

Пермский филиал федерального государственного автономного  
образовательного учреждения высшего образования  
«Национальный исследовательский университет  
«Высшая школа экономики»

Факультет социально-экономических и компьютерных наук

Ларионова Яна

**РАЗРАБОТКА ИССЛЕДОВАТЕЛЬСКОГО ПРОТОТИПА СРЕДСТВ  
РАСПОЗНАВАНИЯ ВИЗУАЛЬНЫХ МОДЕЛЕЙ**

*Выпускная квалификационная работа*

студента образовательной программы «Программная инженерия»  
по направлению подготовки 09.03.04 Программная инженерия

Доцент кафедры  
информационных технологий в  
бизнесе

---

Л. Н. Лядова

Пермь, 2025 год

## **Аннотация**

Ларионова Я.А. Разработка исследовательского прототипа средств распознавания визуальных моделей: Выпускная квалификационная работа. – Направление подготовки: 09.03.04 «Программная инженерия», бакалавриат / Ларионова Яна Андреевна – Пермь: НИУ ВШЭ – Пермь, 2025. 50 с.

Работа посвящена разработке прототипа системы распознавания визуальных моделей на изображениях диаграмм. Цель – разработка прототипа, обеспечивающего поддержку подбора эффективных типов визуализаций данных с учётом задач пользователя на основе автоматического распознавания выбранного им визуального образца.

Во «Введении» обоснована актуальность темы, сформулированы цель и задачи проекта.

В первой главе представлен анализ предметной области, классификаций визуальных моделей, онтологических структур и существующих решений в области распознавания визуализаций. Результатом анализа стало обоснование выбранного подхода и формулирование требований к системе, оформленных в виде технического задания, приведённого в приложении А.

Во второй главе представлены результаты проектирования архитектуры системы, структуры онтологии и логики взаимодействия компонентов.

В третьей главе описана реализация всех основных модулей системы – от обработки изображений до построения интерфейса и формирования пользовательских рекомендаций.

В заключении подведены итоги работы, обозначены ограничения текущей реализации и направления её дальнейшего развития.

Работа содержит 45 страниц основного текста, 4 таблицы, 25 рисунков. Библиографический список включает 23 наименования. Работа включает 1 приложение.

## Оглавление

Введение.....	4
Глава 1 Анализ предметной области.....	6
1.1 Анализ стандартов визуализации данных.....	6
1.2 Анализ онтологического подхода к классификации визуализаций .....	11
1.3 Анализ существующих решений .....	14
1.4 Исследование методов распознавания .....	16
1.4.1 Классические методы компьютерного зрения (CV) .....	16
1.4.2 Методы машинного обучения (ML) .....	17
1.4.3 Методы глубокого обучения (DL).....	17
1.5 Оценка качества распознавания визуальных моделей.....	19
1.6 Выбор технологий и инструментов .....	20
1.7 Разработка требований.....	21
1.7.1 Бизнес–требования.....	21
1.7.2 Пользовательские требования .....	22
1.7.3 Функциональные требования .....	23
1.7.4 Нефункциональные требования .....	25
1.8 Выводы .....	26
Глава 2 Проектирование системы .....	27
2.1 Архитектура системы.....	27
2.2 Проектирование структуры базы знаний .....	29
2.3 Проектирование структуры данных результата распознавания.....	31
2.4 Проектирование метрик оценки соответствия.....	31
2.5 Проектирование поведения системы.....	32
2.6 Формирование рекомендаций.....	36
2.7 Выводы .....	36
Глава 3 Реализация .....	37
3.1 Организация программной среды.....	37
3.2 Разработка модуля выделения элементов диаграмм.....	38
3.3 Сопоставление с онтологией и расчёт степени соответствия .....	39
3.4 Отображение результатов и взаимодействие с пользователем .....	40
3.5 Оценка качества модели распознавания.....	43
3.6 Выводы .....	44
Заключение.....	45
Библиографический список.....	46
ПРИЛОЖЕНИЕ А Техническое задание .....	50

## Введение

**Актуальность темы.** Современные средства визуализации данных предлагают широкий спектр различных типов диаграмм и графиков. По данным многочисленных обзоров [1, 2] и каталогов [3], существует множество видов визуальных моделей, отличающихся как способом представления информации, так и назначением и структурой данных. Это многообразие позволяет подбирать диаграммы для различных аналитических задач, однако одновременно усложняет процесс выбора для конечных пользователей.

Несмотря на существование различных рекомендаций [4 – 6] по выбору типа диаграммы в зависимости от целей анализа, типа данных и характера задачи. Тем не менее, для неопытных пользователей, не обладающих специальной подготовкой в области визуализации данных, применение этих рекомендаций остаётся затруднительным.

В связи с этим возрастает потребность в автоматизированных средствах, которые могли бы поддерживать пользователя в процессе выбора диаграмм. Системы рекомендаций типов визуализаций, а также технологии автоматического определения наиболее подходящих моделей по изображению, становятся востребованным направлением развития средств анализа и представления данных [7, 8].

**Объект и предмет исследования.** Объектом исследования является процесс автоматического распознавания диаграмм, загружаемых пользователем. Предметом исследования являются средства распознавания визуальных моделей на основе анализа их визуальных элементов.

**Цель исследования.** Исследование направлено на разработку средств распознавания типа визуальной модели по изображению с использованием онтологического описания диаграмм и предоставление пользователю рекомендаций по выбору базового типа визуализации для дальнейшей работы.

### Задачи работы

Для достижения поставленной цели были выделены следующие основные задачи:

1. Провести анализ предметной области, включая стандарты построения диаграмм, подходы к их классификации, методы автоматического

распознавания визуальных элементов, а также определить метрики оценки качества и сформулировать требования к системе.

2. Спроектировать архитектуру системы, базу знаний на основе онтологии, структуру представления распознанных элементов и алгоритмы сопоставления с онтологией, а также механизм формирования рекомендаций – списка подходящих визуализаций с указанием степени сходства с образцом, основанного на совпадении визуальных элементов и структурных признаков.
3. Реализовать программный прототип системы и провести оценку качества распознавания на тестовых данных.

В работе применяется объектно-ориентированный подход: для описания структуры и логики системы используются диаграммы UML. Знания о типах диаграмм и их элементах представлены в виде онтологии. Для распознавания элементов диаграмм используются методы компьютерного зрения и глубокого обучения на основе модели YOLOv8. Реализация системы выполнена на языке Python с использованием популярных библиотек для работы с изображениями, машинным обучением и онтологиями.

# **Глава 1 Анализ предметной области**

Для выполнения работы в первую очередь необходимо провести анализ предметной области, связанной с автоматическим распознаванием визуальных моделей. В рамках главы решаются следующие задачи:

1. Провести анализ стандартов визуализации данных с целью выявления подходящих классификаций и практик построения диаграмм, которые могут быть использованы для структурирования типов визуальных моделей.
2. Проанализировать онтологический подход к представлению знаний о диаграммах, определить ключевые сущности, свойства и отношения, которые будут использоваться при построении базы знаний.
3. Проанализировать существующие решения и научные разработки в области автоматического распознавания диаграмм, выявить их сильные и слабые стороны.
4. Исследовать методы распознавания визуальных моделей и определить наиболее подходящие для выявления элементов диаграмм на изображениях.
5. Изучить возможные метрики измерения качества распознавания для оценки точности и полноты распознавания визуальных моделей.
6. Сформулировать функциональные и нефункциональные требования к разрабатываемой системе на основе результатов анализа.
7. Обосновать выбор программных инструментов, библиотек и платформ, необходимых для реализации системы.
8. Сформулировать выводы на основе проведённого анализа, которые будут использованы при проектировании системы.

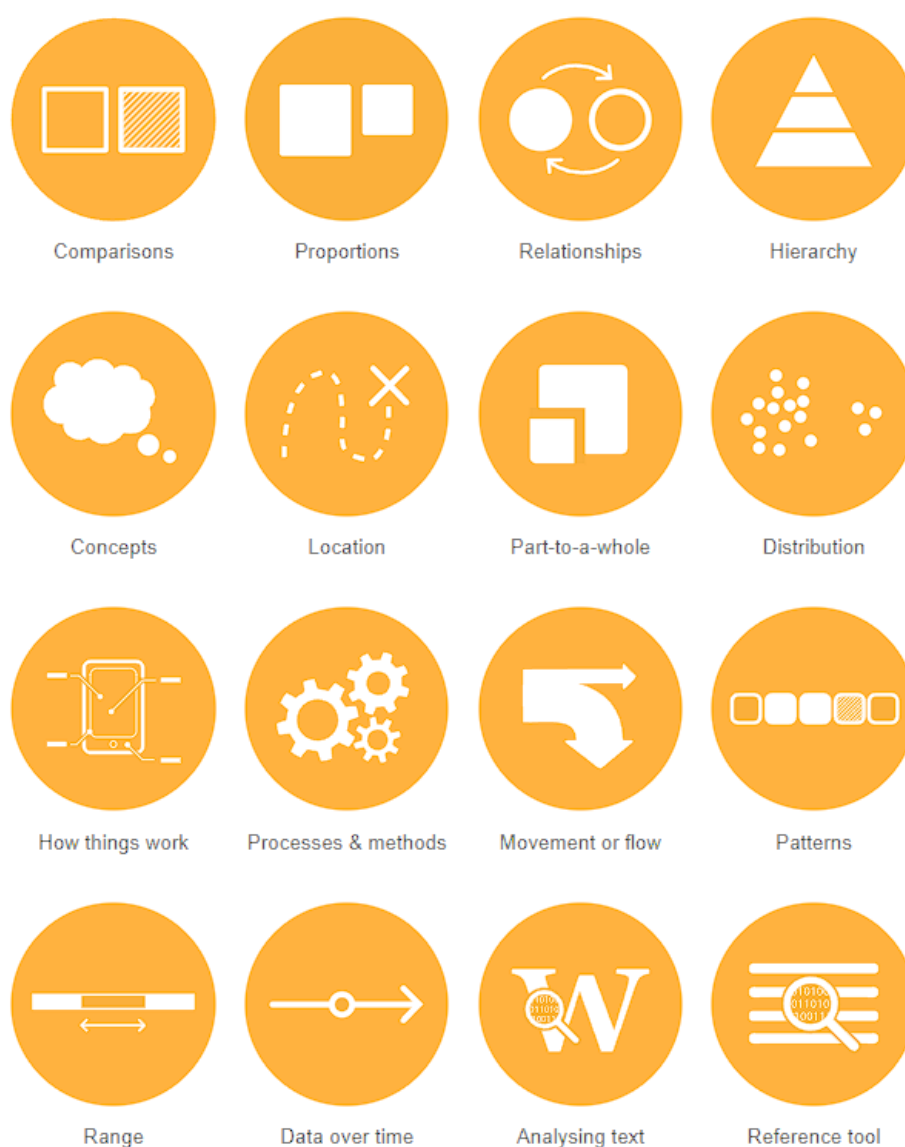
## **1.1 Анализ стандартов визуализации данных**

В современной практике визуализации данных отсутствуют строгие формальные нотации, аналогичные тем, что используются, например, в моделировании бизнес-процессов (BPMN) или в программной инженерии (UML, ER-диаграммы). Вместо этого применяются общепринятые рекомендации и лучшие

практики построения графиков и диаграмм в зависимости от типа представляемой информации и целей визуализации.

Среди существующих классификаций визуализаций данных можно выделить подход, предложенный в каталоге Data Visualization Catalogue [3], где визуальные модели систематизированы по решаемым аналитическим задачам. В рамках данной классификации выделяются 16 основных групп визуализаций, таких как сравнения, пропорции, отношения, иерархии, распределения данных и другие.

На рисунке 1 представлена визуальная схема данной классификации. Иллюстрация заимствована с официального сайта каталога [3].

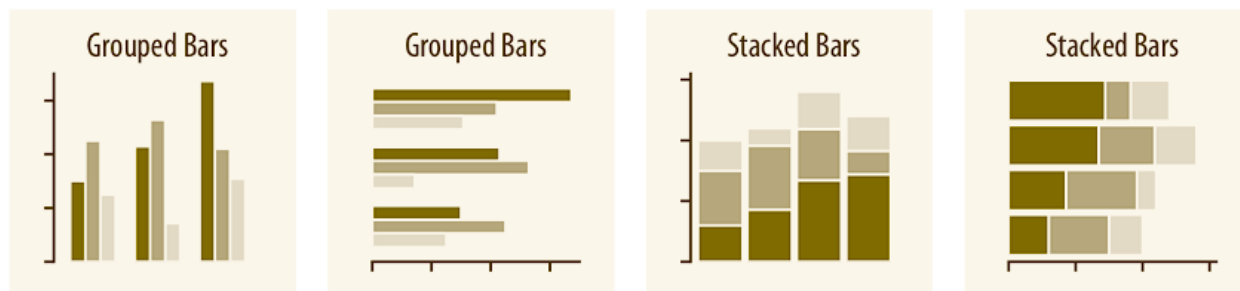


***Рисунок 1 – Классификация визуализаций по аналитическим задачам [3]***

В руководстве Claus Wilke [9] предложена классификация визуализаций, отличающаяся от структуры Data Visualization Catalogue. Автор группирует визуализации не по типу данных или аналитическим задачам, а по характеру

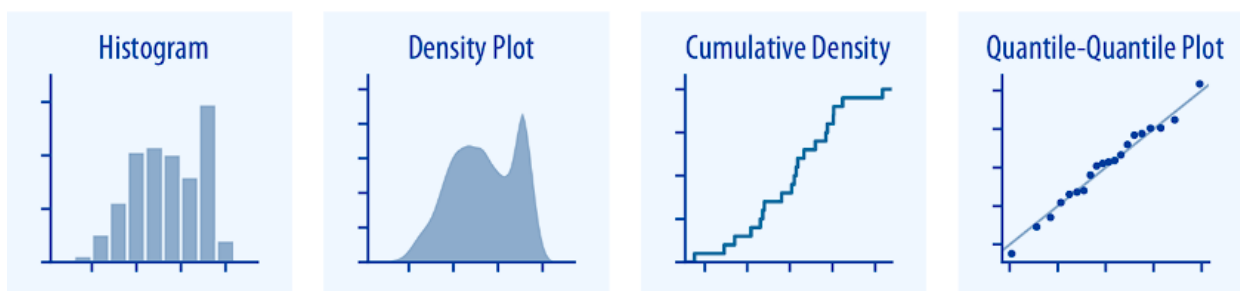
передаваемого сообщения. Такой подход акцентирует внимание на интерпретации данных аудиторией, а не на их структуре.

Категория «Суммы» включает визуализации, предназначенные для отображения совокупных значений. Пример таких визуализаций представлен на рисунке 2.



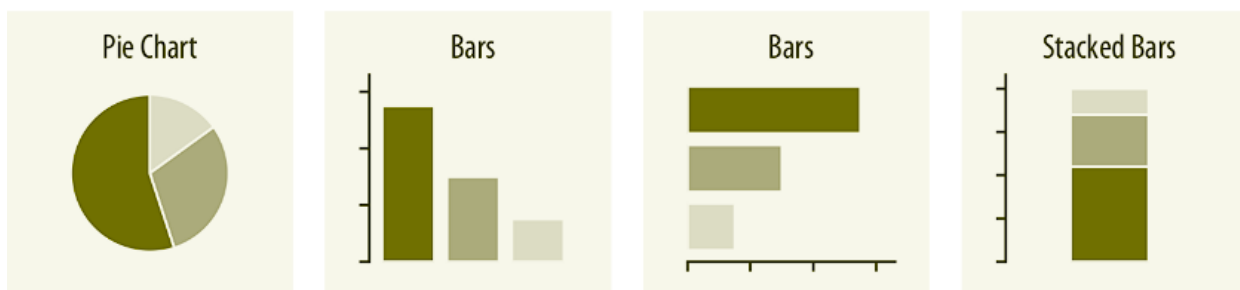
**Рисунок 2 – Примеры визуализаций категории «Суммы» [9]**

Категория «Распределения» охватывает визуализации, показывающие, как значения распределены по диапазону. Рисунок 3 иллюстрирует визуализации, относящиеся к данной категории.



**Рисунок 3 – Примеры визуализаций категории «Распределения» [9]**

Категория «Пропорции» включает круговые и кольцевые диаграммы, treemap и stacked bar, позволяющие сравнивать доли частей от целого. Примеры визуализаций данной категории показаны на рисунке 4.

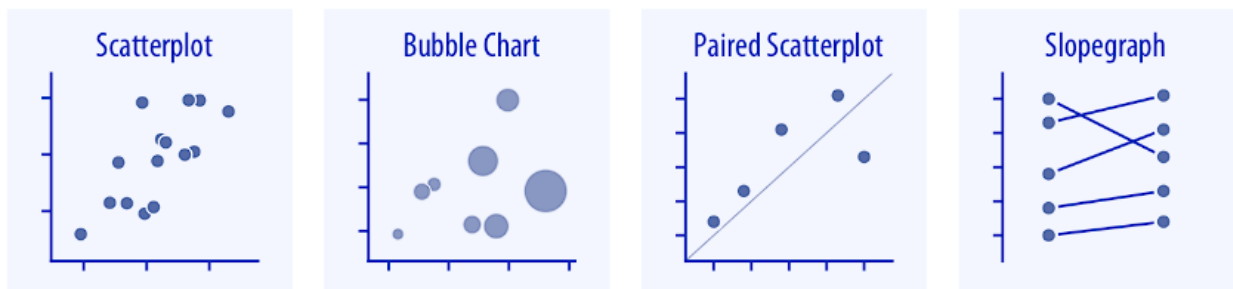


**Рисунок 4 – Примеры визуализаций категории «Пропорции» [9]**

Категория «Соотношения  $X - Y$ » объединяет визуализации для анализа взаимосвязей между переменными. Здесь основное внимание уделяется тому, как

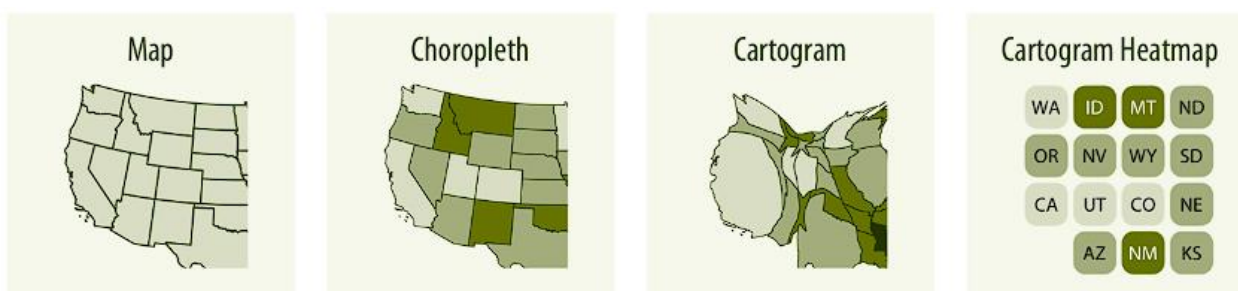


значения одной переменной изменяются в зависимости от другой. Рисунок 5 иллюстрирует визуализации, относящиеся к данной категории.



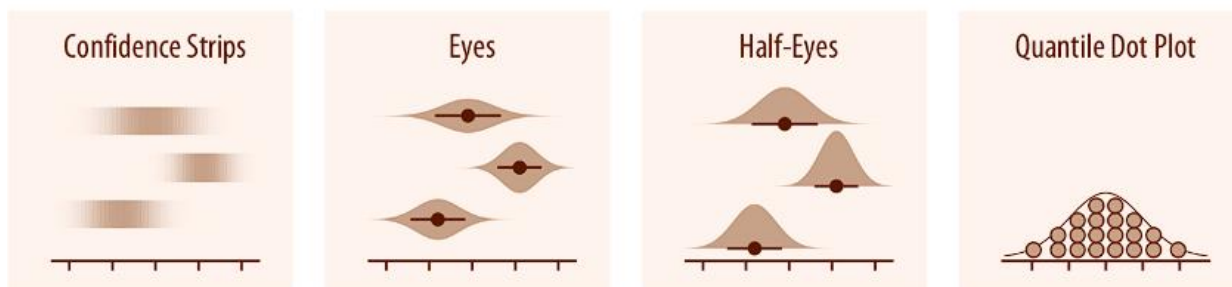
**Рисунок 5 – Примеры визуализаций категории «Соотношения X-Y» [9]**

Категория геопространственных визуализаций включает способы отображения данных, связанных с положением на карте или в пространстве. Ключевым принципом является соответствие координатной привязке и пространственным отношениям между объектами. Примеры таких визуализаций представлены на рисунке 6.



**Рисунок 6 – Примеры визуализаций категории «Геопространственные данные» [9]**

Визуализации из категории «Неопределённость» позволяют передать вариативность, возможные отклонения или границы доверия, помогая лучше интерпретировать неточные или прогнозные значения. Примеры визуализаций этой категории представлены на рисунке 7.



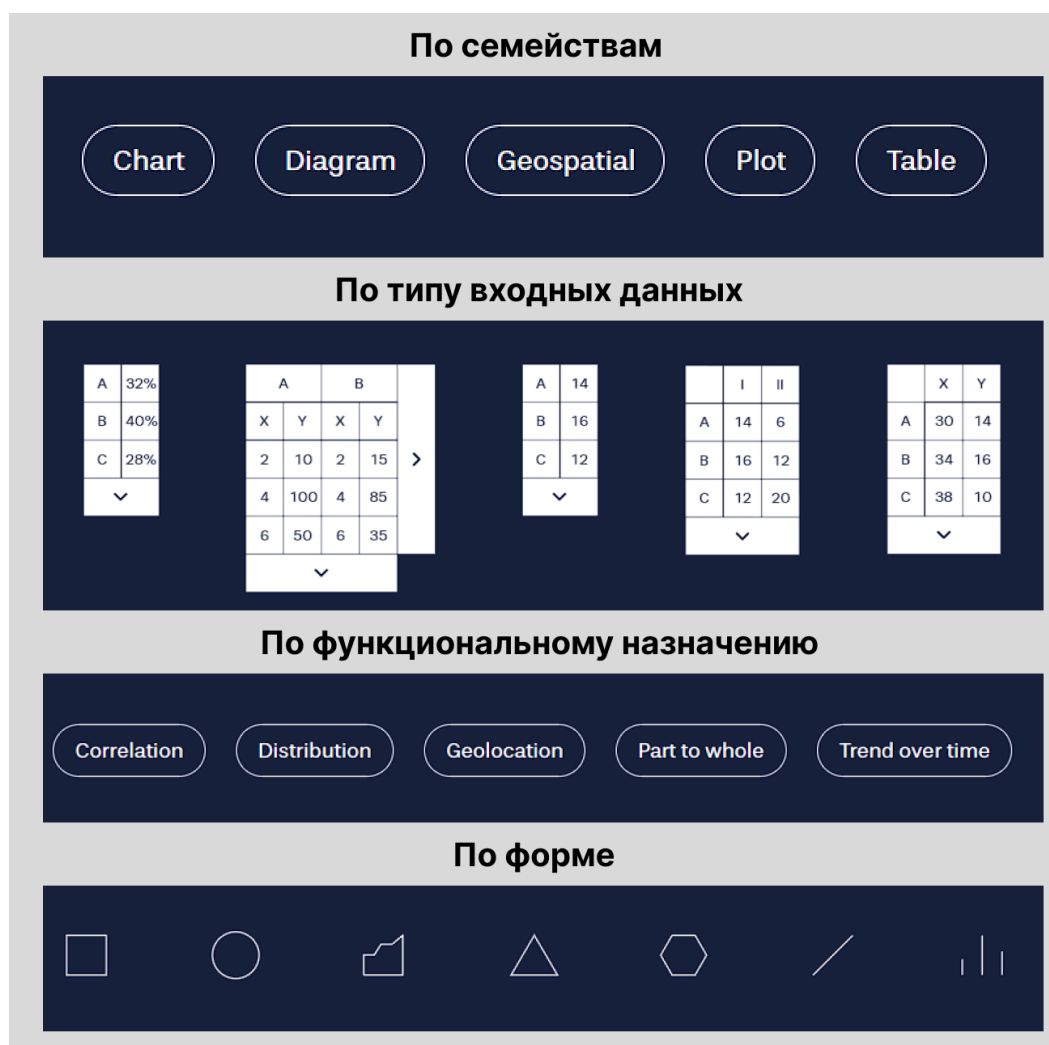
**Рисунок 7 – Примеры визуализаций категории «Неопределённость» [9]**

Ещё одним примером классификации визуализаций данных является проект Data Viz Project [10]. Данный каталог представляет собой обширную базу различных

типов визуальных представлений, которые сгруппированы по нескольким независимым признакам:

- По функциональному назначению (Function).
- По семействам визуализаций (Family).
- По типу входных данных (Input).
- По форме элементов визуализации (Shape).

Такой подход обеспечивает многомерную навигацию по типам визуализаций и позволяет выбирать наиболее подходящую диаграмму как по назначению, так и по форме представления данных. На рисунке 8 представлена многокритериальная схема, иллюстрирующая классификацию визуализаций по этим параметрам.



*Рисунок 8 – Многокритериальная классификация визуализаций [10]*

В ходе анализа существующих подходов к классификации визуализаций данных было выявлено, что различные источники предлагают систематизировать визуальные модели по назначению, типу данных или форме представления информации. Несмотря на ценность этих классификаций, для целей

автоматизированного распознавания диаграмм требуется чёткая формализованная структура с явно выделенными категориями типов диаграмм.

В качестве основы для онтологической классификации визуализаций в данной работе использована структура, предложенная в исследовании [11], опирающемся на подход Энди Кирка. Согласно данной классификации, визуализации делятся на пять основных групп в зависимости от цели и характера представляемых данных:

- сравнение категорий;
- части и целое;
- временные изменения;
- связи и отношения;
- геопространственные данные.

Данная схема легла в основу построения онтологии, отражающей функциональные особенности визуальных моделей.

## **1.2 Анализ онтологического подхода к классификации визуализаций**

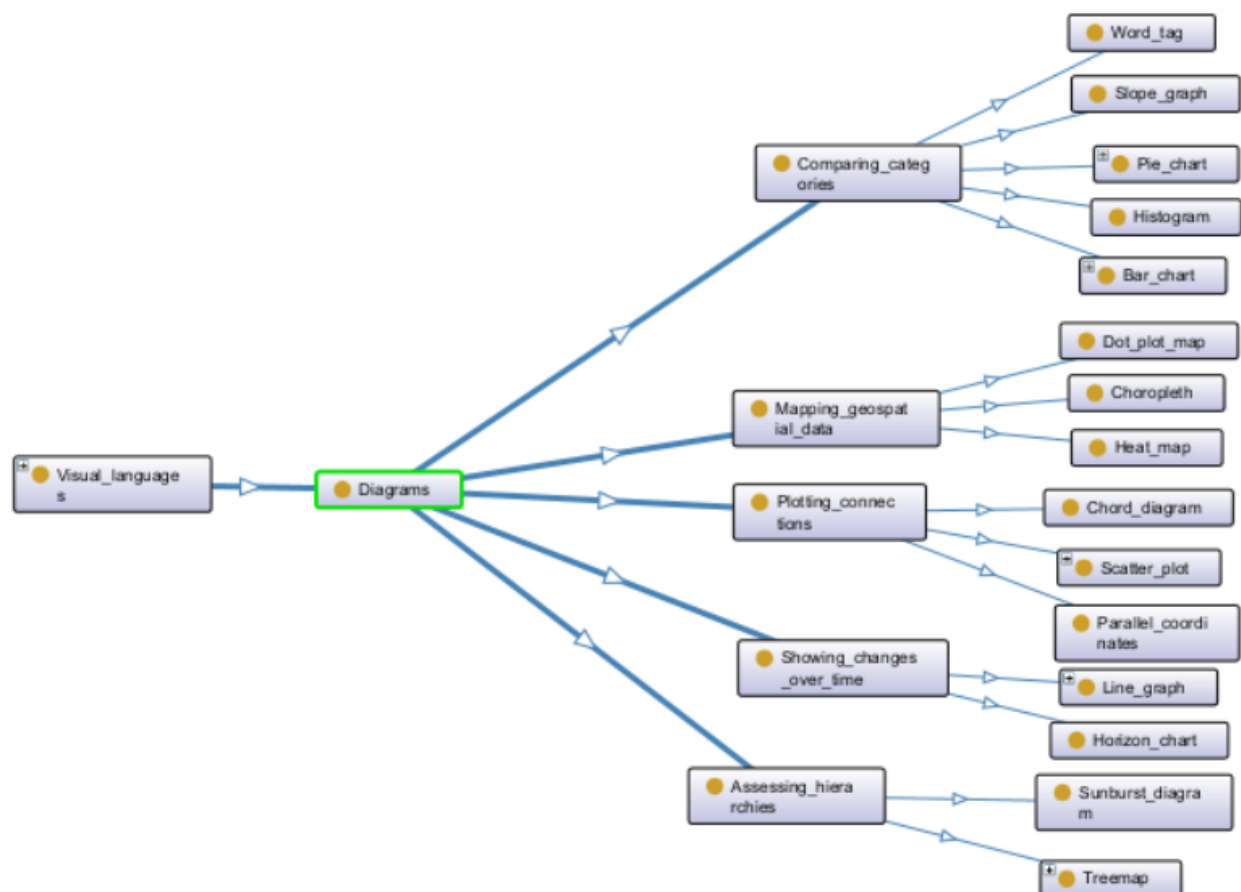
Для формализации знаний о визуальных моделях данных в данной работе рассматривается онтологический подход, основанный на классификации диаграмм по их функциональному назначению.

В качестве основы выбрана структура, представленная в исследовании [11], где визуализации систематизируются по следующим задачам:

- Comparing categories – сравнение категорий.
- Showing changes over time – отображение изменений во времени.
- Assessing hierarchies – визуализация иерархических структур.
- Plotting connections – отображение связей между объектами.
- Mapping geospatial data – визуализация географических данных.

Каждая из указанных категорий включает конкретные типы диаграмм, которые служат для решения соответствующих аналитических задач. Данная структура используется как основа иерархии классов визуализаций в онтологической модели. Структура онтологии приведена на рисунке 9.

Помимо самих типов визуализаций, в модели представлены визуальные элементы, характерные для диаграмм различных типов. Элементы объединены в класс `Diagram_element` и включают как абстрактные компоненты визуализации (например, оси, сетки, легенды), так и графические фигуры.



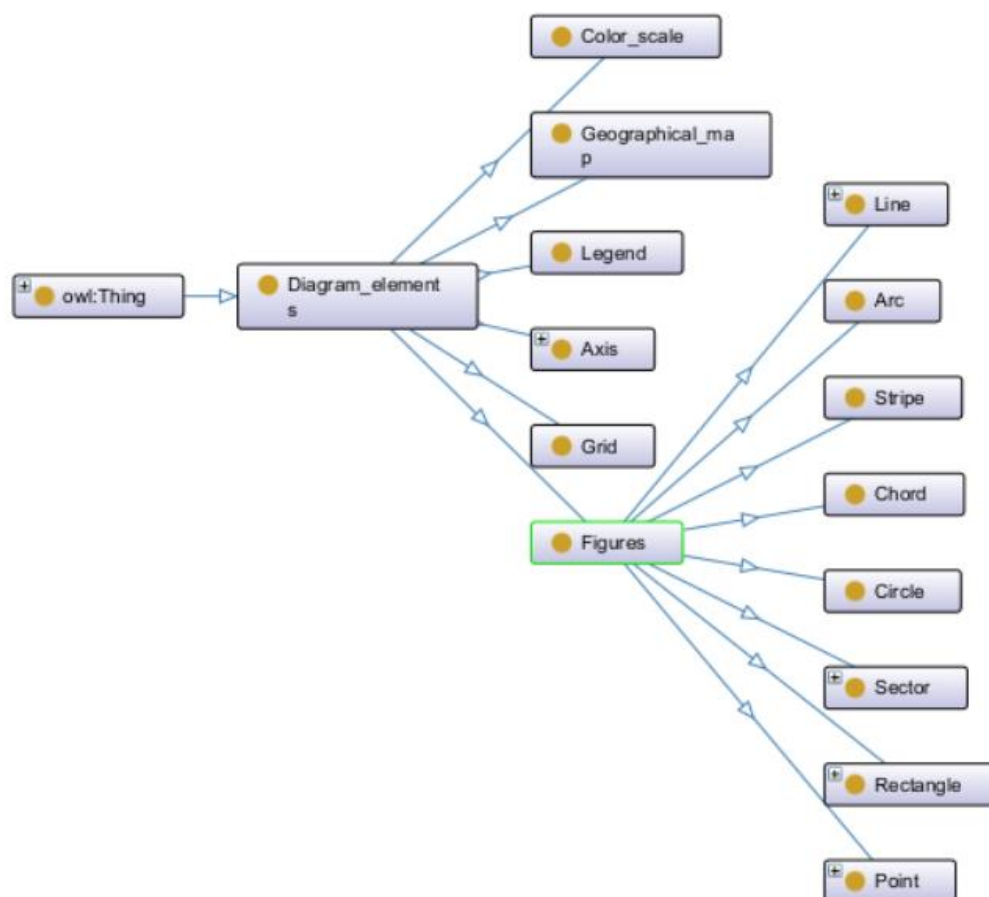
**Рисунок 9 – Иерархия классов диаграмм в онтологии**

К классу Diagram\_element отнесены следующие элементы:

- Axis – используется для представления координатных шкал.
- Grid – применяется для отображения фоновых вспомогательных линий.
- Legend – используется для пояснения значений визуальных элементов.
- Color\_scale – предназначена для визуального кодирования числовых значений цветом.
- Geographical\_map – применяется для визуализации пространственных данных.
- Figures, включающие:
  - Rectangle – элемент, характерный для гистограмм и Treemap.
  - Line – используется в линейных графиках.
  - Point – характерна для Scatter Plot.
  - Circle – применяется в различных круговых визуализациях.
  - Chord – используется в диаграммах связей.
  - Stripe – используется для отображения распределённых величин.
  - Arc – применяется в визуализациях дуговых связей.

- Sector – характерен для круговых диаграмм.

Структура классов элементов визуализаций представлена на рисунке 10.



**Рисунок 10 – Иерархия классов элементов диаграмм в онтологии**

Выбор онтологического подхода обусловлен необходимостью строгой формализации знаний о типах диаграмм и их элементах. Онтология позволяет:

- формально определить сущности (типы диаграмм, элементы визуальных моделей);
- явно задать связи между типами диаграмм и их составными частями;
- обеспечить расширяемость модели при добавлении новых типов визуализаций;
- использовать структуру онтологии для построения правил сопоставления распознанных элементов с типами диаграмм.

Поскольку в рамках проекта реализуется прототип системы распознавания диаграмм, из всей структуры онтологии были выбраны только основные типы визуализаций, соответствующие наиболее распространённым видам диаграмм.

В частности, в проекте рассматриваются следующие типы диаграмм:

- Bar Chart.

- Pie Chart.
- Scatter Plot.
- Line Graph.
- Treemap.

В качестве элементов, по которым определяется структура диаграммы и осуществляется сопоставление с онтологией, были выбраны следующие визуальные объекты:

- Axis – оси координат.
- Line – линии.
- Point – отдельные точки.
- Rectangle – прямоугольники (например, столбцы).
- Sector – сектора (например, в круговых диаграммах).
- Legend – легенда диаграммы (обозначения категорий).

Выбор указанных типов диаграмм и визуальных элементов обусловлен их широкой распространённостью, разнообразием графических характеристик и возможностью стабильного распознавания на изображениях. Эти особенности делают их удобной основой для построения онтологии и расчёта степени соответствия визуальной структуры диаграммы базовым типам визуализации.

### 1.3 Анализ существующих решений

Для оценки существующих решений были выбраны следующие критерии:

1. **Выделение ключевых элементов диаграммы:** способность точно распознавать основные компоненты визуальной модели – оси, графические фигуры (прямоугольники, линии, точки, сектора), текстовые подписи.
2. **Оценка степени соответствия базовым типам диаграмм:** наличие механизма вычисления степени близости между распознанной структурой и описаниями существующих типов визуализаций, что позволяет избежать жёсткой бинарной классификации.
3. **Расширяемость и модифицируемость:** возможность дополнения системы новыми типами диаграмм, элементами или признаками без кардинальной переработки модели.

4. **Качество обработки изображений:** устойчивость методов к различиям в качестве входных изображений: различным форматам диаграмм, зашумлённым или искажённым данным.

5. **Производительность и масштабируемость:** эффективность системы при обработке изображений различного размера и сложности в разумное время.

ChartEye представляет собой фреймворк, основанный на глубоком обучении, для извлечения информации из диаграмм [12]. В модели применяются иерархические визуальные трансформеры для классификации типа диаграммы и роли текста, а также YOLOv7 для обнаружения текстовых элементов. Для повышения качества распознавания используется Super Resolution GAN. К основным особенностям решения можно отнести: выделение ключевых элементов диаграммы (включая текстовые и графические объекты), структурированное представление извлечённой информации, а также высокую точность классификации типов диаграмм (F1-оценка 0.97). Ограничением является отсутствие механизма расчёта степени соответствия распознанной диаграммы формальным моделям или онтологии.

ChartDETR предлагает трансформерную архитектуру для локализации и реконструкции различных графических форм в диаграммах [13]. Система работает за счёт предсказания координат ключевых точек с последующим построением элементов диаграммы. Среди достоинств: выделение множества графических элементов разных форм, высокая точность распознавания, возможность масштабирования на сложные структуры. Ограничением является отсутствие обработки текстовых данных и отсутствующая поддержка формального сопоставления с онтологиями.

ChartOCR представляет собой гибридный подход, сочетающий методы глубокого обучения и алгоритмы на основе правил для извлечения данных из изображений диаграмм [14]. Система ориентирована на выделение ключевых точек и графических объектов с целью упрощения извлечения данных вне зависимости от стиля диаграммы. Основные преимущества включают универсальность подхода и возможность последующей структуризации данных. К ограничениям относится

недостаточная поддержка формализованного представления информации в контексте онтологических моделей.

Swin-Chart использует архитектуру Swin-трансформеров для классификации типов диаграмм [15]. Модель демонстрирует высокую способность к обобщению на различных наборах данных и обеспечивает эффективную классификацию базовых типов диаграмм. Среди преимуществ – высокая скорость обработки и устойчивость к различиям во входных изображениях. Основным ограничением является отсутствие выделения отдельных элементов диаграммы и невозможность вычисления степени соответствия формализованным моделям.

В соответствии с сформулированными критериями был проведён анализ существующих решений в области автоматизированного распознавания диаграмм. Сравнение существующих решений приведено в таблице 1.

Таблица 1 – Сравнение существующих решений

Критерий	ChartEye	ChartDETR	ChartOCR	Swin-Chart
Выделение элементов	Да	Да	Да	Нет
Оценка степени соответствия	Нет	Нет	Нет	Нет
Расширяемость	Средняя	Высокая	Средняя	Средняя
Качество обработки	Высокое	Высокое	Среднее	Высокое
Производительность	Средняя	Высокая	Средняя	Высокая

Наиболее высокую точность выделения элементов демонстрируют ChartEye и ChartDETR, однако механизмы оценки степени соответствия базовым моделям диаграмм отсутствуют.

## 1.4 Исследование методов распознавания

Методы распознавания визуальных моделей данных можно классифицировать по используемым технологиям обработки изображений и анализа структуры объектов. В данном разделе приведено описание ключевых направлений и их применимости к решаемым задачам.

### 1.4.1 Классические методы компьютерного зрения (CV)



**Алгоритм выделения секторов круговых диаграмм.** Авторы исследования [16] разработали метод для 2D и 3D круговых диаграмм, основанный на обработке изображений: удаление шумов, преобразование в полутон, пороговая бинаризация и подсчёт пикселей в сегментах. Метод эффективно находит границы секторов круговой диаграммы, вычисляя их угловые размеры по распределению пикселей; осей как таковых нет, применяется только к круговым диаграммам. По бинарному изображению метод определяет величины (проценты) каждого сектора, но не работает с текстовыми метками – предполагается, что цвета сегментов заранее сопоставлены с легендой вручную. Подходит только для круговых диаграмм; очень высока точность для 2D/3D круговых (97% на синтетических данных), но не извлекает текст и не применим к линиям или точкам.

В работе [17] рассмотрен метод локализации контуров объектов различных масштабов на изображениях с помощью преобразования Хафа. Подход позволяет эффективно выявлять геометрические структуры, такие как прямые линии и окружности, что потенциально может быть использовано для уточнения расположения осей, столбцов и секторов на диаграммах после первичного детектирования элементов. Однако преобразование Хафа само по себе не решает задачу определения типа диаграммы, может давать ложные срабатывания на зашумленных изображениях и требует тщательной настройки параметров.

#### **1.4.2 Методы машинного обучения (ML)**

В работе [18] Savva и соавторов предложена система ReVision для автоматизированной классификации изображений диаграмм на основе традиционных методов машинного обучения. Классификация выполняется с использованием набора вручную извлечённых признаков (геометрических, текстурных и текстовых) и обучения модели SVM. Такой подход позволяет решать задачу определения типа диаграммы (столбчатая, круговая, линейная и др.) без применения методов глубокого обучения. Основным ограничением метода является зависимость от качества предобработки изображений и сложности настройки признаков при работе с сильно зашумлёнными или нестандартными диаграммами.

#### **1.4.3 Методы глубокого обучения (DL)**

В работе [19] тип диаграммы классифицируется напрямую по всему изображению с помощью предобученных сверточных нейронных сетей (CNN) в

парадигме Transfer Learning. Несмотря на высокую точность классификации на ограниченном наборе типов диаграмм, данный подход требует значительных вычислительных ресурсов, плохо масштабируется на новые типы без переобучения и имеет низкую интерпретируемость результатов.

В работе [20] представлен обзор существующих методов классификации типов диаграмм, включая подходы на основе машинного обучения, сверточных нейронных сетей (CNN) и Vision Transformers. Авторы выделяют, что применение CNN, таких как AlexNet, ResNet и VGG, позволяет автоматически извлекать визуальные признаки и эффективно определять тип диаграммы даже при наличии шумов и вариаций в стилях. В экспериментальной части исследования использование Swin Transformer привело к достижению F1-метрики около 93% на реальных диаграммах из датасета UB-PMC. Применение методов на основе сверточных нейронных сетей является перспективным направлением для задач автоматического распознавания визуальных моделей и может быть использовано для решения аналогичных задач в рамках настоящего проекта.

Говоря о CNN для задач детектирования элементов визуальных моделей, дополнительно обратимся к работе [21], где рассмотрено использование алгоритма YOLOv5 для выделения объектов на изображениях. Авторы отмечают, что модель обеспечивает высокую скорость и точность обнаружения объектов по сравнению с классическими сверточными сетями, что особенно важно при обработке реальных данных с возможными искажениями. В контексте поставленных задач автоматического распознавания диаграмм использование подобных методов позволяет надёжно выделять ключевые элементы, такие как оси, прямоугольники и линии.

Проведённый обзор методов распознавания визуальных моделей позволяет выделить три основных подхода: классические методы компьютерного зрения (CV), методы машинного обучения (ML) и методы глубокого обучения (DL). Каждый из них обладает своими преимуществами и ограничениями в контексте задач автоматического анализа диаграмм.

Классические методы обеспечивают высокую точность в узких сценариях (например, при распознавании секторов круговой диаграммы), но не масштабируются на разнообразные типы диаграмм и не учитывают текстовую

информацию. Методы машинного обучения более универсальны, но требуют ручной настройки признаков и чувствительны к качеству входных изображений. Наиболее перспективными являются подходы на основе глубокого обучения, позволяющие автоматически извлекать визуальные признаки и эффективно классифицировать изображения диаграмм. Особенно актуальны методы object detection (например, YOLO), позволяющие не только определить тип диаграммы, но и локализовать ключевые элементы визуальных моделей. В таблице 2 приведено сравнительное обобщение рассмотренных подходов.

Таблица 2 – Сравнение методов распознавания визуализаций.

Метод	Точность	Скорость	Область применимости	Ограничения
CV (выделение секторов, Хафа)	Высокая (до 97%, узкие случаи)	Высокая	Круговые диаграммы, геометрия	Не работает с текстом, плохо масштабируется
ML (ReVision, SVM)	Средняя (зависит от признаков)	Средняя	Классификация диаграмм	Требуется ручной настройки, чувствителен к шуму
DL (CNN, Transfer Learning)	Высокая (до 93% F1)	Низкая – средняя	Классификация и распознавание элементов	Требуется датасета, неинтерпретируемо, ресурсоёмко
DL (YOLO, object detection)	Высокая + локализация	Высокая	Выделение элементов (оси, фигуры, подписи)	Не классифицирует диаграммы целиком, нужен обученный датасет

В рамках проекта выбран подход на основе object detection с использованием YOLOv8, поскольку он обеспечивает баланс между точностью, скоростью и универсальностью при работе с изображениями диаграмм. Это делает возможным автоматическое сопоставление найденных элементов с формализованными типами диаграмм в онтологии.

## 1.5 Оценка качества распознавания визуальных моделей

Для объективного анализа эффективности методов распознавания типов диаграмм и их элементов необходимо определить подходящие метрики качества и рассмотреть существующие подходы к их применению.

На этапе классификации авторы работы [19] используют предобученные CNN-архитектуры, добившись Accurasy свыше 95% по 13 классам типов диаграмм. Для обнаружения диаграмм (и их элементов) в документах применяются детекторы объектов одного и двух этапов (RetinaNet, Faster R-CNN). Оценка модели детекции производится по метрикам, принятым в конкурсе MS COCO [22] – вычисляются Average Precision (AP) и Average Recall (AR) для каждого класса при различных порогах IoU, после чего показатели усредняются. Метрика AP учитывает соотношение precision/recall и требует правильного класса и достаточного перекрытия (IoU) предсказанной рамки с эталонной.

Для повышения точности оценки качества распознавания может быть целесообразно вводить специальные метрики для отдельных типов диаграмм вместо прямого использования стандартного mAP, как это сделано в работе [14]. Например, для линейных графиков определяется точность и полнота совпадения предсказанных точек линии с эталонными точками, на основе которых вычисляется F1-score. Для столбчатых диаграмм предлагается метрика на основе соответствия рамок столбцов (модифицированная мера, учитывающая перекрытие по X, Y и высоте столбца).

Таким образом, для оценки качества распознавания визуальных моделей целесообразно применять не только стандартные метрики классификации и детекции (Accurasy, Precision, Recall, IoU, F1/F1-score, mAP), но и разрабатывать специализированные показатели, учитывающие особенности структуры различных типов диаграмм.

## **1.6 Выбор технологий и инструментов**

На основании проведённого анализа были выбраны технологии, обеспечивающие реализацию ключевых компонентов системы: детектирования визуальных элементов, построения пользовательского интерфейса, аннотирования данных и формального описания онтологии. В таблице 3 представлены выбранные инструменты, соответствующие задачи и краткая характеристика их применимости.

Таблица 3 – Обзор выбранных инструментов

Технология / инструмент	Решаемые задачи	Оценка применимости
Python	Язык разработки, интеграция компонентов, работа с онтологиями и ML-библиотеками	Высокая: универсальный, широко поддерживается, легко интегрируем
YOLOv8	Обнаружение элементов на изображениях диаграмм	Высокая: высокая точность, скорость, актуальная модель
Gradio	Реализация пользовательского интерфейса прототипа	Средняя: простота настройки, интеграция с ML-моделями, но ограниченная поддержка сложных интерактивных элементов
Roboflow	Разметка изображений и генерация датасета для обучения модели YOLO	Средняя: удобно, но требует регистрации, зависит от облачного доступа
Protégé	Разработка онтологии визуализаций в формате OWL	Высокая: удобный графический интерфейс, поддержка RDF/XML
RDFLib	Работа с онтологиями в коде, выполнение SPARQL-запросов	Высокая: библиотека на Python, интеграция с Protégé и OWL

Выбор указанных инструментов обусловлен их соответствием задачам проекта, возможностью быстрой интеграции и активной поддержкой в профессиональном сообществе. Все компоненты поддерживают формат данных и архитектуру, необходимую для построения прототипа системы автоматического распознавания диаграмм и его демонстрации в рамках выпускной квалификационной работы.

## 1.7 Разработка требований

### 1.7.1 Бизнес-требования

Проектируемая система направлена на решение проблемы выбора типа диаграммы для визуализации данных пользователями, не обладающими специальной подготовкой в этой области. Система должна предоставлять пользователю возможность по загруженному изображению определить тип визуальной модели и получить рекомендации по наиболее подходящим базовым типам диаграмм для дальнейшего использования.

Основные бизнес-требования проекта:

- Повысить доступность средств визуализации данных для широкого круга пользователей.

- Упростить процесс выбора подходящего типа диаграммы для представления данных.
- Снизить количество ошибок при самостоятельной попытке построения визуализации без понимания её структуры.

### 1.7.2 Пользовательские требования

Система должна обеспечивать следующие возможности для пользователя:

- Загрузка изображения с изображением диаграммы для анализа.
- Автоматическое определение типа визуализации, представленной на изображении.
- Отображение сопоставленного типа диаграммы и степени уверенности в распознавании.
- Предоставление рекомендаций по базовым типам диаграмм, близким к распознанной.
- Возможность повторной загрузки нового изображения без перезапуска страницы.

На рисунке 11 показана диаграмма вариантов использования, отражающая основные пользовательские требования.



**Рисунок 11 – Диаграмма use case для процесса получения рекомендаций по визуализации**

Ниже приведено описание основного прецедента, отражённого на диаграмме вариантов использования.

**Актор:** пользователь.

**Цель:** получить рекомендации по базовому типу диаграммы на основе структуры загруженного изображения, с возможностью ручной настройки значимости визуальных элементов.

**Основной поток событий:**

1. Пользователь загружает изображение диаграммы через интерфейс системы.
2. Система автоматически:
  - выполняет распознавание визуальных элементов;
  - извлекает значения связей между элементами и типами диаграмм из онтологии;
  - рассчитывает степень соответствия на основе весов;
  - выводит изображение с выделенными элементами (bounding boxes) и список подходящих базовых типов визуализаций.

**Альтернативный поток:**

- 2а. Пользователь изменяет веса элементов.
- 2б. Пользователь инициирует пересчёт.
- 2с. Система обновляет рекомендации.

**Результат сценария:**

Пользователь получает обоснованные рекомендации по подходящим типам визуализаций, соответствующим структуре исходного изображения. При необходимости результат может быть адаптирован с учётом индивидуальных приоритетов за счёт настройки весов.

### **1.7.3 Функциональные требования**

Функциональные требования сгруппированы по основным функциям системы.

1. Работа с изображениями:
  - Система должна позволять пользователю загружать изображение диаграммы в наиболее распространённых форматах (JPEG, PNG).

- Система должна обрабатывать изображения размером до 5 МБ, что обусловлено необходимостью поддержания высокой скорости обработки.

## 2. Выделение элементов диаграммы:

- Система должна автоматически выделять на изображении ключевые элементы:
  - прямоугольники;
  - оси координат;
  - линии;
  - точки;
  - секторы.
- Для выделения элементов должна использоваться модель глубокого обучения на основе сверточных нейронных сетей (YOLOv8).

## 3. Определение типа диаграммы:

- Система должна сопоставлять выделенные элементы с базой знаний (онтологией типов диаграмм).
- Система должна рассчитывать степень соответствия выделенных элементов каждому типу диаграммы на основе заранее определённых признаков и метрик.
- Система должна выводить список наиболее вероятных типов диаграмм с указанием степени уверенности (в процентах) для каждого типа. Результаты следует упорядочить по убыванию уверенности.

## 4. Формирование рекомендаций:

- Система должна предлагать пользователю список базовых типов диаграмм, наиболее близких к распознанной модели.
- Рекомендации должны быть основаны на анализе категории диаграммы в онтологии.

## 5. Ручная настройка весов элементов:

- Система должна предоставлять пользователю возможность изменять значения весов значимости визуальных элементов через таблицу в пользовательском интерфейсе.



- После изменения весов система должна выполнять пересчёт степени соответствия и обновлять рекомендации.

#### **1.7.4 Нефункциональные требования**

##### **1. Производительность**

- Время обработки одного изображения не должно превышать 10 секунд [23], поскольку это предельное время отклика, при котором пользователь сохраняет внимание и контроль над процессом.
- Производительность системы была протестирована в среде, сопоставимой с рекомендуемой конфигурацией (процессор не ниже Intel Core i5 8-го поколения или аналог, не менее 8 ГБ оперативной памяти).

##### **2. Надежность**

- Система должна корректно обрабатывать ошибки загрузки некорректных файлов (например, неподдерживаемый формат, повреждённое изображение).
- При невозможности определить тип диаграммы система должна выдавать соответствующее уведомление пользователю.

##### **3. Совместимость**

- Система должна быть совместима с современными веб-браузерами (Google Chrome, Microsoft Edge, Mozilla Firefox, Safari).
- Для реализации системы должна использоваться среда Python 3.10 и выше (для высокой совместимости используемых библиотек).

##### **4. Масштабируемость**

- Архитектура базы знаний (онтологии) и формат SPARQL-запросов должны обеспечивать возможность добавления новых типов диаграмм без необходимости изменения логики работы системы.
- Алгоритмы сопоставления должны автоматически учитывать новые типы, если они корректно оформлены в онтологии с указанием связей и весов.

##### **5. Удобство использования**

- Взаимодействие с системой должно быть возможно через интуитивно понятный веб-интерфейс без необходимости использования командной

строки.

- Все ключевые действия (загрузка изображения, просмотр результата, редактирование весов, повторный анализ) должны быть доступны на одной странице интерфейса.
- Система должна исключать возможность критических ошибок со стороны пользователя, обеспечивая валидацию загружаемых и редактируемых данных.

## **1.8 Выводы**

В ходе проведённого анализа были исследованы существующие классификации визуальных моделей данных, рассмотрены стандарты и практики их построения, а также описана онтологическая структура типов диаграмм, выбранная в качестве основы для проектирования системы. Проанализированы современные решения в области автоматизированного распознавания визуальных моделей и сформулированы требования к разрабатываемой системе. Были рассмотрены методы компьютерного зрения, машинного и глубокого обучения для выделения и классификации элементов диаграмм, а также подходы к оценке качества распознавания.

На основе полученных результатов сформировано техническое задание на разработку системы автоматического распознавания типа диаграммы и выдачи рекомендаций пользователю. Полный текст технического задания приведён в Приложении А.

## Глава 2 Проектирование системы

Основными задачами проектирования являются:

1. Разработать архитектуру системы, включающую основные модули и интерфейсы их взаимодействия.
2. Спроектировать структуру базы знаний в виде онтологии, отражающей связи между типами диаграмм и визуальными элементами.
3. Определить структуру представления результата распознавания.
4. Разработать формальные метрики для оценки степени соответствия между набором распознанных визуальных элементов и описаниями типовых диаграмм, представленными в онтологии.
5. Описать поведение системы с использованием поведенческих диаграмм (активностей и последовательностей) для визуализации логики работы и взаимодействия модулей.
6. Определить логику формирования рекомендаций на основе рассчитанных показателей соответствия.

Также предусматривается использование веб-интерфейса на базе Gradio для взаимодействия пользователя с системой. Проработка интерфейса выполняется на этапе программной реализации. Результаты проектирования послужат основой для последующей реализации системы.

### 2.1 Архитектура системы

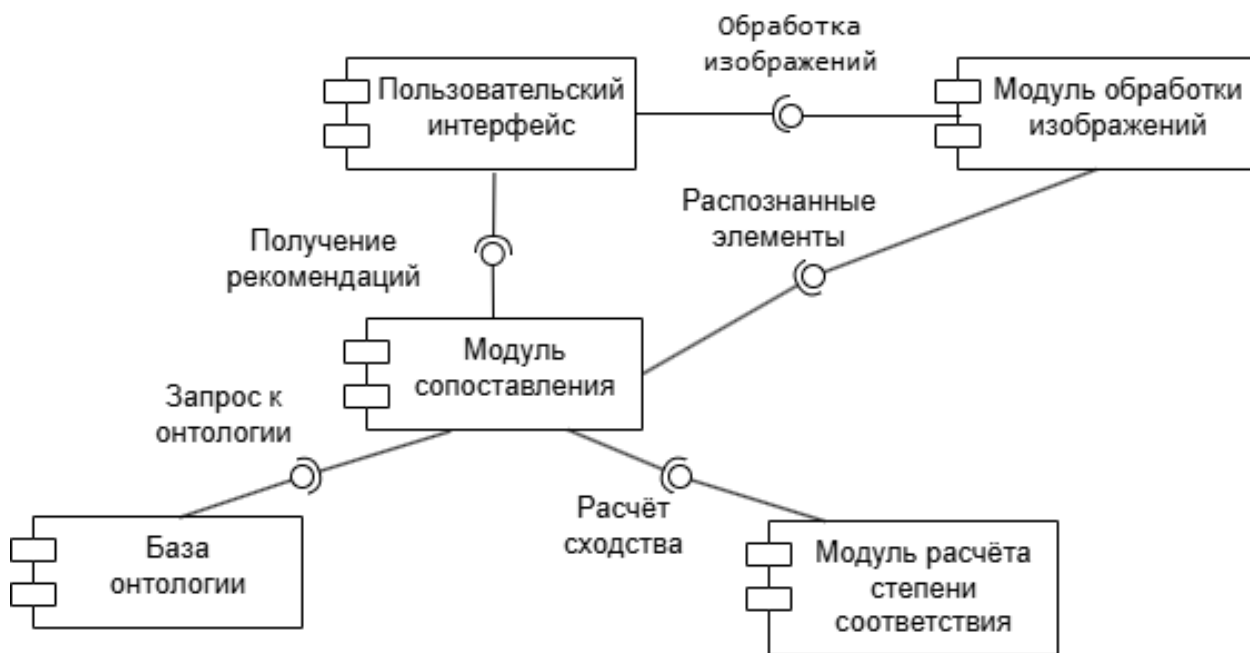
Для описания логической структуры системы и взаимодействия между её основными частями используется диаграмма компонентов UML. Такой подход позволяет отразить планируемые программные модули и интерфейсы, обеспечивающие их взаимодействие в рамках проектируемой архитектуры.

На диаграмме представлены следующие компоненты:

- Модуль сопоставления – осуществляет сопоставление распознанных элементов с онтологией диаграмм, используя библиотеку RDFLib и SPARQL-запросы. Он запрашивает связи из базы знаний, передаёт данные в модуль расчёта степени соответствия и возвращает рекомендации в пользовательский интерфейс.

- Модуль обработки изображений – реализует детекцию графических элементов на изображении с использованием модели YOLOv8. Принимает изображение от интерфейса и возвращает список распознанных элементов.
- Модуль сопоставления – осуществляет сопоставление распознанных элементов с онтологией диаграмм, используя библиотеку RDFLib и SPARQL-запросы. Выдаёт рекомендации по типам диаграмм и предоставляет интерфейс для запроса структуры онтологии.
- База онтологии – содержит RDF-описание визуальных моделей и отвечает на запросы о связях между элементами и типами диаграмм.
- Модуль расчёта степени соответствия – рассчитывает значения степени соответствия между распознанной структурой и эталонными типами диаграмм.

На рисунке 12 представлена UML диаграмма компонентов системы.



**Рисунок 12 – Диаграмма компонентов архитектуры системы**

Каждый компонент реализует или использует интерфейсы, отображённые на диаграмме. Например, модуль сопоставления реализует интерфейс «ПолучениеРекомендаций» и обращается к «ЗапросКОнтологии» и «РасчетСходства».

## 2.2 Проектирование структуры базы знаний

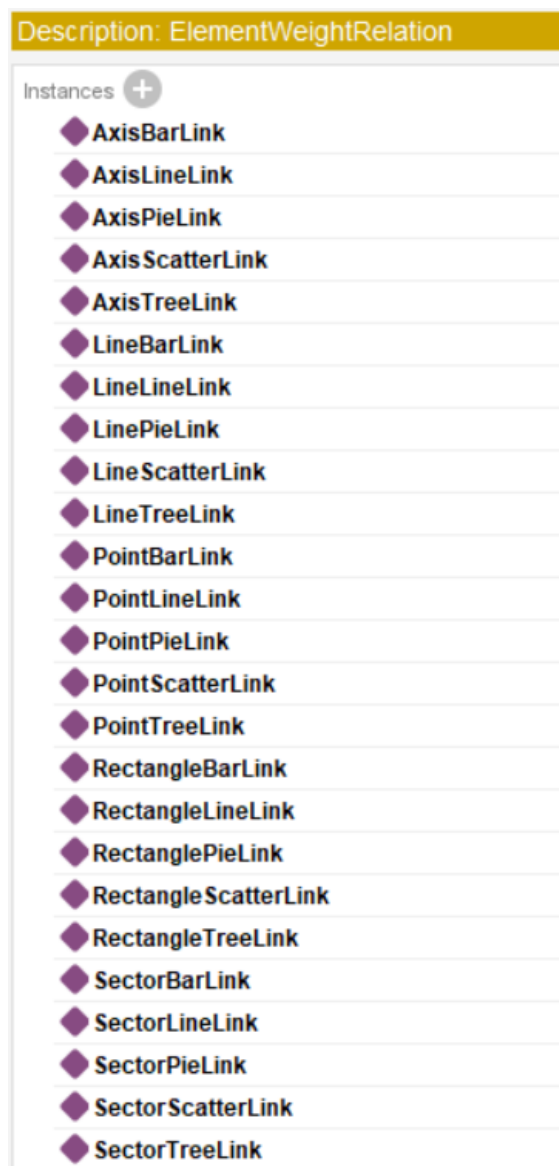
Для поддержки сопоставления распознанных визуальных элементов с типами диаграмм в системе используется база знаний, представленная в виде онтологии. В качестве основы выбрана структура, разработанная в [11], включающая иерархию категорий визуализаций и набор базовых элементов. В рамках проектирования данная онтология была доработана и расширена с целью формального описания связей между элементами и типами диаграмм, а также добавления метрик, необходимых для оценки степени соответствия.

Для хранения весов элементов в зависимости от типа диаграммы введён промежуточный класс `ElementWeightRelation`. Экземпляры этого класса описывают, насколько важен тот или иной элемент визуализации (например, прямоугольник или ось) для конкретной метрики типа диаграммы (например, `Bar Chart`).

На рисунке 13 представлен список экземпляров класса `ElementWeightRelation`, созданных для комбинаций элементов (`Axis`, `Line`, `Point`, `Rectangle`, `Sector`, `Legend`) и типов диаграмм (`Bar_chart`, `Pie_chart`, `Line_graph`, `Scatter_plot`, `Treemap`).

Для задания связей между экземплярами `ElementWeightRelation`, элементами диаграмм и метриками введены два объектных свойства:

- `forElement` – указывает, к какому элементу диаграммы относится вес (domain: `ElementWeightRelation`, range: `Diagram_elements`);
- `forMetric` – указывает, к какому типу диаграммы относится метрика (domain: `ElementWeightRelation`, range: `SimilarityMetrics`).



*Рисунок 13 – Список экземпляров класса ElementWeightRelation*

На рисунке 14 представлен пример настройки этих свойств.



*Рисунок 14 – Свойства forElement и forMetric*

Для хранения числовых значений весов используется дата-свойство hasWeightValue. Оно применяется к каждому экземпляру ElementWeightRelation и

задаёт значимость конкретного элемента в рамках конкретной метрики. Значения весов элементов для каждого типа диаграммы, установленные по умолчанию показаны в таблице 4.

Таблица 4 – Значения весов элементов для каждого типа диаграммы

Элемент	Bar Chart	Pie Chart	Line Graph	Scatter Plot	Treemap
Axis	0.9	0.0	0.8	0.6	0.0
Line	0.0	0.0	0.9	0.6	0.0
Point	0.0	0.0	0.6	0.9	0.0
Rectangle	0.9	0.0	0.0	0.0	0.9
Sector	0.0	0.9	0.0	0.0	0.0
Legend	0.0	0.8	0.0	0.0	0.0

Каждый вес добавляется в виде числового значения к соответствующему экземпляру `ElementWeightRelation`. Веса элементов выбраны на основе визуального анализа типичных образцов диаграмм.

### 2.3 Проектирование структуры данных результата распознавания

Результатом этапа распознавания визуальной модели является структурированное представление выявленных элементов диаграммы. Для дальнейшего сопоставления с базой знаний и расчета степени соответствия разработана единая структура данных.

Распознанные элементы сохраняются в виде словаря следующего вида:

- «axis»: количество найденных осей;
- «rectangle»: количество найденных прямоугольников;
- «line»: количество найденных линий;
- «point»: количество найденных точек;
- «sector»: количество найденных секторов;
- «legend»: количество найденных легенд.

Эта структура обеспечивает унифицированный формат передачи данных между модулем распознавания и модулем сопоставления.

### 2.4 Проектирование метрик оценки соответствия

Для оценки степени соответствия между структурой распознанной диаграммы и описанием типов визуализаций, представленных в онтологии, предлагается использовать модифицированную формулу взвешенной метрики. Она дополнена

штрафом за отсутствующие, но ожидаемые элементы, что позволяет точнее оценивать соответствие структуре диаграммы.

Метрика рассчитывается по следующей формуле:

$$S = \frac{\sum_{i=1}^n \omega_i \cdot m_i - \lambda \sum_{i=1}^n \omega_i (1 - m_i)}{\sum_{i=1}^n \omega_i} \quad (1)$$

где  $S$  – итоговая степень соответствия (от 0 до 1),

$\omega_i$  – вес  $i$ -го элемента диаграммы (например, оси, прямоугольники, линии),

$m_i$  – бинарное значение совпадения по элементу (1 – найден, 0 – не найден),

$\lambda \in [0,1]$  – коэффициент штрафа,

$n$  – количество типов элементов, учитываемых при сравнении.

Подход основывается на взвешенном агрегировании по признакам, аналогично методам, применяемым в рекомендательных системах и задачах классификации с неоднородными признаками. Схожие принципы используются в работе ChartOCR [14], где оценка качества распознавания визуальных моделей осуществляется на основе индивидуальных метрик, адаптированных под каждый тип диаграммы.

Такой метод позволяет реализовать небинарную классификацию, при которой система может возвращать список подходящих типов диаграмм с указанием степени соответствия, а не единственный результат. Это особенно важно в случаях, когда структура изображения может частично совпадать с несколькими типами визуализаций. Кроме того, данный подход легко расширяется за счёт задания пользовательских весов или автоматической настройки на конкретные задачи.

## 2.5 Проектирование поведения системы

Для анализа логики взаимодействия компонентов и отображения сценариев работы системы используются поведенческие диаграммы UML. В данном разделе представлены ключевые сценарии поведения системы в виде диаграммы активностей и диаграммы последовательностей.

### 2.5.1 Алгоритм получения рекомендаций по типу диаграммы

Процесс начинается с загрузки изображения пользователем через веб-интерфейс. Система инициирует выполнение детекции визуальных элементов на

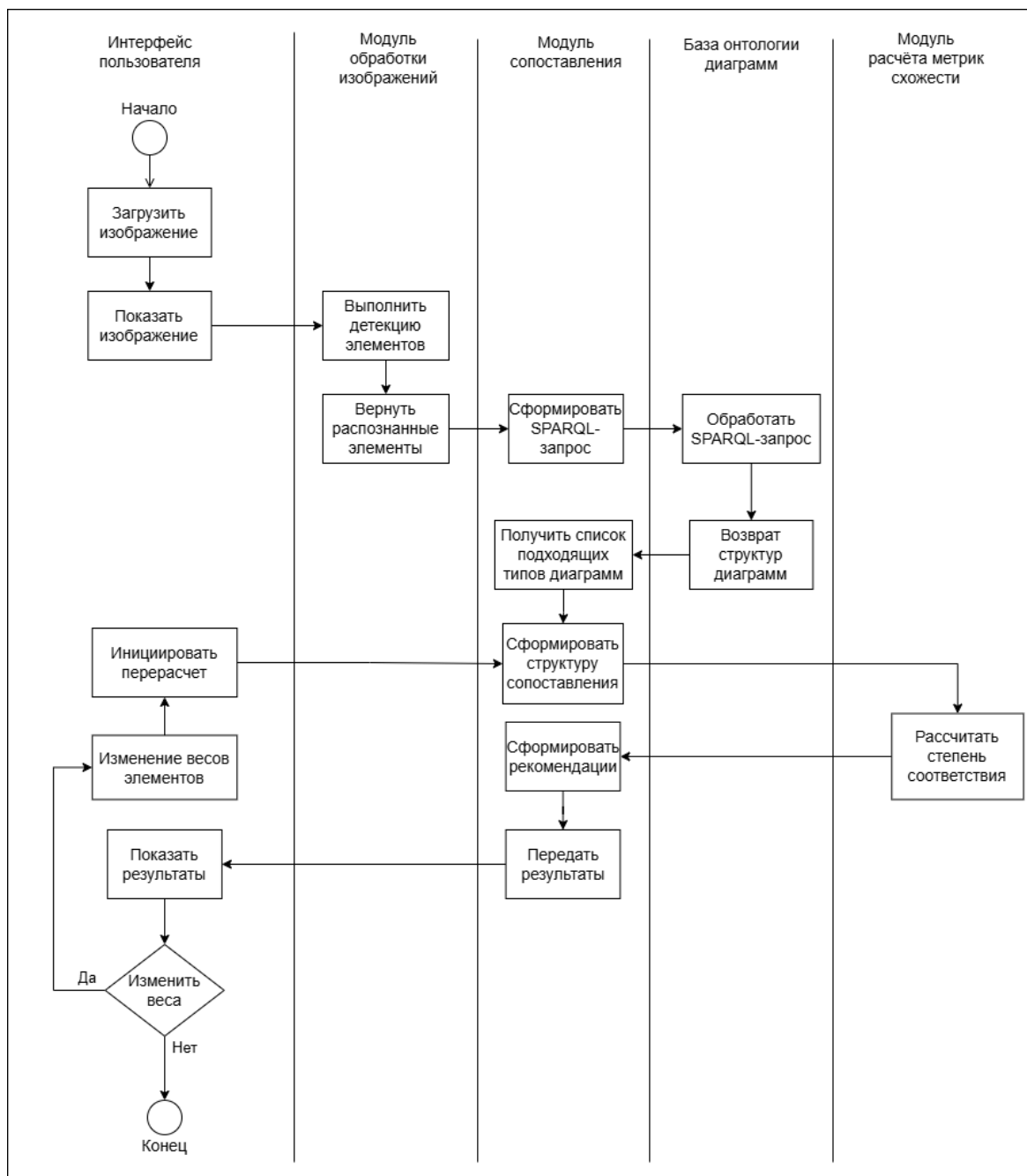


изображении с помощью модели YOLOv8, результатом чего становится список найденных объектов (оси, прямоугольники, точки и др.).

Далее модуль сопоставления формирует SPARQL-запрос к онтологической базе, содержащей структуры типовых диаграмм. После получения данных система строит структуру сопоставления и передаёт информацию в модуль расчёта метрик. На основании вычисленной степени соответствия формируются рекомендации по типам диаграмм, наиболее близким к распознанной структуре.

Пользователь может просмотреть результат, а при необходимости изменить веса значимости отдельных элементов. В этом случае инициируется перерасчёт рекомендаций с учётом новых параметров.

На рисунке 15 представлена диаграмма активностей для алгоритма получения рекомендаций на основе загруженного изображения.



**Рисунок 15 – Диаграмма активностей для алгоритма получения рекомендаций**

Диаграмма иллюстрирует взаимодействие между основными модулями системы, начиная с загрузки изображения и заканчивая формированием рекомендаций по типу диаграммы. Помимо основного процесса внимание уделено возможности ручной настройки весов, что позволяет пользователю уточнить результат.

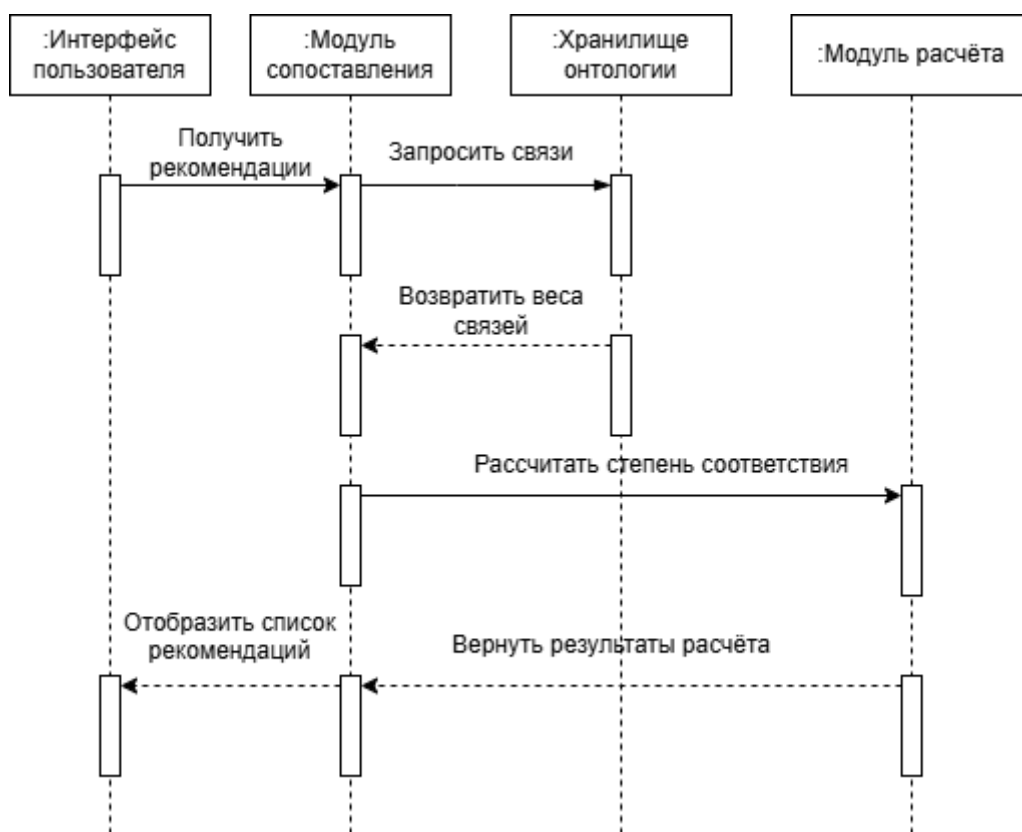
### 2.5.2 Алгоритм расчета степени соответствия

На данном этапе система сопоставляет распознанные элементы диаграммы с эталонными структурами из базы знаний, чтобы определить, насколько каждый тип диаграммы соответствует входным данным.

Процесс основан на формуле (см. раздел 2.4), учитывающей веса элементов и коэффициент штрафа за несоответствие. На практике реализация этой логики требует взаимодействия между несколькими модулями:

- пользователь инициирует получение рекомендаций;
- модуль сопоставления формирует SPARQL-запрос и извлекает из базы онтологии весовые структуры;
- модуль расчёта использует полученные данные для оценки степени совпадения, применяя формулу;
- результат возвращается пользователю в виде отсортированного списка наиболее подходящих базовых визуализаций.

На рисунке 16 представлена диаграмма последовательности, иллюстрирующая вызовы и обмен сообщениями между компонентами системы.



**Рисунок 16 – Алгоритм сопоставления и расчета метрик**

Процесс включает:

- обращение к базе онтологии для получения связей между элементами и типами диаграмм;
- итерацию по каждому типу диаграммы с вычислением показателя  $s$  по формуле;

- выбор пользовательских весов, если они указаны;
- сортировку и возврат списка типов диаграмм с максимальной степенью соответствия.

Такой подход обеспечивает гибкую и настраиваемую систему рекомендаций, опирающуюся как на автоматически извлечённые признаки, так и на пользовательские предпочтения.

## **2.6 Формирование рекомендаций**

После расчёта степени соответствия система формирует ранжированный список рекомендаций на основе всех типов диаграмм, присутствующих в онтологии, в которых хотя бы один из распознанных элементов имеет ненулевой вес.

Таким образом, рекомендации представляют собой базовые типы визуализаций, частично совпадающие с распознанной структурой, отсортированные по убыванию рассчитанного показателя соответствия. Это позволяет отображать не только наиболее вероятный тип диаграммы, но и альтернативные варианты, учитывающие возможные неточности в распознавании и вариативность структуры визуальных моделей.

Такой подход делает систему более устойчивой и гибкой, а также позволяет пользователю самостоятельно интерпретировать результат с учётом специфики задачи.

## **2.7 Выводы**

В результате проектирования была сформирована целостная архитектура системы, включающая модули детекции, сопоставления, расчёта метрик и формирования рекомендаций. Разработана структура онтологической базы знаний, позволяющая учитывать важность различных визуальных элементов для каждого типа диаграммы. Предложена формальная модель представления результатов распознавания, а также адаптированная метрика оценки соответствия, обеспечивающая более точную классификацию диаграмм. Поведенческие диаграммы позволили формализовать сценарии взаимодействия модулей и алгоритмы работы системы. Полученные проектные решения обеспечивают основу для эффективной реализации.

## **Глава 3 Реализация**

На основе проведённого проектирования была осуществлена реализация основных компонентов системы автоматизированного распознавания типов диаграмм.

В рамках реализации решаются следующие задачи:

1. Настроить программную среду и подготовить исходные данные для обучения модели и запуска системы.
2. Реализовать модуль выделения ключевых визуальных элементов на изображениях с использованием модели YOLOv8.
3. Разработать механизм сопоставления распознанных элементов с онтологией и реализовать алгоритм расчёта степени соответствия между набором элементов и типами диаграмм.
4. Обеспечить отображение результатов пользователю и реализовать интерфейс для взаимодействия, включая возможность ручной настройки весов.
5. Провести оценку качества модели распознавания с использованием стандартных метрик.

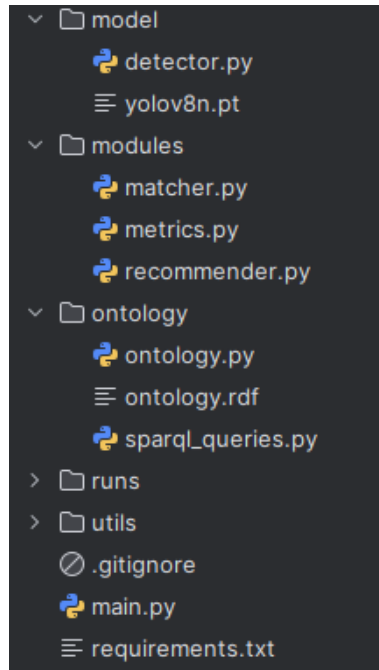
### **3.1 Организация программной среды**

Для реализации прототипа использовались ранее выбранные технологии и программные средства (см. раздел 1.6 «Выбор технологий и инструментов»), включая язык программирования Python, библиотеку rdflib для работы с онтологиями и модель детекции объектов YOLOv8.

Все разработки велись в среде PyCharm. Структура проекта организована по основным функциональным компонентам:

- модуль загрузки и предобработки изображения;
- модуль распознавания ключевых элементов диаграммы;
- модуль сопоставления распознанной структуры с онтологией типов диаграмм;
- модуль расчета метрик степени соответствия;
- пользовательский интерфейс для демонстрации результатов.

Структура проекта представлена на рисунке 17.



*Рисунок 17 – Структура проекта разрабатываемой системы*

### **3.2 Разработка модуля выделения элементов диаграмм**

Для решения задачи автоматического обнаружения ключевых элементов диаграмм был разработан модуль на основе модели YOLOv8, предназначенный для распознавания компонентов визуальных моделей, таких как прямоугольники, оси, линии, сектора, точки и легенды на изображениях, загружаемых пользователем.

Для обучения модели потребовался специально подготовленный датасет. Его создание и аннотирование было выполнено с помощью сервиса Roboflow, который значительно упростил процесс разметки и экспорта данных в совместимом с YOLOv8 формате. В частности, Roboflow автоматически сгенерировал структуру проекта с аннотациями в виде .txt файлов и конфигурационным файлом data.yaml, необходимым для запуска обучения.

В итоговый датасет вошло 676 изображений (около 130 для каждого типа диаграмм), вручную размеченных по ключевым элементам: Rectangle, Axis, Line, Point, Sector и Legend.

На рисунке 18 приведён фрагмент кода, демонстрирующий загрузку обученной модели и выполнение предсказания ключевых элементов визуальной модели на входном изображении. В качестве результата функция возвращает изображение с нанесёнными рамками и список найденных классов.

```

# Загрузка модели
model = YOLO("../runs/detect/train7/weights/best.pt")

def detect_elements(image): 2 usages new *
    results = model.predict(source=image, save=False, conf=0.25)
    classes_detected = []
    for r in results:
        for box in r.bboxes:
            cls_id = int(box.cls[0])
            label = model.names[cls_id]
            classes_detected.append(label)
    # Обработанное изображение и список найденных элементов
    img_with_boxes = results[0].plot()
    return img_with_boxes, classes_detected

```

*Рисунок 18 – Фрагмент кода модуля обнаружения элементов*

### 3.3 Сопоставление с онтологией и расчёт степени соответствия

Для сопоставления распознанных элементов визуальной модели с базовыми типами диаграмм была разработана база знаний с использованием системы Protégé и экспортирована в формате RDF. Основу базы знаний составляют типы диаграмм, их характерные элементы и веса этих элементов, отражающие их значимость при определении принадлежности изображения к тому или иному типу.

В структуре онтологии введён промежуточный класс `ElementWeightRelation`, который связывает:

- тип диаграммы (`forMetric`);
- элемент диаграммы (`forElement`);
- числовое значение веса (`hasWeightValue`).

Такое представление позволяет более гибко учитывать вклад каждого элемента в итоговую оценку соответствия распознанной структуры определённому типу диаграммы.

Для извлечения информации о типах диаграмм, их характерных элементах и соответствующих весах используется SPARQL-запрос, представленный на рисунке 19.

```

query = """
PREFIX : <http://www.semanticweb.org/yana/ontologies
SELECT ?metric ?element ?weight
WHERE {
    ?rel a :ElementWeightRelation .
    ?rel :forElement ?element .
    ?rel :forMetric ?metric .
    ?rel :hasWeightValue ?weight .
    FILTER (?weight > 0)
}
"""

```

**Рисунок 19 – SPARQL-запрос для получения весов элементов диаграмм**

На основании результатов запроса реализован алгоритм расчёта степени соответствия. Для каждого типа диаграммы распознанные элементы сравниваются с ожидаемыми элементами, определёнными в онтологии. Суммируются веса найденных совпадений, и из этой суммы вычитается штраф за элементы, отсутствующие в структуре данного типа диаграммы. Итоговая степень соответствия вычисляется по формуле:

$$\text{Степень соответствия} = \frac{\text{Сумма всех найденных весов} - \lambda \times \text{Сумма штрафов}}{\text{Общая сумма всех весов типа диаграммы}} \times 100\% \quad (2)$$

Данная формула эквивалентна ранее представленной (1), но оформлена в другом виде для наглядности.

Реализация функции расчёта степени соответствия типа диаграммы приведена в фрагменте кода, который изображен на рисунке 20.

```

raw_score = (matched_weight - penalty_lambda * penalty_weight)
similarity = max((raw_score / total_possible_weight) * 100
                 if total_possible_weight > 0 else 0, 0)
final_score = round(similarity, 2)

```

**Рисунок 20 – Функция расчёта степени соответствия с учётом весов**

В результате работы функции формируется упорядоченный по убыванию степени соответствия словарь с возможными типами диаграмм и рассчитанными значениями уверенности.

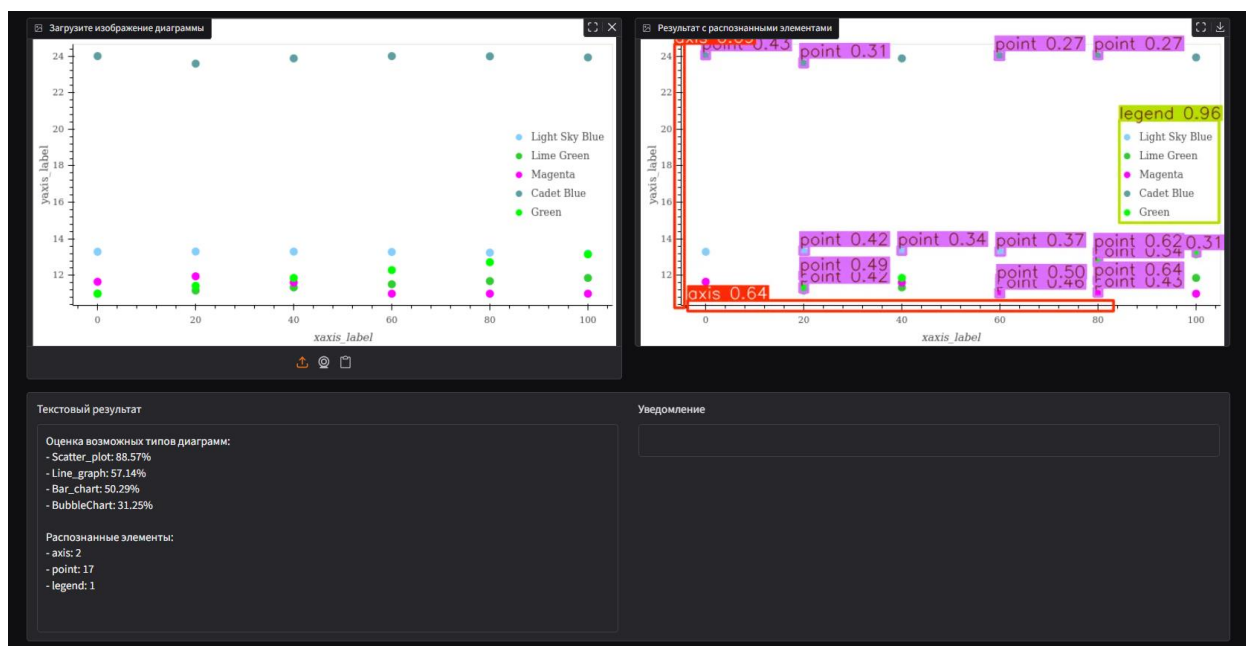
### 3.4 Отображение результатов и взаимодействие с пользователем

Разработанный пользовательский интерфейс обеспечивает прозрачное отображение этапов обработки изображения и удобное взаимодействие с результатами распознавания и рекомендациями. Интерфейс реализован с



использованием библиотеки Gradio, что позволило упростить интеграцию визуальных компонентов и обеспечить интерактивность.

На рисунке 21 представлены два блока: слева – загруженное пользователем изображение диаграммы, справа – результат работы модели распознавания, где визуальные элементы выделены цветными bounding box с подписями классов и вероятностями. Данное отображение позволяет наглядно проверить корректность работы модели.



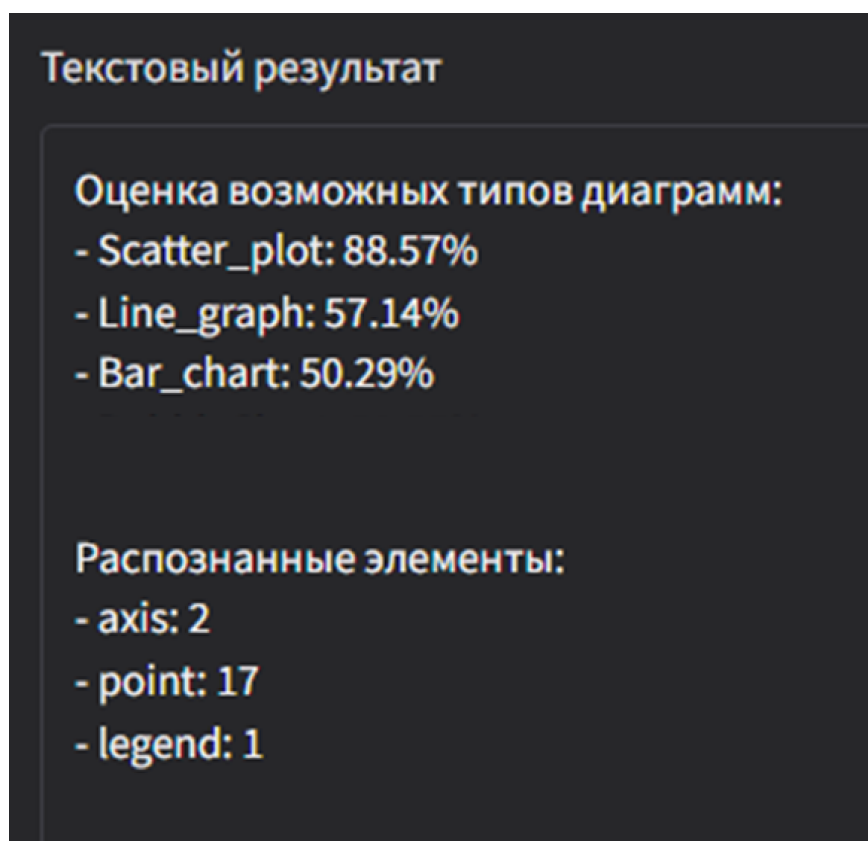
**Рисунок 21 – Исходное изображение и визуализация распознанных элементов**

Под визуализацией находится текстовый блок, представленный на рисунке 22, содержащий две группы данных:

- список возможных типов диаграмм, отсортированный по степени соответствия (в процентах), рассчитанной на основе формулы из раздела 2.4;
- перечень распознанных элементов с указанием их количества по классам.

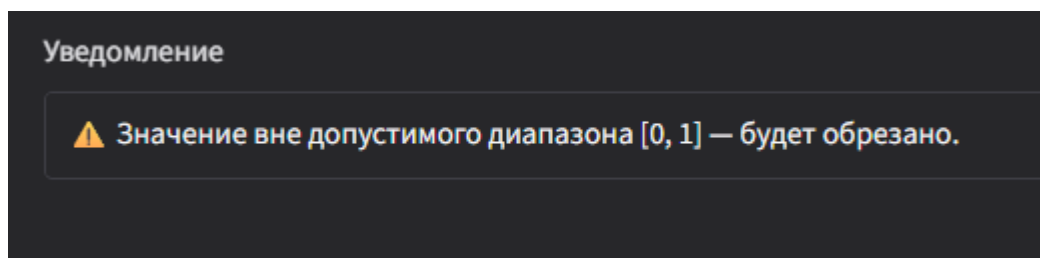
Рядом с текстовым блоком размещена панель уведомлений, в которой отображаются сообщения об ошибках. Например, при попытке изменить значение в защищённой ячейке таблицы пользователь получает соответствующее уведомление, или при вводе некорректного значения система выводит сообщение об ошибке. Это позволяет избежать сбоев в логике расчёта и делает интерфейс более надёжным и понятным для пользователя.

Панель уведомлений с сообщением об ошибке показана на рисунке 23.



*Рисунок 22 – Текстовый результат обработки изображения*

Такая структура результатов позволяет пользователю быстро оценить наиболее вероятные типы диаграмм, а также проверить наличие и корректность распознанных элементов.



*Рисунок 23 – Панель уведомлений*

Дополнительно пользователь может вручную скорректировать значимость отдельных типов элементов при расчёте рекомендаций. Для этого предоставлена таблица ручной настройки весов, отображённая на рисунке 24. Пользователь может отредактировать значения и инициировать повторный расчёт степени соответствия с новыми весами. Предусмотрена также возможность сброса настроек к значениям по умолчанию.

Ручная настройка весов элементов

Таблица весов

Элемент	BarChart	BubbleChart	LineGraph	ScatterPlot
axis	0.9	0.4	0.8	0.6
legend	0	0	0	0
point	0	0.1	0.4	0.95

Перераспознать с новыми весами      Сбросить веса по умолчанию

**Рисунок 24 – Интерфейс настройки весов визуальных элементов**

Такой подход к отображению результатов обеспечивает прозрачность работы алгоритма, гибкость для пользовательских предпочтений и возможность интерактивного уточнения рекомендаций.

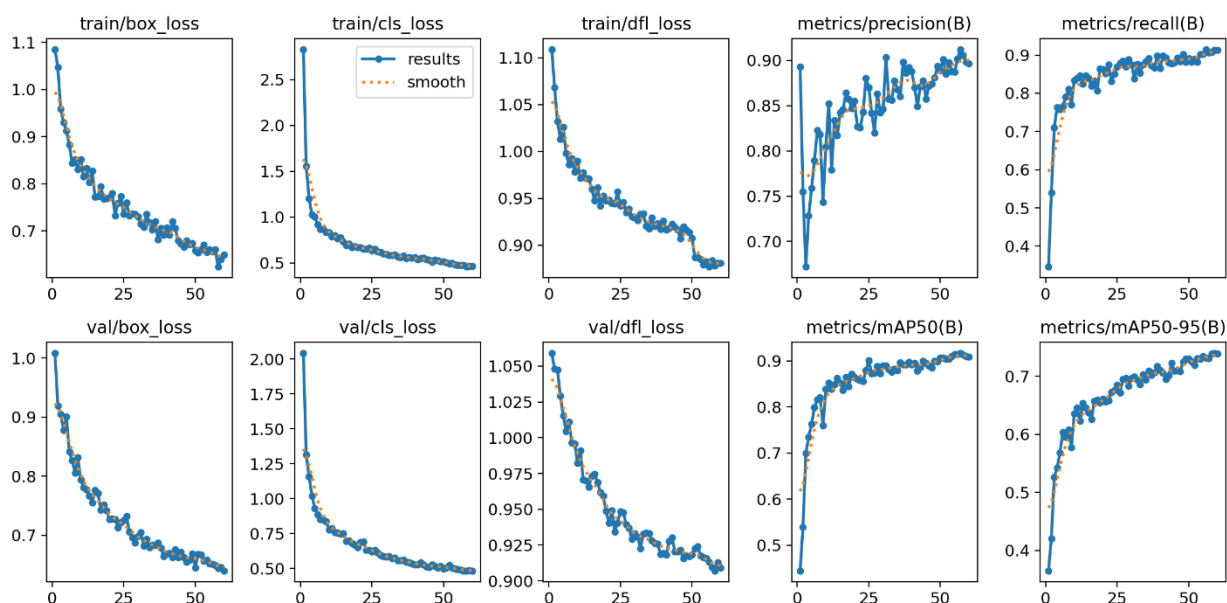
### 3.5 Оценка качества модели распознавания

Разработанный модуль на базе YOLOv8 обеспечивает обнаружение визуальных элементов диаграмм, таких как прямоугольники, оси, точки, линии и сектора. Модель была обучена на вручную аннотированном наборе изображений, содержащем примеры основных типов диаграмм, и продемонстрировала высокие показатели точности распознавания.

Поскольку основная цель проекта – разработка прототипа системы, детальное тестирование модели на независимых валидационных выборках и кросс-валидация не проводились. Тем не менее, динамика ключевых метрик в процессе обучения свидетельствует о хорошей способности модели к обобщению.

Следует отметить, что в дальнейшем предполагается дополнить систему более специализированными метриками, которые будут учитывать структурную значимость элементов для разных типов диаграмм (см. раздел 1.5). На текущем этапе для оценки эффективности использовались встроенные метрики обучения YOLOv8, включая precision, recall, mAP50 и mAP50-95. Графики обучения приведены на рисунке 25. Итоговые значения метрик составили:

- precision: около 90%;
- recall: около 91%;
- mAP50: около 91%;
- mAP50-95: около 73%.



**Рисунок 25 – Динамика метрик обучения модели**

Эти показатели подтверждают, что модель успешно решает задачу детектирования визуальных элементов диаграмм и может быть эффективно использована в составе системы рекомендаций по типу визуализации.

### 3.6 Выводы

Результатом этапа реализации стал работоспособный прототип системы, включающий модули распознавания визуальных элементов, сопоставления с онтологией и формирования рекомендаций. Обученная модель YOLOv8 продемонстрировала высокую точность в выделении ключевых компонентов диаграмм, что позволило эффективно проводить последующее сопоставление. Реализован алгоритм оценки степени соответствия, опирающийся на формальные веса и логически оформленный в коде.

Пользовательский интерфейс, созданный с использованием библиотеки Gradio, обеспечивает прозрачное отображение результатов, возможность ручной настройки параметров и интерактивное взаимодействие. Проведённая оценка метрик подтверждает пригодность модели для использования в задачах автоматического анализа визуальных структур. Полученные результаты демонстрируют готовность системы к практическому применению и дальнейшему расширению.

## Заключение

В ходе исследования была разработана система, позволяющая автоматически определять тип визуальной модели по изображению диаграммы. Актуальность темы обусловлена необходимостью поддержки пользователей в выборе подходящих диаграмм, особенно при отсутствии специализированной подготовки в области визуализации данных.

Работа включает аналитический обзор существующих типов диаграмм, методов их структурного описания, подходов к распознаванию визуальных моделей и способов оценки качества. На основе анализа была доработана онтология типов диаграмм и реализован прототип системы, включающий модуль детекции визуальных элементов с использованием модели YOLOv8 и компонент сопоставления с базой знаний.

Модель успешно распознаёт ключевые визуальные элементы диаграмм. Онтологическая структура обеспечивает сопоставление с базовыми типами визуализаций, а расширение обучающей выборки позволило значительно повысить точность. В интерфейс добавлена возможность ручной настройки весов значимости элементов, что позволяет адаптировать рекомендации под пользовательские задачи.

Перспективы развития системы следующие:

- повышение качества аннотаций и обучение на сложных визуализациях;
- внедрение дополнительных эвристик в процесс сопоставления;
- дальнейшее развитие механизма рекомендаций с учётом целей анализа и структуры данных [8];
- возможная интеграция прототипа в аналитическую платформу в качестве рекомендательного модуля.

## Библиографический список

1. Types Sharma S. Data Storytelling 101: 72 Types of Data Visualization to Design Stories [Электронный ресурс] // Gramener. – URL: <https://blog.gramener.com/types-of-data-visualization-for-data-stories/> (дата обращения: 18.12.2024).
2. Hess K. 80 types of charts & graphs for data visualization (with examples) [Электронный ресурс] // Datylon. – URL: <https://www.datylon.com/blog/types-of-charts-graphs-examples-data-visualization> (дата обращения: 18.12.2024).
3. The Data Visualization Catalogue [Электронный ресурс] // Data Visualization Catalogue. – URL: <https://datavizcatalogue.com> (дата обращения: 20.12.2024).
4. Moran K. Choosing Chart Types: Consider Context [Электронный ресурс] // Nielsen Norman Group. – URL: <https://www.nngroup.com/articles/choosing-chart-types/> (дата обращения: 17.12.2024).
5. Yi M., Restori M. How to choose the right data visualization [Электронный ресурс] // Atlassian. – URL: <https://www.atlassian.com/data/charts/how-to-choose-data-visualization> (дата обращения: 08.01.2025).
6. Types of Charts and Graphs [Электронный ресурс] // ChartExpo. – URL: <https://chartexpo.com/blog/types-of-charts-and-graphs#> (дата обращения: 08.01.2025).
7. Kubernatova P., Friedjungova M., van Duijn M. Constructing a Data Visualization Recommender System // ResearchGate. – URL: <https://www.researchgate.net/publication/334577081> (дата обращения: 12.02.2025).
8. Kubernatova P., Friedjungova M., van Duijn M. Knowledge at First Glance: A Model for a Data Visualization Recommender System Suited for Non-expert Users [Электронный ресурс] // ResearchGate. – URL: <https://www.researchgate.net/publication/326874617> (дата обращения: 12.02.2025).

12.02.2025).

9. Claus Wilke. Fundamentals of Data Visualization [Электронный ресурс] // Claus Wilke. – URL: <https://clauswilke.com/dataviz/> (дата обращения: 18.12.2024).
10. The Data Viz Project [Электронный ресурс] // Data Viz Project. – URL: <https://datavizproject.com/> (дата обращения: 18.12.2024).
11. Dzheiranian A., Ermakov I., Proskuryakov K., Lyadova L.N. An Approach to Developing Data Visualization Tools Based on Domain Specific Modeling // ResearchGate. – URL: [https://www.researchgate.net/publication/386194082\\_An\\_Approach\\_to\\_Developing\\_Data\\_Visualization\\_Tools\\_Based\\_on\\_Domain\\_Specific\\_Modeling](https://www.researchgate.net/publication/386194082_An_Approach_to_Developing_Data_Visualization_Tools_Based_on_Domain_Specific_Modeling) (дата обращения: 16.12.2024).
12. Mustafa O., Ali M.K., Moetesum M., Siddiqi I. ChartEye: A Deep Learning Framework for Chart Information Extraction [Электронный ресурс] // arXiv. – URL: <https://arxiv.org/abs/2408.16123> (дата обращения: 20.01.2025).
13. Xue W., Chen D., Yu B., Chen Y., Zhou S., Peng W. ChartDETR: A Multi-shape Detection Network for Visual Chart Recognition [Электронный ресурс] // arXiv. – URL: <https://arxiv.org/abs/2308.07743> (дата обращения: 20.01.2025).
14. Luo J., Li Z., Wang J., Lin C.-Y. ChartOCR: Data Extraction from Charts Images via a Deep Hybrid Framework [Электронный ресурс] // WACV 2021. – URL: [https://openaccess.thecvf.com/content/WACV2021/papers/Luo\\_ChartOCR\\_Data\\_Extraction\\_From\\_Charts\\_Images\\_via\\_a\\_Deep\\_Hybrid\\_WACV\\_2021\\_paper.pdf](https://openaccess.thecvf.com/content/WACV2021/papers/Luo_ChartOCR_Data_Extraction_From_Charts_Images_via_a_Deep_Hybrid_WACV_2021_paper.pdf) (дата обращения: 23.01.2025).
15. Dhote A., Javed M., Doermann D.S. Swin-chart: An efficient approach for chart classification [Электронный ресурс] // Pattern Recognition Letters. – URL: <https://www.sciencedirect.com/science/article/abs/pii/S0167865524002447>

(дата обращения: 25.01.2025).

16. Bajić F., Job J. Data Extraction of Circular-Shaped and Grid-like Chart Images [Электронный ресурс] // PubMed Central. – URL: <https://pmc.ncbi.nlm.nih.gov/articles/PMC9147858/> (дата обращения: 30.01.2025).
17. Fursov V.A., Bibikov S.A., Yakimov P. Localization of objects contours with different scales in images using hough transform [Электронный ресурс] // ResearchGate. – URL: <https://www.researchgate.net/publication/283667712> (дата обращения: 05.02.2025).
18. Savva M., Kong N., Chhajta A., Fei-Fei L., Agrawala M., Heer J. ReVision: Automated Classification, Analysis and Redesign of Chart Images [Электронный ресурс] // ACM. — URL: <https://dl.acm.org/doi/abs/10.1145/2047196.2047247> (дата обращения: 05.02.2025).
19. Araujo T., Chagas P., Alves J., Santos C., Sousa Santos B., Meiguins B.S. A Real-World Approach on the Problem of Chart Recognition Using Classification, Detection and Perspective Correction [Электронный ресурс] // Sensors. — URL: <https://www.mdpi.com/1424-8220/20/16/4370> (дата обращения: 10.02.2025).
20. Dhote A., Javed M., Doermann D.S. A Survey and Approach to Chart Classification [Электронный ресурс] // ResearchGate. — URL: <https://www.researchgate.net/publication/372247991> (дата обращения: 11.02.2025).
21. Mohanapriya S., Saranya S.M., Kumaravel T., Sumithra P. Image Detection and Segmentation using YOLO v5 for Surveillance [Электронный ресурс] // ResearchGate. — URL: <https://www.researchgate.net/publication/372823216> (дата обращения: 12.02.2025).
22. Vinyals O., Toshev A., Bengio S., Erhan D. Show and Tell: Lessons Learned from the 2015 MSCOCO Image Captioning Challenge [Электронный ресурс] // IEEE. — URL: <https://ieeexplore.ieee.org/abstract/document/7505636> (дата



обращения: 10.02.2025).

23. Nielsen J. Response Times: The 3 Important Limits [Электронный ресурс] // Nielsen Norman Group. — URL: <https://www.nngroup.com/articles/response-times-3-important-limits/> (дата обращения: 25.04.2025).

# **ПРИЛОЖЕНИЕ А**

## **ТЕХНИЧЕСКОЕ ЗАДАНИЕ**