксперимента для customer-support-agent-prod
    - 12345
```

Затем запустите сервер с этой конфигурацией, либо через переменную окружения, либо с помощью флага `--trace-archival-config`:

```
export RSSTREAMML_TRACE_ARCHIVAL_CONFIG=/etc/mlflow/trace-archival.yaml

mlflow server \
  --backend-store-uri postgresql://user:pass@localhost/mlflow \
  --default-artifact-root s3://mlflow-artifacts

# Или передайте путь к конфигурации напрямую
mlflow server \
  --backend-store-uri postgresql://user:pass@localhost/mlflow \
  --default-artifact-root s3://mlflow-artifacts \
  --trace-archival-config /etc/mlflow/trace-archival.yaml
```

- `interval_seconds` определяет, как часто планировщик проверяет необходимость выполнения архивации.
- `max_traces_per_pass` ограничивает объем работы за один проход. Если опустить этот параметр, проход будет неограниченным, и RS-StreamML заархивирует все подходящие трассировки, которые сможет обработать за этот запуск.
- `long_retention_allowlist` содержит идентификаторы экспериментов, которым разрешено хранить трассировки дольше, чем это предусмотрено общей политикой сервера или рабочего пространства.

Когда рабочие пространства включены, настройки архивации трассировок разрешаются в следующем порядке:

- Глобальные значения по умолчанию предоставляют базовое расположение архивации и срок хранения.
- Параметр `trace_archival_config` на уровне рабочего пространства переопределяет эти значения по умолчанию для всех трассировок в данном рабочем пространстве.
- Настройки на уровне эксперимента могут дополнительно сократить срок хранения для конкретного эксперимента.
- Если эксперимент запрашивает больший срок хранения, чем общая политика сервера или рабочего пространства, этот больший срок соблюдается только в том случае, когда идентификатор эксперимента присутствует в `long_retention_allowlist`.

4.3.3.3 Удаленное хранилище артефактов

4.3.3.3.1 Использование Tracking Server для проксированного доступа к артефактам (по умолчанию)

По умолчанию tracking server хранит артефакты в своей локальной файловой системе в директории `./mlartifacts`.

Чтобы настроить tracking server на подключение к удаленному хранилищу и обслуживание артефактов, запустите сервер с флагом `--artifacts-destination`.

```
mlflow server \
  --host 0.0.0.0 \
  --port 8885 \
  --artifacts-destination s3://my-bucket
```

При такой настройке сервер RS-StreamML работает как прокси для доступа к удаленным артефактам.

Tracking Server преобразует URI `mlflow-artifacts:/` в запросе отслеживания от клиента в явную цель объектного хранилища (например, `s3://my_bucket/mlartifacts`) для взаимодействия с артефактами.

Следующие шаблоны будут указывать на настроенное прокси-хранилище объектов (в приведенном выше примере `s3://my-root-bucket/mlartifacts`):

- `https://<host>:<port>/mlartifacts`
- `http://<host>/mlartifacts`
- `mlflow-artifacts://<host>/mlartifacts`
- `mlflow-artifacts://<host>:<port>/mlartifacts`
- `mlflow-artifacts:/mlartifacts`

4.3.3.3.2 Использование Tracking Server без проксирования

Для получения прямого доступа к удаленному хранилищу без проксирования через tracking server, запустите сервер с флагом `--no-serve-artifacts`, установив `--default-artifact-root` в URI удаленного хранилища, на которое вы хотите перенаправлять запросы.

```
mlflow server --no-serve-artifacts --default-artifact-root s3://my-bucket
```

При такой настройке клиент RS-StreamML по-прежнему отправляет минимальные HTTP-запросы на tracking server для получения правильного URI удаленного хранилища, но может напрямую загружать артефакты в удаленное хранилище или скачивать их из него.

Используйте правильные ключи конфигурации

Настройте опций `--serve-artifacts` и `--default-artifact-root`.

- Для того чтобы сервер проксировал загрузки, укажите опцию `--artifacts-destination` и не устанавливайте `--default-artifact-root`. Новые эксперименты получают URI запусков `mlflow-artifacts:/...`, и клиенты будут загружать и скачивать артефакты через сервер.

- Для того чтобы клиенты писали непосредственно в хранилище, установите опцию `--default-artifact-root` и `--no-serve-artifacts`. Запуски получают URI `s3://.../gs://...`, и клиенты должны иметь учетные данные для доступа к хранилищу.

4.3.4 Tracking API

RS-StreamML предлагает два основных метода отслеживания экспериментов, каждый из которых оптимизирован для различных случаев использования:

1. Автоматическое логирование. Без настройки, максимальное покрытие

Подходит для быстрого начала работы или при использовании поддерживаемых ML-библиотек.

```
import mlflow

mlflow.autolog()

# существующий код обучения работает без изменений
model.fit(X_train, y_train)
```

Что логируется автоматически:

- Параметры модели и гиперпараметры
- Метрики обучения и валидации

- Артефакты модели и контрольные точки
- Графики и визуализации обучения
- Специфичные для фреймворка метаданные

Поддерживаемые библиотеки: Scikit-learn, XGBoost, LightGBM, PyTorch, Keras/TensorFlow, Spark и другие.

2. Ручное логирование. Настраиваемые рабочие процессы

Подходит для пользовательских циклов обучения, продвинутого экспериментирования или когда вам нужен точный контроль над тем, что отслеживается.

```
import mlflow

with mlflow.start_run():
    # Логирование параметров
    mlflow.log_param("learning_rate", 0.01)
    mlflow.log_param("batch_size", 32)

    # логика обучения здесь
    for epoch in range(num_epochs):
        train_loss = train_model()
        val_loss = validate_model()

        # Логирование метрик с отслеживанием шага
        mlflow.log_metrics({"train_loss": train_loss, "val_loss": val_loss},
                           step=epoch)

    # Логирование финальной модели
    mlflow.sklearn.log_model(model, name="model")
MlflowClient client = new MlflowClient();
RunInfo run = client.createRun();

// Логирование параметров
client.logParam(run.getRunId(), "learning_rate", "0.01");
client.logParam(run.getRunId(), "batch_size", "32");

// Логирование метрик с временными шагами
for (int epoch = 0; epoch < numEpochs; epoch++) {
    double trainLoss = trainModel();
    client.logMetric(run.getRunId(), "train_loss", trainLoss,
                    System.currentTimeMillis(), epoch);
}
library(mlflow)

with(mlflow_start_run(), {
  # Логирование параметров
  mlflow_log_param("learning_rate", 0.01)
  mlflow_log_param("batch_size", 32)

  # Цикл обучения
  for (epoch in 1:num_epochs) {
    train_loss <- train_model()
    mlflow_log_metric("train_loss", train_loss, step = epoch)
  }
})
```

4.3.4.1 Основные функции логирования

4.3.4.1.1 Настройка и конфигурация

Функция	Назначение	Пример
<code>mlflow.set_tracking_uri()</code>	Подключение к tracking server или базе данных	<code>mlflow.set_tracking_uri("http://localhost:5000")</code>

Функция	Назначение	Пример
<code>mlflow.get_tracking_uri()</code>	Получение текущего URI отслеживания	<code>uri = mlflow.get_tracking_uri()</code>
<code>mlflow.create_experiment()</code>	Создание нового эксперимента	<code>exp_id = mlflow.create_experiment("my-experiment")</code>
<code>mlflow.set_experiment()</code>	Установка активного эксперимента	<code>mlflow.set_experiment("fraud-detection")</code>

4.3.4.1.2 Управление запусками (Run Management)

Функция	Назначение	Пример
<code>mlflow.start_run()</code>	Начать новый запуск (с менеджером контекста)	<code>with mlflow.start_run(): ...</code>
<code>mlflow.end_run()</code>	Завершить текущий запуск	<code>mlflow.end_run(status="FINISHED")</code>
<code>mlflow.active_run()</code>	Получить текущий активный запуск	<code>run = mlflow.active_run()</code>
<code>mlflow.last_active_run()</code>	Получить последний завершённый запуск	<code>last_run = mlflow.last_active_run()</code>

4.3.4.1.3 Логирование данных

Функция	Назначение	Пример
<code>mlflow.log_param()</code> / <code>mlflow.log_params()</code>	Логирование гиперпараметров	<code>mlflow.log_param("lr", 0.01)</code>
<code>mlflow.log_metric()</code> / <code>mlflow.log_metrics()</code>	Логирование метрик производительности	<code>mlflow.log_metric("accuracy", 0.95, step=10)</code>
<code>mlflow.log_input()</code>	Логирование информации о датасете	<code>mlflow.log_input(dataset)</code>
<code>mlflow.set_tag()</code> / <code>mlflow.set_tags()</code>	Добавление метаданных-тегов	<code>mlflow.set_tag("model_type", "CNN")</code>

4.3.4.1.4 Управление артефактами

Функция	Назначение	Пример
<code>mlflow.log_artifact()</code>	Логирование одного файла/директории	<code>mlflow.log_artifact("model.pkl")</code>
<code>mlflow.log_artifacts()</code>	Логирование целой директории	<code>mlflow.log_artifacts("./plots/")</code>
<code>mlflow.get_artifact_uri()</code>	Получение расположения хранилища артефактов	<code>uri = mlflow.get_artifact_uri()</code>

4.3.4.1.5 Управление моделями

Функция	Назначение	Пример
<code>mlflow.initialize_logged_model()</code>	Инициализация логируемой модели в состоянии PENDING	<code>model = mlflow.initialize_logged_model(name="my_model")</code>
<code>mlflow.create_external_model()</code>	Создание внешней модели	<code>model = mlflow.create_external_model(name="agent")</code>
<code>mlflow.finalize_logged_model()</code>	Обновление статуса модели на READY или FAILED	<code>mlflow.finalize_logged_model(model_id, "READY")</code>
<code>mlflow.get_logged_model()</code>	Получение логируемой модели по ID	<code>model = mlflow.get_logged_model(model_id)</code>
<code>mlflow.last_logged_model()</code>	Получение самой последней логируемой модели	<code>model = mlflow.last_logged_model()</code>
<code>mlflow.search_logged_models()</code>	Поиск логируемых моделей	<code>models = mlflow.search_logged_models(filter_string="name='my_model'")</code>

Функция	Назначение	Пример
<code>mlflow.log_model_params()</code>	Логирование параметров для конкретной модели	<code>mlflow.log_model_params({"param": "value"}, model_id)</code>
<code>mlflow.set_logged_model_tags()</code>	Установка тегов на логируемой модели	<code>mlflow.set_logged_model_tags(model_id, {"key": "value"})</code>
<code>mlflow.delete_logged_model_tag()</code>	Удаление тега из логируемой модели	<code>mlflow.delete_logged_model_tag(model_id, "key")</code>

4.4 Модели

4.4.1 Реестр моделей

Реестр моделей RS-StreamML – это централизованное хранилище моделей, набор API и пользовательский интерфейс, предназначенные для совместного управления полным жизненным циклом модели машинного обучения.

Реестр моделей обеспечивает отслеживание происхождения (т.е. какой эксперимент и запуск RS-StreamML привели к созданию модели), версионирование, псевдонимы, тегирование метаданных и поддержку аннотаций.

4.4.1.1 Зачем нужен реестр моделей

Использование Реестра моделей дает следующие преимущества:

- **Контроль версий:** реестр автоматически отслеживает версии каждой модели, позволяя командам сравнивать итерации, откатываться к предыдущим состояниям и управлять несколькими версиями параллельно (например, для промежуточного (staging) и производственного (production) окружений).
- **Происхождение и прослеживаемость модели:** каждая зарегистрированная версия модели связана с запуском RS-StreamML, залогированной моделью или ноутбуком, которые ее создали, что обеспечивает полную воспроизводимость. Вы можете проследить, как именно была обучена модель, с какими данными и параметрами.
- **Готовые к production рабочие процессы:** такие функции, как псевдонимы моделей (например, `@champion`) и теги, упрощают управление рабочими процессами развертывания, позволяя продвигать модели в экспериментальное, промежуточное или производственное окружение контролируемым и аудируемым способом.
- **Управление и соответствие требованиям:** благодаря структурированным метаданным, тегированию и управлению доступом на основе ролей, Реестр моделей поддерживает требования управления, критически важные для ML-операций корпоративного уровня.

4.4.1.2 Основные понятия

Основные понятия Реестра моделей:

Понятие	Описание
Модель (Model)	Модель RS-StreamML создается с помощью одного из методов <code>RSStreamML.<model_flavor>.log_model()</code> или API <code>RSStreamML.create_external_model()</code> . После логирования эта модель может быть зарегистрирована в Реестре моделей.
Зарегистрированная модель (Registered Model)	Модель RS-StreamML может быть зарегистрирована в Реестре моделей. Зарегистрированная модель имеет уникальное имя, содержит версии, псевдонимы, теги и другие метаданные.
Версия модели (Model Version)	Каждая зарегистрированная модель может иметь одну или несколько версий. Когда новая модель добавляется в Реестр моделей, она добавляется как версия 1. Каждая новая модель,

Понятие	Описание
	зарегистрированная под тем же именем, увеличивает номер версии. Версии моделей могут иметь теги, которые полезны для отслеживания атрибутов версии модели (например, <code>pre_deploy_checks: "PASSED"</code>).
URI модели (Model URI)	Вы можете ссылаться на зарегистрированную модель, используя URI следующего формата: <code>models:/<имя-модели>/<версия-модели></code> . Например, если у вас есть зарегистрированная модель с именем «MyModel» и версией 1, URI, ссылающийся на эту модель, будет: <code>models:/MyModel/1</code> .
Псевдоним модели (Model Alias)	Псевдонимы моделей позволяют назначить изменяемую именованную ссылку на определенную версию зарегистрированной модели. Назначив псевдоним конкретной версии модели, вы можете использовать этот псевдоним для ссылки на данную версию модели через URI модели или API реестра. Например, вы можете создать псевдоним с именем <code>champion</code>, указывающий на версию 1 модели с именем <code>MyModel</code>. Затем вы можете ссылаться на версию 1 <code>MyModel</code>, используя URI <code>models:/MyModel@champion</code>. Псевдонимы удобно использовать для развертывания моделей. Например, вы можете назначить псевдоним <code>champion</code> версии модели, предназначенной для производственного трафика, и использовать этот псевдоним в производственных нагрузках. Затем вы можете обновить модель, обслуживающую производственный трафик, переназначив псевдоним <code>champion</code> на другую версию модели.
Теги (Tags)	Теги – это пары «ключ-значение», которые вы связываете с зарегистрированными моделями и версиями моделей, позволяя маркировать и категоризировать их по функциям или статусу. Например, вы можете применить тег с ключом <code>"task"</code> и значением <code>"question-answering"</code> (отображается в интерфейсе как <code>task:question-answering</code>) к зарегистрированным моделям, предназначенным для задач ответа на вопросы. На уровне версии модели вы можете пометить версии, проходящие предразверточную валидацию, тегом <code>validation_status:pending</code> , а те, которые допущены к развертыванию – тегом <code>validation_status:approved</code> .
Аннотации и описания (Annotations and Descriptions)	Вы можете аннотировать модель верхнего уровня и каждую версию индивидуально, используя Markdown, включая описание и любую релевантную информацию, полезную для команды, такую как описания алгоритмов, использованные наборы данных или общая методология, задействованная в подходе к моделированию данной версии.

4.4.1.3 Регистрация модели в RS-StreamML

Откройте страницу с деталями запуска RS-StreamML, содержащего залогированную модель, которую вы хотите зарегистрировать.

Далее выберите папку модели, содержащую нужную модель RS-StreamML, в разделе «**Artifacts**» (Артефакты).

Нажмите на кнопку «**Register Model**» (Зарегистрировать модель), после чего появится модальная форма.

В данной форме в выпадающем меню «**Model**» выберите «**Create New Model**» (Создать новую модель) – это создаст новую зарегистрированную модель с вашей моделью RS-StreamML в качестве начальной версии.

Затем нажмите на кнопку регистрации.

4.4.2 Работа с зарегистрированными моделями

В данном разделе описана функциональность Реестра моделей – от регистрации модели и изучения ее структуры до загрузки конкретной версии модели для дальнейшего использования.

4.4.2.1 Реестр моделей

ПРИМЕЧАНИЕ: в описании указан локальный сервер отслеживания (tracking server) и реестр моделей. Для производственных сценариев рекомендуется использовать удаленный сервер отслеживания.

4.4.2.1.1 Шаг 0: Установка зависимостей

```
pip install --upgrade RSStreamML
```

4.4.2.1.2 Шаг 1: Регистрация модели

Чтобы использовать реестр моделей RS-StreamML, нужно добавить в него ваши модели. Это делается путем регистрации данной модели с помощью одной из следующих команд:

- `RSStreamML.<model_flavor>.log_model(registered_model_name=<model_name>):` регистрирует модель при ее записи (логировании) на сервер отслеживания.
- `RSStreamML.register_model(<model_uri>, <model_name>):` регистрирует модель после ее записи на сервер отслеживания. Обратите внимание, что перед выполнением этой команды нужно будет записать модель, чтобы получить URI модели.

У RS-StreamML есть множество разновидностей моделей («флейворов»). В приведенном ниже примере используется `RandomForestRegressor` из библиотеки `scikit-learn`, чтобы продемонстрировать простой способ регистрации модели. Вы можете использовать любую поддерживаемую разновидность модели.

В следующем фрагменте кода описан запуск `RSStreamML run` и обучение модели, логирование некоторых гиперпараметров, логирование и регистрация модели.

```
from sklearn.datasets import make_regression
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split

import RSStreamML
import RSStreamML.sklearn

with RSStreamML.start_run() as run:
    X, y = make_regression(n_features=4, n_informative=2, random_state=0,
                           shuffle=False)
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
                                                         random_state=42)

    params = {"max_depth": 2, "random_state": 42}
    model = RandomForestRegressor(**params)
    model.fit(X_train, y_train)

    # Логирование параметров и метрик с помощью API RSStreamML
    RSStreamML.log_params(params)

    y_pred = model.predict(X_test)
    RSStreamML.log_metrics({"mse": mean_squared_error(y_test, y_pred)})

    # Логирование модели sklearn и регистрация как версии 1
    RSStreamML.sklearn.log_model(
        sk_model=model,
        name="sklearn-model",
        input_example=X_train,
```

```

    registered_model_name="sk-learn-random-forest-reg-model",
)

```

Пример вывода

```

Successfully registered model 'sk-learn-random-forest-reg-model'.
Created version '1' of model 'sk-learn-random-forest-reg-model'.

```

Особенности реализации:

- Для регистрации модели вы можете использовать параметр

registered_model_name в `RSStreamML.sklearn.log_model()` ИЛИ вызвать `RSStreamML.register_model()` после логирования модели.

- Сигнатуры моделей обеспечивают проверку входных и выходных данных модели. Параметр `input_example` в `log_model()` автоматически определяет и записывает сигнатуру.

4.4.2.2 Изучение зарегистрированной модели

В данном разделе описано хранение информации в пользовательском интерфейсе RS-StreamML и в локальной директории.

4.4.2.2.1 Шаг 1: Изучение директории `mlruns`

Рассмотрим структуру директории, созданную при выполнении python-скрипта.

ПРИМЕЧАНИЕ: при удаленных развертываниях, которые рекомендуются для производственных сценариев, сервер отслеживания будет находиться в объектном хранилище, а реестр моделей – в реляционной базе данных (PostgreSQL, MySQL и т.д.).

```

mlruns/
├── 0/
│   ├── bc6dc2a4f38d47b4b0c99d154bbc77ad/
│   │   ├── metrics/
│   │   │   └── mse
│   │   │       # Пример файла метрики для
│   │   │       # среднеквадратичной ошибки
│   │   ├── artifacts/
│   │   │   #Arteфакты, связанные с запуском
│   │   │   ├── sklearn-model/
│   │   │   │   ├── python_env.yaml
│   │   │   │   ├── requirements.txt
│   │   │   │   ├── MLmodel
│   │   │   │   ├── model.pkl
│   │   │   │   ├── input_example.json
│   │   │   │   └── conda.yaml
│   │   │   # Требования к пакетам Python
│   │   │   # файл модели с метаданными
│   │   │   # Сериализованный файл модели
│   │   ├── tags/
│   │   │   ├── RSStreamML.user
│   │   │   ├── RSStreamML.source.git.commit
│   │   │   ├── RSStreamML.runName
│   │   │   ├── RSStreamML.source.name
│   │   │   ├── RSStreamML.log-model.history
│   │   │   └── RSStreamML.source.type
│   │   ├── params/
│   │   │   ├── max_depth
│   │   │   └── random_state
│   │   └── meta.yaml
│   └── meta.yaml
└── models/
    # Директория Реестра моделей
    ├── sk-learn-random-forest-reg-model/
    │   # Имя зарегистрированной модели
    │   ├── version-1/
    │   │   # Директория версии модели
    │   │   └── meta.yaml
    │   └── meta.yaml

```

Сервер отслеживания организован по ID эксперимента и ID запуска и отвечает за хранение артефактов, параметров и метрик эксперимента. С другой стороны, реестр моделей хранит только метаданные с указателями на сервер отслеживания.

По умолчанию есть три способа управления окружением модели: `python_env.yaml` (python virtualenv), `requirements.txt` (требования PyPi) и `conda.yaml` (conda окружение).

4.4.2.2 Шаг 2: Запуск сервера отслеживания

В той же директории, где находится ваша папка `mlruns`, выполните приведенную ниже команду.

```
RSStreamML server --host 127.0.0.1 --port 8080
INFO:      Started server process [26393]
INFO:      Waiting for application startup.
INFO:      Application startup complete.
INFO:      Uvicorn running on http://localhost:8080 (Press CTRL+C to quit)
```

4.4.2.3 Шаг 3: Просмотр сервера отслеживания

Если нет ошибок, вы можете перейти в веб-браузер и перейти на `http://localhost:8080`, чтобы увидеть UI RS-StreamML.

Сначала перейдите в реестр моделей.

Далее добавьте теги и псевдоним версии модели, чтобы облегчить развертывание модели. Вы можете добавлять или редактировать теги и псевдонимы, нажав на соответствующую ссылку «Add» или значок карандаша в таблице версий модели.

- Добавьте тег версии модели с ключом `problem_type` и значением `regression`.
- Добавьте псевдоним версии модели `the_best_model_ever`.

4.4.2.3 Загрузка зарегистрированной модели

Чтобы выполнить вывод (inference) на зарегистрированной версии модели, нужно загрузить ее в память.

Следующий фрагмент кода показывает простой способ загрузить модель из реестра моделей через конкретный URI модели и выполнить вывод.

```
import RSStreamML.sklearn
from sklearn.datasets import make_regression

model_name = "sk-learn-random-forest-reg-model"
model_version = "latest"

# Загрузка модели из Реестра моделей
model_uri = f"models:{model_name}/{model_version}"
model = RSStreamML.sklearn.load_model(model_uri)

# Генерация нового набора данных для прогнозирования и сам прогноз
X_new, _ = make_regression(n_features=4, n_informative=2, random_state=0,
                           shuffle=False)
y_pred_new = model.predict(X_new)

print(y_pred_new)
```

4.4.2.3.1 Пример 1. Загрузка через сервер отслеживания

URI модели – это уникальный идентификатор для сериализованной модели. Учитывая, что артефакт модели хранится вместе с экспериментами на сервере отслеживания, вы можете использовать приведенные ниже URI модели, чтобы обойти реестр моделей и загрузить артефакт в память.

- Абсолютный локальный путь:

```
RSStreamML.sklearn.load_model("/Users/me/path/to/local/model")
```

- Относительный локальный путь:

```
RSStreamML.sklearn.load_model("relative/path/to/local/model")
```

- ID запуска: `RSStreamML.sklearn.load_model(f"runs:{RSStreamML._run_id}/{run_relative_path_to_model}")`

Если вы находитесь не в том же окружении, в котором логируют модель, в этом случае следует загружать модель, используя имя и версию модели.

4.4.2.3.2 Пример 2. Загрузка по имени и версии

Чтобы загрузить модель в память, используя `model_name` и `model_version`, примените следующий метод:

```
model =  
RSStreamML.sklearn.load_model(f"models:{model_name}/{model_version}")
```

Можно также использовать псевдоним версии модели (`model version alias`).

4.4.2.3.3 Пример 3. Загрузка через псевдоним версии модели

Псевдонимы версий моделей – это определяемые пользователем идентификаторы для версии модели. Поскольку они изменяемы после регистрации модели, они отделяют версии моделей от кода, который их использует.

Пример:

```
import RSStreamML.sklearn  
from RSStreamML import RSStreamMLClient  
  
client = RSStreamMLClient()  
  
# Установка псевдонима версии модели  
model_name = "sk-learn-random-forest-reg-model"  
model_version_alias = "the_best_model_ever"  
client.set_registered_model_alias(model_name, model_version_alias, "1") #  
Дублирование шага в UI  
  
# Получение информации о модели  
model_info = client.get_model_version_by_alias(model_name,  
model_version_alias)  
model_tags = model_info.tags  
print(model_tags)  
  
# Получение версии модели с использованием URI модели  
model_uri = f"models:{model_name}@{model_version_alias}"  
model = RSStreamML.sklearn.load_model(model_uri)  
  
print(model)
```

4.4.3 Использование реестра моделей

В данном разделе описано использование Реестра моделей RS-StreamML: как регистрировать модели, управлять версиями, применять псевдонимы и теги, а также организовывать модели для развертывания.

Прежде чем добавить модель в Реестр моделей, вы должны залогировать ее с помощью методов `log_model` соответствующих разновидностей моделей (`flavors`). После того как модель залогирована, вы можете добавить, изменить, обновить или удалить модель в Реестре моделей через UI или API.

4.4.3.1 Регистрация модели

Выполните следующие шаги, чтобы зарегистрировать модель в Реестре моделей.

Откройте страницу с деталями запуска (Run) RS-StreamML, содержащего залогированную модель, которую вы хотите зарегистрировать. Выберите папку модели, содержащую нужную модель, в разделе Артефакты (Artifacts).

Нажмите на кнопку **Register Model**, после чего появится всплывающая форма.

В выпадающем меню **Model** на форме вы можете либо выбрать **Create New Model** (создает новую зарегистрированную модель с вашей моделью в качестве ее начальной версии), либо выбрать существующую зарегистрированную модель (регистрирует вашу модель как новую версию внутри нее).

4.4.3.2 Поиск зарегистрированных моделей

После регистрации ваших моделей в Реестре моделей вы можете перейти к ним следующими способами:

- Перейдите на страницу **Registered Models**, которая содержит ссылки на ваши зарегистрированные модели и соответствующие версии моделей.
- Перейдите в раздел **Artifacts** на странице деталей вашего запуска, нажмите на папку модели, а затем нажмите на версию модели в правом верхнем углу, чтобы просмотреть версию, созданную из этой модели.

4.4.3.3 Развертывание и организация моделей

Вы можете развертывать и организовывать модели в Реестре моделей, используя псевдонимы (*aliases*) и теги (*tags*) моделей. Чтобы установить псевдонимы и теги для версий модели в вашей зарегистрированной модели, перейдите на страницу обзора вашей зарегистрированной модели.

Вы можете добавить или редактировать псевдонимы и теги для конкретной версии модели, нажав на соответствующую ссылку **Add** или значок карандаша в таблице версий модели.

4.5 Промпты

Реестр промптов RS-StreamML – это инструмент, упрощающий разработку и управление промптами в LLM-приложениях и AI-агентах. Он позволяет версионировать, отслеживать и повторно использовать промпты внутри всей организации, обеспечивая единообразие и улучшая совместную работу над промптами.

4.5.1 Начало работы

4.5.1.1 Создание промпта

1. Выполните `RSStreamML server` в терминале, чтобы запустить веб-интерфейс RS-StreamML.
2. Перейдите на вкладку **«Prompts»** в интерфейсе RS-StreamML.
3. Нажмите на кнопку **«Create Prompt»**.
4. Заполните информацию о промпте: имя, текст шаблона и сообщение коммита (необязательно).
5. Нажмите **«Create»**, чтобы зарегистрировать промпт.

ПРИМЕЧАНИЕ: текст шаблона промпта может содержать переменные в формате `{{variable}}`. Эти переменные будут заполняться динамическим содержимым при использовании промпта в LLM-приложении или AI-агенте. RS-StreamML также предоставляет API `to_single_brace_format()` для преобразования шаблонов в формат с одиночными фигурными скобками для фреймворков, которые требуют интерполяции именно в таком формате.

4.5.1.2 Обновление промпта

1. После создания промпта откроется страница созданного промпта. (Если вы закрыли страницу, перейдите на вкладку «Prompts» в интерфейсе RS-StreamML и нажмите на имя промпта.)
2. Нажмите на кнопку **«Create Prompt Version»**.
3. Во всплывающем диалоговом окне будет предварительно заполнен существующий текст промпта. Измените его по своему усмотрению.
4. Нажмите **«Create»**, чтобы зарегистрировать новую версию.

4.5.1.3 Сравнение версий промпта

Если у вас есть несколько версий промпта, вы можете сравнить их, чтобы понять изменения между версиями. Для сравнения версий промптов в веб-интерфейсе RS-StreamML нажмите на вкладку «**Compare**» на странице сведений о промпте.

4.5.1.4 Загрузка и использование промпта

Чтобы использовать промпт в LLM-приложении или AI-агенте, вы можете загрузить его с помощью API `RSStreamML.genai.load_prompt()` и заполнить переменные, используя метод `format()` объекта `RSStreamML.entities.Prompt`.

```
import RSStreamML
import openai

target_text = """
RS-StreamML - крупнейшая открытая платформа AI-инжиниринга для агентов и LLM.
"""

# Загрузка промпта
prompt = RSStreamML.genai.load_prompt("prompts:/summarization-prompt/2")

# Использование промпта с LLM
client = openai.OpenAI()
response = client.chat.completions.create(
    messages=[
        {
            "role": "user",
            "content": prompt.format(num_sentences=1, sentences=target_text),
        }
    ],
    model="gpt-4o-mini",
)

print(response.choices[0].message.content)
```

4.5.2 Объект Prompt

Объект **Prompt** – это центральная сущность в Реестре промптов RS-StreamML. Он представляет собой версионированный текст шаблона, который может содержать переменные для динамического содержимого.

Ключевые атрибуты объекта Prompt:

- **`name`** (имя): уникальный идентификатор промпта.
- **`template`** (шаблон): содержимое промпта, которое может быть:
 - Строкой с переменными в формате `{{variable}}` (текстовые промпты)
 - Списком словарей, представляющих сообщения чата, с ключами `'role'` и `'content'` (чат-промпты)
- **`version`** (версия): порядковый номер, представляющий ревизию промпта.
- **`commit\_message`** (сообщение коммита): описание изменений, сделанных в данной версии промпта (аналогично сообщениям коммитов в Git).
- **`tags`** (теги): необязательные пары "ключ-значение", присваиваемые версии промпта для категоризации и фильтрации. Например, вы можете добавить теги для имени проекта, языка и т.д. (применяются ко всем версиям промпта).
- **`alias`** (псевдоним): изменяемая именованная ссылка на промпт. Например, вы можете создать псевдоним `production`, указывающий на версию, используемую в вашей производственной системе.
- **`is\_text\_prompt`**: свойство, указывающее, является ли промпт текстовым (`True`) или чат-промптом (`False`).

- ``response_format`` (формат ответа): необязательное свойство, содержащее спецификацию ожидаемой структуры ответа. Может использоваться для валидации или структурирования выходных данных LLM.
- ``model_config`` (конфигурация модели): необязательный словарь, содержащий конфигурацию, специфичную для модели: имя модели, температура, `max_tokens` и другие параметры вывода.

4.5.2.1 Типы промптов

4.5.2.1.1 Текстовые промпты

Текстовые промпты используют простой строковый шаблон с переменными, заключенными в двойные фигурные скобки.

4.5.2.1.2 Чат-промпты

Чат-промпты используют список словарей сообщений, каждый с ключами `'role'` и `'content'`.

4.5.2.2 Формат ответа

Свойство ``response_format`` позволяет указать ожидаемую структуру ответов от LLM. Это может быть класс модели Pydantic или словарь, определяющий схему:

```
from pydantic import BaseModel

class SummaryResponse(BaseModel):
    summary: str
    key_points: list[str]
    word_count: int

# Или в виде словаря
response_format_dict = {
    "type": "object",
    "properties": {
        "summary": {"type": "string"},
        "key_points": {"type": "array", "items": {"type": "string"}},
        "word_count": {"type": "integer"},
    },
}
```

4.5.2.3 Управление тегами промпта и версии

RS-StreamML позволяет изменять и просматривать теги после регистрации промпта. Теги могут применяться как на уровне самого промпта, так и к отдельным версиям промпта.

```
import RSStreamML

# Операции с тегами на уровне промпта
RSStreamML.genai.set_prompt_tag("summarization-prompt", "language", "ru")
RSStreamML.genai.get_prompt_tags("summarization-prompt")
RSStreamML.genai.delete_prompt_tag("summarization-prompt", "language")

# Операции с тегами на уровне версии промпта
RSStreamML.genai.set_prompt_version_tag("summarization-prompt", 1, "author",
"alice")
RSStreamML.genai.load_prompt("prompts:/summarization-prompt/1").tags
RSStreamML.genai.delete_prompt_version_tag("summarization-prompt", 1,
"author")
```

4.5.3 Конфигурация модели

Реестр промптов RS-StreamML позволяет хранить конфигурацию модели вместе с промптами, обеспечивая воспроизводимость и ясность в отношении того, какая модель и с какими параметрами использовалась для данной версии промпта.

Это позволяет выполнять следующие действия:

- Версионировать шаблоны промптов и параметры модели совместно.
- Делиться промптами с рекомендуемыми настройками модели внутри команды.
- Воспроизводить точные конфигурации вывода из предыдущих экспериментов.
- Поддерживать разные конфигурации модели для разных версий промпта.

4.5.3.1 Базовое использование

Вы можете прикрепить конфигурацию модели к промпту, передав параметр `model_config` при регистрации:

```
import RSStreamML

# Использование словаря
model_config = {
    "model_name": "gpt-4",
    "temperature": 0.7,
    "max_tokens": 1000,
    "top_p": 0.9,
}

RSStreamML.genai.register_prompt(
    name="qa-prompt",
    template="Ответь на следующий вопрос: {{question}}",
    model_config=model_config,
    commit_message="QA промпт с конфигурацией модели",
)

# Загрузка и доступ к конфигурации модели
prompt = RSStreamML.genai.load_prompt("qa-prompt")
print(f"Модель: {prompt.model_config['model_name']}")
print(f"Температура: {prompt.model_config['temperature']}")
```

4.5.3.2 Поддерживаемые параметры конфигурации

В `PromptModelConfig` поддерживаются следующие стандартные параметры:

- ``model_name`` (str): имя или идентификатор модели (например, "gpt-4", "claude-3-opus").
- ``temperature`` (float): температура дискретизации для управления случайностью (обычно 0.0–2.0).
- ``max_tokens`` (int): максимальное количество токенов для генерации в ответе.
- ``top_p`` (float): параметр ядерной дискретизации (nucleus sampling) (обычно 0.0–1.0).
- ``top_k`` (int): параметр top-k дискретизации.
- ``frequency_penalty`` (float): штраф за частоту токенов (обычно от -2.0 до 2.0).
- ``presence_penalty`` (float): штраф за наличие токенов (обычно от -2.0 до 2.0).
- ``stop_sequences`` (list[str]): список последовательностей, которые остановят генерацию модели.
- ``extra_params`` (dict): дополнительные параметры, специфичные для провайдера или экспериментальные.

4.5.3.3 Управление конфигурацией модели

Конфигурация модели доступна для редактирования (**mutable**) и может быть обновлена после создания версии промпта. Это позволяет исправлять ошибки или изменять параметры модели без создания новых версий промпта.

4.5.3.3.1 Установка или обновление конфигурации модели

Используйте `RSStreamML.genai.set_prompt_model_config()` для установки или обновления конфигурации модели для версии промпта:

```
import RSStreamML
from RSStreamML.entities.model_registry import PromptModelConfig

# Регистрация промпта без конфигурации модели
RSStreamML.genai.register_prompt(
    name="my-prompt",
    template="Анализ: {{text}}",
)

# Позже добавляем конфигурацию модели
RSStreamML.genai.set_prompt_model_config(
    name="my-prompt",
    version=1,
    model_config={"model_name": "gpt-4", "temperature": 0.7},
)

# Или обновляем существующую конфигурацию
RSStreamML.genai.set_prompt_model_config(
    name="my-prompt",
    version=1,
    model_config={"model_name": "gpt-4-turbo", "temperature": 0.8,
"max_tokens": 2000},
)

# Проверяем обновление
prompt = RSStreamML.genai.load_prompt("my-prompt", version=1)
print(prompt.model_config)
```

4.5.3.3.2 Удаление конфигурации модели

Используйте `RSStreamML.genai.delete_prompt_model_config()` для удаления конфигурации модели из версии промпта:

```
import RSStreamML

# Удаление конфигурации модели
RSStreamML.genai.delete_prompt_model_config(name="my-prompt", version=1)

# Проверка удаления
prompt = RSStreamML.genai.load_prompt("my-prompt", version=1)
assert prompt.model_config is None
```

4.5.4 Кэширование промптов

RS-StreamML автоматически кэширует загруженные промпты в памяти для повышения производительности и сокращения количества повторных вызовов API. Поведение кэширования различается в зависимости от того, загружаете ли вы промпт по версии или по псевдониму.

4.5.4.1 Поведение кэширования по умолчанию

- **Промпты на основе версий** (например, `prompts:/summarization-prompt/1`): кэшируются с бесконечным TTL (Time To Live – время жизни) по умолчанию. Поскольку версии промптов неизменяемы, их можно безопасно кэшировать неограниченно долго.
- **Промпты на основе псевдонимов** (например, `prompts:/summarization-prompt@latest` ИЛИ `prompts:/summarization-prompt@production`): кэшируются с TTL 60 секунд по умолчанию. Псевдонимы могут со временем указывать на разные версии, поэтому более короткий TTL гарантирует, что приложение получит актуальные обновления.

4.5.4.2 Настройка кэширования

4.5.4.2.1 Управление кэшем для запроса

Вы можете управлять кэшированием для каждого запроса с помощью параметра `cache_ttl_seconds`:

```
import RSStreamML

# Пользовательский TTL: кэшировать на 5 минут
prompt = RSStreamML.genai.load_prompt("prompts:/summarization-prompt/1",
cache_ttl_seconds=300)

# Полностью обойти кэш: всегда загружать из реестра
prompt = RSStreamML.genai.load_prompt("prompts:/summarization-
prompt@production", cache_ttl_seconds=0)

# Использовать бесконечный TTL даже для промптов на основе псевдонимов
prompt = RSStreamML.genai.load_prompt(
    "prompts:/summarization-prompt@latest", cache_ttl_seconds=float("inf")
)
```

4.5.4.2.2 Глобальная настройка кэша

Вы можете установить значения TTL по умолчанию глобально с помощью переменных окружения:

```
# Установить TTL для промптов на основе псевдонимов в 5 минут
export RSSTREAMML_ALIAS_PROMPT_CACHE_TTL_SECONDS=300

# Установить TTL для промптов на основе версий в 1 час (вместо бесконечного)
export RSSTREAMML_VERSION_PROMPT_CACHE_TTL_SECONDS=3600

# Полностью отключить кэширование глобально
export RSSTREAMML_ALIAS_PROMPT_CACHE_TTL_SECONDS=0
export RSSTREAMML_VERSION_PROMPT_CACHE_TTL_SECONDS=0
```

4.5.4.3 Инвалидация кэша

Кэш автоматически инвалидируется (сбрасывается) при изменении версии промпта или псевдонима, включая следующие операции:

- `RSStreamML.genai.set_prompt_version_tag`
- `RSStreamML.genai.set_prompt_alias`
- `RSStreamML.genai.delete_prompt_version_tag`
- `RSStreamML.genai.delete_prompt_alias`

4.6 Настройки

В разделе **Настройки** вы можете задать параметры для настройки RS-StreamML.

4.6.1 Раздел «Общие»

На вкладке **Общие** вы можете настроить внешний вид системы, телеметрию и демонстрационные данные рабочего пространства.

4.6.1.1 Элементы интерфейса

Параметр	Описание	Доступные значения
Предпочтение темы	Выбор цветовой схемы интерфейса.	«Светлая», «Темная»
Включить телеметрию	Сбор анонимной статистики использования пользовательского интерфейса.	«Вкл.» / «Выкл.»
Демонстрационные данные	Кнопка для удаления всех демо-данных, сгенерированных на главной странице (эксперименты, трассы, оценки, промпты).	

4.6.1.2 Пошаговая инструкция для общих настроек

4.6.1.2.1 Изменение темы интерфейса

Для того чтобы изменить цвет интерфейса, перейдите в раздел **Общие**. Далее в поле «Предпочтение темы» выберите:

- «Светлая» – стандартная светлая тема.
- «Темная» – темная тема для работы в условиях низкой освещенности.

Тема применится сразу без перезагрузки страницы.

4.6.1.2.2 Управление телеметрией

Для того чтобы настроить телеметрию, в разделе **Общие** установите переключатель в поле «Включить телеметрию» в интересующее состояние:

- «Вкл.» – разрешить отправку анонимных данных об использовании UI.
- «Выкл.» – запретить сбор телеметрии.

4.6.1.2.3 Очистка демонстрационных данных

Для того чтобы очистить демонстрационные данные, в разделе **Общие** нажмите на кнопку «Удалить все демонстрационные данные» и подтвердите действие.

В результате демонстрационные эксперименты, трассы, оценки и промпты будут удалены.

4.6.2 Раздел «Подключения LLM»

На вкладке **Подключения LLM** вы можете создавать и управлять API-ключами для аутентификации у внешних провайдеров LLM.

4.6.2.1 Интерфейс раздела

На вкладке **Подключения LLM** отображается список подключений. Для каждой записи в списке показаны следующие сведения:

- Key name – имя ключа (например, openai_prod, anthropic_staging).
- Provider – провайдер LLM.
- Endpoints – используемые эндпоинты.

- Used by – какие модели или сервисы используют этот ключ.
- Last updated – дата последнего изменения.

На данной странице вы можете выполнить следующие действия:

- Выполнить поиск по API-ключам в поле Search.
- Отфильтровать данные по провайдеру в поле Provider.
- Настроить отображаемые колонки в таблице в поле Columns.
- Создать новый API-ключ, нажав на кнопку **Create**.
- Удалить выбранные ключи, нажав на кнопку **Delete**.

4.6.2.2 Пошаговая инструкция для подключения LLM

4.6.2.2.1 Создание API-ключа

Для того чтобы создать API-ключ, перейдите в раздел **Подключения LLM** и нажмите на кнопку **Create**.

Далее в открывшейся форме заполните следующие данные:

- В поле «Provider» из выпадающего списка выберите провайдера.
- В поле «API key name» введите уникальное имя для API-ключа.
- В поле «API Key» вставьте секретный ключ.
- В поле «API Base URL» введите кастомный базовый URL API провайдера.

После того как все поля заполнены, нажмите **Create API Key**.

Если вы хотите отменить создание API ключа и закрыть форму, нажмите **Cancel**.

4.6.2.2.2 Редактирование и удаление

Для того чтобы отредактировать запись, нажмите на строку ключа. Далее измените имя, провайдера или другие поля (кроме самого ключа).

Для того чтобы удалить запись, выберите один или несколько ключей (чекбоксы). Затем нажмите «Delete» и подтвердите удаление.

4.6.3 Раздел «Вебхуки»

На вкладке **Вебхуки** вы можете настроить получение HTTP-уведомлений, которые отправляются при наступлении определенных событий в RS-StreamML.

4.6.3.1 Пошаговая инструкция для настройки вебхуков

4.6.3.1.1 Создание вебхука

Для того чтобы создать новый вебхук, перейдите в раздел **Вебхуки** и нажмите на кнопку **Create webhook**.

Далее в открывшейся форме заполните следующие поля:

- В поле «Name» введите уникальное имя для webhook. Рекомендуется использовать осмысленное название, отражающее назначение.
- В поле «URL» укажите конечный адрес (endpoint), на который будут отправляться HTTP-запросы при наступлении событий. Адрес должен начинаться с http:// или https:// и быть доступным из сети, где запущен RS-StreamML.
- В поле «Description» при необходимости введите краткое описание назначения webhook. Данное поле не является обязательным.
- В поле «Secret» при необходимости укажите секретный ключ, который будет использоваться для HMAC-подписи входящих webhook -запросов.

- В поле «Active» установите переключатель в соответствующее положение, чтобы указать, должен ли webhook быть активным. Во включенном состоянии webhook будет отправлять уведомления; в выключенном — не будет.

- В разделе «Events» отметьте флажками те события, которые будут инициировать отправку webhook -уведомления.

После того как Вы заполнили все поля, нажмите на кнопку **Create**. В результате будет создан новый webhook с указанными параметрами.

Если Вы передумали создавать webhook, нажмите на кнопку **Cancel**. В результате форма будет закрыта без сохранения изменений.

4.6.3.1.2 Редактирование и удаление вебхука

Для того чтобы отредактировать вебхук, нажмите на существующий вебхук в списке. Далее измените параметры и сохраните.

Для того чтобы удалить вебхук, выберите интересующий вебхук в списке, нажмите на кнопку **Delete**.

4.7 Завершение работы

Для того чтобы завершить работу с программой RS-StreamML, закройте окно браузера.