МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ И РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Конструирование признаков из контекста динамики показателей процессов

Ульяновск
УлГТУ
2023

УДК 004.891.2
ББК 32.813
К 65

Коллектив авторов: Ярушкина Н.Г., Мошкин В.С., Гуськов Г.Ю., Романов А.А., Филиппов А.А., Андреев И.А., Корунова Н.В., Эгов Е.Н.

Рецензенты:
Генеральный директор ООО ГК «Агросервис-ИТ» Кудасов Г. В.
Доцент кафедры «Вычислительная техника» УлГТУ, канд. техн. наук Святов К. В.

Научный редактор:
д-р техн. наук, профессор Н.Г. Ярушкина

УДК 004.891.2

Конструирование признаков из контекста динамики показателей процессов [Электронный ресурс] / Н.Г. Ярушкина, В.С. Мошкин, Г.Ю. Гуськов, и др.; под ред. Н.Г. Ярушкиной. Электр. данные – Ульяновск : УлГТУ, 2023. – 183 с.

В монографии авторы изложили методы и алгоритмы конструирования признаков из контекста динамики показателей процессов. Рассмотрены подходы к получению, синтезу, подготовке, анализу таких данных, подходы к интерпретации результатов машинного обучения. Теоретические разработки поддержаны практическими примерами, иллюстрирующими принципы работы предложенных подходов. Описаны программные средства, реализующие представленные в книге методы и прошедшие экспериментальную апробацию.

Монография адресована научным работникам и прикладным пользователям.

Представлена в авторской редакции.

ISBN 978-5-9795-2356-9

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

За последние двадцать лет активно применяются разнообразные методы анализа данных в производстве с целью улучшения управляемости процессами, повышения эффективности работы и увеличения объемов производства. Использование эффективных подходов к анализу данных позволяет предприятию принимать решения оперативно и иногда применять методы предиктивной аналитики. Развитие современного производства, наличие различных датчиков, привело к значительному росту объема данных, которые необходимо анализировать для принятия решений. Роль анализа данных в производстве быстро растет, благодаря различным инструментам, основанным на автоматическом машинном обучении и автоматическом создании признаков, что уменьшает требования к квалификации аналитика данных.

В настоящее время компании стремятся полностью использовать потенциал методов анализа данных и машинного обучения для улучшения качества производственных процессов. Однако, отсутствуют методы и инструменты для формирования контекста анализа динамики показателей, полученных из разных источников, учитывая высокую степень неопределенности в описании объектов и процессов в данной области, а также поставленной задачи анализа.

Данная тема становится актуальной, поскольку для эффективного предиктивного анализа данных, полученных из различных источников, необходимо найти решение основных проблем, связанных с учетом нечеткости и фрагментарности данных, описывающих объекты и процессы в конкретной предметной области. Также следует решить проблему отсутствия универсальных механизмов надежного и интерпретируемого искусственного интеллекта и методов автоматического создания признаков

для данных, принимая во внимание высокую степень неопределенности и ограничений в предметной области.

Определение правильного обучающего набора данных является критически важной задачей для успешного процесса машинного обучения. Часто успех модели, полученной методами машинного обучения, зависит от правильного формирования обучающего набора данных. Ошибки, допущенные при формировании обучающего набора данных, могут серьезно повлиять на результаты алгоритмов обучения. Качество обучающих данных намного важнее особенностей алгоритма обучения.

С появлением глубоких нейронных сетей стало особенно важно правильно формировать обучающий набор данных, поскольку во многих задачах глубокие нейронные сети значительно превосходят другие алгоритмы машинного обучения, но для достижения такого качества требуется использовать большие обучающие наборы данных и специальные методы синтеза и генерации данных.

Благодарности

Монография выполнена в рамках государственного задания № 075-03-2023-143 по проекту «Исследование интеллектуальной предиктивной аналитики на базе интеграции методов конструирования признаков гетерогенных динамических данных для машинного обучения и методов предиктивного мультимодального анализа данных».

1 ИССЛЕДОВАНИЕ МЕТОДОВ И СРЕДСТВ ФОРМИРОВАНИЯ КОНТЕКСТА АНАЛИЗА ДИНАМИКИ ПОКАЗАТЕЛЕЙ ПРОЦЕССОВ

1.1 Машинное обучение в задачах предиктивной аналитики

В задачах предиктивной аналитики, связанных с использованием машинного обучения и глубокого обучения, важно учитывать структуру наборов данных и проводить предварительную обработку данных. Это позволяет моделям лучше представлять проблемную ситуацию и использовать соответствующие эвристики проблемной области. Кроме того, методы онтологического инжиниринга могут быть эффективными для решения задач предиктивной аналитики. **Онтологический инжиниринг** — это процесс создания и использования формальных моделей для представления знаний о предметной области. Онтологический инжиниринг может включать в себя создание онтологии, то есть структуры, которая описывает основные понятия, отношения и правила в данной проблемной области. Онтологический инжиниринг может эффективно помочь в организации данных, построении моделей и решении задач предиктивной аналитики.

Задачи предиктивной аналитики не ограничиваются только использованием регрессионного и корреляционного анализа. Они также требуют качественных суждений о будущей ситуации в целом. Выбор временных рядов переменных, характеризующих прогнозную ситуацию, является важным шагом в решении задач предиктивной аналитики. Здесь необходимо учитывать особенности и правила предметной области, к которой относится объект наблюдения. Для этого может быть использован онтологический инжиниринг, который позволяет описать проблемную область и выразить ее с помощью формальных моделей. Таким образом, онтологический инжиниринг является важным инструментом для

организации данных и построения моделей в задачах предиктивной аналитики.

Общая схема решения задач в машинном обучении включает следующие шаги:

1. **Подготовка данных.** На этом шаге данные, которые будут использоваться для обучения модели, подвергаются предварительной обработке. Подготовка данных может включать в себя очистку данных от выбросов и ошибок, заполнение пропущенных значений, масштабирование данных и другие методы предварительной обработки.

2. **Выбор модели**. На этом шаге выбирается подходящая модель машинного обучения для решения конкретной задачи. Это может быть линейная регрессия, дерево решений, нейронная сеть или другая модель, в зависимости от характеристик задачи и доступных данных.

3. **Обучение модели**. На этом шаге модель обучается на обучающих данных. Обучение модели включает в себя настройку ее параметров и оптимизацию функции потерь для достижения наилучшей производительности.

4. **Оценка модели**. После обучения модели ее необходимо оценить на проверочных данных, которые не использовались в процессе обучения. Это позволяет оценить точность и производительность модели на новых данных.

5. **Настройка модели.** Если модель показывает неудовлетворительные результаты на проверочных данных, может потребоваться настройка ее параметров или изменение модели целиком. Этот шаг может повторяться несколько раз, чтобы достичь наилучшей производительности модели.

6. **Применение модели**. После успешного обучения и оценки модели она может быть использована для предсказания или классификации новых данных.

Важно отметить, что эта общая схема может варьироваться в зависимости от конкретной задачи и доступных данных.

Такие шаги, как выбор и предобработка признаков или использование ансамблевых методов машинного обучения критически важны для решения задачи. Реальные объекты для анализа/классификации задаются целостно, а не списком признаков. Главный шаг в формировании модели машинного обучения – конструирование признаков. Базовые признаки обычно задаются разработчиком, исходя из его знаний о предметной области и целей задачи, то есть из контекста задачной ситуации.

1.2 Конструирование признаков модели машинного обучения и контекст задачи предиктивной аналитики

Конструирование признаков — это процесс создания новых признаков на основе имеющихся данных. Этап конструирования признаков может включать в себя использование статистических методов, функций преобразования или других техник для создания новых признаков, которые могут быть более информативными для модели.

Глубокое обучение, включая нейронные сети, позволяет модели самой комбинировать признаки, строить представления объектов, извлекать сложные и абстрактные признаки, которые могут быть полезны для решения задачи.

Таким образом, выбор и конструирование признаков являются важными шагами в формировании модели машинного обучения, и глубокое обучение позволяет модели самой извлекать и комбинировать признаки для достижения наилучшей производительности.

Конструирование признаков — это процесс создания новых признаков на основе имеющихся данных, включающий в себя использование статистических методов, функций преобразования или других техник для создания новых признаков, которые могут быть более информативными для модели.

Перечислим основные методы формирования признаков в машинном обучении. Вот более подробное описание каждого из них:

Инженерия признаков. Этот метод включает разработку и создание новых признаков вручную, основываясь на знаниях эксперта в предметной области. Инженерия признаков может включать в себя использование статистических методов, функций преобразования или других техник для создания новых признаков, которые могут быть более информативными для модели. Этот метод особенно полезен, когда у нас мало данных или когда имеющиеся признаки не содержат достаточно информации для решения задачи.

Обучение представлений. В архитектурах глубокого обучения модель сама обучается комбинировать и преобразовывать признаки, чтобы создавать более сложные и абстрактные представления объектов. Например, в нейронных сетях низкоуровневые признаки входных данных передаются через несколько слоев, позволяя модели извлекать более высокоуровневые и информативные признаки. Это позволяет модели более эффективно работать с данными и решать сложные задачи.

Обучение с подкреплением. Этот метод относится к обучению интеллектуального агента в процессе исследования среды. В процессе обучения с подкреплением агент принимает решения и получает обратную связь в виде награды или штрафа. Через многократные попытки и устранение ошибок агент учится принимать оптимальные решения и максимизировать полученную награду. В этом контексте **обобщение**

признаков означает, что агент учится воспринимать и использовать различные признаки и контексты для принятия решений.

Выбор и конструирование признаков являются важными шагами в формировании модели машинного обучения, и различные методы могут быть использованы в зависимости от конкретной задачи и доступных данных.

Тезис «От инженерии признаков к обучению представлений: от старого ИИ к новому ИИ» отражает важный сдвиг в развитии искусственного интеллекта (ИИ). Раньше, в эпоху «старого ИИ», основное внимание уделялось разработке и инженерии признаков, где эксперты вручную создавали и конструировали признаки, чтобы модель могла решать задачи. Однако с развитием глубокого обучения и нейронных сетей фокус сместился на обучение представлений, где модель сама обучается комбинировать и преобразовывать признаки, чтобы создавать более сложные и абстрактные представления объектов.

Этот сдвиг от инженерии признаков к **обучению представлений** отражает прогресс в развитии ИИ и позволяет моделям более эффективно работать с данными и решать сложные задачи. Обучение представлений позволяет моделям самостоятельно извлекать и комбинировать признаки, что может привести к более высокой производительности и способности к обобщению.

В ходе исследования разработана **технологическая концепция конструирования признаков как концепция гибридизации методов инженерии признаков в форме онтологического инжиниринга и обучения представлений в форме извлечения высокоуровневых признаков из низкоуровневых в форме глубокого обучения нейронных сетей.**

1.3 Классификация форм признаков

Конструирование признаков – это процесс извлечения признаков из данных и приведения их к формату, пригодному для обработки моделью машинного обучения. Признак, как правило, представляет собой числовое представление сырых данных, которое может быть использовано для обучения модели.

Существует множество методов обработки и конструирования признаков, которые могут быть применены для улучшения производительности модели машинного обучения. Некоторые из этих методов включают:

1. **Квантование**: разделение значений признака на равные диапазоны или интервалы для создания категориальных признаков.
2. Разбиение на группы (**квантильное разбиение**): разделение значений признака на несколько групп на основе статистических характеристик, таких как квартили или процентили.
3. **Логарифмическое преобразование**: применение логарифмической функции к значениям признака для учета нелинейных зависимостей.
4. **Степенное преобразование**: применение степени к значениям признака для учета нелинейных зависимостей.
5. **Масштабирование** (нормализация) признаков: приведение значений признаков к диапазону от 0 до 1 или стандартизация для устранения различий в масштабе и ускорения обучения модели.
6. **Стандартизация**: приведение значений признаков к стандартному нормальному распределению для улучшения обобщающей способности модели.

7. **Отбор признаков**: выбор наиболее информативных признаков для использования в модели, с целью уменьшения размерности данных и улучшения производительности модели.

8. **Кодирование категориальных переменных**: преобразование категориальных признаков в числовые значения, например, с помощью фиктивного кодирования или кодирования состояний.

9. **Хэширование** признаков: преобразование значений признаков в хэш-функции для учета их категориальной природы и улучшения производительности модели.

10. **Уменьшение размерности** (линейная проекция): снижение размерности данных путем применения линейного преобразования для улучшения обобщающей способности модели.

11. **Преобразование признаков**: применение различных преобразований, таких как полиномиальная интерполяция или рациональные функции, для создания новых признаков на основе существующих.

12. **Кластеризация и классификация признаков**: разделение признаков на группы или категории на основе их сходства или содержания для улучшения обобщающей способности модели.

Эти методы обработки и конструирования признаков могут быть использованы в зависимости от конкретной задачи и доступных данных, чтобы улучшить производительность модели машинного обучения.

Для большинства приложений при создании прикладных интеллектуальных систем этап сбора данных, их предобработки, выбора признаков, формирования модели выполняется как методами инженерии знаний (экспертом), так и с помощью **обучения представлений**.

1.4 Пример формирования обучающей выборки для разработки интеллектуальной системы предиктивной аналитики

Приведем пример формирования обучающей выборки для разработки интеллектуальной системы предиктивной аналитики, и именно для программного модуля выбора метода прогнозирования временных рядов.

Частью задачи прогнозирования временных рядов является выбор метода прогнозирования, адекватного имеющимся данным. Методы прогнозирования различаются по подходу, сложности и требуемым входным данным, а их количество продолжает расти. Каждый из методов прогнозирования работает со своей моделью временного ряда, выделяющей существенные для данного метода характеристики временного ряда.

Таким образом, зная необходимые характеристики временного ряда, возможно определить адекватные методы прогнозирования. На практике такая работа часто выполняется экспертом с использованием числовых значений свойств временного ряда, его графического представления, а также опыта эксперта.

В то же время подобная задача может быть решена с использованием нейронных сетей при наличии необходимых данных, формирующих обучающую выборку.

Обучающая выборка состоит из двух основных частей: входных данных и соответствующих им выходных данных. В качестве входных данных выступают числовые значения характеристик временного ряда, а в качестве выходных данных – адекватный метод прогнозирования временного ряда.

При отборе характеристик для формирования входных данных важно охватить как можно больше разнообразных аспектов временного ряда, не

ограничиваясь стандартными для многих моделей трендом, сезонностью и периодичностью.

Начать можно с простых статистических значений, таких как: длина временного ряда, медиана значений, среднее, дисперсия, стандартное отклонение. (Далее будет дано пояснение, почему не используются минимальное и максимальное значения).

Среди методов математической статистики можно рассмотреть такие параметры, как асимметрия (*skewness*) и эксцесс (*kurtosis*). Асимметрия при рассмотрении случайной величины измеряет отклонение величины от симметричного распределения, такого как нормальное распределение. Эксцесс в свою очередь можно рассматривать как меру приближения случайной величины к среднему значению. Значения асимметрии и эксцесса могут быть получены следующим образом

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \, , sm_2 = \sum_{i=1}^{n} (x_i - \bar{x})^2 \, , sm_3 = \sum_{i=1}^{n} (x_i - \bar{x})^3 \, , sm_4 = \sum_{i=1}^{n} (x_i - \bar{x})^4,$$

$$Skew = \frac{\sqrt{n} \times sm_3}{sm_2 \times \sqrt{sm_2}}, Kurt = \frac{n \times sm_4}{sm_2 \times sm_2},$$

где n – длина временного ряда, x_i – значение временного ряда, $Skew$ – асимметрия, $Kurt$ – эксцесс.

Минимальное значение эксцесса равно 1, а для нормально распределенной величины эксцесс равен 3. На практике используют скорректированную величину, называемую «избыточный эксцесс» (*Excess kurtosis*), рассчитываемую как

$$ExKurt = Kurt - 3,$$

для которой минимальное значение равно –2, а значение для нормально распределенной последовательности равно 0.

Значения асимметрии и избыточного эксцесса применяются в тесте Харке-Бера. Данная математическая статистика используется для проверки

наличия у набора данных стандартного распределения. Статистическая значимость (p-значение) статистики Харке-Бера при увеличении длины выборки асимптотически стремится к обратной функции распределения хи-квадрат с двумя степенями свободы. Значения рассчитываются следующим образом

$$JB = \frac{n}{6}\left(Skew^2 + \frac{1}{4}ExKurt^2\right), p = e^{-\frac{JB}{2}},$$

где JB – статистика Харке-Бера, p – статистическая значимость.

Для коротких последовательностей рассчитанное таким образом значение будет содержать значительную ошибку. Чтобы уменьшить ошибку, для коротких последовательностей применяется скорректированная версия теста Харке-Бера, рассчитываемая следующим образом

$$c_1 = \frac{6 \times (n-2)}{(n+1) \times (n+3)},$$

$$c_2 = \frac{3 \times (n-1)}{n+1}, c_3 = \frac{24 \times n \times (n-2) \times (n-3)}{(n+1)^2 \times (n+3) \times (n+5)},$$

$$AJB = \frac{Skew^2}{c_1} + \frac{(Kurt - c_2)^2}{c_3}, p = e^{-\frac{AJB}{2}},$$

где AJB – скорректированная статистика Харке-Бера, p – статистическая значимость.

Тест Дики-Фуллера используется для анализа временных рядов на стационарность, а также является одним из тестов для проверки на единичные корни. При помощи этого теста проверяют значение коэффициента a в авторегрессионном уравнении первого порядка

$$x_t = a \times x_{t-1} + \varepsilon_t,$$

где ε_t – ошибка. Если $a = 1$, то процесс имеет единичный корень. Приведенное авторегрессионное уравнение можно представить в виде

$$\Delta x_t = b \times x_{t-1} + \varepsilon_t,$$

где $b = a - 1$, а Δ – оператор разности первого порядка $\Delta x = x_t - x_{t-1}$. В тесте выполняется проверка нулевой гипотезы о равенстве нулю коэффициента b. Статистика для проверки значимости использует собственное распределение Дики-Фуллера.

Существуют три версии теста:

– Без константы и тренда:

$$\Delta x_t = b \times x_{t-1} + \varepsilon_t.$$

– С константой, но без тренда:

$$\Delta x_t = b_0 + b \times x_{t-1} + \varepsilon_t.$$

– С константой и линейным трендом:

$$\Delta x_t = b_0 + b_1 \times t + b \times x_{t-1} + \varepsilon_t.$$

Для каждой из версий используются свои критические значения статистики Дики-Фуллера. Если значение статистики лежит левее критического значения при данном уровне значимости, то нулевая гипотеза о единичном корне отклоняется.

Если в тестовые регрессии добавить лаги первых разностей временного ряда, то распределение статистики не изменится. Такой тест называют расширенным тестом Дики-Фуллера. Необходимость включения лагов первых разностей связана с тем, что процесс может быть авторегрессией не первого, а более высокого порядка. Данную модель можно представить в виде

$$\Delta x_t = (a_1 + a_2 - 1) \times x_{t-1} - a_2 \times \Delta x_{t-1} + \varepsilon_t.$$

Для проверки наличия единичных корней в данной модели следует провести стандартный тест для коэффициента при x_{t-1}, причем в тестовую регрессию должен быть добавлен лаг первой разности зависимой переменной.

В качестве альтернативы тесту Дики-Фуллера можно рассмотреть критерий KPSS. Критерий назван по первым буквам фамилий ученых

Квятковский-Филлипс-Шмитд-Шин (*Kwiatkowski-Phillips-Schmitd-Shin*). Он также основан на методах математической статистики и доступен в большинстве математических пакетов.

Для проверки тренда средних и дисперсий может использоваться критерий Фостера-Стюарта. Статистики критерия вычисляются как

$$S = \sum_{t=2}^{n} S_t \, , d = \sum_{t=2}^{n} d_t,$$

$$d_t = u_t - l_{t,} \, S_i = u_t + l_t.$$

Если $x_i > x_{i-1}, \dots, x_1$, то $u_i = 1$, иначе $u_i = 0$.

Если $x_i < x_{i-1}, \dots, x_1$, то $l_i = 1$, иначе $l_i = 0$.

Статистика S используется для проверки тренда дисперсий, а статистика d – для тренда средних. С их помощью рассчитываются следующие величины:

$$t = \frac{d}{f}, \tilde{t} = \frac{S - f^2}{l}, l = \sqrt{2 \ln n - 3.4253}, f = \sqrt{2 \ln n - 0.8456}.$$

При отсутствии тренда данные величины имеют распределение Стьюдента с n степенями свободы. Если $|t|, |\tilde{t}| > t_{\frac{1+\alpha}{2}}$, то с доверительной вероятностью α гипотеза существования тренда принимается.

В качестве альтернативы для проверки тренда средних и дисперсий может использоваться критерий Кокса-Стюарта.

Эти и другие величины вычисляются для каждого временного ряда в датасете. На текущем этапе решено использовать датасет соревнований по прогнозированию временных рядов *Computational Intelligence in Forecasting* (CIF) 2015.

Датасет содержит временные ряды из различных предметных областей. Также данные временные ряды имеют различную длину и частоту измерений: ежегодные, ежеквартальные, ежемесячные.

Дополнительным плюсом данного датасета, не считая того, что он подготовлен для профессиональных соревнований, является нацеленность соревнований на использование вычислительного интеллекта. Это ведет к тому, что не все подобранные временные ряды прогнозируются с одинаковой эффективностью классическими методами.

Данные в датасете заранее разделены на обучающую и проверочную выборки, поскольку на время проведения соревнований вторая часть была сокрыта. Этот факт упрощает подготовку данных перед началом исследования.

Для формирования выходных данных необходимо отобрать набор рассматриваемых методов прогнозирования. В рамках исследования принято использовать методы прогнозирования, представленные в пакете *Darts* для *Python* (таблица 1.1). Такое решение позволяет: во-первых, охватить большой набор различных методов, во-вторых, убедиться, что на точность прогнозов не влияют различия в реализации. Используются следующие модели:

Таблица 1.1. Методы прогнозирования, представленные в пакете *Darts* для *Python*

Метод прогнозирования временных рядов		
ARIMA	AutoARIMA	StatsForecastAutoARIMA
ExponentialSmoothing	StatsForecastAutoCES	Theta
FourTheta	StatsForecastAutoTheta	FFT
KalmanForecaster	Croston	RandomForest
RegressionModel	LinearRegressionModel	LightGBMModel
CatBoostModel	XGBModel	RNNModel
BlockRNNModel	NBEATSModel	NHiTSModel
TCNModel	TransformerModel	TFTModel
DLinearModel	NLinearModel	-

В дополнение к данным методам рассматривается набор методов прогнозирования на основе нечеткой логики, реализованный на кафедре «Информационные системы» УлГТУ [1]. Используются следующие модели:

– нечеткая модель Direct Set Assignment;

– нечеткая модель по [2] без тренда и сезонности;

– нечеткая модель по [2] аддитивный тренд, без сезонности;

– нечеткая модель по [2] без тренда аддитивная сезонность;

– нечеткая модель по [2] аддитивный тренд аддитивная сезонность;

– нечеткая модель по [3] без тренда и сезонности;

– нечеткая модель по [3] без тренда аддитивная сезонность;

– нечеткая модель по [3] без тренда мультипликативная сезонность;

– нечеткая модель по [3] аддитивный тренд, без сезонности;

– нечеткая модель по [3] аддитивный тренд аддитивная сезонность;

– нечеткая модель по [3] аддитивный тренд мультипликативная сезонность;

– нечеткая модель по [3] мультипликативный тренд, без сезонности;

– нечеткая модель по [3] мультипликативный тренд, аддитивная сезонность;

– нечеткая модель по [3] мультипликативный тренд, мультипликативная сезонность.

Определившись с характеристиками временных рядов и набором методов прогнозирования, можно перейти непосредственно к формированию обучающей выборки. Методика формирования включает следующие шаги.

1. **Нормализация данных.** Принято решение масштабировать временные ряды в диапазон [0.5; 1.5]. Такое решение позволяет характеристикам временных рядов различной природы быть в одинаковых диапазонах. Диапазон смещен от стандартных [0; 1], чтобы позволить применять мультипликативные модели прогнозирования. Также такое масштабирования делает характеристики минимума и максимума незначимыми.

2. **Расчет характеристик временных рядов.** Для каждого временного ряда выполняется расчет характеристик. Для математических статистик вычисляется разница между значением статистики и критическим значением таким образом, чтобы положительные значения подтверждали наличие соответствующих свойств (для единообразия). Эти данные формируют входные данные для нейронной сети.

3. **Прогнозирование временных рядов.** Для каждого временного ряда выполняется прогнозирование каждым из отобранных методов. Затем выполняется оценка точности прогнозов на основании проверочной выборки рассматриваемого датасета. В рамках текущей работы применяется SMAPE. Для каждого временного ряда выполняется ранжирование методов прогнозирования в соответствии с оценкой точности. Эти данные формируют выходные данные для нейронной сети.

Полученная обучающая выборка предназначена для обучения нейронных сетей различных архитектур для выбора адекватного метода прогнозирования временных рядов. Ранжированные по точности методы прогнозирования позволят проводить сравнения точности выбора адекватного метода.

На базе самой эффективной архитектуры в дальнейшем может быть построено программное обеспечение помощи эксперту и полный комплекс прогнозирования временных рядов.

1.5 Особенности конструирования признаков в задачах обработки текстов

В задачах **обработки текстов** признаки могут быть разделены на две основные категории: непосредственно наблюдаемые свойства и признаки для текста.

Непосредственно наблюдаемые свойства включают буквы, слоги, леммы и основы, N-граммы и другие лингвистические свойства, которые могут быть извлечены непосредственно из текста. Например, буквы и слоги представляют собой базовые элементы текста, которые могут быть использованы в качестве признаков. Леммы и основы — это лексические признаки, которые могут помочь в понимании значения слов. N-граммы — это последовательности из *n* соседних слов, которые могут быть использованы для представления контекста слов.

Признаки для текста включают мешок слов (*bag-of-words*), веса (*TF-IDF*), признаки слов в контексте (окна, позиция) и признаки, учитывающие отношения слов. Мешок слов представляет собой простой способ представления текста, где каждое слово представляется как отдельный признак. Веса (*TF-IDF*) используются для учета важности слов в контексте документа или корпуса. Признаки слов в контексте, такие как окна и позиция, позволяют учесть окружающую среду (контекст) слова и его роль в предложении. Признаки, учитывающие отношения слов, могут включать синтаксические структуры на уровне предложения, такие как фразово-структурное дерево или дерево зависимостей, а также дискурсные отношения, такие как кластеризация или идентификаторы кластера.

Производные лингвистические свойства включают классы слов, такие как метки частей речи или синтаксические роли, которые могут быть использованы для дополнительной информации о словах. Синтаксические структуры на уровне предложения, такие как фразово-структурное дерево

или дерево зависимостей, могут быть использованы для понимания связей между словами в предложении. Дискурсные отношения, такие как кластеризация или идентификаторы кластера, могут быть использованы для анализа связей между предложениями в тексте.

Выбор признаков в задачах обработки текстов зависит от конкретной задачи и доступных данных. Различные признаки могут быть использованы для улучшения производительности модели и извлечения полезной информации из текста (таблица 1.2).

Таблица 1.2. Соответствие задач обработки текстов и типа признаков

Задача обработки текстов	**Признаки**
Классификация документов: определение языка	Мешок буквенных биграмм, мешок байтовых биграмм (кодировки)
Классификация документов: тематическая	Мешок слов, мешок словесных биграмм
Классификация документов: установление авторства	Метки частей речи, плотность неполнозначных слов
Слово в контексте: частеречная разметка	Идентификатор слова, префиксы, суффиксы, орфографические формы слова
Слово в контексте: распознавание именованных сущностей	Окна фокусного слова, индикаторы, префиксы, суффиксы, орфографические формы слова
Слово в контексте, лингвистические признаки: разрешение лингвистической многозначности	Поверхностная форма слова, лемма, префиксы и суффиксы, признаки WordNet (индикаторы), гиперонимы первого синсета, всех синсетов, синонимы
Отношение между словами в контексте: анализ методом разложения на дуги (анализ зависимостей)	Комбинации признаков: слова, части речи, расстояние между словами, направление между словами

При использовании нейронных сетей для задач обработки текста используется специальный компонент нейронной сети – слой погружения.

Слой погружения — это компонент нейронной сети, который отображает дискретные символы (например, слова или предложения) на

непрерывные векторы в пространстве небольшой размерности. Он используется для представления текстовых данных в виде числовых векторов, что позволяет модели машинного обучения работать с текстом так же, как и с другими типами данных.

Существует несколько методов погружения, которые могут быть использованы для представления текстовых данных в виде векторов. Два наиболее распространенных метода погружения — это унитарное кодирование и плотное кодирование.

Унитарное кодирование представляет каждый символ в тексте в виде одного из N возможных состояний, где N — количество возможных символов. Каждое состояние представляется в виде вектора фиксированной длины, и каждый символ в тексте представляется в виде вектора, где значение каждого элемента соответствует наличию или отсутствию соответствующего состояния. Таким образом, унитарное кодирование позволяет представить текст в виде последовательности векторов фиксированной длины.

Плотное кодирование представляет каждый символ в тексте в виде вектора фиксированной длины, где каждый элемент вектора представляет собой вероятность появления соответствующего состояния. Это позволяет модели учитывать вероятность появления каждого символа и его контекст в тексте.

Кроме того, перед применением слоя погружения можно использовать различные методы предобработки текстовых данных. Например, непрерывный мешок слов (*bag-of-words*) представляет каждый документ в виде вектора, где каждый элемент соответствует наличию или отсутствию определенного слова в документе. Взвешенный мешок слов — это расширение непрерывного мешка слов, где каждый элемент вектора представляет собой вес или важность соответствующего слова в

документе. Эти методы позволяют представить текстовые данные в виде числовых векторов, которые могут быть использованы для обучения модели машинного обучения.

Отображение текстовых данных на вещественные векторы – это представление в **пространстве признаков**.

1.6 Предобученные языковые модели – метод и средства получения векторных представлений слов и предложений

Для создания классификатора текстов в современных системах часто используют **предобученную модель** для получения векторных представлений слов и предложений. Использование предобученной модели для получения векторных представлений слов и предложений является эффективным подходом для работы с текстовыми данными.

Процесс начинается с использования предобученной модели, такой как Word2vec, BERT или BERTRu, для получения векторных представлений слов и предложений. Эти модели обучаются на больших объемах текстовых данных и способны улавливать семантические и синтаксические связи между словами. Векторные представления слов и предложений могут быть использованы для представления текстовых данных в виде числовых векторов.

Затем для создания классификатора можно использовать простой алгоритм, такой как логистическая регрессия или метод опорных векторов (*SVM*), обученный на этих векторных представлениях. Векторы для каждого предложения или документа будут использоваться в качестве признаков для классификации.

Слова или предложения, которые имеют близкое **семантическое значение**, будут иметь близкие векторные представления в пространстве признаков. Расстояние между векторами может использоваться для определения близости или сходства между словами или предложениями.

Использование предобученных моделей и простых классификаторов позволяет эффективно работать с текстовыми данными и решать различные задачи обработки текста, такие как классификация, кластеризация или анализ тональности.

Предобученная модель представляет собой языковую модель. **Языковая модель** — это статистическое распределение, которое описывает вероятность появления следующего слова в предложении, исходя из его начального состояния. Она позволяет модели предсказывать следующее слово или предложение на основе предыдущих слов или предложений.

Существует несколько архитектур, которые могут быть использованы для построения языковых моделей, но **рекуррентные нейронные сети**, такие как *LSTM* (*Long Short-Term Memory*) и *GRU* (*Gated Recurrent Unit*), являются наиболее успешными. Эти модели способны улавливать долгосрочные зависимости в тексте и хорошо работают с задачами генерации текста, машинного перевода и другими задачами обработки естественного языка.

Предобученные языковые модели, такие как *ULMFit*, *ELMo* и *RuBERT*, позволяют переносить знания из одного языка или задачи на другой. Они обучаются на больших объемах текстовых данных и способны улавливать семантические и синтаксические связи между словами. Эти модели могут быть использованы для представления текстовых данных в виде векторных представлений, которые могут быть использованы для решения различных задач обработки текста.

Использование предобученных языковых моделей позволяет эффективно работать с текстовыми данными и решать различные задачи обработки текста, такие как классификация, генерация текста или машинный перевод. Они предоставляют мощный инструмент для работы с

естественным языком и продолжают развиваться и улучшаться с каждым годом.

Рекуррентные нейронные сети RNN являются популярным выбором для обработки естественного языка, так как они хорошо работают с последовательными данными, такими как текст. Однако одной из проблем RNN является проблема исчезающего и взрывоопасного градиента, которая может возникнуть при обработке длинных последовательностей.

В 2013 году Илья Суцкевер предложил схему ячейки LSTM-сети, которая позволяет RNN эффективно работать с длинными последовательностями. LSTM-сети имеют механизмы для контроля информации, которая проходит через них, и могут сохранять и использовать информацию из предыдущих состояний для принятия решений в будущем (рис. 1.1). Это делает их подходящими для задач обработки текста, где важно учитывать контекст и последовательность слов.

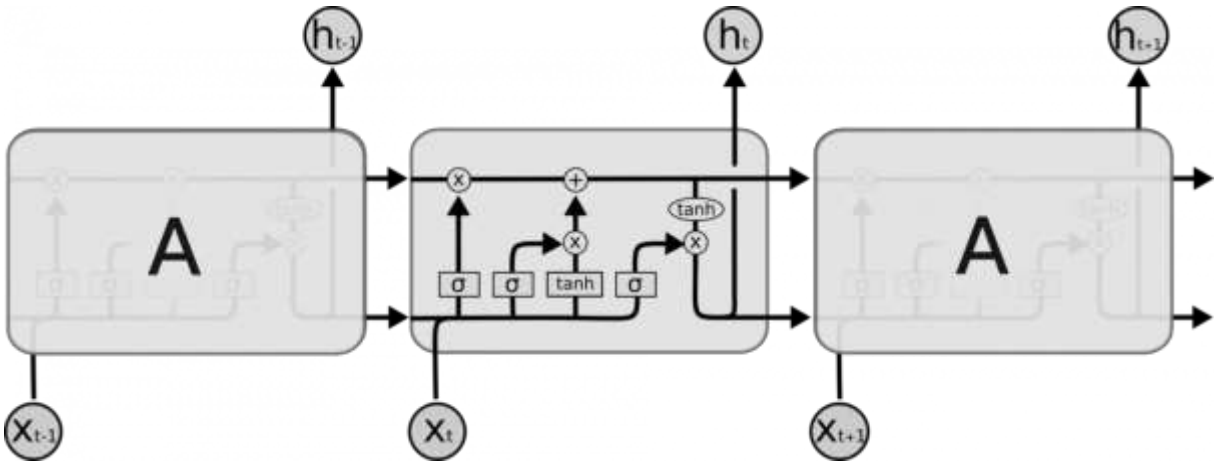

Рис. 1.1. Схема ячейки LSTM-сети

Двунаправленная LSTM-сеть, которая может работать одновременно с левым и правым контекстом, также является популярным вариантом для обработки текста. Она может использовать информацию из предыдущих и последующих слов для лучшего понимания смысла предложения или текста.

LSTM-сети и другие типы рекуррентных нейронных сетей являются мощным инструментом для обработки естественного языка и продолжают развиваться и улучшаться с каждым годом. Их способность учитывать последовательность и контекст делает их подходящими для широкого спектра задач обработки текста, включая классификацию, генерацию текста и машинный перевод.

***Universal Language Model Fine-tuning* (*ULMFiT*)** — это языковая модель, предложенная в 2018 году. Она основана на трехслойной архитектуре LSTM и имеет ряд преимуществ. Одним из ключевых преимуществ *ULMFiT* является небольшое количество параметров нейронной сети, составляющее около 3,5 миллиона. Это делает модель относительно легкой для обучения и позволяет использовать ее на видео-ускорителях архитектуры *Pascal GTX 1080 Ti* для корпусов, состоящих из нескольких миллионов текстов, в течение разумного времени.

Качество языковой модели *ULMFiT* также является значительным преимуществом. **Перплексия** предварительно обученной модели для русского языка на корпусе новостей интернет-ресурса «Лента.ру» составляет 21,98, а точность — 43%. Это говорит о том, что модель способна улавливать семантические и синтаксические связи в тексте и хорошо работает с задачами обработки естественного языка.

В обработке естественного языка перплексия — это способ оценки качества языковых моделей. Языковая модель — это распределение вероятностей по целым предложениям или текстам. Перплексия *PP* дискретного распределения вероятности *p* определяется как

$$PP(h) = 2^{H(h)} = 2^{-\sum_x p(x)\log_2 p(x)},$$

где *H(p)* — энтропия (в битах) распределения, *x* — диапазон событий.

Перплексия — это показательная функция от энтропии, которая является более точно определенной величиной. Энтропия — это мера ожидаемого, или «среднего» количества битов, необходимых для кодирования результата случайной переменной, например, при использовании теоретически оптимального кода переменной длины. Перплексия случайной переменной X может быть определена как перплексия распределения по ее возможным значениям x.

Реализация модели *ULMFiT* доступна в популярном пакете машинного обучения *FastAI* и входит в стандартный набор подключенных библиотек ресурса *GoogleColab*, который является крупнейшим бесплатным ресурсом для исследователей в области машинного обучения. Это делает модель доступной для широкого круга исследователей и разработчиков.

Кроме того, реализации модели *ULMFiT* позволяют проводить дополнительное обучение всех слоев модели, что дает возможность максимальной настройки модели под конкретную задачу. Это позволяет использовать модель для широкого спектра задач обработки естественного языка, включая классификацию, генерацию текста и машинный перевод.

Еще одним преимуществом модели *ULMFiT* является значительный объем ее словаря. Предварительно обученная модель для русского языка содержит словарь из 60 000 словоформ. Хотя лемматизация не используется, словарь модели может быть изменен и расширен без существенных потерь качества модели. Это делает модель гибкой и способной адаптироваться к различным задачам и данным.

Модель *ULMFiT* представляет собой высококачественную языковую модель, которая имеет ряд преимуществ, таких как небольшое количество параметров, высокое качество, доступность реализации и гибкость в настройке и расширении словаря. Она может быть использована для

широкого спектра задач обработки естественного языка и продолжает развиваться и улучшаться с каждым годом (рис. 1.2).

***ELMo* (*Embeddings from Language Model*)** — это модель, которая предоставляет **контекстуализированные векторные представления слов**, предварительно обучая языковую модель без контроля. Она состоит из двухслойного двунаправленного кодера *LSTM* и основного модуля прогнозирования (рис. 1.3). Главное преимущество *ELMo* заключается в том, что она способна генерировать векторные представления слов, которые учитывают контекст фразы, в которой они были использованы. Это делает ее подходящей для широкого спектра задач обработки естественного языка, таких как классификация, генерация текста и машинный перевод.

Словарь *ELMo* состоит из букв языка, на котором обучается модель. Если требуется обучить мультиязычную модель, можно просто объединить алфавиты всех языков, на которых будет использоваться модель. Это делает модель гибкой и способной адаптироваться к различным языкам и задачам.

Для русского языка существуют как мультиязычные, так и специально обученные варианты модели *ELMo*. Это означает, что можно использовать модель, предварительно обученную на большом объеме текстов на русском языке, для получения лучших результатов на этом языке.

В целом, *ELMo* — это мощная модель, которая способна генерировать контекстуализированные векторные представления слов и хорошо работает с различными задачами обработки естественного языка. Ее гибкость и способность адаптироваться к различным языкам делают ее популярным выбором для многих приложений, рис. 1.3.

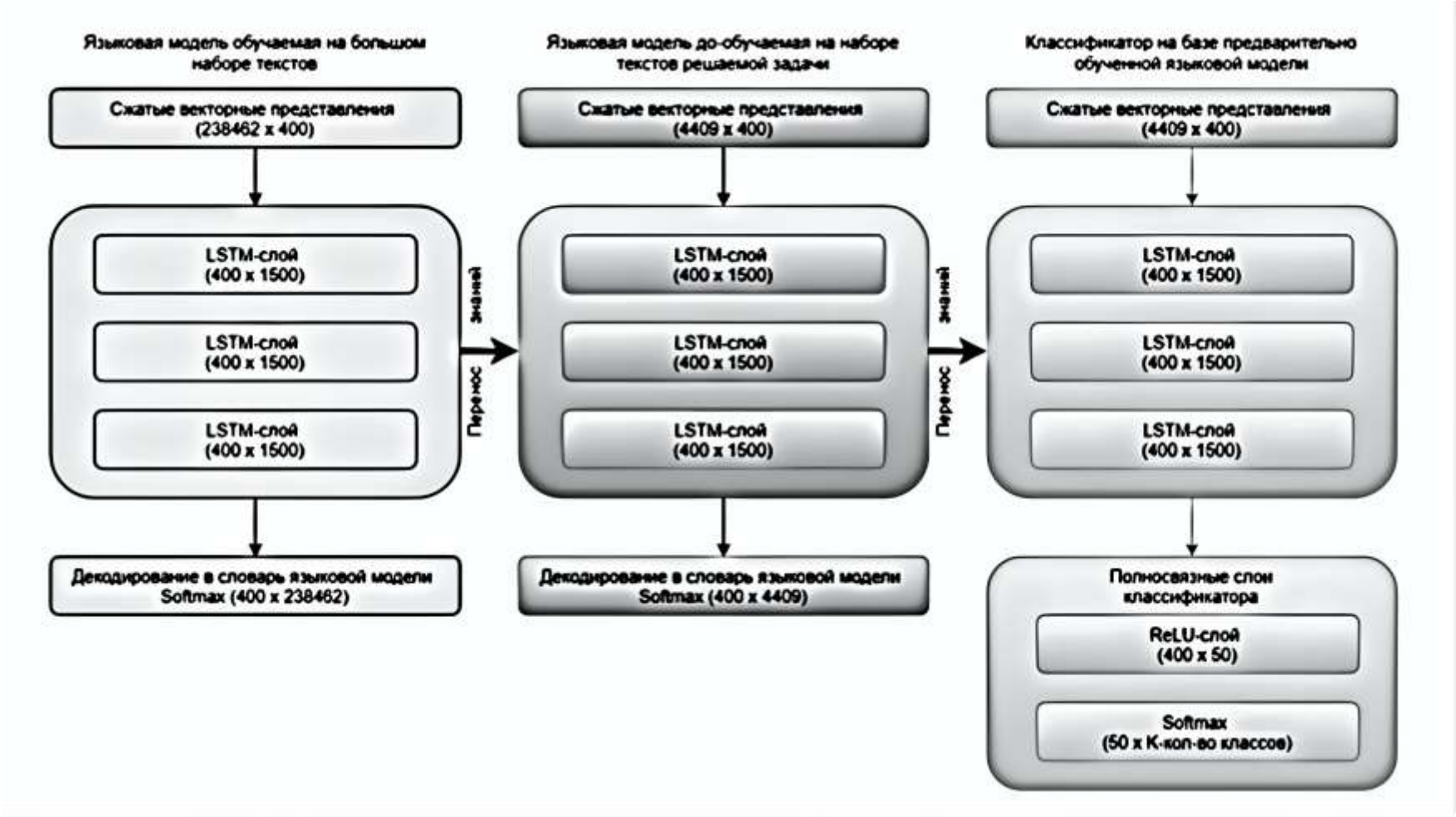

Рис. 1.2. Схема работы с предобученной языковой моделью.

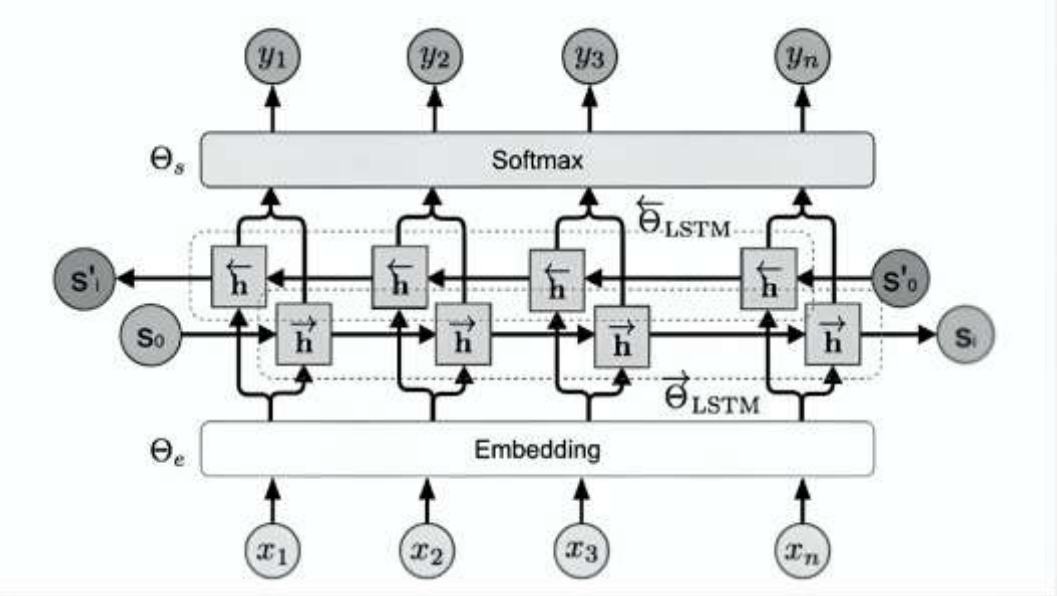

Рис. 1.3. Ячейка *ELMo*

BERT **(*Bidirectional Encoder Representations from Transformers*)** — это модель, разработанная и обученная компанией *Google*, которая использует архитектуру **трансформера** для создания контекстуализированных векторных представлений слов (рис. 1.4, рис 1.5). Она состоит из двух взаимосвязанных компонентов: **кодеров и декодеров**, которые позволяют преобразовывать последовательность входных слов в последовательность выходных слов.

Модель *BERT* была обучена на двух больших наборах данных: одном на английском языке и другом на нескольких языках. Обучение проводилось без контроля, что позволило модели изучить широкий спектр семантических и синтаксических отношений в языке. Основное преимущество модели *BERT* заключается в ее способности генерировать векторные представления слов, которые учитывают контекст предложения, в котором они были использованы. Это делает ее подходящей для широкого спектра задач обработки естественного языка, таких как классификация, генерация текста и машинный перевод.

В модели *BERT* используется уникальный подход к представлению слов, который включает разделение каждого слова на отдельные морфемы и представление каждой морфемы в виде вектора. Это позволяет модели работать с ранее неизвестными словами, распознавая в них известные морфемы. Модель BERT имеет огромное количество параметров (100 миллионов для базового варианта и 340 миллионов для расширенного варианта), что позволяет ей строить наиболее полную языковую модель для английского языка. На этой модели были получены рекордные результаты по основным задачам обработки естественного языка.

В целом, модель *BERT* — это мощный инструмент для обработки естественного языка, который способен генерировать контекстуализированные векторные представления слов и хорошо работает с различными задачами. Ее способность работать с различными языками и адаптироваться к различным задачам делает ее популярным выбором для многих приложений.

***RuBERT* – средство генерации контекстуализированных векторных представлених слов на русском языке**

RuBERT — это мультиязычная версия модели BERT, которая была дополнительно обучена на русскоязычном корпусе интернет-ресурса «Лента.ру». Она использует ту же архитектуру и компоненты, что и модель *BERT*, но с некоторыми изменениями в конфигурации и обучении.

Основным преимуществом *RuBERT* является то, что она способна генерировать **контекстуализированные векторные представления** слов на русском языке, учитывая семантические и синтаксические отношения в предложении. Это делает ее подходящей для широкого спектра задач обработки естественного языка на русском языке, таких как классификация, генерация текста и машинный перевод.

RuBERT обучена на большом объеме текстов на русском языке, что позволяет ей улавливать особенности и закономерности языка. Это делает ее более подходящей для работы с русскоязычными данными по сравнению с моделями, которые не были обучены на русском языке. В 2020 году мультиязычная модель *BERT* на 12 слоев кодеров дополнительно обучена на русскоязычном наборе текстов интернет-ресурса «Лента.ру» и доступна под названием *RuBERT*. Основными достоинствами модели *RuBERT* являются:

1. Использование токенов-морфем в качестве словаря языковой модели. Каждое слово на входе в модель раскладывается на ряд токенов – морфем специфичных для используемого естественного языка. Потенциально такой подход позволяет модели работать с ранее неизвестными для модели словами за счет распознавания в них известных морфем.

2. Большое число параметров нейронной сети (100 миллионов для базового варианта и 340 миллионов для расширенного варианта модели) позволило построить наиболее полную языковую модель. На данной модели были получены рекордные результаты по основным задачам обработки естественного языка

В целом, *RuBERT* — это мощная модель для обработки естественного языка на русском языке, которая способна генерировать контекстуализированные векторные представления слов и хорошо работает с различными задачами. Ее способность улавливать семантические и синтаксические отношения в предложении делает ее ценным инструментом для исследователей и разработчиков, работающих с русским языком, рис. 1.4, 1.5.

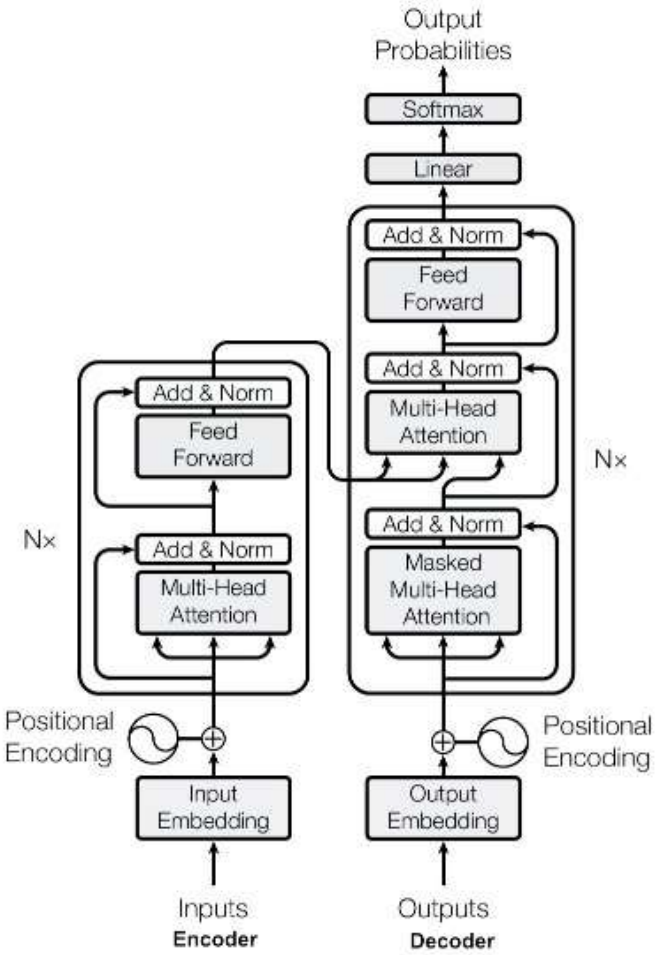

Рис. 1.4. Архитектура трансформера – взаимодействие кодеров и декодеров

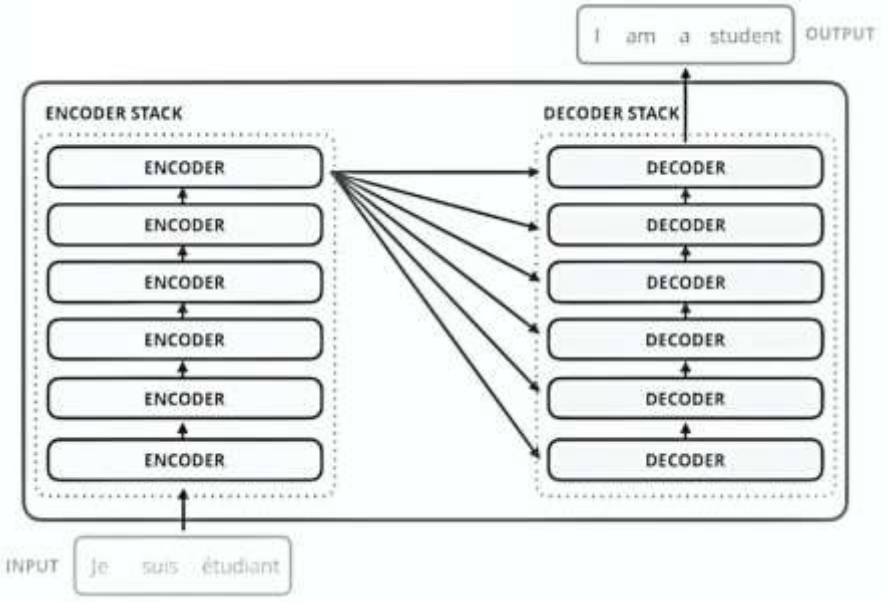

Рис. 1.5. Архитектура трансформера для создания контекстуализированных векторных представлений слов

2 ПОДХОД К КОНСТРУИРОВАНИЮ ПРИЗНАКОВ

Для получения качественных моделей с использованием методов машинного обучения требуется объем данных для обучения, достаточный для выявления в данных скрытых закономерностей [4–10]. Если данных будет недостаточно, то модель не сможет достичь необходимого уровня обобщения, что скажется на снижении качества модели. Также на качество полученной модели влияет и качество самих данных.

Однако самой важной с точки зрения получения качественной модели является стадия конструирования признаков [11]. Под конструированием признаков понимается определение значимых с точки зрения решаемой задачи сущностей и их атрибутов. Далее для каждого такого атрибута осуществляется сопоставление с той частью данных, которая описывает его исторические значения. От аналитика требуется знание предметной области, существенный опыт в области интеллектуального анализа данных и машинного обучения.

Большинство методов машинного обучения работают только с представлением признаков в виде векторов с числовыми значениями [11]. Таким образом, на этапе конструирования признаков аналитику требуется сформировать таблицу с данными, каждый столбец которой представляет признак, а строка – отдельное наблюдение (состояние анализируемого объекта в некоторый момент времени). Каждый столбец такой таблицы должен быть представлен числом. В случае категориальных данных можно использовать унитарное кодирование, преобразующее категориальный атрибут в одно или несколько бинарных значений. Для текстовых данных и изображений используются отдельные подходы. Для текстовых данных можно использовать мешок слов, хэширование признаков, тематическое моделирование или погружение слов (word embeddings).

В настоящее время существуют программные средства для автоматического конструирования признаков для реляционных данных, например, Featuretools [12]. При использовании таких средств новые признаки формируются на основе комбинирования существующих и применения различных функций преобразования данных (получение логарифма, возведение в степень, получение количества уникальных значений и т. д.) и функций агрегации (сумма, среднее, максимальное значение и т. д.). При конструировании признаков таким образом не учитываются особенности предметной области и значительно повышается размерность пространства признаков.

В случае отсутствия обучающей выборки данных достаточного объема применяются различные методы синтезирования и аугментации данных [4–11, 13–24].

Для изображений применяются алгоритмы поворота и смещения изображения, добавление шумов и т. д. Для реляционных данных используются методы на основе скрытых цепей Маркова, симуляторов процессов для определенных предметных областей, моделей машинного обучения и т. д. [6–9].

Отдельно можно выделить применение методов Zero/Few-Shot Learning и Transfer Learning [25–29], когда ранее обученную модель применяют для решения похожей проблемы с дополнительным обучением на небольшом количестве примеров.

При конструировании признаков также может возникнуть проблемы связанные с большой размерностью пространства признаков, несбалансированностью выборки и переобучением [6, 11, 30–35]. Размер пространства признаков влияет на скорость и вычислительную сложность процесса обучения модели, а также на скорость работы полученной модели. Большое количество признаков может привести к переобучению

модели и снижению ее качества. Несбалансированная выборка приводит к искусственному завышению результатов оценки качества работы модели за счет наличия класса с подавляющем числом примеров.

Следовательно, существуют отдельные области, в которых применение методов машинного обучения затруднено отсутствием обучающей выборки достаточной для обучения. Однако возможно привлечь эксперта предметной области, который на основе априорных знаний о предметной области сможет построить модель анализируемого объекта для синтеза обучающей выборки достаточного объема.

Таким образом, требуется разработка подхода к конструированию признаков в машинном обучении для данных с учетом высокой степени неопределенности и ограничений предметной области с учетом вышеозначенных проблем.

2.1 Получение и подготовка данных для анализа

Для решения задач с помощью методов машинного обучения на первом шаге необходимо получить и подготовить данные для анализа, на основе которых модель будет обучаться. Для большинства задач доступны открытые источники данных, представленные на специализированных ресурсах в сети Интернет, например Kaggle [36] или Hugging Face [37]. Также данные ресурсы содержат уже обученные модели, которые можно использовать как есть, или применять в совокупности с методами Zero/Few-Shot Learning и Transfer Learning.

Однако для отдельных задач и/или предметных областей могут отсутствовать обученные модели и подготовленные данные для обучения.

Например, для решения задач в рамках производственного предприятия необходимо собрать данные, а затем подготовить их для конструирования признаков.

Современные предприятия в большинстве случаев имеют фрагментарную автоматизацию, при которой исторические данные атрибутов сущностей предметной области хранятся в различных информационных системах. Для решения задач в области машинного обучения такие данные необходимо извлечь из отдельных информационных систем, отобразить в модель данных, пригодную для проведения анализа, а также предобработать данные для исключения пропусков, выбросов и других ошибок.

Таким образом, для решения задачи конструирования признаков необходимо автоматизировать процессы извлечения, трансформации и загрузки данных (ETL, Extract, Transform, Load) из информационных систем в некоторое локальное хранилище.

Системы интеграции данных обеспечивают интеграцию на разных уровнях [38]: физическом, логическом и семантическом. Интеграция на физическом уровне позволяет преобразовать данные из разных источников в унифицированный формат представления. Интеграция на логическом уровне позволяет получать доступ к данным из разных источников доступны через единую точку входа. Интеграция на семантическом уровне позволяет представлять данные с учетом их семантики.

Каждый из представленных выше уровней интеграции имеет свои проблемы. На физическом уровне могут использоваться разные форматы представления данных. На логическом уровне может присутствовать неоднородность использованных моделей и/или схем данных. На семантическом уровне различным источникам данных могут соответствовать разные предметные области или фрагменты областей.

Для решения задачи построения интегрирующей модели данных часто применяются методы Linked Data. Эти методы включают использование унифицированных идентификаторов ресурсов, организацию доступа

к ресурсам по протоколу HTTP, использование стандартных технологий Semantic Web (RDF, OWL, SWRL, SPARQL) и использование гиперссылок для идентификации сущностей предметной области, включая не только веб-документы.

Для реализации процедуры ETL для получения и подготовки данных была разработана следующая информационная модель представления метаданных источников и приемников данных:

$$M^{ETL} = \langle S^{ETL}, D^{ETL}, F^{ETL}, R^{ETL} \rangle,$$

где $S^{ETL} = \{S_1^{ETL}, S_2^{ETL}, \dots, S_i^{ETL}, \dots, S_n^{ETL}\}$ – множество источников данных;

$S_i^{ETL} = \langle url, rpc, scheme \rangle$ – i-й источник данных, в котором указан адрес для подключения к адаптеру url, удаленная процедура rpc, которую необходимо вызвать для получения данных, и схема $scheme$, которая описывает формат возвращаемых данных. Для каждого источника данных требуется создание адаптера. Адаптер позволяет скрыть детали реализации источника данных и упростить процесс доступа к ним;

$D^{ETL} = \{D_1^{ETL}, D_2^{ETL}, \dots, D_j^{ETL}, \dots, D_n^{ETL}\}$ – множество точек назначения в локальном хранилище данных (приемнике) для загрузки данных;

$D_j^{ETL} = \langle table, column, transformation \rangle$ – j-й приемник данных в локальном хранилище, в котором указана таблица назначения $table$, колонка назначения таблицы $column$ (содержит название колонки и тип данных) и функция трансформации данных $transformation$. Функция трансформации $transformation$ позволяет привести данные к нужному типу и выполнить дополнительные простые преобразования, например изменение регистра, возведение в степень и т. д.;

$F^{ETL} = \{F_1^{ETL}, F_2^{ETL}, \dots, F_k^{ETL}, \dots, F_n^{ETL}\}$ – множество функций для постобработки локального хранилища после выполнения операций извлечения и трансформации данных;

$F_k^{ETL} = \langle table, processing \rangle$ – k-я функция для постобработки загруженных в локальное хранилище данных, которая применяет алгоритм постобработки $processing$ к таблице $table$ локального хранилища. Функции постобработки позволяют решать задачи, связанные с удалением строк таблицы с пропущенными значениями, заполнять пропущенные значения средними значениями, удалять строки с аномальными значениями и выбросами и т. д.;

$R^{ETL} = \langle R_{DS}^{ETL}, R_{DF}^{ETL} \rangle$ – множество отношений для указания связей R_{DS}^{ETL} между источниками и приемниками, а также между приемниками и функциями постобработки R_{DF}^{ETL}.

Для выполнения процедуры ETL был разработан алгоритм, состоящий из следующих шагов:

1. Создание пустого множества следующего вида:

$$Dest = \{\langle table1, D_i^{ETL} \rangle, \langle table1, D_j^{ETL} \rangle, \dots, \langle table2, D_k^{ETL} \rangle\}.$$

Каждый элемент множества $Dest$ содержит название таблицы локального хранилища $table$, в которую должны быть записаны полученные данные, и множество приемников из множества D^{ETL} для определения перечня колонок данной таблицы и функций для трансформации данных перед записью.

1. Каждый приемник D_i^{ETL} из множества D^{ETL} добавляется в множество $Dest$ для определения таблиц локального хранилища для заполнения и перечня их колонок. Перечень колонок определяется путем группировки словаря по названию таблицы. Затем происходит создание таблиц и их колонок. Если таблица или колонка уже существуют в локальном хранилище, то выводится сообщение об ошибке.

2. Для каждого приемника D_i^{ETL} из множества D^{ETL} производится поиск источника из множества S^{ETL} на основе отношений R_{DS}^{ETL}. Если для приемника не указан источник, то выводится сообщение об ошибке.

3. Для каждой таблицы из множества $Dest$ производится загрузка данных из источника S_i^{ETL} и запись в приемник D_i^{ETL}, при этом выполняется соответствующая функция трансформации данных источника D_i^{ETL}.

4. Для каждой таблицы из множества $Dest$ производится выполнение функций постобработки из множества F^{ETL} с учетом отношений R_{DF}^{ETL}.

Представленный алгоритм может выполняться через определенные интервалы времени для формирования данных в формате многомерных временных рядов.

На рисунке 2.1 представлена схема, иллюстрирующая процесс загрузки и подготовки данных на основе предложенного метода ETL.

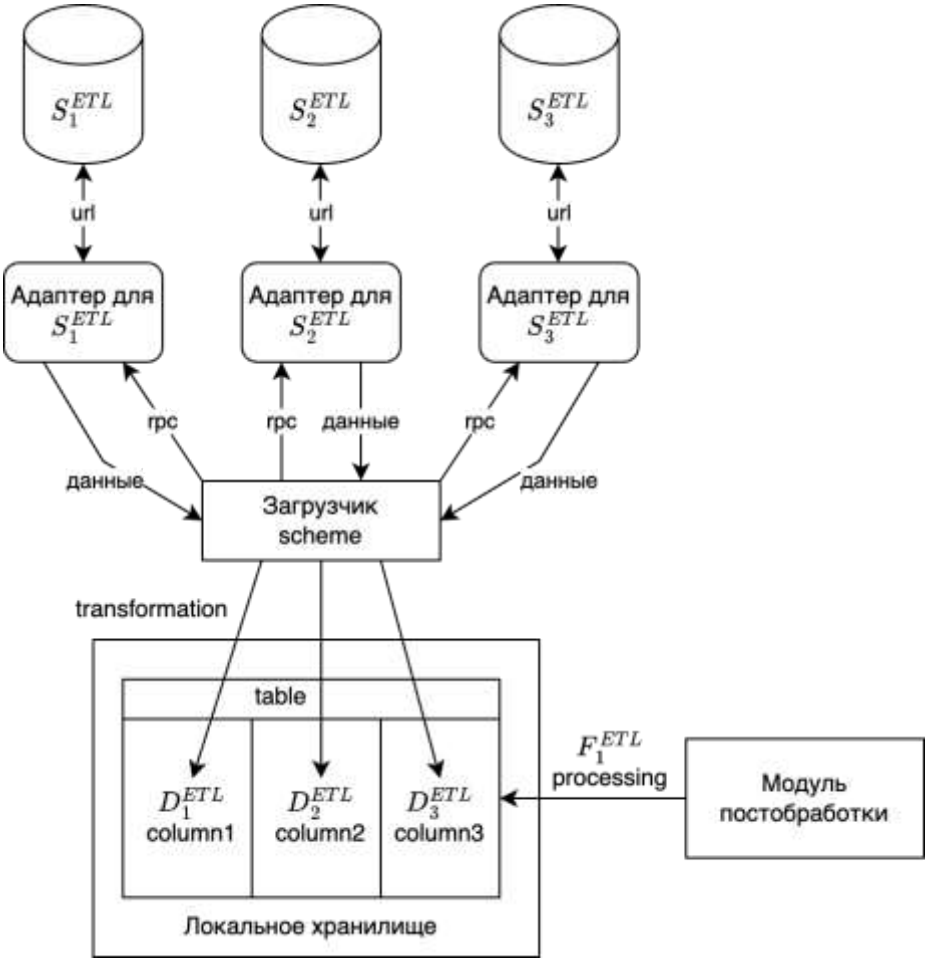

Рис. 2.1. Схема загрузки и подготовки данных для анализа

Таким образом, после выполнения данного алгоритма локальное хранилище содержит подготовленные данные для анализа методами машинного обучения.

2.2 Конструирование признаков с учетом высокой степени неопределенности и ограничений предметной области

Конструирование признаков является творческим процессом, который требует от аналитика глубокого знания предметной области и значительного опыта в области интеллектуального анализа данных и машинного обучения [11]. При решении новой задачи аналитик может находиться в ситуации с высокой степенью неопределенности, что требует от него применение современных методов конструирования, версионирования, документирования и хранения признаков. Например, в компании Uber используется платформа для машинного обучения Michelangelo, которая включает хранилище признаков [11]. Хранилища признаков позволяют фиксировать изменения как в самих признаках, так и в полученных моделях машинного обучения. При ухудшении качества модели всегда можно отследить внесенные изменения и проанализировать их.

Как было сказано ранее, современные средства для автоматического конструирования признаков, например Featuretools [12], основаны на применении простых операций преобразования данных, их агрегирования и комбинирования, что позволяет получить большое количество признаков. Однако полученные признаки могут не соответствовать ограничениям предметной области, повышают размерность пространства признаков и затрудняют процесс выбора признаков для формирования модели повышая степень неопределенности. В качестве существенного преимущества подобных инструментов можно выделить минимальные временные затраты и повышение степени полноты полученных признаков.

Следовательно, конструирование признаков предполагает учет особенностей, как самого анализируемого объекта, так и предметной области, в условиях ограничений которой функционирует объект.

Для решения задачи конструирования признаков в условиях высокой степени неопределенности и с учетом ограничений предметной области была разработана информационная модель контекста, которая позволяет эксперту предметной области выделить важные с его точки зрения сущности предметной области и их атрибуты.

Для представления предложенной модели будем использовать дескрипционную логику [39]. В таблице 2.1 представлены операторы и аксиомы дескрипционной логики, которые используются для описания предложенной модели контекста, а также их соответствие операторам языка представления онтологий OWL 2 [40].

Таблица 2.1. Операторы и аксиомы дескрипционной логики

Описание	**Дескприпционная логика**	**OWL**
Класс верхнего уровня (предок всех классов)	$\top$	owl:Thing
Пустой класс	$\bot$	owl:Nothing
Аксиома включения классов	$A \sqsubseteq B$	A owl:SubClassOf B
Аксиома непересекающихся классов	$A \sqcap B \sqsubseteq \bot$	[A, B] owl:DisjointClasses
Аксиома эквивалентности классов (или определение классов с необходимыми и достаточными условиями)	$A \equiv B$	[A, B] owl:equivalentClasses
Пересечение или соединение классов	$A \sqcap B$	A and B
Аксиома универсального ограничения	$\forall R.A$	R only A
Аксиома экзистенциального ограничения	$\exists R.A$	R some A
Аксиома ограничения кардинальности	$\leq n\, R.A$	R exactly n A
Аксиома определения индивидуальности (a является экземпляром класса A)	$a : A$	a: A
Аксиома определения роли	$(a, b) : R$	a R b

С точки зрения дескрипционной логики предложенную модель контекста можно представить следующим образом:

$$D = \langle D^{TBox}, D^{ABox} \rangle,$$

где D^{TBox} – множество аксиом определения терминологии модели;

D^{ABox} – множество фактов (утверждений, аксиом) модели.

Рассмотрим более подробно терминологию D^{TBox} предложенной модели контекста.

Определим общие для всех классов модели свойства (роли):

$$\top \sqsubseteq \exists hasName.String \sqcap \forall hasName.String \sqcap= 1\ hasName.String,$$

где $hasName$ –функциональная роль «имеет название».

Терминология D^{TBox}содержит следующие классы:

– $Object$ – класс для описания анализируемых объектов;

– $Attribute$ – класс для описания атрибутов анализируемого объекта;

– $Dependency$ – класс для описания зависимостей между атрибутами нескольких анализируемых объектов;

– $DependencyType$ – класс для описания типов зависимостей (степень влияния): позитивная, негативная:

$$Object \sqsubseteq \top$$

$$Attribute \sqsubseteq \top$$

$$Dependency \sqsubseteq \top$$

$$DependencyType \equiv \{positive, negative\}.$$

Классы $Object$, $Attribute$, $Dependency$ и $DependencyType$ объявлены непересекающимися:

$$Object \sqcap Attribute \sqcap Dependency \sqcap DependencyType \sqsubseteq \bot.$$

Для класса $Object$ указана роль $hasAttribute$ для определения перечня атрибутов анализируемого объекта:

$$Object \sqsubseteq \forall hasAttribute.Attribute.$$

Класс $Object$ имеет следующие роли:

– $hasMin$ и $hasMax$ – функциональные роли для определения минимального и максимального значения атрибута соответственно;

– $hasSource$ – функциональная роль для определения колонки таблицы локального хранилища, в которой хранятся значения данного атрибута;

– $hasDependency$ – роль для определения зависимостей между атрибутами:

$$Attribute \sqsubseteq \forall hasMin.Double \sqcap= 1\ hasMin.Double \sqcap$$
$$\sqcap \forall hasMax.Double \sqcap= 1\ hasMax.Double \sqcap$$
$$\sqcap \exists hasSource.String \sqcap \forall hasSource.String \sqcap= 1\ hasSource.String \sqcap$$
$$\sqcap \forall hasDependency.Dependency.$$

Для класса $Dependency$ были созданы следующие роли:

– $dependsOn$ – функциональная роль для определения зависимости между атрибутами;

– $hasType$ – функциональная роль для определения типа зависимости между атрибутами;

– $hasWeight$ – функциональная роль для определения веса зависимости:

$$Dependency \sqsubseteq \exists dependsOn.Attribute \sqcap \forall dependsOn.Attribute \sqcap$$
$$\sqcap = 1\ dependsOn.Attribute \sqcap$$
$$\sqcap \exists hasType.DependencyType \sqcap \forall hasType.DependencyType \sqcap$$
$$\sqcap = 1\ hasType.DependencyType \sqcap$$
$$\sqcap \exists hasWeight.Double \sqcap \forall hasWeight.Double \sqcap=$$
$$1\ hasWeight.Double.$$

Множество фактов D^{ABox} предложенной модели контекста формируются автоматизировано с применением специальных экранных форм для облегчения работы эксперта.

Множество фактов D^{ABox} формируется при выполнении следующих шагов:

1. Определение перечня анализируемых объектов:

$$object1 : Object$$

$$(object1, 'Some\ object'): hasName.$$

2. Создание списка атрибутов для каждого анализируемого объекта с указанием источника данных и допустимого диапазона значений:

$$attribute1 : Attribute$$

$$(attribute1, 'Some\ attribute'): hasName$$

$$(attribute1, 0.5): hasMin$$

$$(attribute1, 11.6): hasMax$$

$$(attribute1, 'table1.column2'): hasSource$$

$$(object1, attribute1): hasAttribute.$$

3. Определение зависимости анализируемых объектов между собой на уровне атрибутов:

$$dependency1 : Dependency$$

$$(dependency1, attribute2): dependsOn$$

$$(dependency1, positive): hasType$$

$$(dependency1, 0.85): hasWeight$$

$$(attribute1, dependency1): hasDependency.$$

Для хранения сведений о признаках на основе предложенной модели контекста используется язык представления онтологий OWL 2 [40]. Для хранения истории изменения сведений о признаках используется отдельная таблица локального хранилища.

На рисунке 2.2 представлена схема, иллюстрирующая фрагмент предложенной модели контекста.

Таким образом, после выполнения шагов, представленных выше, эксперт сформирует такое представление признаков, которое учитывает

ограничения предметной области, и которое позволяет реализовать функции хранения, версионирования и документирования признаков. Разработанная программная система для автоматизации работы эксперта позволяет снизить временные затраты процесса конструирования признаков, а возможность хранения признаков в локальном хранилище позволяет использовать их повторно в похожих задачах.

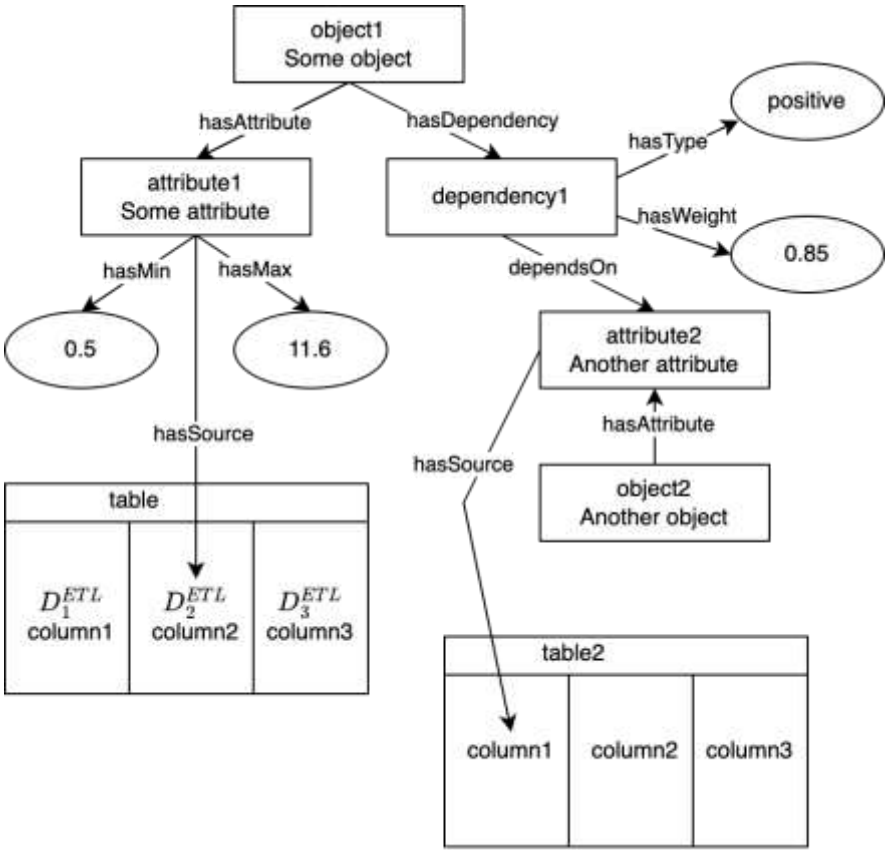

Рис. 2.2. Фрагмент предложенной модели контекста для конструирования признаков

2.3 Синтез обучающей выборки на основе мультиагентного подхода

После конструирования признаков может возникнуть ситуация, когда обучающая выборка мала или не сбалансирована. В результате получить модель машинного обучения с приемлемым показателем качества становится затруднительно [4, 9]. В современной практике используются различные подходы для решения данной проблемы: аугментация [13–15],

синтез обучающей выборки [16–24], zero/few-shot learning [25], transfer learning [26–29] и т. д.

Предложенный метод синтеза обучающей выборки основан на мультиагентном подходе. Мультиагентный подход позволяет создавать универсальные и гибкие системы, способные адаптироваться к изменениям как в предметной области, так и во внешней относительно системы среде [41–43].

В рамках мультиагентной системы каждый агент может представлять собой сущность целиком или некоторые ее атрибуты для синтеза обучающей выборки.

Формально систему для синтеза обучающей выборки на основе теоретико-множественного представления можно записать как:

$$Synth = \langle A^{Synth}, E^{Synth} \rangle,$$

где A^{Synth} – множество агентов, каждый из которых предмтавляет некоторое простое явление с точки зрения предметной области, в котором участвует сущность, или который влияет на атрибуты сущности. Каждый агент содержит реализацию алгоритма, который принимает на вход необходимые внешние параметры, а на выходе выдает значения отдельных признаков;

E^{Synth} – множество связей между агентами. Связи позволяют связывать вход и выход отдельных агентов, также к связям может быть привязан отдельный алгоритм, работа которого может вносить корректировки в передаваемые между агентами значения. Таким образом, появляется возможность эмуляции воздействия внешней среды на систему.

Фактически представленная система агентов для синтеза обучающей выборки представляет собой ориентированный ациклический граф и позволяет моделировать поведение сущностей и процессов некоторой предметной области.

Так как каждый объект представляет собой изолированную сущность появляется возможность использовать в качестве агентов различные системы симуляции и моделирования, что позволяет синтезировать обучающую выборку максимально приближенную к реальным данным.

Для учета нечеткости и неопределенности, а также для облегчения создания отдельных объектов предусмотрена возможность использования механизмов нечеткой логики и нечетких множеств при создании агентов и связей системы синтеза обучающей выборки. Например, нечеткие множества могут использоваться для преобразования категориальных значений в числовые при выполнении дефаззификации. Также предусмотрена возможность использования механизмов нечеткого логического вывода для реализации агентов с учетом особенностей контекста [44].

На рисунке 2.3 представлен иллюстративный пример системы для синтеза обучающей выборки на основе предложенного мультиагентного подхода.

Как видно из рисунка 2.3, система состоит из трех уровней. На третьем уровне находится три агента: агент 3.1 выполняет некоторый алгоритм генерации значений, агент 3.2 взаимодействует со средой моделирования для получения значений, а агент 3.3 получает данные с датчика. Далее агенты третьего уровня передают данные на второй уровень агенту 2.1. При передаче данных от агента 3.1 агенту 2.1 в случайный момент времени в передаваемое значение могут быть внесены помехи, что также реализовано алгоритмически.

Агент 2.1 выполняет алгоритм для агрегации полученных от агентов третьего уровня значений. Агент 2.2 взаимодействует с симулятором некоторого физического явления. Агенты второго уровня передаю значения на первый уровень агенту 1.

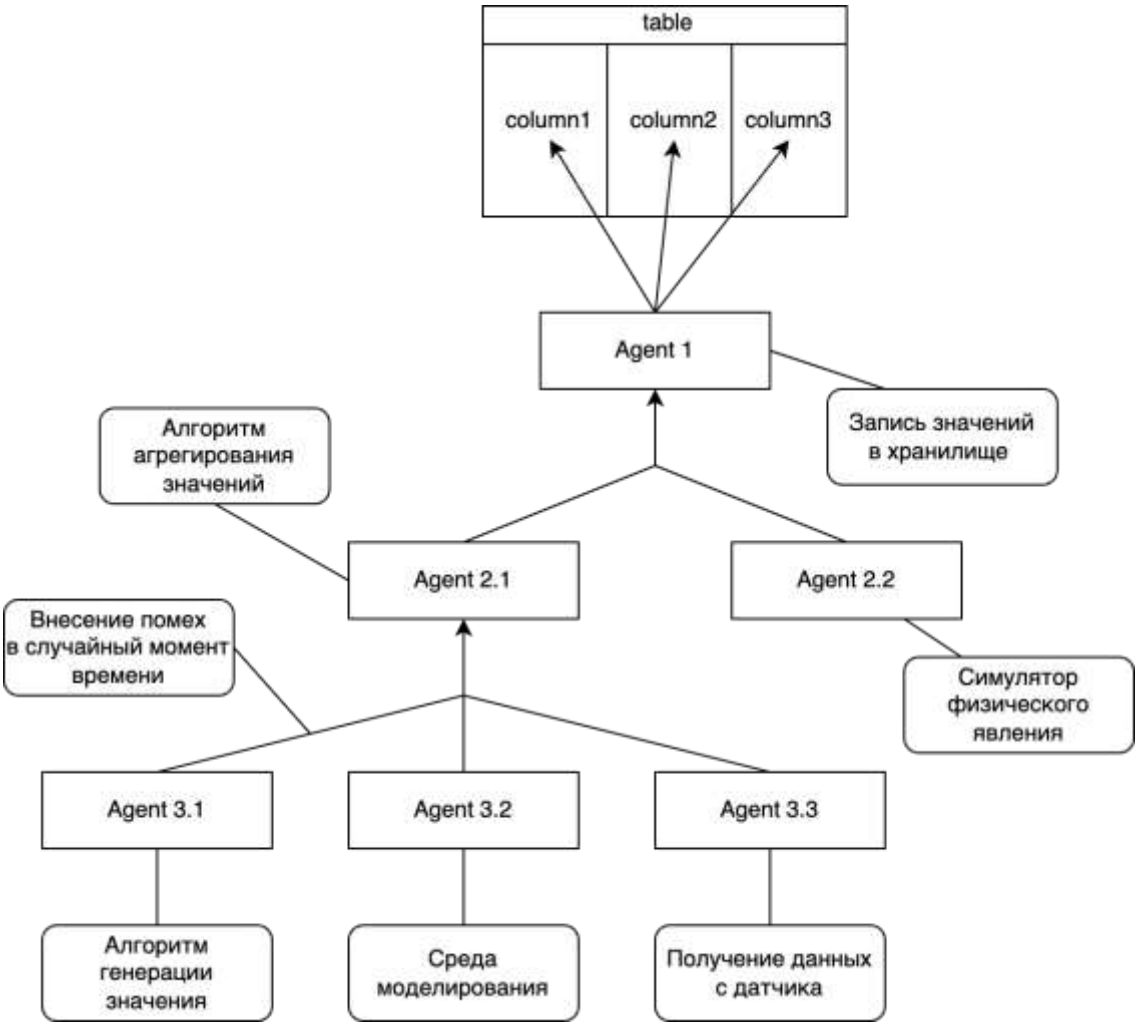

Рис. 2.3. Иллюстративный пример мультиагентной системы для синтеза обучающей выборки

Агент 1 дожидается получения необходимых данных от агентов второго уровня и записывает полученные значения в хранилище.

Таким образом, предложенный подход позволяет создавать мультиагентные системы для моделирования поведения отдельных сущностей и бизнес-процессов предметной области. Такая система позволяет синтезировать обучающую выборку в случае отсутствия достаточного количества данных, или в случае, если выборка не сбалансирована.

В рамках данной главы описан предложенный подход к конструированию признаков в машинном обучении для данных с учетом высокой степени неопределенности и ограничений предметной области.

В подразделе 2.1 представлен метод получения и подготовки данных для анализа на основе выполнения процедуры ETL для взаимодействия с внешними источниками и записи полученных данных в локальное хранилище. Предложенный метод может быть использован в ситуации, когда обучающая выборка для решения задачи методами машинного обучения отсутствует в подготовленном виде.

В подразделе 2.2 предложен метод автоматизированного конструирования признаков для машинного обучения в условиях неопределенности и с учетом ограничения предметной области. Была предложена модель контекста и инструментарий для автоматизации процесса конструирования признаков. Предложенный метод может быть полезен в случае, когда задача решается в условиях неопределенности, но может быть привлечен эксперт, который знаком с особенностями и ограничениями предметной области. Полученный признаки сохраняются в виде онтологии на языке OWL 2, что делает возможным хранение, документирование и версионирование признаков. Признаки из хранилища можно использовать при решении похожих задач.

В подразделе 2.3 предложен метод синтеза обучающей выборки на основе мультиагентного подхода. В отдельных случаях данных может быть недостаточно для обучения моделей машинного обучения, или данные могут быть не сбалансированы. Предложенный метод позволяет построить модель некоторой сущности или бизнес-процесса с учетом особенностей и ограничений предметной области и синтезировать необходимое количество данных.

3 МОДЕЛИ, МЕТОДЫ И АЛГОРИТМЫ, ОБЕСПЕЧИВАЮЩИЕ ДОВЕРИЕ И ИНТЕРПРЕТАЦИЮ РЕЗУЛЬТАТОВ ПРЕДИКТИВНОЙ АНАЛИТИКИ

3.1 Понятия доверительного и объяснительного ИИ

Доверительный ИИ относится к системам искусственного интеллекта, которые предназначены для выполнения задач или принятия решений на основе заданного набора правил или параметров. Эти системы основаны на наборе заранее определенных правил или параметрах и не могут адаптироваться к новым ситуациям или изменениям в данных.

С другой стороны, **объяснительный ИИ** относится к системам, которые предназначены для объяснения своих выводов и решений. Эти системы стремятся предоставить понятное объяснение того, как они пришли к своим выводам или решениям, и почему они считают их правильными.

Объяснительный ИИ обычно включает в себя использование алгоритмов, которые прозрачны по своей природе, то есть они предоставляют информацию о том, как они пришли к своим выводам. Это может включать использование методов, таких как локальная интерпретируемость моделей, которые позволяют анализировать влияние отдельных факторов на решение системы.

Доверительный ИИ и объяснительный ИИ представляют собой разные подходы к разработке систем искусственного интеллекта.

Доверительный ИИ предназначен для выполнения задач или принятия решений на основе заранее определенных правил или параметров. Масштаб внедрения ИИ определяется процессами регламентации определения надежности и стандартами качества интеллектуальных систем. Задача решается как международными, так и национальными организациями стандартизации. В составе

Международной организации по стандартизации (ISO) над данной задачей работает несколько комитетов.

3.2 Основные подходы к формированию доверительного и объяснительного ИИ. Стандартизация

ISO/IEC JTC 1/SC 42 – это подкомитет Международной организации по стандартизации (ISO) и Международной электротехнической комиссии (IEC), который отвечает за разработку стандартов в области искусственного интеллекта (AI). Подкомитет разделен на несколько групп, которые занимаются различными аспектами AI, включая вычислительные подходы и характеристики систем AI, доверие к системам AI и варианты использования и приложения.

ISO/IEC JTC 1/SC 42/SG 1 - Вычислительные подходы и характеристики систем искусственного интеллекта: Эта группа занимается разработкой стандартов, которые описывают различные вычислительные подходы, используемые в системах AI, а также характеристики, которые должны быть у этих систем. Группа стандартов может включать такие вещи, как алгоритмы машинного обучения, методы обработки естественного языка и техники глубокого обучения.

ISO/IEC JTC 1/SC 42/SG 2 - Доверие к системам искусственного интеллекта: Эта группа занимается разработкой стандартов, которые описывают, как обеспечить доверие к системам AI. Стандарт может включать такие вещи, как требования к прозрачности и объяснимости, оценка надежности и точности систем AI и методы тестирования и оценки производительности.

ISO/IEC JTC 1/SC 42/SG 3 - Варианты использования и приложения: Эта группа занимается разработкой стандартов, которые описывают различные варианты использования и приложения систем AI, что может

включать такие подходы, как автоматизированное принятие решений, автономные транспортные средства и интеллектуальные помощники.

ISO/IEC JTC 1/SC 42/WG1 - Основополагающие стандарты: Эта группа занимается разработкой стандартов, которые описывают основные понятия и термины, связанные с AI, а также принципы и методы, используемые в системах AI. Стандарт может включать такие вещи, как определение AI, стандартные тесты для оценки производительности систем AI и принципы этики и безопасности, связанные с AI.

ISO/IEC JTC 1/SC 42 является важным форумом для разработки стандартов в области AI. Работа группы направлена на создание четких и всеобъемлющих стандартов, которые описывают различные аспекты AI, включая вычислительные подходы, доверие, варианты использования и приложения, а также основополагающие стандарты. Эти стандарты будут полезны для организаций и частных лиц, которые хотят использовать AI в своих приложениях, обеспечивая четкие рекомендации и лучшие практики для разработки и развертывания систем AI.

ISO/IEC JTC 1/SC 42, подкомитет по искусственному интеллекту, занимается разработкой стандартов в области искусственного интеллекта (таблица 3). Одна из задач подкомитета — изучить подходы к обеспечению доверия к системам искусственного интеллекта через прозрачность, проверяемость, объяснимость, управляемость и другие факторы.

В рамках этой задачи подкомитет будет изучать различные методы и техники, которые могут быть использованы для обеспечения доверия к системам искусственного интеллекта. Это может включать такие вещи, как требования к прозрачности и объяснимости, оценку надежности и точности систем искусственного интеллекта и методы тестирования и оценки производительности.

Другая задача подкомитета — изучить потенциальные риски и угрозы, связанные с системами искусственного интеллекта, а также способы и методы их смягчения. Стандарт может включать такие вещи, как оценка безопасности систем искусственного интеллекта, управление рисками и разработка стратегий для смягчения потенциальных угроз.

Еще одна задача подкомитета — изучить типы отклонений, которые могут возникнуть в системах искусственного интеллекта, с целью их минимизации. А решения задачи опираются на такие вещи, как изучение статистических отклонений в системах искусственного интеллекта и отклонений в принятии решений с помощью искусственного интеллекта, таблица 3.1.

Таблица 3.1 Структура отчета подкомитета по искусственному интеллекту ISO/IEC JTC 1/SC 42.

Глава отчета	**Комментарии**
1. Подходы	Объяснение понятия доверия и подхода к построению отчета
2. Высокоуровневые проблемы	Разные заинтересованные стороны понимают под доверенностью разные вещи
3. Уязвимости ИИ	Использование систем ИИ для мошенничества, непредсказуемость, ошибки в обучающих данных, переобучение, человеческий фактор и т. д.
4. Меры по парированию	Обеспечение доверия посредством прозрачности, объяснимости, контролируемости. Обеспечение доверия путем сравнения с функциональными возможностями человека
5. Нетехнические меры	Этические вопросы, управление ожиданиями от систем ИИ, маркировка решений с применением систем ИИ, отождествление с человеком
6. Заключение	

3.3 Модель объяснительного (интерпретируемого) ИИ на основе гибридизации нейросетевого подхода (черный ящик) и системы суррогатных правил (белый ящик)

Объяснительный (интерпретируемый) ИИ предназначен для объяснения своих выводов и решений. Оба подхода имеют свои

преимущества и недостатки, и выбор подхода будет зависеть от конкретной задачи или приложения.

Широкими возможностями объяснений распознавания или классификации обладают **нечеткие нейронные сети**, которые позволяли интерпретировать слои нейронной сети, как этапы логического нечеткого вывода.

В частности, разработана **модель мягкой экспертной (рекомендательной) системы** сочетающей работу обученной нейронной сети с нечетким логическим выводом заключений.

Мягкие экспертные системы (МЭС) — это тип нечетких экспертных систем, которые используют статистические данные для представления знаний и используют нечеткие нейронные сети для обработки данных. Основная идея МЭС заключается в использовании генетических алгоритмов для эволюции структуры и весов нейронных сетей, таким образом, создавая систему, которая может адаптироваться к различным задачам и данным.

МЭС состоит из нескольких основных блоков, включая фаззификатор с базой данных функций принадлежности, нейроимитатор, который используется для обучения, распознавания и прогнозирования, система принятия решений и система нечеткого вывода. Фаззификатор отвечает за представление знаний в виде набора функций принадлежности, которые определяют вероятность принадлежности объекта к определенному классу. Нейроимитатор используется для обработки данных и извлечения признаков из входных данных. Система принятия решений отвечает за выбор наилучшего решения на основе представленных знаний, а система нечеткого вывода используется для рассуждений на основе нечетких правил.

МЭС были применены в различных областях, включая компьютерное зрение, обработку естественного языка и управление. Они могут использоваться для решения сложных задач, таких как классификация, кластеризация и прогнозирование. МЭС могут быть полезны в ситуациях, когда данные не являются четко определенными или когда требуется использовать нечеткие правила для принятия решений [45-46].

МЭС являются мощным инструментом для обработки сложных задач и могут быть использованы для различных приложений, включая интеллектуальное проектирование и анализ сложных систем. Они объединяют в себе возможности нечетких нейронных сетей и экспертных систем, что делает их мощным инструментом для обработки сложных задач и данных.

Нечеткие генетические нейронные сети (*Fuzzy Genetic Neural Networks, FGNN*) — это тип искусственных нейронных сетей, который объединяет генетические алгоритмы и нейронные сети для создания системы, способной обрабатывать сложные задачи восприятия и рассуждения.

Основная идея FGNN заключается в том, чтобы использовать генетические алгоритмы для эволюции структуры и весов нейронной сети, таким образом, создавая систему, которая может адаптироваться к различным задачам и данным. FGNN состоит из двух основных типов нейронов: радиально-базисных нейронов и логических нейронов И-ИЛИ-.

Радиально-базисные нейроны в FGNN предназначены для обработки входных данных и выявления паттернов в данных. Эти нейроны могут принимать на вход несколько входных данных и обрабатывать их с помощью функции активации, которая вычисляет сумму или произведение входных данных. Это позволяет FGNN обрабатывать

неструктурированные данные, такие как изображения или тексты, и извлекать важные признаки из этих данных.

Область объяснительного (интерпретируемого) ИИ является не только предметом активных исследований, но и областью создания прикладных интеллектуальных систем за счет общедоступного инструментария разработки.

За 2014–2023 г. выполнено большое количество исследований, позволившее говорить о глубоком обучении нечетких нейронных сетей, сохраняющих потенциал формирования объяснений (таблица 3.2).

Таблица 3.2. Хронология формирования методов глубокого обучения нечетких нейронных сетей

Год	**Модель**	**Свойства**
2014	Deep Belief Networks with Fuzzy Granulated Inputs	Первая архитектура нейронной сети с глубокими слоями нейронов, в которой используются нечеткие входные данные временных рядов. Эффективно уменьшает шум и избыточную информацию в данных
2014	Deep Belief Network with fuzzified outputs	Расширяет модель глубокой бинарной классификации путем реализации нечеткой принадлежности
2015	Fuzzy Restricted Boltzmann Machine	Расширяет концепцию RBM путем внедрения нечетких весов – строительный блок для более глубоких архитектур
2016	Fuzzy Deep Learning for tumor variation prediction	Использует оператор нечеткой логики вместо ANN. Уменьшенное количество параметров значительно сокращает время обучения
2016	Deep Learning with Fuzzy Feature Points	Использует радиальную базисную функцию для фаззификации входных данных. Использует традиционные глубокие CNN
2017	Restricted Boltzmann Machine with Fuzzy Inputs	Использует нечеткую логику, чтобы понять семантическое сходство между входными векторами и назначить веса для входных векторов
2017	Pythagorean Fuzzy Deep Boltzmann machine	Расширение нечеткой ограниченной машины Больцмана Использование пифагорейских нечетких чисел в качестве весов. Может эффективно обрабатывать зашумленные или неполные данные. Реализация обучения на основе биогеографии

Окончание табл. 3.2

Год	Модель	Свойства
2017	Hierarchical Fused Fuzzy Deep Neural Network	Извлекает глубокое представление и нечеткие значения принадлежности данных. Реализует мультимодальное обучение
2017	Takagi Sugeno Fuzzy Deep Network	Использует систему нечеткого вывода Takagi Sugeno для сопоставления с результатами в соответствии с правилами «ЕСЛИ-ТО»
2018	Stacked Auto Encoder trained using Fuzzy Logic	Система нечеткой логики используется для настройки нескольких параметров нейронной сети. Нечеткая логика также используется для настройки скорости обучения
2018	Deep Convolutional Neural Networks using Weighted Fuzzy Active Shape Model	Использует специальную свертку для фаззификации входных данных и снижения шума. Использует традиционные глубокие CNN
2018	Fuzzy Convolutional Neural Network for Text Sentiment Analysis	Новая модель представляет собой интеграцию модифицированной сверточной нейронной сети (CNN) и нечеткой логики
2018	Two phase approach to detection of software projects with similar architecture based on clustering and ontological methods	Разработан инструмент для анализа архитектуры проекта, позволяющий повторно использовать большие части проектов и избежать концептуально неправильных решений
2019	An approach to vocabulary expansion for neural network language model by means of hierarchical clustering	Предложен подход к получению качественного пространства признаков для произвольной иерархии
2018	Fuzzy Deep Neural Network for Classification of Overlapped Data	Предложен метод преобразования исходных значений атрибутов в центры кластеров с помощью нечеткой глубокой нейронной сети. Тестовые данные проверяются с помощью модели Fuzzy Deep Neural Network на производительность
2020	An approach to user feedback processing in order to increase quality of clustering results	Представлен подход к построению метода интерактивной кластеризации с учетом обратной связи на базе современных непрерывных методов кластеризации с использованием глубоких нейронных сетей. В частности, представлена реализация на основе метода кластеризации DEC

3.4 Некоторые из возможностей интерпретируемого машинного обучения на *Python*

1. **Оценка важности признаков и их влияния** с помощью модели *RuleFit*: Модель *RuleFit* — это интерпретируемая модель машинного обучения, которая может оценивать важность признаков и их влияние на модель. Это может быть полезно для понимания того, какие признаки наиболее важны для модели и как они влияют на ее предсказания.

2. **Глобальное суррогатное моделирование** машинного обучения с помощью библиотеки *Skater*: Глобальное суррогатное моделирование — это метод интерпретации, который позволяет анализировать высоко-неопределенные модели машинного обучения, такие как черные ящики, с помощью моделей типа белого ящика. Библиотека *Skater* предоставляет инструменты для выполнения этого и может помочь в понимании того, как модель принимает решения.

3. **Визуализация сверточных нейронных сетей**. Визуализация сверточных нейронных сетей может быть сложной из-за их сложной архитектуры. Однако существуют методы, такие как визуализация активации и градиентные карты активации классов (*Grad-CAM*), которые могут помочь в понимании того, как сверточные нейронные сети воспринимают и обрабатывают данные.

4. **Пертурбационные методы атрибуции**. Пертурбационные методы атрибуции — это методы, которые позволяют анализировать вклад отдельных признаков или шагов в модель машинного обучения. Эти методы могут быть полезны для понимания того, какие признаки или шаги наиболее важны для модели и как они влияют на ее предсказания.

5. Методы исследования **окклюзивной чувствительности**. Методы исследования окклюзивной чувствительности — это методы,

которые позволяют анализировать, как модель машинного обучения реагирует на изменения в данных. Эти методы могут быть полезны для понимания того, как модель обрабатывает различные сценарии и как она может быть улучшена.

6. **Глубокий объяснитель**, основанный на обратном распространении (*SHAP*): *SHAP* — это метод интерпретации, который предоставляет объяснения предсказаний модели машинного обучения на основе ее модели. Он основан на идее, что объяснение предсказания модели является линейным комбинацией вкладов отдельных признаков. *SHAP* может быть полезен для понимания того, какие признаки наиболее важны для предсказаний модели и как они взаимодействуют друг с другом.

3.5 Пример архитектуры прикладной интеллектуальной системы поддержки автоматизированной диагностики меланомы, интегрирующей нейронные сети и систему правил

3.5.1 Проблемная область и описание контекста для конструирования признаков и интерпретации результатов системы поддержки автоматизированной диагностики

В Российской Федерации с меланомой ежегодно впервые сталкивается 12 тысяч человек (около 5 тысяч мужчин и 7 тысяч женщин), что составляет 5–8 человек на 100 000 населения для обоих полов. Этот показатель так же, как и в остальном мире, ежегодно увеличивается. С 2011 по 2021 годы заболеваемость меланомой в Российской Федерации выросла на 30–45% [47]**.** Среднегодовой темп прироста заболеваемости за 10 лет составил 3,74 % у мужчин и 3,04 % у женщин [48]**.**

Стоит учесть, что чем больше возраст человека, тем выше вероятность возникновения меланомы. Кроме того, методы диагностики становятся все более совершенными и эффективными, растет осведомленность пациентов о заболевании [48]. Доля меланомы в общем числе

злокачественных новообразований в Российской Федерации сопоставима с мировыми показателями и составляет 1,8%. Средний возраст больных составил 61,6 года (оба пола), 61,3 года (мужчины), 61,8 года (женщины). Кумулятивный риск развития меланомы кожи (период 2007–2017 гг., возраст 0–74 года) составил 0,55% [49].

Хотя на меланому приходится лишь 4% всех случаев рака кожи, она является причиной 80% случаев смерти от рака кожи. В абсолютном выражении меланома ежегодно вызывает смерть 57 тысяч человек в мире [50].

Представленные показатели показывают, что эффективность работы медицинских работников «первого контакта» по выявлению злокачественных новообразований довольно мала. Диагностика меланомы на ранней стадии крайне важна: чем раньше она будет выявлена, тем эффективнее будет результат лечения.

Тенденция последних лет, обычно не связанная строго с системами диагностики новообразований кожи, заключается в том, что исследователи разрабатывают глубокие сети с большим количеством скрытых слоев (либо сверточных, либо полностью связанных слоев) для получения лучших результатов. В результате большинство работ, связанных с обнаружением, сегментацией и/или классификацией меланомы, основано на таких архитектурах нейронных сетей, как ResNet [51], Inception/GoogLeNet, U-Net [52], DenseNet [53], AlexNet, VGG [54] и Mask R_CNN.

3.5.2 Использование критериев наружного осмотра для интерпретации результатов и контроля распознавания новообразований нейронными сетями

Меланома кожи – злокачественная опухоль нейроэктодермального происхождения, исходящая из меланоцитов (пигментных клеток) кожи.

Это одна из самых агрессивных и смертельно опасных видов опухолей, именно поэтому своевременно диагностировать ее крайне важно.

Для выявления меланомы используют два вида методов: инвазивный и неинвазивный. Пациенты зачастую не соглашаются на проведение биопсии кожного новообразования без веских причин. Перед биопсией всегда применяется метод наружного осмотра, в ходе которого врач оценивает новообразование по нескольким критериям, распространенным в медицинской практике.

Систем критериев существует множество, однако одна из самых популярных – ABCDE, где:

• A (*Asymmetry*). Обычно проявление пигментации круглые и симметричные. Формы ранней меланомы зачастую бывают асимметричными: ось симметрии, проводимая через центр очага пигментации, разделит очаг на две неодинаковые половины;

• B (*Border*). У обычных границы четкие и ровные, у ранней меланомы же зачастую они нечеткие и изрезанные;

• C (Цвет) Как правило, родинки и пигментные пятна имеют равномерную коричневую окраску одного тона, злокачественные новообразования отличаются неравномерной окраской разных тонов черного или коричневого;

• D (диаметр). Обычные родинки или пигментные пятна обычно имеют диаметр до 5–6 мм, тогда как меланома на ранней стадии обычно больше 6 мм;

• E (*Evolution*) Изменения в родинке или пигментном пятне. Пигментные пятна, которые с течением времени меняют свою форму, размер или окраску, могут оказаться злокачественными.

В настоящее время ведутся работы по автоматизации анализа кожных новообразований. Однако чаще всего в исследованиях используется

технология нейронных сетей, характеризуемая сложностью интерпретации результатов и их обоснованности. Подобные ограничения для области медицины крайне нежелательно: врач должен понимать, как именно система приняла то или иное решение.

В связи с этим возникает задача разработки метода автоматизации проведения и интерпретации ABCDE-анализа новообразования на изображении. Такой подход может быть использован как специалистом на приеме, так и системой поддержки автоматизированной диагностики для интерпретации результатов и контроля распознавания новообразований нейронными сетями.

3.5.3 Показатели ABCDE-анализа. Предобработка изображения для ABCDE-анализа

Каждый из существующих методов автоматизированной диагностики имеет этап предварительной обработки. Он заключается в применении следующих основных операций: удаление шума, выделение значимой части данных, изменение размера, преобразование яркости в оттенки серого или коррекция яркости, усиление интенсивности и контраста [55]**.** Поскольку изображения новообразований кожи имеют высокую вариабельность содержания, этап сегментации является широко обсуждаемой темой и сложной задачей. Этот шаг является частью алгоритма, который позволяет разделить изображение на несколько наборов пикселей с выделением областей интереса с использованием автоматического или полуавтоматического процесса в качестве конечного результата. К числу наиболее часто используемых методов обнаружения и сегментации новообразований относятся методы, основанные на искусственных нейронных сетях.

3.5.4 Пример предварительной обработки изображения

Пиксели изображения группируются по цвету в зависимости от их близости друг к другу (рис. 3.1).

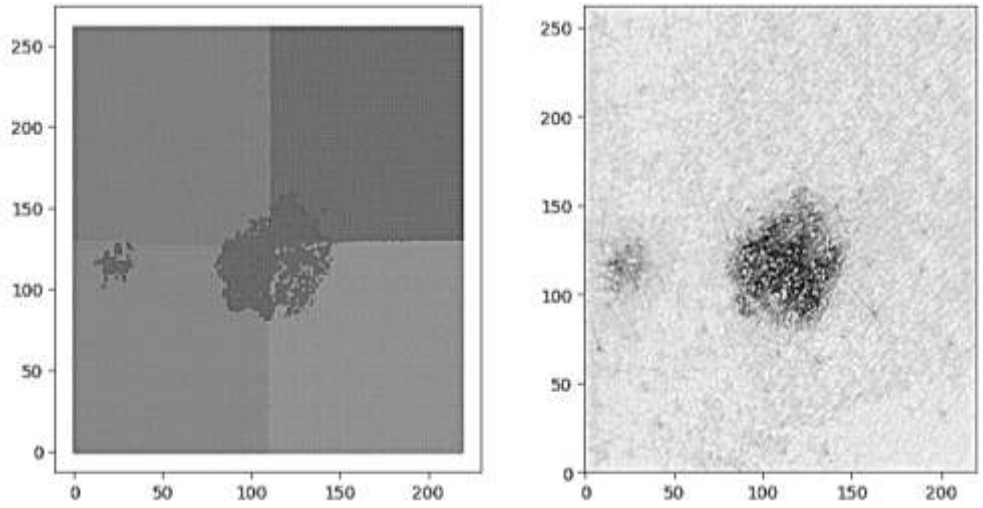

Рис. 3.1. Результат кластеризации изображений

На основании данных каждого кластера можно построить гистограмму рис. 3.2. Кластер, наиболее отличающийся от остальных по данным гистограммы, является кластером новообразования.

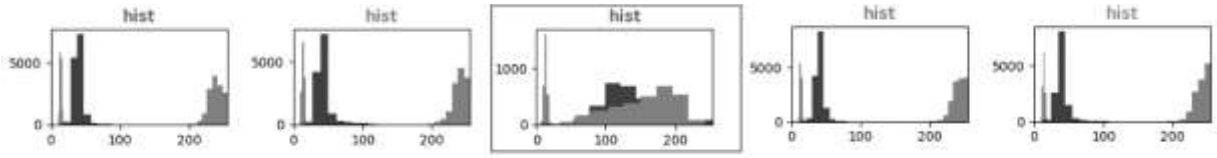

Рис. 3.2. Выделение кластера с новообразованием при анализе гистограммы

После этого на изображении выделяется новообразование кожи и убирается шум встроенной в *OpenCV* функцией для цветных изображений *fastNlMeansDenoisingColored*. Применяются морфологические трансформации: *ERODE*, удаляющая лишние пиксели вне контура новообразования, и *DILATE*, восстанавливающая все пиксели на контуре и внутри новообразования.

В этом случае на фотографии больше одного новообразования. Для обработки и оценки используется больший из них, поэтому на изображении находится и выделяется самый большой контур (рис. 3.3).

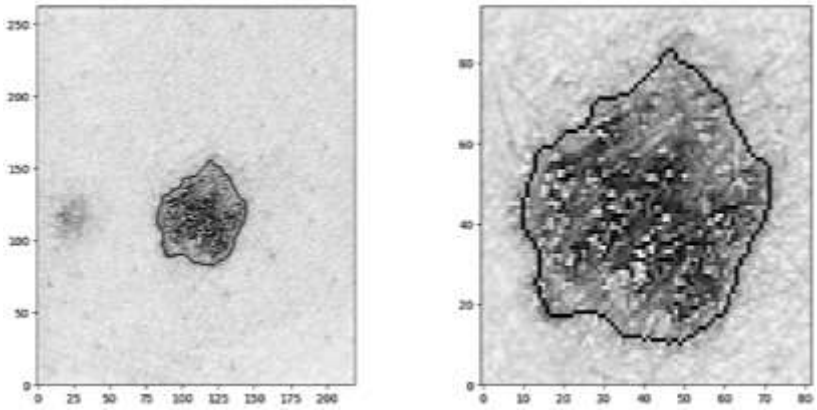

Рис. 3.3. Выделение основного очага новообразования

И только после проведения операций предобработки становится возможным рассчитать метрики. Далее по полученным результатам рассчитываются показатели ABCDE-анализа, после чего рассчитывается общий результат в формате вероятности перерождения новообразования в меланому в процентах.

3.5.5 Пример расчета ABCDE-метрик

Асимметрия. Выбирается точка из массива точек контура новообразования, показанного на рисунке синим цветом. Относительно него берутся пиксели справа и слева, показанные на рисунке, и от них рассчитывается расстояние до геометрического центра контура новообразования. (рис. 3.4).

Дан набор, содержащий n точек $x_1, x_2 \dots x_n$, где $x_i \in R^n$, геометрический центр образования определяется

$$c = \arg\min_{y \in R^n} \sum_{i=1}^{n} \left|\left|x_i - y\right|\right|_2,$$

где *arg min* означает значение аргумента *y*, при котором достигается минимальная сумма. Это точка *y*, для которой сумма всех евклидовых расстояний до x_i минимальна.

Затем вычисляется модуль разности этих расстояний и прибавляется к конечному значению асимметрии. Чем выше это значение, тем более асимметрично новообразование.

Расчет происходит с шагом в один пиксель, значения метрик нормированы относительно длины контура новообразования.

В процессе расчета асимметрии также находится главная ось симметрии, где значение метрики асимметрии наименьшее, и выводится график рассчитанной асимметрии относительно каждой точки (рис. 3.4).

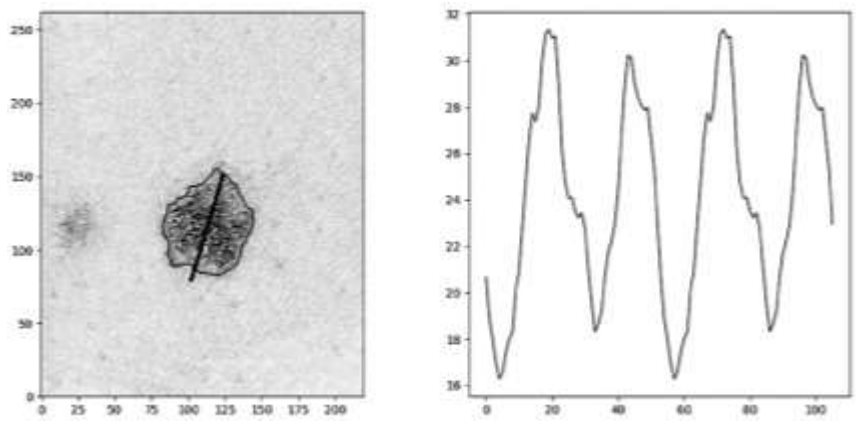

Рис. 3.4. Ось симметрии, на которой значение метрики асимметрии наименьшее. График рассчитанной асимметрии относительно каждой точки

$$\mathrm{A} = \arg\min_{j \in n} \sum_{i=1}^{n/2} |s_{(j-i)\%n} - s_{(j+i)\%n}|,$$

где *arg min* означает значение аргумента j, при котором достигается минимальное значение метрики, j – выбранная точка, i – индекс смещения, n – количество точек, $s_{(j-i)\%n}$ – расстояние от левой точки до центра, $s_{(j+i)\%n}$ – расстояние от правой точки до центра. На рисунке 3.4 представлено изображение с выделенной осью сечения относительно которой новообразование представляется наименее симметричным, что позволит специалистам интерпретировать результаты автоматизированного анализа.

Граница. Для оценки ровности границ создается контурная маска и по этой маске вычисляется градиент, нормализованный по количеству пикселей. Значение метрики представляет собой сумму заданного градиента. Чем больше число, тем более неровной будет граница, поскольку градиент показывает разницу между соседними пикселями.

Граница на рис. 3.5 включает 15 краевых пикселей.

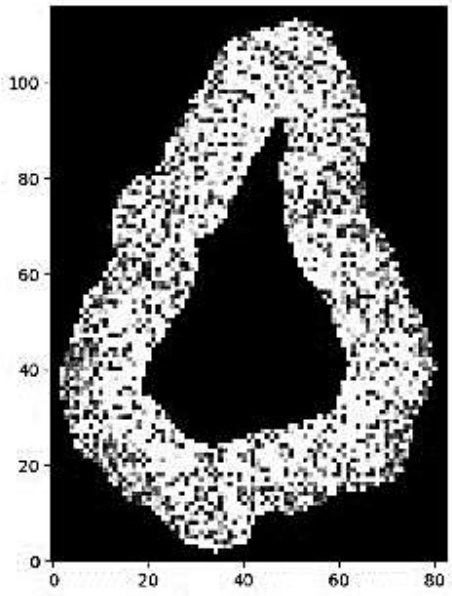

Рис. 3.5. Граница, состоящая из 15 пикселей

$$B = \sum_{i=1}^{n} grad\ x_i,$$

где n – количество пикселей на границе, x_i - конкретный пиксель.

Цвет. Для расчета цветовой метрики используется тот же метод, что и для границ, только расчет ведется по маске всего новообразования (рис. 3.6).

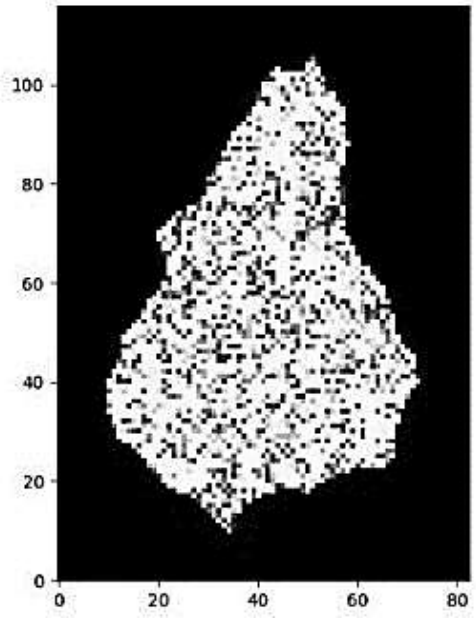

Рис. 3.6. Маска новообразования для расчета метрики цвета

$$C = \sum_{i=1}^{n} grad\ x_i$$

где n – количество пикселей новообразования, x_i- конкретный пиксель.

Размер. Вычисляется радиус наименьшей описанной окружности в пикселях, затем умножается на размер пикселя, полученное значение удваивается (рис. 3.7).

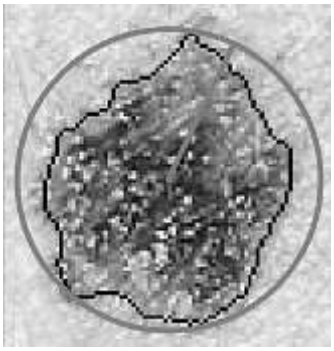

Рис. 3.7. Метрика диаметра: удвоенный радиус наименьшей описанной окружности

$$D = 2 * \min_{\vec{c} \in Z}(\max_{\vec{x} \in X} \|\vec{c} - \vec{x}\|),$$

где Z – множество двумерных векторов, X – множество точек, принадлежащих новообразованию. На рисунке 3.7 представлено изображение, позволяющее специалистам интерпретировать результаты

автоматизированного анализа. Так как на изображении поверх новообразования отрисована окружность, в которую вписано новообразование.

3.5.6 Структура системы и процесс анализа изображений новообразований

Система представляет собой WEB-приложение, разработанное на Python с использованием OpenCV. Входная информация поступает от пользователей системы в виде изображений в форматах .jpg и .png.

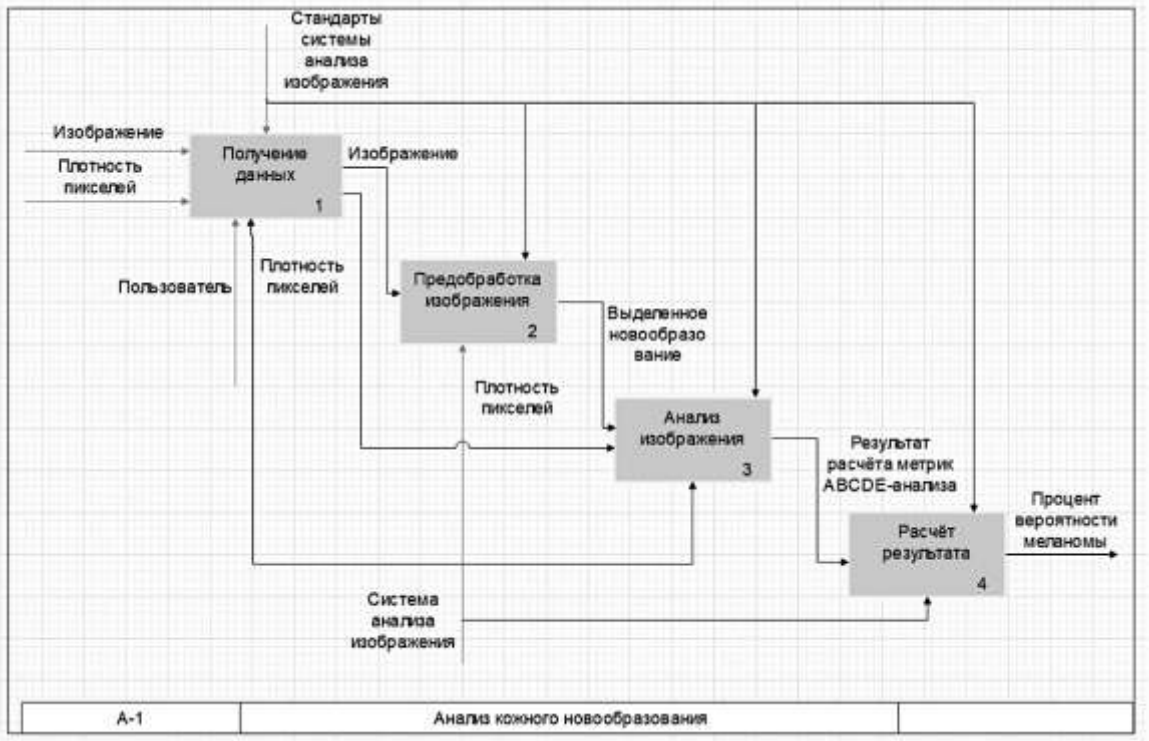

Рис. 3.8. Диаграмма IDEF0 level 1

На схеме (рис. 3.8) подробно описан процесс анализа кожных новообразований. Этот процесс состоит из четырех этапов, на каждом из которых элементом управления являются эталоны системы анализа изображений, а необходимыми механизмами – система анализа изображений:

– **Входные данные**. Ввод и вывод изображения и метаданных изображения. В этом процессе есть еще один необходимый механизм: пользовательские настройки.

– **Предварительная обработка изображений.** На входе – изображение, полученное от пользователя, на выходе – изображение изолированного новообразования кожи.

– **Анализ изображения новообразования**. Вход – выбранное новообразование и плотность пикселей, выход – результат расчета метрик ABCDE-анализа.

– **Формирование прогноза эволюции новообразования**. На входе – показатели анализа ABCDE, на выходе - процент вероятности принадлежности новообразования к меланоме.

3.5.7 Вычислительные эксперименты

Для определения эффективности предложенной модели и выбора порогового значения были проведены вычислительные эксперименты на открытом наборе данных. Под пороговым значением понимается граница, после превышения которой новообразование считается злокачественным, и система предлагает специалисту изучить новообразование еще раз с учетом визуализации. Значения показателей суммируются с учетом коэффициентов, подобранных при помощи модели линейной регрессии на основе изображений из размеченного датасета. Результаты опытов представлены на графике на рис. 3.9.

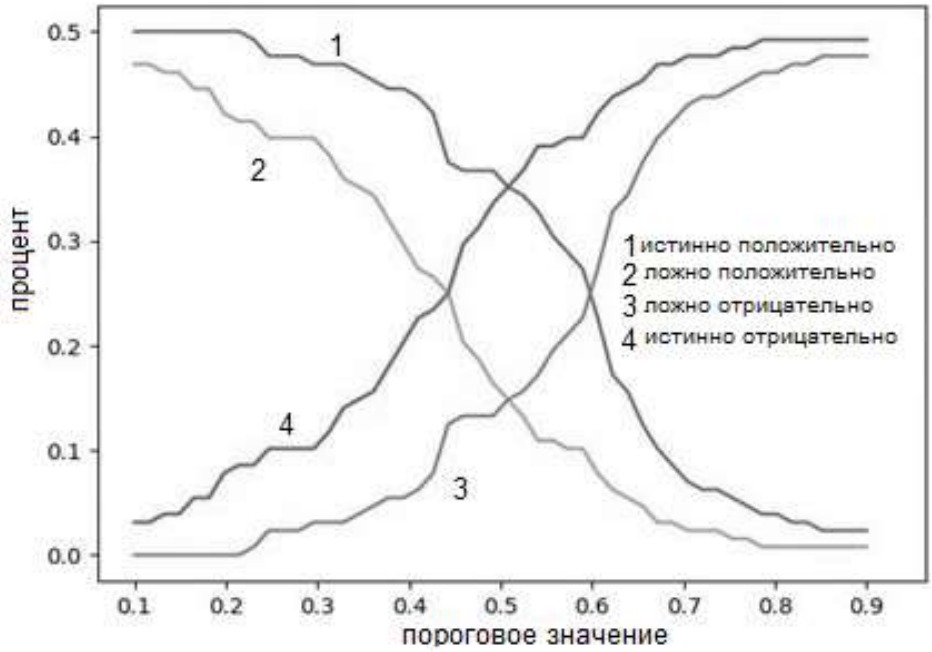

Рис. 3.9. График пороговой оценки

В ходе исследования был разработан программный продукт, позволяющий оценивать новообразование по изображению и интерпретировать результат распознавания на основе критериев ABCDE-анализа. Подобное решение позволяет не просто рассчитать нормализованные показатели, привязанные к критериям ABDCE-анализа, но и позволить интерпретировать результаты в процессе работы специалистов, в том числе за счет визуализации дополнительных построений на изображении. Метрика «Эволюция» представлена в информационной системе последовательностью изображений, по каждому из которых проведен анализ. В данный момент разработан набор правил, фиксирующий изменения новообразования, а обучить модель для поиска изменений невозможно по причине отсутствия подобного набора данных, отражающего развитие новообразования в динамике.

4 ИЗВЛЕЧЕНИЕ ПРИЗНАКОВ ОБЪЕКТОВ ИЗ БОЛЬШИХ НЕСТРУКТУРИРОВАННЫХ ТЕКСТОВЫХ ДАННЫХ

В современном мире огромное количество информации, в том числе и пригодной для использования при конструировании признаков, находится в виде неструктурированного текста на естественном языке – статей, монографий, технических заданий и другого рода текстов. В рамках разработки моделей и алгоритмов конструирования признаков из контекста динамики показателей процессов предметной области необходимо решить задачу извлечения признаков объектов, что, в свою очередь, порождает задачу извлечения самих объектов из текстов на естественном языке. Общая схема представлена на рисунке 4.1.

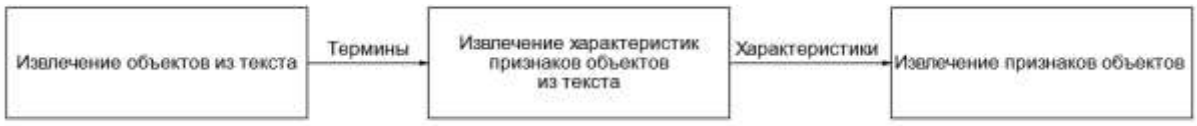

Рис. 4.1. Порядок обработки текста на естественном языке для извлечения признаков объектов

4.1 Извлечение терминологии в задаче извлечения признаков объектов

В настоящее время для решения задачи извлечения терминологии применяются статистические методы, такие как анализ временных рядов, корреляционный анализ, регрессионный анализ, а также семантические методы, деревья решений, нечеткая логика, нейронные сети и онтологический подход.

Любые статистические методы для получения результатов требуют наличия большого количества входных данных для обучения, что является их слабой стороной. К статистическим методам, которые могут быть применены для извлечения признаков объектов, можно отнести такие, как коэффициент взаимной информации (mutual information), T-Score, LogLikelihood, C-Value и TF-IDF.

Mutual Information – метод, который применим только к двусловиям. Данный метод основан на вычислении встречаемости составных частей двусловия вместе и раздельно. По результатам работы метода можно отнести двусловие к одной из трех категорий – статистически значимая встречаемость «вместе», статистически незначимая встречаемость или делается вывод, что статистическая значимость не установлена. Расчет Mutual Information производится по формуле

$$\mathrm{MI} = \log_2 \frac{p_{ab} \times \mathrm{S}}{p_a \times p_b},$$

где p_{ab} – встречаемость составных частей двусловия вместе,

p_a, p_b – встречаемость каждого слова отдельно,

S – общий размер текста или количество уникальных слов, встречающихся в тексте (в зависимости от выбранной вариации метода).

Результаты работы метода ранжируются следующим образом:

– MI ∈ (1; ∞) – статистически значимое словосочетание

– MI ∈ [0;1] – статистически незначимое словосочетание

– MI ∈ (-∞;0) – статистическая значимость не установлена

Метод T-Score определяет степень взаимосвязи слов. По сути, это вероятность наличие некоторой связи между двумя словами.

Метод T-Score применяется для преобразования биномиального распределения дискретной случайной величины в нормальное распределение непрерывной случайной величины N. Суть метода заключается в использовании нулевой гипотезы о независимости. Важно помнить, что метод T-Score приближает нормальное распределение к биноминальному распределению, поэтому он имеет некоторые ограничения, связанные с предположением о нормальности распределения.

Данный метод может быть применим только к двусловиям.

$$\mathrm{TS} = \frac{p_{ab} - \frac{\mathrm{p}_a \times \mathrm{p}_\mathrm{b}}{s}}{\mathrm{p}_{\mathrm{ab}}^2},$$

где p_{ab} – встречаемость двусловия,

p_a, p_b – встречаемость каждого из слов отдельно

n – общий объем двусловий в тексте

Метод наибольшего правдоподобия, или Log-Likelihood – популярный статистический метод, в котором оценивается неизвестный параметр при помощи максимизации функции правдоподобия. Метод основан на совместной вероятности встречаемости двух слов, поэтому применим только к двусловиям

$$\begin{gathered}\mathrm{LL} = \mathrm{a} \times \log_2(\mathrm{a}+1) + \mathrm{b} \times \log_2(\mathrm{b}+1) + \mathrm{c} \times \log_2(\mathrm{c}+1) + \\ + \mathrm{d} \times \log_2(\mathrm{d}+1) - (\mathrm{a}+\mathrm{b}) \times \log_2(\mathrm{a}+\mathrm{b}+1) - \\ - (\mathrm{a}+\mathrm{c}) \times \log_2(\mathrm{a}+\mathrm{c}+1) - (\mathrm{b}+\mathrm{d}) \times \\ \times \log_2(\mathrm{b}+\mathrm{d}+1) - (\mathrm{c}+\mathrm{d}) \times \log_2(\mathrm{c}+\mathrm{d}+1) + (\mathrm{a}+\mathrm{b}+\mathrm{c}+\mathrm{d}) \times \\ \times \log_2(\mathrm{a}+\mathrm{b}+\mathrm{c}+\mathrm{d}+1),\end{gathered}$$

где a – частотность данной биграммы;

b – суммарная частотность других (отличных от данной) биграмм с той же самой левой леммой;

c – суммарная частотность других биграмм с той же самой правой леммой;

d – суммарная частотность остальных биграмм текста.

Метод C-Value, является статистическим, однако качество его оценки зависит от того, насколько хорошо определены термины большей размерности, поэтому в реальности применим только после совместного применения лингвистических методов и других статистических методов. В основе метода лежит определение вложенности n-граммы в n+1-грамму. Метрика C-Value определяется по формуле:

$$\mathrm{CV} = \left\{ \frac{\log_2|A| \times \mathrm{freq(A)}, \text{если не вложен}}{\log_2|\mathrm{A}| \times \mathrm{freq(A)} - \frac{1}{\mathrm{P(T_A)}} \times \sum_{\mathrm{b} \in \mathrm{T_A}} \mathrm{freq(B)}} \right\},$$

где A – словосочетание;

|A| – количество слов в словосочетании;

freq(A) – частота встречаемости словосочетания в тексте;

T_A – множество словосочетаний, определенных как термины, которые содержат данное словосочетание;

$P(T_A)$ – количество словосочетаний, содержащих данное словосочетание.

Метод TF (Term Frequency – частота слова) является наиболее простым методом, т. к. считает, что чем чаще слово встречается – тем с большей вероятностью оно является термином. В реальных задачах в чистом виде не применяется, так как обладает крайне низкой точностью. Однако применяется в совокупности с некоторыми лингвистическими методами, такими как фильтрация по стоп-словам, и является входными данными для ряда других методов.

$$\mathrm{TF} = \frac{p_a}{\mathrm{S}},$$

где p_a – абсолютная частота слова в тексте;

S – количество слов в тексте.

Метод TF-IDF – это расширение метода TF, которое вводит такое понятие, как обратная частота документа, т. е. инверсию частоты встречаемости слова в коллекции текстов. Применяется в первую очередь для фильтрации общеупотребительных слов из текущего текста, при этом IDF для русского языка обычно берется из национального корпуса русского языка.

$$\mathrm{TF} \times \mathrm{IDF(a)} = \mathrm{TF(a)} \times log_2 \frac{M - Q(M)}{Q(M)},$$

где TF(a) – абсолютная частота слова a в текущем тексте;

M – размер контрастной коллекции (обычно – размер коллекции национального корпуса русского языка);

Q(M) – число документов, в которых употреблялось слово a в контрастной коллекции M [57].

К лингвистическим методам выделения объектов из текста можно отнести классический лексический метод, при котором происходит отбор

слов и словосочетаний по используемым в данном языке шаблонам, основанных на лексике и грамматике данного языка [58]. Для русского языка наиболее распространенными шаблонами являются шаблоны, описанные в таблице 4.1.

Таблица 4.1. Основные шаблоны терминов для русского языка

Первое слово	**Второе слово**	**Третье слово**	**Четвертое слово**
Существительное	Нет	Нет	Нет
Существительное	Существительное	Нет	Нет
Прилагательное	Существительное	Нет	Нет
Причастие	Существительное	Нет	Нет
Существительное	Существительное	Существительное	Нет
Прилагательное	Прилагательное	Существительное	Нет
Прилагательное	Причастие	Существительное	Нет
Причастие	Прилагательное	Существительное	Нет
Причастие	Причастие	Существительное	Нет
Прилагательное	Предлог	Существительное	Нет
Причастие	Предлог	Существительное	Нет
Существительное	Существительное	Существительное	Существительное

Кроме того, к лингвистическим методам относится метод увеличения и уменьшения весов по характеристике открытости словосочетаний. Вес словосочетания в ранжировании словосочетаний по вероятности терминологичности увеличивается (т. е. увеличивается вероятность данного словосочетания как объекта поиска), если словосочетание чаще всего встречается как «закрытое», и уменьшается, если чаще всего встречается как «открытое». Закрытое словосочетание – это словосочетание, с обеих сторон от которого стоит существительное, прилагательное или причастие, полуоткрытое словосочетание – это если существительное, прилагательное или причастие стоит только с одной

стороны, а открытое словосочетание – если существительных, прилагательных и причастий нет с обеих сторон [59].

Расширением лексического метода можно считать морфологический анализ текста, при котором происходит определение основных морфологических признаков слова, определение начальной формы слова и выделение лексико-грамматических классов. Данное дополнение используется в первую очередь для приведения слов в нормализованный вид, чтобы разные варианты употребления слова не считались разными словами.

Другим расширением лексического метода, который, в свою очередь, использует результат работы морфологического расширения, является стоп-лист, который содержит часто встречающиеся слова, наличие которых в многословии однозначно определяет данное слово как не являющееся термином вообще или в данной конкретной предметной области.

Для извлечения многословных терминов, относящихся к предметной области, может также применяться онтологический подход. Для этого необходимо иметь хорошо подготовленную и заполненную однословными терминами онтологию с правильно выставленными связями. Возможен вариант использования нескольких онтологий одной или разных предметных областей для более точного определения семантической метрики «термин/не термин». На основе онтологического подхода можно выделить два критерия отнесения кандидата в термины:

– Тезаурусный критерий;
– Критерий вложенных связей.

Степень близости слова/сочетания слов терминам рассматриваемой предметной области в соответствии с тезаурусным критерием оценивается по следующей формуле:

$$k_{Ont} = \frac{\mathrm{k_t}}{\mathrm{O+1}},$$

где k_t – результат первого этапа анализа;

O – число отношений, связывающих опорный объект онтологии с ближайшим объектом, имеющим истинное значение свойства «являетсяТермином».

Критерий вложенных связей – метрика, которая позволяет при помощи связи слов в онтологии определить, является ли данное словосочетание термином для данной конкретной области.

Таким образом, при сопоставлении входных сочетаний и объектов предметной области, связанных между собой однонаправленными отношениями, термином рассматриваемой предметной области будет считаться многословие, все слова которого встречаются в данной онтологии и соединены набором отношений.

4.2 Извлечение характеристик и признаков объектов для терминов, выделенных в тексте проблемной области

Контекст – это определение, которое имеет несколько трактовок. Контекст может быть локальным, сущностным, тематическим и глобальным. Рассмотрим все виды контекста на примере статьи «Построение унифицированной базы знаний на основе данных профилей социальных сетей» [60].

Локальный контекст – это слова и смыслы, которые окружают текущее словосочетание. Например, если взять из данной статьи термин «сбор данных», то локальным контекстом для данного термина будут являться определения «набор инструментов», «обеспечение», «профиль пользователя», «автоматический», «социальных сетей», т. к. это определения, окружающие данный термин [61].

Тематический контекст – это слова и термины, которые находятся в данном тексте, но не окружают данное словосочетание, однако могут быть связаны с ним по смыслу текста.

Глобальный контекст – это слова и термины, независимые от текущего текста, которые могут использоваться как характеристики данного определения.

Для термина «сбор данных» можно подобрать очень большое количество сущностных контекстов, например «медицинские данные», «физические характеристики перекрытий в многоквартирных домах», «психологический портрет» и т. п.

Для определения контекста существует несколько методов, большинство из которых основаны на семантических тезаурусах и онтологическом подходе [62].

Одним из наиболее полных, а потому наиболее часто используемым семантическим тезаурусом считается WordNet, однако данный тезаурус существует только для английского языка, поэтому для русского обычно используется один из аналогов, которые имеют такую же структуру, как и оригинальный тезаурус.

Самыми популярными являются RuWordNet (создан при поддержке РФФИ на основе РуТез, около 120 тыс. слов и выражений) и Russian WordNet.

Структурно RuWordNet повторяет WordNet, который, в свою очередь, представляет из себя наборов синонимов (синсетов), имеющие такие отношения, как «мероним - холоним», «часть - целое», «гипоним – гипероним» и других. Синсеты разных частей речи имеют разные виды отношений. Например, «гипоним – гипероним» относится к существительным и глаголам.

Первая версия RuWordNet представляла собой автоматический перевод оригинального англоязычного тезауруса WordNet с небольшими доработками [63]. Позднее данный тезаурус был доработан как при помощи ручного редактирования файлов, составляющих тезаурус, так и ряда программ, предоставляющих удобный пользовательский интерфейс – TenDrow. Обобщенная структура тезаурусов семейства WordNet представлена на рисунке 4.2.

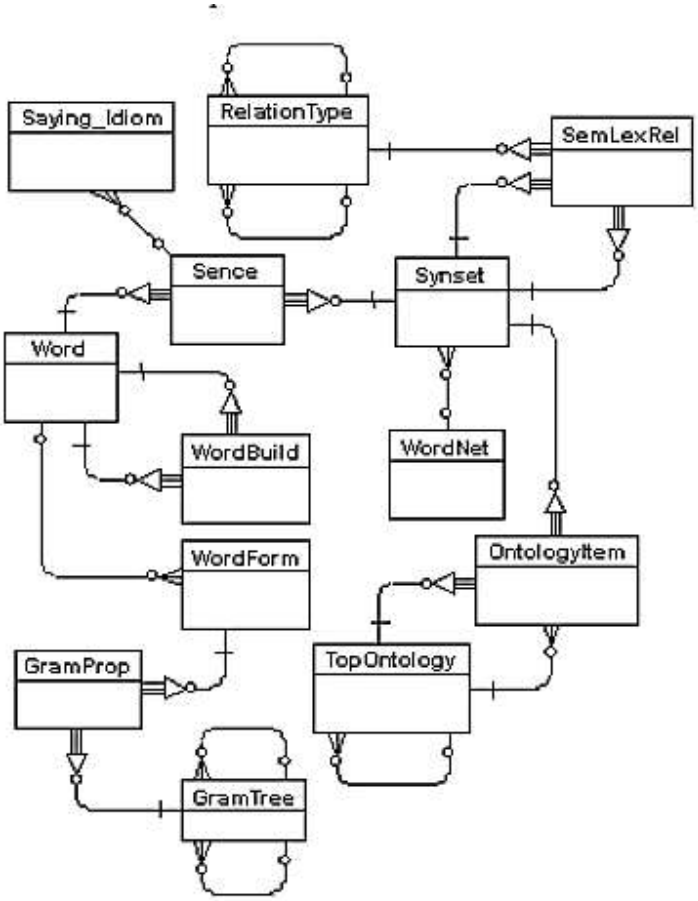

Рис. 4.2. Структура тезауруса семейства WordNet

Существуют также тезаурусы, которые наследуют структуру WordNet, но обычно добавляя свои метки или меняют структуру, например тезаурус WordNet-Affect, который добавляет каждому слову эмоцию или эмоции, которые данное слово несет в себе.

Другим известным тезаурусом является SenticNet, который, в первую очередь, служит для определения эмоциональной окраски слова, так как в

его основе лежат наборы эмоциональных понятий. Однако в то же время он содержит и смысловую составляющую, что позволяет использовать его в ряде других задач.

Данный тезаурус так же отличает от других мультиязычность (включая русский) и система распространения – в виде API, открытого для использования всем желающим.

К тезаурусам так же можно отнести лингвистическую онтологию РуТез, особенностью, которую являются специализированные для русского языка отношения между объектами онтологии, являющимися также терминами. Размер данной онтологии – более 160 тыс. слов и выражений а также более 200 тыс. связей. Для РуТез существует расширение РуСентиЛекс, добавляющая ряд полей для каждого слова.

Существуют работы Е.Б. Козеренко, описывающие модели аналитической обработки слабоструктурированных текстов естественного языка, одной из поставленных в данной работе задач является поиск терминов (в работах Е.Б. Козеренко используется термин «информационные объекты») и контекста данных терминов (в работах Е.Б. Козеренко - «взаимосвязей»).

В ряде работ, связанных с задачей категоризации текстов по ключевым словам, например, в «Выявление интересов пользователей Интернета на основе ассоциативного подхода» описываются методы извлечения терминов из текстов на естественном языке. При этом слова, отнесенные к терминам, сортируются по категориям и им присваивается некоторый вес, увеличивающийся по мере встречаемости термина в обрабатываемом тексте [64].

Категории классификаторов представлены в виде кортежа:

<Ключевое слово, категория, вес>.

Эксперты могут подтверждать отнесение ключевого слова к категории.

В работах идет расчет вероятности отнесения слова, а также текста, в котором встречается данное слово, к определенной категории. Вероятность правильного решения i-го эксперта

$$P_i = p\,(\text{Категория} \mid \text{слово}),$$

где P_i – вероятность правильного решения i-го эксперта.

Вес рассчитывается по формуле

$$W_i = \ln(\frac{P_i}{1-P_i}).$$

Как следствие, строятся предметно-ориентированные знания, которые включают в себя справочники (классификаторы) объектов и процессов, ассоциативные портреты с указанной областью знаний, методы создания моделей, фрагментов семантических сетей, описывающих взаимосвязи между объектами, отношениями, процессами.

В работах Дмитриева А. С. под «контекстной информацией» подразумевается набор слов, окружающих данный объект, а также терминов, их связывающих. Для поиска контекста в работах используется модификация метода «Снежный шар», который извлекает кортежи слов из обрабатываемого текста. Метод «снежного шара» может определять так называемые «левые», «правые» и «средние» контексты, связанные одним шаблоном [65]. По итогам работы методы «снежный шар» формируется кортеж вида (рис. 4.3)

$$K = < lv, t\_1, mv, t\_2, rv >,$$

где *lv* – вектор контекста слева от определения,

mv – вектор контекста посередине,

rv – вектор контекста справа от определения,

t_1 и *t_2* – теги объекта и термина.

Рис. 4.3. Визуальное представление кортежа по методу «снежный шар»

Получение синтаксического дерева предложения – хороший способ получить отношения слов внутри предложения. Синтаксическое дерево предложения – это набор слов данного предложения, представленный в виде набора узлов, имеющих потомков и предков. Корнем дерева обычно является слово, обозначающее действие с каким-либо объектом, т.е. сказуемое.

Пример визуального отображения подобного дерева представлен на рисунке 4.4.

Для построения синтаксического дерева обычно используются готовые библиотеки - Stanza (обученный на специализированном наборе данных, содержащих только тексты на русском языке), SpaCy (адаптированный для русского языка), UdPipe, DeepPavlov (при использовании пакета syntax_ru_syntagrus_bert), SyntaxNet, Natasha и ряда других [66].

4.3 Подход для решения задачи извлечения признаков объектов и определяемых их переменных

Необходимо внести ясность, что под признаками объекта (также используется синоним «характеристики») понимается конкретное описание части объекта. Так, например, для слова «шарик» признаками будут являться слова «деревянный», «стальной», «маленький», «большой», «холодный», «горячий» и т. п. Под переменными понимается совокупность признаков, относящихся к одной категории – например, у

слова «шар» переменными будут являться категории «материал», «размер», «температура».

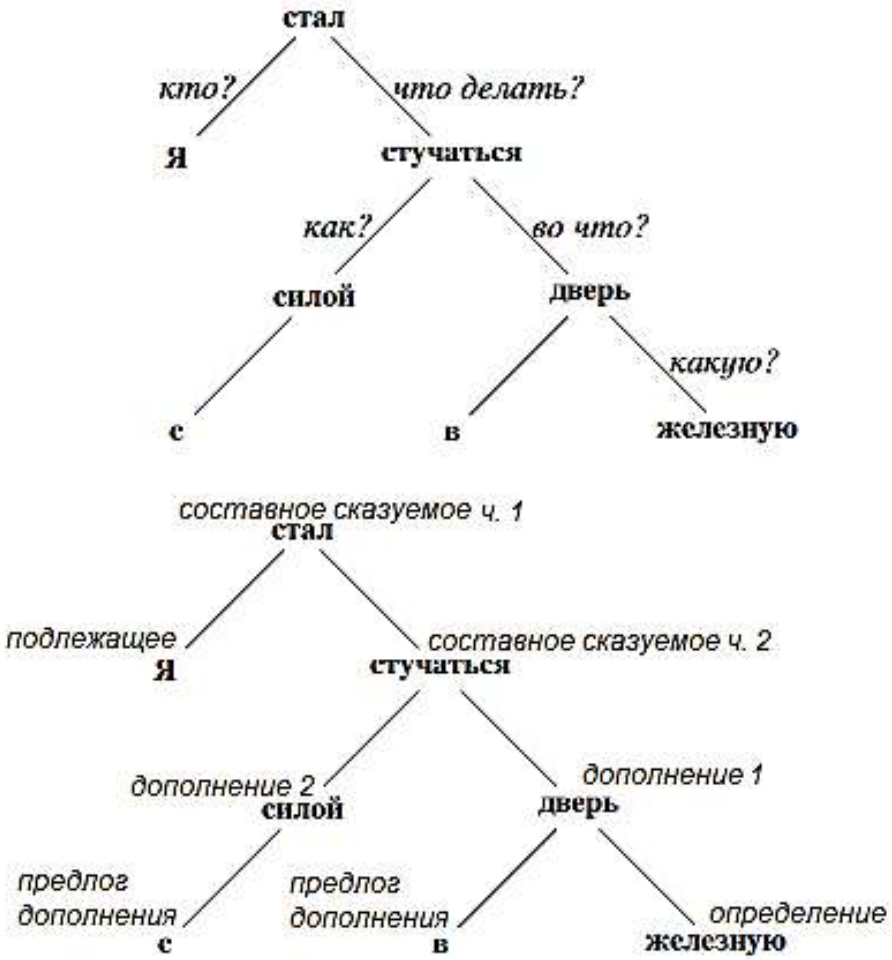

Рис. 4.4. Синтаксическое дерево для предложения «я стал силой стучаться в железную дверь»

Учитывая вышеизложенное, для решения задачи извлечения признаков объектов предлагается использовать следующий алгоритм, включающий извлечение терминов, с использованием гибридного подхода извлечения терминов, основанный на лингвистических и статистических методах, синтаксический разбор предложения и использование семантического тезауруса:

1. Извлечение всех одно-, дву-, трех- и четырех- словий из текста. Для извлечения многословий применяется следующий алгоритм:

А. Морфологический анализ слова и приведение слов к начальной форме. Например, возьмем словосочетание из текста: «стальных слитков». По результатам морфологического анализа мы получаем следующий вывод:

стальных {н.ф. стальной, прилагательное, множественное число, родительный падеж, полная форма}

слитков {н.ф. слиток, существительное, родительный падеж, множественное число, нарицательное, неодушевленное, мужской род, второе склонение}

Б. Согласование слов в многословии. Например, для получившегося из начальной формы словосочетания «стальной слиток» изменений не нужно, а вот словосочетания «стальной дверь» нужно провести согласование для получения в результате написания «стальная дверь».

2. Фильтрация многословий согласно описанным лингвистическим методам – шаблоны, стоп-слова, критерий открытости.

3. Применение статистических методов для ранжирования вероятности терминологичности и фильтрация многословий с вероятностью терминологичности больше 75%, при этом часть статистических критериев применяется не для подсчета вероятности терминологичности, а для фильтрации явно не подходящих словосочетаний. Так, например, Mutual Information применяется для фильтрации статистически незначимых словосочетаний.

4. Получение локального контекста каждого термина в существующем тексте.

5. Получение синтаксического дерева предложения, в котором содержится анализируемый термин. Для построения синтаксического дерева предложения используется библиотека «Natasha».

6. Выделение признаков, указанных явно.

7. Выделение признаков, указанных неявно, с использованием онтологического тезауруса «RuWordNet».

8. Сверка признаков, указанных явно, по правилам выделения признаков, указанных неявно.

Для понимания, что такое «явно» и «неявно», указанные признаки рассмотрим предложения «Балка красного цвета сломалась под тяжестью давления» и «Красная балка сломалась под тяжестью давления» в разрезе объекта «балка». У объекта «балка» в данных предложениях имеется переменная «цвет», конкретный признак – «красный». В первом случае признак указан явно, так как слово «цвет» уже фигурирует в предложении. Во втором случае слова «цвет» нет. Поэтому необходимо обратиться к тезаурусу RuWordNet для извлечения гиперонима «цветовой», из которого можно при помощи тезауруса РуТез получить понятие «цвет».

9. Построение дерева терминов, их признаков и конкретных характеристик.

Схематично описанный подход можно представить в виде схемы, отображенной на рисунке 4.5.

По итогу работы алгоритма получается дерево в формате XML следующего вида (рис. 4.6)

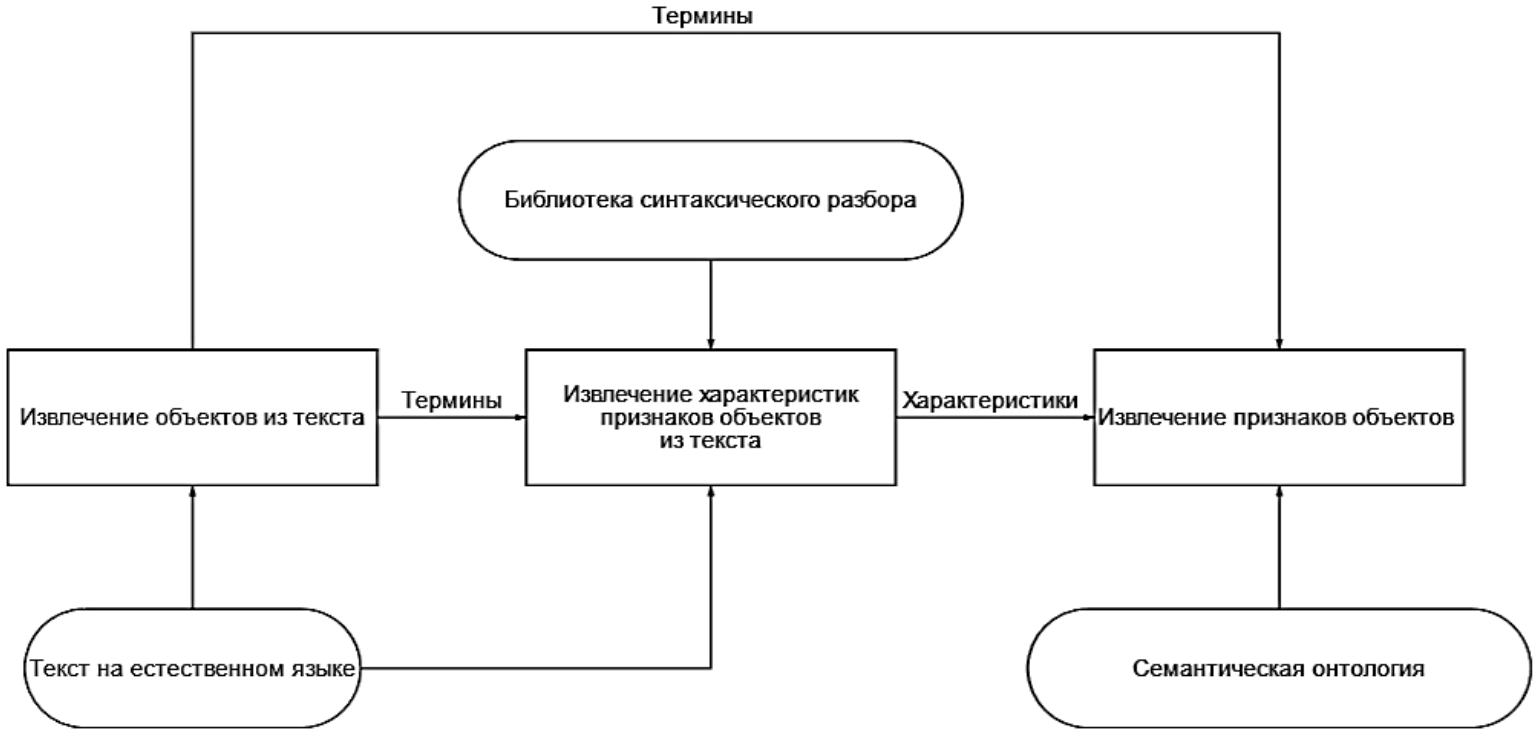

Рис. 4.5. Предлагаемый подход извлечения признаков объектов

```
<XML>
    <term name="Шарик" probability="73,22">
        <attribute type="Цвет">
            Красный
        </attribute>
        <attribute type="Цвет">
            Синий
        </attribute>
        <attribute type="Состав">
            Сталь
        </attribute>
        <attribute type="Состав">
            Дерево
        </attribute>
    </term>
</XML>
```

Рис. 4.6. Результат работы описанного подхода, содержащий объекты, переменные и признаки

Для доказательства работы алгоритма извлечения терминов в задаче извлечения признаков объектов был проведен эксперимент с привлечением экспертов. В рамках эксперимента были проведены пункты 1–3 описанного алгоритма. Корпус текстов был взят из предметной области «Сентимент-анализ». Пункты алгоритма 1–3 проводились автоматически и параллельно выполнялись экспертами вручную.

План эксперимента:

1. Эксперты извлекли из корпуса текстов 154 термина, при этом эксперт №1 извлек 132 термина, а эксперт №2 извлек 112 терминов. После объединения выборок и повторной проверки было оставлено 154 неповторяющихся термина, подтвержденных обоими экспертами.

2. Извлечение провелось программой-прототипом, реализующей пункты 1–3. По результатам работы было извлечено 211 терминов.

3. Было проанализированы выборки из пунктов 1 и 2, по результатам которых получились результаты, описанные в таблице 4.2.

Таблица 4.2. Результаты работы первого эксперимента

Показатели	**Количество терминов**	**Процент от общего числа**
Извлечено экспертами вручную	154	90%
Всего извлечено экспертами	173	100%
Извлечено программно (включая ложно-положительные)	211	Не оценивается
Извлечено программно после фильтрации экспертами	141	81%
Ложноположительные извлечено программно	70	Не оценивается

Из данных таблицы 4.2 видно, что программно удалось извлечь 81% от общего числа терминов, извлеченных экспертами. Важно уточнить, что точность работы самого алгоритма на данном этапе составила 67%, если учитывать ложноположительные срабатывания.

Для доказательства работы алгоритма извлечения признаков и переменных объектов по входному корпусу текстов и извлеченных из него объектов (терминов) был проведен второй эксперимент, охватывающий пункты 4–9 описанного алгоритма. Корпус текстов был взят из предметной области «Сентимент-анализ», объекты – отфильтрованные экспертами результаты предыдущего эксперимента (173 объекта). Пункты алгоритма 4–9 проводились автоматически и параллельно выполнялись экспертами вручную.

План эксперимента:

1. Эксперты извлекли из корпуса текстов все признаки и их переменные, опираясь только на 173 объекта, включенных в эксперимент. Результат был представлен в виде таблицы, пример для термина «эмоциональная окраска» представлен в таблице 4.3. Всего для 173

объектов из корпуса текстов было извлечено 224 параметра и 418 конкретных признаков.

Таблица 4.3. Пример результатов работы экспертов по выделению параметров и признаков

Термин	**Параметр**	**Признак**
окраска	эмоция	эмоциональный
		веселый
		злой

2. Результаты, извлеченные экспертами, были приведены в дерево формата XML, структурно повторяющего результат работы программного прототипа.

3. Извлечение провелось программой-прототипом, реализующей пункты 4–9. Всего для 173 объектов из корпуса текстов было извлечено 203 параметра и 430 конкретных признаков.

4. Была проведена экспертиза по сравнению результатов для нахождения ложноположительных результатов. Из 203 параметров, извлеченных программно, после фильтрации экспертами осталось 149 параметров, из чего можно сделать вывод, что точность работы алгоритма для выделения параметров составляет 73%. Из 430 конкретных признаков, выделенных программно, экспертами было оставлено 318, что составляет 74% от общего числа выделенных признаков.

Исходя из вышеизложенного, можно сделать вывод, что предлагаемый алгоритм к извлечению признаков объектов и определяемых их переменных из больших неструктурированных текстовых данных может быть применен на реальных текстах. В будущем планируется доработка данного алгоритма для повышения точности определения текста, а также разработка пользовательского интерфейса для удобного редактирования списка объектов и дерева признаков и переменных.

5 АЛГОРИТМ ИЗВЛЕЧЕНИЯ АТОМОВ ПРОДУКЦИОННЫХ ПРАВИЛ БАЗЫ ЭКСПЕРТНЫХ ЗНАНИЙ ИЗ НЕСТРУКТУРИРОВАННЫХ ТЕКСТОВ

5.1 Постановка задачи формирования правил из текстовых ресурсов

Для решения задачи логического вывода новых знаний зачастую используются системы правил. Однако формирование базы знаний, состоящей из экспертных правил, требует больших временных и трудозатрат. В связи с этим возникает задача автоматизированного или автоматического извлечения продукционных структур из текстовых ресурсов [67].

Текстовые ресурсы являются богатым источником знаний в различных предметных областях, однако имеют ряд особенностей, которые усложняют процесс их извлечения для решения конкретных задач:

– Неструктурированность.

– Синтаксические и морфологические особенности разных естественных языков. В первую очередь, это касается разнообразия синтаксических структур предложений и моделей повествования.

– Синонимия и омонимия, многозначность слов и словосочетаний. Данная проблема чаще всего решается посредством анализа лексического окружения встречаемых слов, а также анализа контекста.

– Множество жанров. Каждый жанр предполагает особые правила построения предложений, особенные речевые обороты и пр., что безусловно усложняет процесс анализа естественного языка на уровне унификации данных.

– Орфографические, речевые и синтаксические ошибки в тексте.

Задачи корректировки ошибок данного типа решаются с использованием специализированных программных библиотек (например, PySpellChecker), в которых априори заложены правила написания слов и предложений на выбранном языке. Помимо этого, необходимо учитывать контекст словоупотребления [68].

В связи с этим актуальной является задача разработки оригинальных подходов извлечения атомов продукционных правил базы экспертных знаний из неструктурированного текста посредством гибридизации методов обработки естественного языка и семантико-когнитивного анализа текстовых ресурсов, которые бы обеспечили объективность полученных результатов продукционного моделирования.

5.2 Продукционная модель представления знаний

Любое правило можно представить следующим образом:

$at_1\ Oper\ at_2\ Oper\ ...\ Oper\ at_n \rightarrow Con_1\ Oper\ Con_2\ Oper\ ...\ Oper\ Con_m,$

где $at_1 ... at_n$ – атомы условия (предикаты антецедента);

Oper – логические операторы ($\cap$, $\cup$, Not);

Con – атомы следствия (консеквента) правила.

Продукционную модель представления знаний можно представить, используя следующую информационную модель:

$$Name = <Sc, Cond^{Sc}, Core, Post, Com>$$

где *N* – это наименование продукции;

Sc – это предметная область продукции;

Cond – это условие применимости продукции в рамках предметной области *Sc*;

Core – это ядро продукции;

Post – это множество постусловий продукции, принимающие актуальные значения при реализации продукции;
Com – это комментарий к продукции, дополнительная информация о способе, форме и времени применения [69].

Любое правило в рамках продукционной модели представления знаний состоит из одной или нескольких пар «атрибут-значение», соединенных знаками логических операций. В рабочей памяти экспертных систем, основанных на продукционных моделях, хранятся пары «атрибут-значение», истинность которых доказана в результате решения конкретной задачи. Содержимое рабочей памяти экспертной системы изменяется в процессе решения задачи, по мере срабатывания правил.

Правило срабатывает, если при сопоставлении фактов, содержащихся в рабочей памяти, с антецедентом анализируемого правила имеет место совпадение, при этом заключение сработавшего правила заносится в рабочую память [70].

Поэтому в процессе логического вывода объем фактов в рабочей памяти увеличивается. В процессе логического вывода каждое правило из базы правил может сработать только один раз, но при этом результат логического вывода одного правила может стать входом для одного или нескольких правил из используемой системы продукции (рис. 5.1).

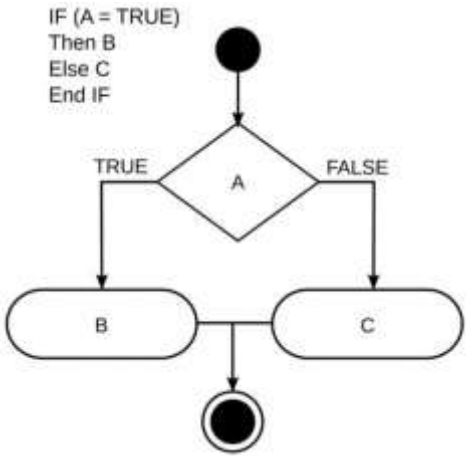

Рис. 5.1. Общая схема правила

Продукционная модель имеет недостаток: при накоплении большого числа правил, они могут противоречить друг другу. В связи с этим, важной составляющей процесса валидации базы знаний является проверка на полноту и непротиворечивость совокупности правил с использованием машины логического вывода (ризонера) [71].

В настоящее время применяются три вида экспертных систем, использующих продукционную форму представления знаний:

– Системы с прямым логическим выводом.

Системы с прямым выводом предполагают реализацию логического вывода от фактов к заключениям.

– Системы с обратным логическим выводом.

Работа систем с обратным выводом предполагает формулировку гипотезы вероятностных заключений. Эти заключения могут быть опровергнуты или подтверждены на основании фактов, которые поступают на вход правил.

– Системы, реализующие двунаправленные логические выводы [72].

К достоинствам экспертных систем, основанных на продукционных моделях, можно отнести:

- возможность реализации логического вывода новых знаний;
- модульность;
- единообразие структуры (элементы продукционной системы могут применяться для построения экспертных систем различных предметных областей);
- наглядность и естественность (логический вывод знаний на основе продукций аналогичен процессу рассуждений эксперта);
- легкость корректировки продукционных структур и внесения дополнений;

– гибкость родовидовой иерархии понятий, которая поддерживается только как связи между правилами (изменение правила влечет за собой изменение в иерархии) [73].

К недостаткам экспертных систем, основанных на продукционных моделях, можно отнести:

– отличие от структур знаний, свойственных человеку;

– неясность взаимных отношений правил;

– сложность оценки целостного образа знаний;

– низкая эффективность обработки знаний.

– Процесс логического вывода трудно поддается управлению;

– сложность оценки целостного образа знаний [74].

Если база знаний экспертной системы состоит из небольшого числа правил (до нескольких десятков), то эффективность их применения относительно высокая. Однако значительное увеличение числа правил ведет к появлению противоречий и усложняет процесс логического вывода, а процесс валидации базы знаний становится все более трудозатратным.

5.3 Языки представления правил

Одной из важных задач в рамках построения интеллектуальных систем является выбор эффективной модели и языка представления системы правил базы знаний. Приведем основные языки, используемые

Prolog. Все знания на языке Prolog представляются в виде набора фактов и правил. Все правила описывают абстрактные знания, которые можно вывести из других правил и фактов.

Любое правило на языке Prolog состоит из следующих частей:

– Голова правила. Голова правила включает имя правила и объявление списка аргументов.

– Тело правила. Тело правила включает множество предикатов (атомов), которые соединяются между собой лексическими аналогами логических операций (логическое «И» задается запятой, логическое «ИЛИ» задается точкой с запятой).

Атомы правил могут состоять из фактов, иных правил или встроенных предикатов, таких как арифметические операции, операции сравнения и пр.

Правило на языке Prolog имеет два состояния: истинное и ложное. Истинным является правило в том случае, когда истинно логическое выражение, хранимое в теле правила.

Примеры записи правил на языке Prolog представлены ниже:

```
–unpaidBill(BillId, ClientId, Date, AmoutToPay, AmountPaid) :- bill(BillId, ClientId, Date, AmoutToPay, AmountPaid), AmoutToPay < AmountPaid.
```

Данное правило предполагает, что все счета, у которых сумма к оплате меньше оплаченной суммы, являются неоплаченными.

```
–debtor(ClientId, Name, Email) :- client(ClientId, Name, Email), unpaidBill(BillId, ClientId, _, _, _).
```

В этом правиле предполагается, что должником является клиент, у которого есть хотя бы один неоплаченный счет [75].

CLIPS. CLIPS *(C Language Integrated Production System)* — среда разработки продукционной модели.

CLIPS включает в язык представления порождающих правил и язык описания процедур. Основными компонентами языка описания правил являются база фактов (fact base) и база правил (rule base). При этом, база фактов представляет исходное состояние проблемы, а база правил включает операторы, которые преобразуют состояние проблемы, приводя его к решению.

Встроенная машина логического вывода (ризонер) CLIPS сопоставляет эти факты и правила и выясняет, какие из правил можно активизировать. Это выполняется циклически, причем каждый цикл состоит из трех шагов:

– сопоставление фактов и правил;

– выбор правила, подлежащего активизации;

– выполнение действий, предписанных правилом.

В языке CLIPS правила можно представить с использованием следующей модели:

```
(defrule < наименование правила>
< комментарий (необязательный) >
< объявление (необязательное>
< условие_1 >
< условие_m > ›
< следствие_1 >
< следствие_n >)
```

Приведем пример:

```
(defrule chores
"Things to do on Sunday"
(salience 10)
(today is Sunday)
(weather is warm) ›
(assert (wash car))
(assert (chop wood))
```

Также в определении правила присутствуют и переменные. Например, правило:

```
(defrule pick-a-chore
"Allocating chores to days"
(today is?day)
(chore is?job) ›
(assert (do?job on?day)))
```

В данном случае ?job и ?day являются переменными, значения которых устанавливаются в области объявления фактов [76].

SWRL. В рамках концепции Semantic Web популярность получила форма представления правил в формате SWRL (Semantic Web Rule Language).

Главным преимуществом данного вида представления знаний является возможность беспрепятственной интеграции с семантическими структурами – онтологиями в формате OWL.

SWRL – это технология, которая помогает описать абстрактный механизм оперирования объектами предметной области, а также закономерности предметной области. Помимо возможности интеграции с OWL-онтологиями, формат SWRL позволяет выводить новые факты из существующих утверждений.

SWRL-правила – это простейшие логические структуры, связывающие условия (антецедент) со следствием (консеквентом). Представление рассматриваемой области знаний в виде причинно-следственных связей объясняется тем, что рекомендации и выводы в области анализа состояния сложной технической системы имеют форму составных условных высказываний.

Архитектура SWRL-правила представлена на рисунке 5.2.

SWRL-правила представляют собой дизъюнкты Хорна и могут быть представлены в следующем виде:

$$Ax : C_1(?x) \wedge C_2(?y) \wedge r_1.(?x, ?y) \wedge C_3(?x, ?z) \rightarrow C_2(?z, ?y),$$

где $(C_1, C_2, C_3) \in C$, $r_1 \in R$, x, y – экземпляры или переменные,

z –переменные или значения.

Набор правил, которые обеспечивают логический вывод результатов анализа, можно разделить на две группы:

1. SWRL-правила, следствием которых является конкретные

факты, которые представляют собой новые знания. Модель данного правила может быть представлена в следующем виде:

$$p_1(x, d) \wedge p_2(d, s) \wedge \ldots p_i(x, c_1) \wedge g(c_1, c_2) \wedge \ldots \rightarrow p_n(x, h) .$$

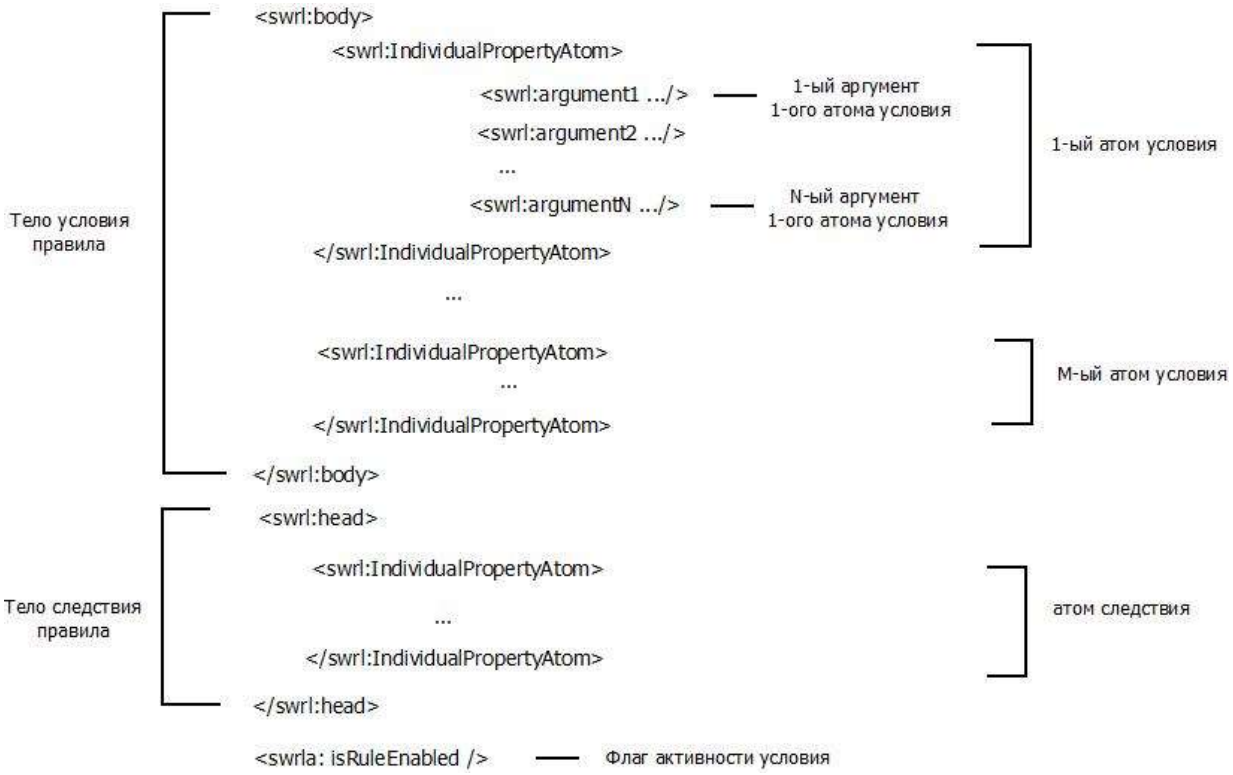

Рис. 5.2. Общая схема SWRL-правила

2. SWRL-правила, консеквентом которых является логическое присвоение одному из объектов-характеристик сети конкретного значения. Далее это значение может выступать в роли входных данных других SWRL-правил. Модель данного правила может быть представлена в следующем виде:

$$p_1(x, d) \wedge p_2(d, s) \wedge \ldots p_i(x, c_1) \wedge g(c_1, c_2) \wedge \ldots \rightarrow p_n(x, c_n) \wedge g_n(c_n, c_{n+1}),$$

где $x, h \in H, p_1, p_2, \ldots, p_i \ldots p_n \in P, g \ldots g_n \in G, d \in D, c_1, c_2 \ldots c_n, c_{n+1} \in C.$

Описанные выше группы SWRL-правил представлены на рисунке 5.3.

Примеры представления правил на языке SWRL:

- *Имеет_тенденцию(Инвестиции_в_основной_капитал,Рост)* ∩ *Имеет_тенденцию(инвестиции_в_основной_капитал_на_душу_населения,Стабильность)* ∩ *Имеет_тенденцию(Индекс_физического_объема_инвестиций_в_основной_капитал,Стабильность)* → *Потенциал_Средний(Инвестиционный_потенциал)*
- *Ситуация(?x)* ∩ *включаетПоказатель(?x, Фоновая_нагрузка)* ∩ *имеетТенденцию(Фоновая_нагрузка, Рост)* ∩ *включаетПоказатель(?x,Утилизация_канала_связи)* ∩ *имеетТенденцию(Утилизация_канала_связи, Спад)* ∩ *включаетПоказатель(?x, Число_коллизий_в_сети)* ∩ *имеетТенденцию(Число_коллизий_в_сети, Рост)* → *предполагает(?x, Проблема_архитектуры)*
- *Имеет_тенденцию(ВРП, Спад)* ∩ *Имеет_тенденцию(ВРП_на_душу_населения,Стабильность)* ∩ *Имеет_тенденцию(Индекс_физического_объема_ВРП, Спад)* → *Потенциал_Низкий(Производственный_потенциал)*

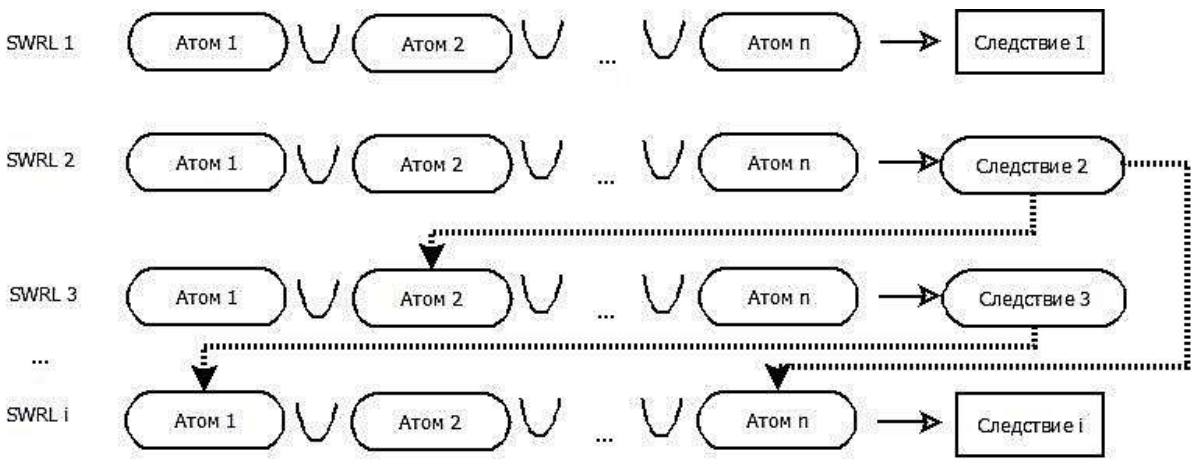

Рис. 5.3. Группы SWRL-правил

5.4 Подход к извлечению атомов продукционных структур в формате SWRL из неструктурированных текстов

Задача извлечения продукционных структур из неструктурированных ресурсов является нетривиальной и предполагает использование гибридных алгоритмов, включающих подходы обработки естественного языка и семантико-когнитивные методы с использованием онтологических структур. Перечислим базовые этапы предложенного подхода:

1. На первом этапе предполагается разбиение анализируемого текста на предложения в соответствии со знаками препинания.

В дальнейшем для анализа используются только предложения, оканчивающиеся на знаки из множества:

$$PM= \{“.”,\ “!”,\ “...”\}.$$

2. На следующем этапе предполагается детектирование правил с использованием лингвистических шаблонов. Базовые лингвистические шаблоны:

Если <Ant>, то <Cons>;

Когда <Ant>, тогда <Cons>,

где *Ant* – предполагаемое условие (антецедент) правила,

Cons – предполагаемое следствие (консеквент) правила.

3. Следующие этап включает детектирование и выделение структурных шаблонов внутри антецедента и консеквента вида:

<Если Субъект Свойство Объект>, то < Субъект Свойство Объект>

В качестве знаков логических операций между атомами выступают лексические элементы «и», «или», «не, нет».

4. Сопоставление выделенных объектов и классов с соответствующими элементами OWL-онтологии.

Интеграция онтологической и продукционной моделей знаний предполагает использование в качестве объектов SWRL аналогичные объекты классов, описанных в построенной OWL-онтологии.

Сопоставление выделенных из текста объектов и классов с соответствующими элементами OWL-онтологии обеспечивается с помощью формирования запросов к онтологии, генерируемых системой анализа при выполнении набора правил.

Модель представления экспертных знаний в виде набора SWRL-правил имеет ряд преимуществ перед остальными подобными технологиями:

– правила SWRL не содержат конкретных объектов, а только ссылаются на них, что дает возможность применять одно и то же правило к ряду групп объектов;

– правила SWRL могут быть добавлены к OWL-описанию, т. е. включены в онтологию.

Схематично процесс сопоставления элементов OWL-онтологии и компонентов SWRL-правил представлен на рисунке 5.4.

5. Следующий этап предполагает применение семантических структур (в частности, OWL-онтологии) для извлечения признаков и характеристик объектов с учетом вероятности синонимии и использования встроенных отношений.

Данный этап расширяет результаты предыдущего этапа посредством поиска объектов OWL-онтологии, обладающих свойствами синонимии и омонимии, к тем терминам, которые были выделены из текстового фрагмента.

В результате сокращается терминологическое облако, с которым нужно работать алгоритму при формировании атомов правил.

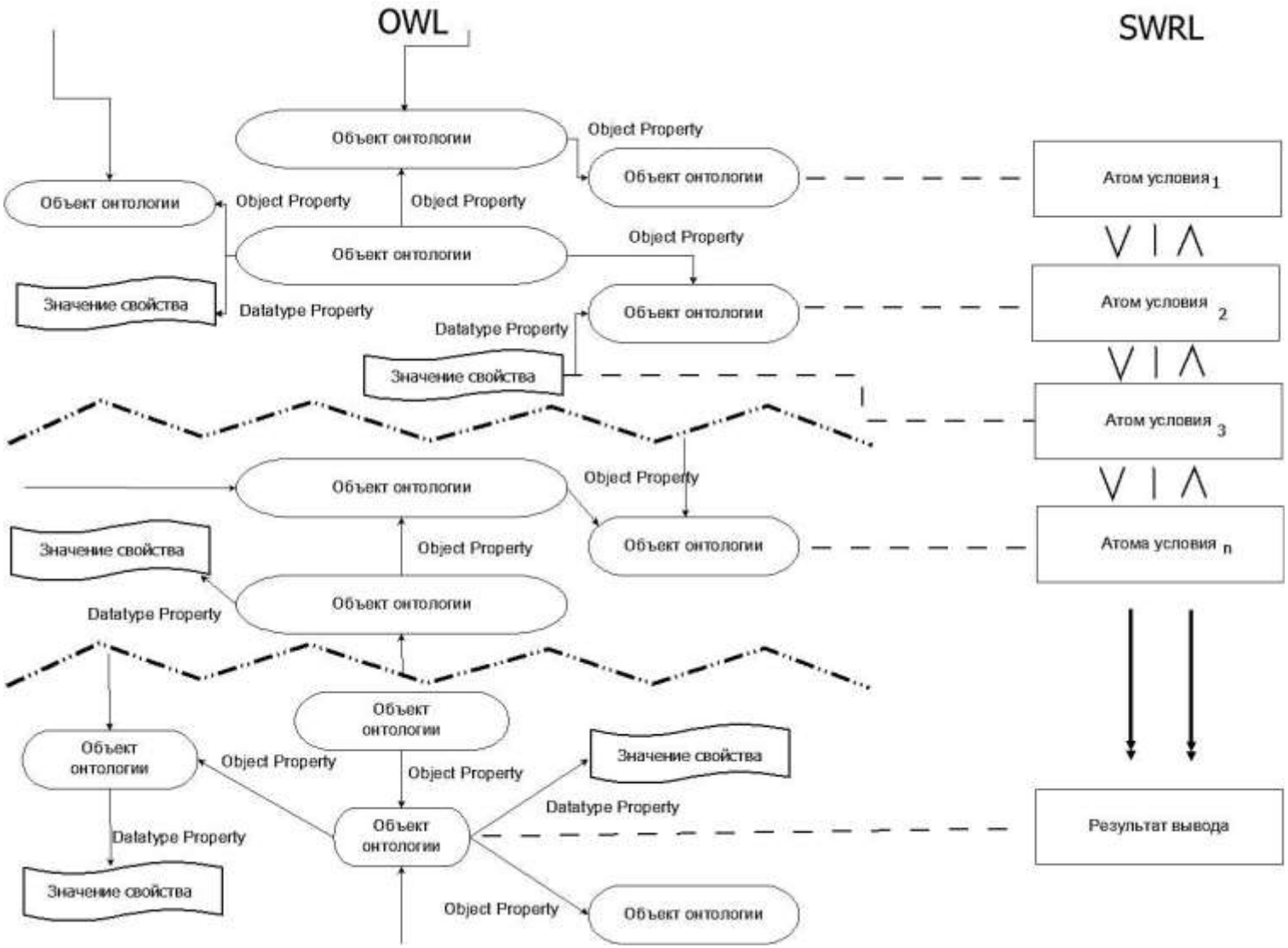

Рис. 5.4. Процесс сопоставления элементов OWL-онтологии и компонентов SWRL-правил

6. После получения конечного набора классов, субъектов, объектов, их свойств и значений этих свойств, предполагается итоговая трансляция полученных структур в унифицированную форму SWRL-правил по априорным шаблонам трансляции.

В результате применения данного подхода формируется набор SWRL-правил, которые могут быть задействованы в процессе логического вывода знаний с помощью следующих ризонеров (рассуждателей, решателей, машин логического вывода):

– Pellet
– Fact++
– HermiT и пр.

Приведем примеры лексического представления правил и результатов их трансляции в SWRL-формат.

Правило 1:

Текстовое представление: «Если топливо поступает в двигатель и двигатель вращается, то проблема в свечах зажигания».

SWRL-формат: *ПоступаетВ(Топливо, Двигатель)* ∩ *Вращается(Двигатель, Истина)* → *ПроблемаВСвечахЗажигания(Проблема)*

Правило 2:

Текстовое представление: «Если двигатель не вращается и фары не горят, то проблема в аккумуляторе или проводке».

SWRL-формат: *Вращается(Двигатель, Истина)* ∩ *Горят(Фары, Ложь)* → *ПроблемаВАккумуляторе(Проблема)* ∪ *ПроблемаВПроводке(Проблема)*

Правило 3:

Текстовое представление: «Если двигатель не вращается и фары горят, то проблема в стартере».

SWRL-формат: *Вращается(Двигатель, Ложь)* ∩ *Горят(Фары, Истина)* → *ПроблемаВСтартере(Проблема)*

Правило 4:

Текстовое представление: «Если в баке есть топливо и топливо поступает в карбюратор, то топливо поступает в двигатель».

SWRL-формат: *НаходитсяВ(Топливо,Бак)* ∩ *ПоступаетВ(Топливо,Карбюратор)* → *ПоступаетВ(Топливо,Двигатель).*

Особенностью предложенного подхода является необходимость описания объектов, субъектов и свойств, содержащихся в атомах полученных правил в интегрируемой OWL-онтологии. Это необходимо для реализации условия семантической целостности системы правил.

6 ИНТЕЛЛЕКТУАЛЬНАЯ ПРОГРАММНАЯ СИСТЕМА ДЛЯ ФОРМИРОВАНИЯ КОНТЕКСТА АНАЛИЗА ДАННЫХ ДИНАМИКИ ПОКАЗАТЕЛЕЙ

6.1 Формирование контекста для анализа динамики показателей

По результатам статистики в 1994 году количество успешных проектов программных систем достигало значения 16%. К настоящему времени данный показатель повысился примерно в два раза [77].

При разработке новой программной системы от разработчика требуется погружение в особенности предметной области проекта для определения перечня сущностей и бизнес-процессов, которые важны с точки зрения решаемой задачи [78,79]. Выделенные сущности и бизнес-процессы формируют операционное пространство проектной деятельности проекта.

Далее разработчики формируют концептуальную картину миру, в рамках которого будет происходить функционирование разрабатываемой системы. Концептуальная картина мира формируется в рамках умственной деятельности разработчиков с учетом выделенных ранее сущностей и бизнес-процессов предметной области, но наиболее важное место в данном случае занимает опыт разработчика и степень понимания особенностей операционного пространства. Концептуальная картина мира формирует концептуальное пространство проектной деятельности проекта.

После формирования операционного и концептуального пространств проекта разработчики переходят к процессу разработки – формирование множества артефактов проектирования и физическое воплощение проекта.

В настоящее время существует базовая теория программной инженерии, которая определяет общие требования и подходы к разработке программного обеспечения, но не дает вопрос на конкретные вопросы, возникающие в процессе формирования операционного и концептуального пространства, а также при разработке программной системы [78,79].

Также при разработке любого проекта программной системы требуется осуществлять контроль и планирование деятельности разработчиков. Ресурсы любого проекта всегда ограничены [80]. Для успешного управления проектом требуется оперативно анализировать различные данные, полученные из различных источников, представленные различными форматами. Принимаемые на любом этапе жизненного цикла программной системы решения непосредственно влияют на качество этой системы в целом и отдельных ее артефактов проектирования в частности.

К основным проблемам современной разработки программных систем можно отнести:

1. Высокий уровень неопределенности при разработке проектов для новой проблемной области или с использованием новых архитектурных подходов или технологий.
2. Влияние внешней среды на процесс разработки, в том числе резкое сокращение ресурсов и сроков.
3. Отсутствие необходимых компетенций у членов команды.
4. Необходимость оперативной оценки большого числа факторов, влияющих на успешность проекта и качество принимаемых решений.

Следовательно, требуется разработка новых моделей, методов и алгоритмов для формализации и накопления опыта разработчиков, полученных в рамках предыдущих проектов, для его повторного использования при работе над новыми проектами, а также для автоматизации процесса поддержки принятия управленческих решений [79,81–86]. Также следует учитывать не только удачные или неудачные проектные решения, но и особенности процесса разработки проекта. Все необходимые данные для решения поставленной задачи могут быть извлечены из систем контроля версий и систем управления проектами в процессе индексирования для последующей формализации [88–93].

6.1.1 Предложенные модели, методы и алгоритмы

Современные практики разработки ПС в основном базируются на гибких методологиях разработки [79,80], позволяющих:

- оперативно реагировать на изменения требований заказчиков;
- демонстрировать заказчикам срезы полученной в процессе разработки ПС функциональности для оценки и возможных уточнений и корректировок;
- повысить оперативность принятия управленческих решений.

Также процесс разработки ПС часто производится с применением методологии Design Thinking (DT) [77–80]. Основной особенностью DT является решение инженерных, деловых и прочих задач, основываясь на творческом, а не аналитическом подходе. При использовании DT задачи решаются не на основе критического анализа, а рассматриваются как творческий процесс, что позволяет находить неожиданные и неочевидные решения. Обычно в рамках методологии DT выделяют следующие этапы работы над решением проблемы [77–80]:

1. Определение проблемы.
2. Исследование.
3. Формирование идей.
4. Прототипирование.
5. Выбор лучшего решения.
6. Внедрение решения.
7. Оценка результатов.

Будем рассматривать каждую итерацию процесса разработки на основе гибких методологий как совокупность следующих этапов: планирование, проектирование, конструирование (рис. 6.1).

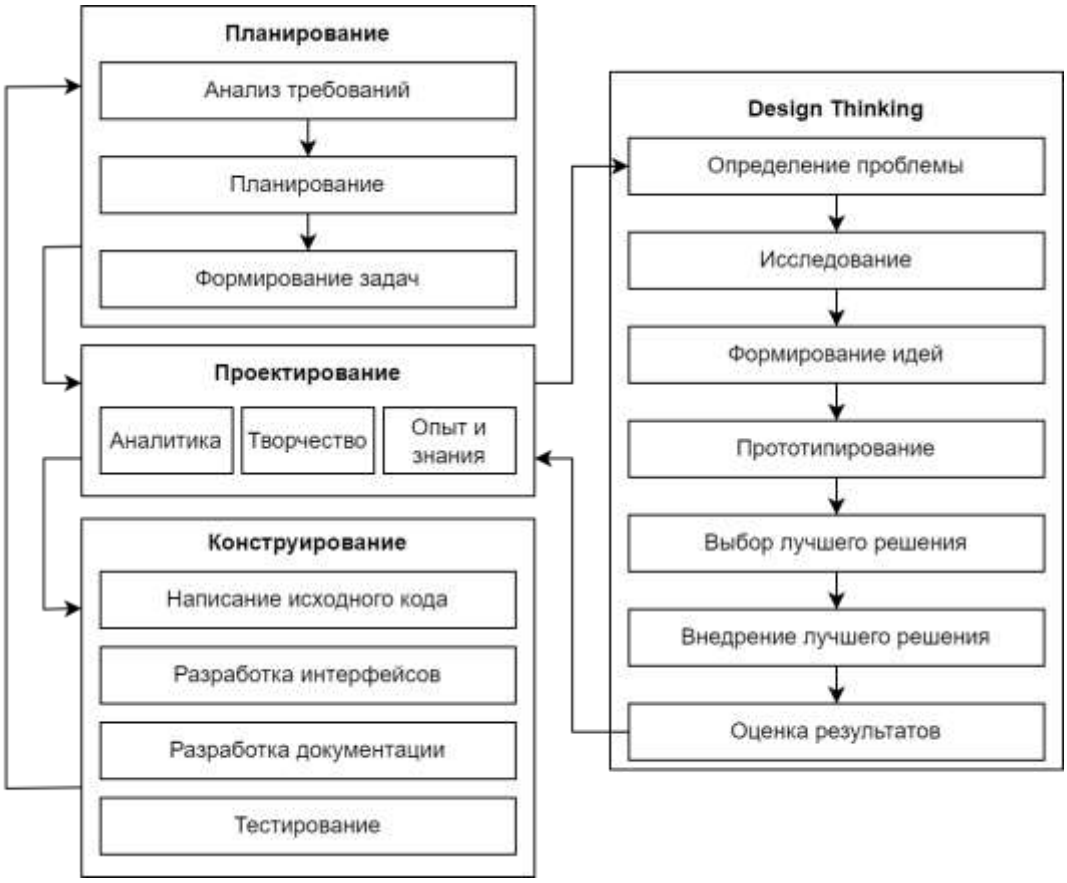

Рис. 6.1. Структура итерации процесса разработки на основе гибких методологий

Как видно из рисунка 6.1, на качество этапа конструирования ПС влияет качество этапов планирования и проектирования:

1. Результат этапа планирования во многом зависит от качества проработки функциональных и нефункциональных требований [80], а также от качества управленческих решений. Управленческие решения можно выразить в виде множества сформированных задач для разработчиков, а также в виде множества кадровых решений. При этом на этапе планирования необходимо учитывать ограничения ресурсов, выделенных на выполнение проекта, ограничения реального мира, а также требования к качеству разрабатываемой ПС.

2. Результат этапа проектирования зависит от этапа планирования и квалификации проектировщика. Как видно из обзора публикаций по тематике исследования, применение методов интеллектуального анализа и

инженерии знаний позволяет автоматизировать этап проектирования на основе формализации опыта предыдущих проектов. Также этап проектирования представляет собой творческий процесс с точки зрения методологии DT, что требует разработки средств автоматизированного формирования КП [78,79].

6.1.2 Модель базы знаний для учета опыта предыдущих проектов

В рамках данного исследования модель базы знаний для учета опыта предыдущих проектов имеет вид

$$B = \{B_1, B_2, \dots, B_i, \dots, B_n\},$$

где B_i – i-й проиндексированный проект:

$$B_i = \langle L^B, P^B, T^B, D^B, W^B, R^B \rangle,$$

где L^B – представление процесса разработки. Данное представление позволяет учесть специфику жизненного цикла проекта. Кроме того, это представление позволяет менеджеру проекта оценить влияние управленческих решений на динамику показателей программного проекта, например, как изменение числа разработчиков повлияло на активность разработки или качество определенных артефактов проектирования и т. д.;

P^B – представление структуры программного проекта (каталоги и файлы). Данное представление позволяет получить информацию о структуре файлов и каталогов программного проекта, чтобы классифицировать файлы по следующим типам: исходный код, документация, инструменты для сборки/компиляции кода, тесты, каталог дополнительных данных, внешние зависимости (библиотеки), каталог с исполняемыми файлами и т. д. Эта информация позволяет использовать необходимые методы анализа файлов разных типов, а также учитывать структуру проекта при извлечении шаблонов проектирования;

T^B – представление среды исполнения программного проекта (набор технологических компонентов). Данное представление позволяет получить

информацию о среде программной системы: зависимости (библиотеки), внешние компоненты (сервисы), среды исполнения и т. д. Кроме того, такое представление позволяет учитывать различные стили архитектуры и шаблоны проектирования, которые разработчики использовали в проекте. Информация о среде является важной, поскольку неправильно настроенная среда может вызвать ошибки в программной системе. Информация о среде также позволяет исследовать только те завершенные проекты, которые соответствуют требованиям текущего проекта;

D^B – представление особенностей предметной области. Данное представление позволяет определить операционное и концептуальное пространства проекта;

W^B – представление языковой среды проекта. Языковая среда проекта позволяет установить соответствие между объектами, имеющими разные названия, но имеющие одинаковую семантику, например, работник и персонал, девелопмент и строительство и т. д;

R^B – множество отношений между представлениями.

Представление процесса разработки проекта L^B представим как

$$L^B = \langle M, I, R, B, C, D, Cm, Cn, R_L \rangle,$$

где M – множество этапов;

I – множество задач;

R – множество запросов на слияние;

B – множество веток;

C – множество изменений;

D – множество текстовых описаний, представленных текстом на ЕЯ;

Cm – множество комментариев, представленных текстом на ЕЯ;

Cn – множество разработчиков;

R_L – множество отношений между компонентами представления L (рис. 6.2).

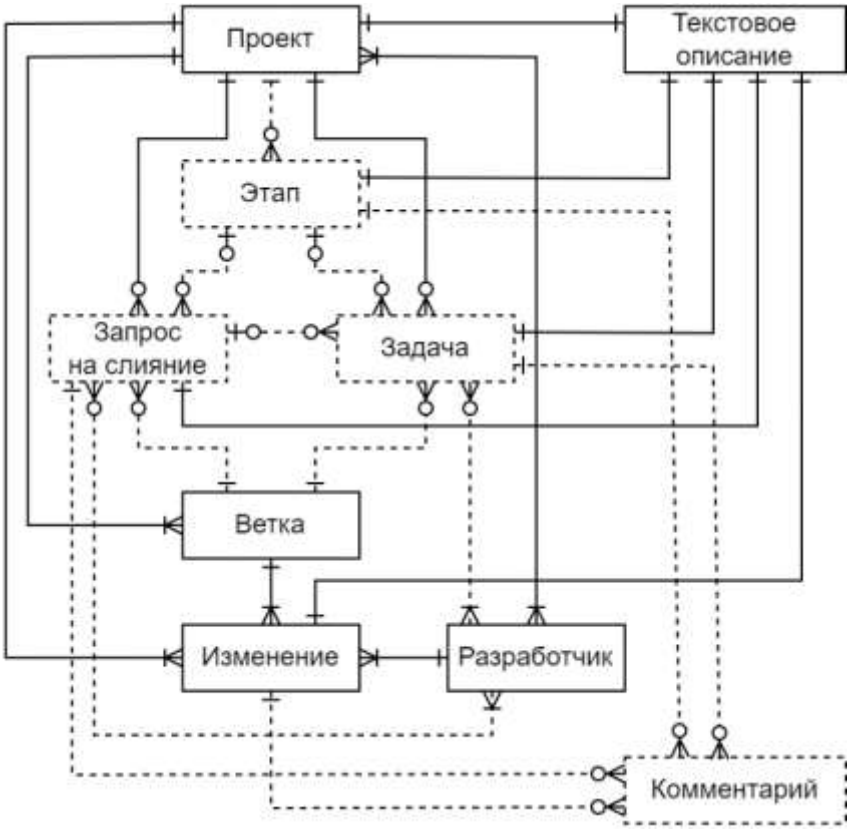

Рис. 6.2. ER-диаграмма представления процесса разработки проекта *L*

Как видно из рисунка 6.2, для иллюстрации некоторых сущностей и отношений используются пунктирные линии. Такие сущности и отношения могут не содержаться в индексируемом репозитории и, следовательно, могут быть не представлены в базе знаний.

Формирование фрагментов базы знаний для представления процесса разработки L^B происходит в процессе индексации репозитория в разрезе изменений и связанных с ними сущностей представления [94].

Представление организации файлов P опишем с помощью выражения:

$$P = \langle D, F, R_P \rangle,$$

в котором D – каталоги файловой системы;

F – фалы, в которых представлен исходный код проекта ПС и различные конфигурационные файлы при наличии;

R_P – отношение для формирования иерархии файлов и каталогов.

Представление структуры программного проекта P^B формируется в процессе структурного анализа проекта ПС [95].

Представление среды исполнения программного проекта имеет вид:

$$T^B = \langle T^{Arch}, T^{Dep}, Arch, Dep \rangle,$$

где $T^{Arch} = \{T_1^{Arch}, T_2^{Arch}, \dots, T_j^{Arch}, \dots, T_l^{Arch}\}$ – множество обнаруженных в проекте архитектурных стилей (MVC, REST и т. д.) и/или шаблонов проектирования (фабрика, фасад и т. д.);

$T^{Dep} = \{T_1^{Dep}, T_2^{Dep}, \dots, T_k^{Dep}, \dots, T_m^{Dep}\}$ – множество используемых в проекте сторонних зависимостей (библиотеки, фреймворки, внешние сервисы, системы управления базами данных и т. д.);

$Arch$ – компонент для поиска архитектурных стилей и/или шаблонов проектирования;

Dep – компонент для поиска используемых сторонних зависимостей.

Компоненты $Arch$ и Dep необходимо настраивать вручную для каждого языка программирования, фреймворка, типа приложения и других особенностей окружения проекта ПС. Более подробно данная задача и перспективы по ее автоматизации буду рассмотрены в одной из следующих статей.

Представление особенностей предметной области D^B опишем выражением:

$$D^B = \langle E, P, R_D \rangle,$$

в котором E – множество сущностей предметной области, в рамках которой разрабатывается проект ПС;

P – множество бизнес-процессов предметной области, в которые вовлечены сущности из множества E;

$R_D = \langle R_D^{EE}, R_D^{EP} \rangle$ – множество отношений R_D^{EE} между сущностями, а также множество отношений R_D^{EP} между сущностями и бизнес-процессами соответственно.

Множества сущностей E и бизнес-процессов P предметной области данного представления формируются в процессе индексирования исходного кода ПС [95].

Языковую среду проекта W^B представим как:

$$W^B = \langle C, T, R_W \rangle,$$

где C – множество понятий предметной области;

T – множество терминов, которые описывают понятия предметной области;

$R_W = \langle R_W^{CC}, R_W^{CT} \rangle$ – множество отношений R_W^{CC} для построения иерархии понятий, а также множество отношений R_W^{CT} для организации связи между понятиями и описывающими их терминами.

Терминологическое окружение извлекается в процессе анализа [96,97] различных текстовых описаний на ЕЯ, содержащихся в репозитории проекта ПС, с применением методов статистического и лингвистического анализа.

Таким образом, полученная база знаний представляет собой источник проектного опыта, на основе которого можно формировать методы автоматизации построения КП для автоматизации этапа проектирования.

Привязка сущностей представлений базы знаний к отдельным изменениям проекта вносит в формализованное описание проекта ПС динамическую компоненту. Представление сведений о проекте в динамике позволяет использовать различные методы диагностической и предиктивной аналитики для повышения качества и оперативности принятия управленческих решений.

6.1.3 Формализация опыта предыдущих проектов

Формализация опыта предыдущих проектов осуществляется на основе следующего алгоритма.

Рассмотрим пример индексирования проекта программной системы для планирования научной деятельности ng-tracker. Данный проект написан на языке Java с использованием фреймворка Spring Boot. Для компактности будем рассматривать не весь исходный код проекта, а только пакет ru.ulstu.conference, и связанную с ним задачу №57 «Создание классов для модуля Конференции» этапа №681923«Конференции».

Шаг 1. Извлечение представления процесса разработки L^B

Представление процесса разработки L^B формируется из двух источников:

– API веб-хостинга ИТ-проектов (GitLab, GitHub и т. д.);

– git-репозиторий проекта.

Git-репозиторий проекта считается более предпочтительным источником, так как является более стабильным с точки зрения смены API, а также является постоянно доступным для чтения. Однако из git-репозитория невозможно извлечь сведения об этапах процесса разработки: этапы, задачи, запросы на слияние.

В настоящий момент разработан модуль для работы с API хостинга GitLab. Взаимодействие между модулем и GitLab API осуществляется через следующие HTTP-запросы:

1. GET https://gitlab.com/api/v4/projects/romanov73%2Fng-tracker/milestones – получение списка этапов проекта ng-tracker.

2. GET https://gitlab.com/api/v4/projects/romanov73%2Fng-tracker/milestones/681923/issues – получение списка задач этапа «Конференции» проекта ng-tracker.

3. GET https://gitlab.com/api/v4/projects/romanov73%2Fng-tracker/issues/57/related_merge_requests – получение списка запросов на слияние для задачи «Создание классов для модуля Конференции» проекта ng-tracker.

4. GET https://gitlab.com/api/v4/projects/romanov73%2Fng-tracker/merge_requests/42/commits – получение списка изменений для запроса на слияние с идентификатором 42 проекта ng-tracker.

Далее для каждого изменения с помощью библиотеки JGit выполняется извлечение следующих сведений:

– хэш изменения;
– дата изменения;
– описание изменения;
– ветка;
– автор изменения;
– файлы, затронутые изменением.

Если в проекте отсутствуют этапы и задачи, то происходит получение сведений только об изменениях.

Для проекта ng-tracker, этапа «Конференции» и задачи «Создание классов для модуля Конференции» получен фрагмент представления L^B, изображен на рисунке 6.3.

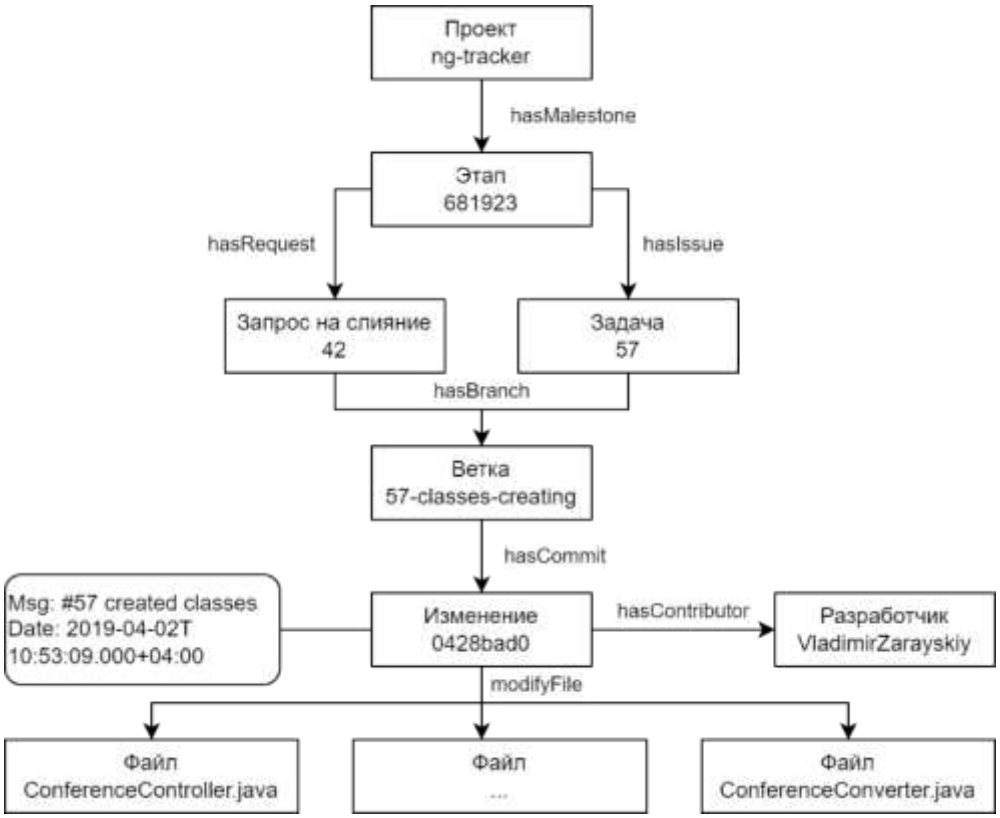

Рис. 6.3. Пример фрагмента базы знаний для представления L^B

Все примеры имеет иллюстративный характер, в хранилище для каждой сущности генерируется первичный ключ и все связи между сущностями формируются на основе этих ключей.

Шаг 2. Извлечение представления организации файлов P^B

Представление организации файлов P^B формируется в процессе анализа иерархии файлов и каталогов исходного кода проекта. Для разных языков программирования и систем сборки в индексаторе содержатся типовые пути для поиска исходного кода. Для проекта ng-tracker получен фрагмент представления P^B, изображенный на рисунке 6.4.

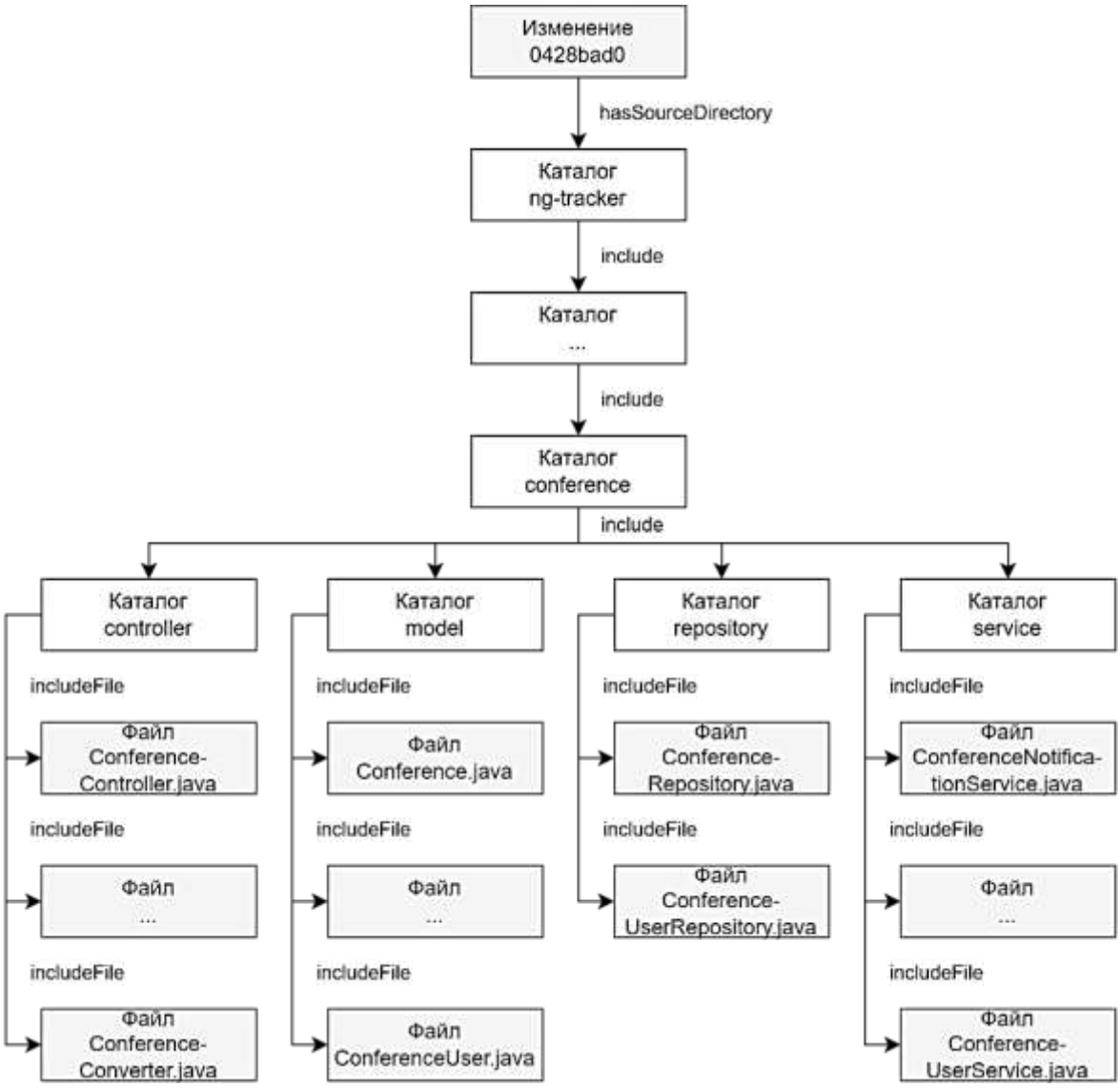

Рис. 6.4. Пример фрагмента базы знаний для представления P^B

На рисунке 6.4 серым цветом отмечены сущности представления P^B, которые также присутствуют в представлении L^B, а именно сущности с типом «Файл» и сущность «0428bad0» с типом «Изменение».

Шаг 3. Извлечение представления технологической составляющей T^B

Представление технологической составляющей T^B формируется в процессе поиска определенных шаблонов $Arch$ и Dep в репозитории программной системы.

Например, для проектов на Spring определение использования шаблона MVC заключается в поиске аннотации @Controller для класса, и

аннотаций @GetMapping, @PostMapping, @RequestMapping и т. д. для его методов (листинг 1). Для поиска составных шаблонов проектирования необходимо учитывать также сведения о структуре проекта из представления организации файлов P^B.

Листинг 1. Фрагмент MVC-контроллера приложения ng-tracker

```
@Controller()
@RequestMapping(value = "/conferences")
@ApiIgnore
public class ConferenceController {
    …
    @GetMapping("/conferences")
    public void getConferences(ModelMap modelMap) {
        modelMap.put("filteredConferences",
            new
ConferenceFilterDto(conferenceService.findAllDto()));
    }
    …
}
```

Компонент для поиска используемых сторонних зависимостей Dep позволяет находить файлы конфигурации систем автоматизации сборки и извлекать из них название библиотек-зависимостей. Например, для Java проектов осуществляется поиск файлов build.gradle, pom.xml и т. д. Если найден файл build.gradle, то в качестве среды сборки проекта указывается Gralde, а затем из секции dependencies этого файла извлекаются названия и версии библиотек-зависимостей (листинг 2).

Листинг 2. Фрагмент файла *build.gradle* проекта ng-tracker

```
dependencies {
    compile group: 'org.springframework.boot', name: 'spring-boot-
starter-web'
    compile group: 'org.springframework.boot', name: 'spring-boot-
starter-security'
    compile group: 'org.postgresql', name: 'postgresql', version:
'42.2.5'
    compile group: 'org.liquibase', name: 'liquibase-core', version:
'3.6.3'
}
```

Иллюстративный пример обнаруженных в проекте архитектурных стилей и сторонних зависимостей представления T изображены на рисунке 6.5(a) и 6.5(b).

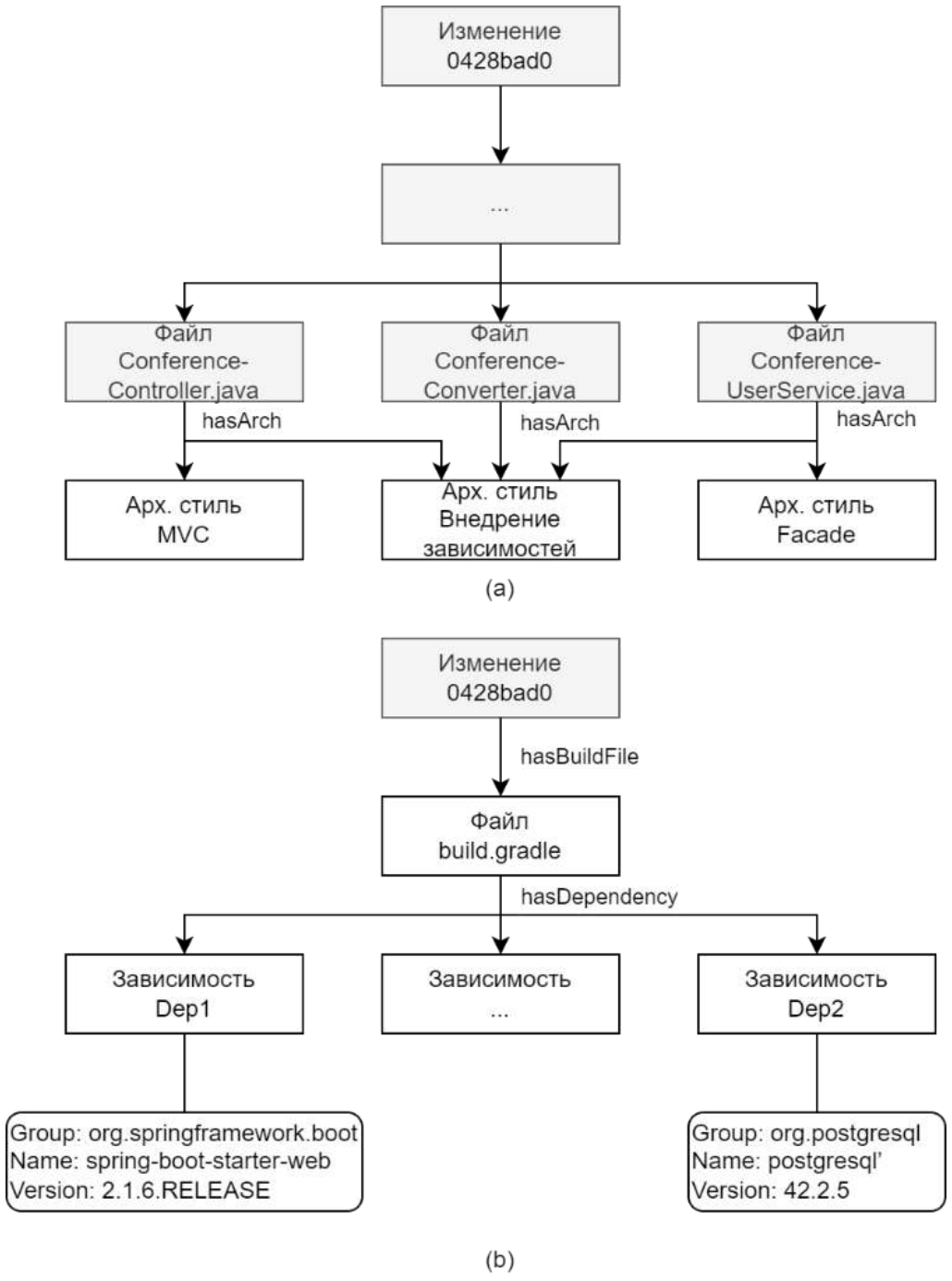

Рис. 6.5. Пример фрагмента базы знаний для представления *T*

На рисунке 6.5 серым цветом отмечены сущности, которые были использованы в фрагментах других представлений базы знаний (рис. 6.3 и 6.4).

Шаг 4. Извлечение представления особенностей предметной области D^B

Представление особенностей предметной области D^B формируется в процессе поиска в файлах исходного кода проекта классов, которые описывают модели данных и бизнес-логику. Для каждого языка программирования и фреймворка требуется настройка индексатора для поиска соответствующих операторов языка или элементов фреймворка. Например, для проектов на языке программирования Java и фреймворке Spring Boot классы с описанием моделей данных помечаются аннотацией @Entity (листинг 3), а классы с реализацией бизнес-логики аннотацией @Service (листинг 4).

Листинг 3. Фрагмент класса-сущности проекта ng-tracker

```
@Entity
@Table(name = "conference")
@DiscriminatorValue("CONFERENCE")
public class Conference extends BaseEntity implements UserActivity,
EventSource {
…
}
```

Листинг 4. Фрагмент файла с реализацией бизнес-логики проекта ng-tracker

```
@Service
public class ConferenceService extends BaseService {
…
public List<Conference> findAll() {
        return conferenceRepository
            .findAll(new Sort(Sort.Direction.DESC, "beginDate"));
    }
…
}
```

Также подобные классы можно искать, например, путем поиска определенных ключевых слов в названиях классов и их путей или с помощью анализа методов класса. Затем из классов с реализацией бизнес-логики извлекаются публичные методы, в которых в качестве аргументов или возвращаемых значений используются найденные ранее сущности.

Пример фрагмента представления D^B изображен на рисунке 6.6.

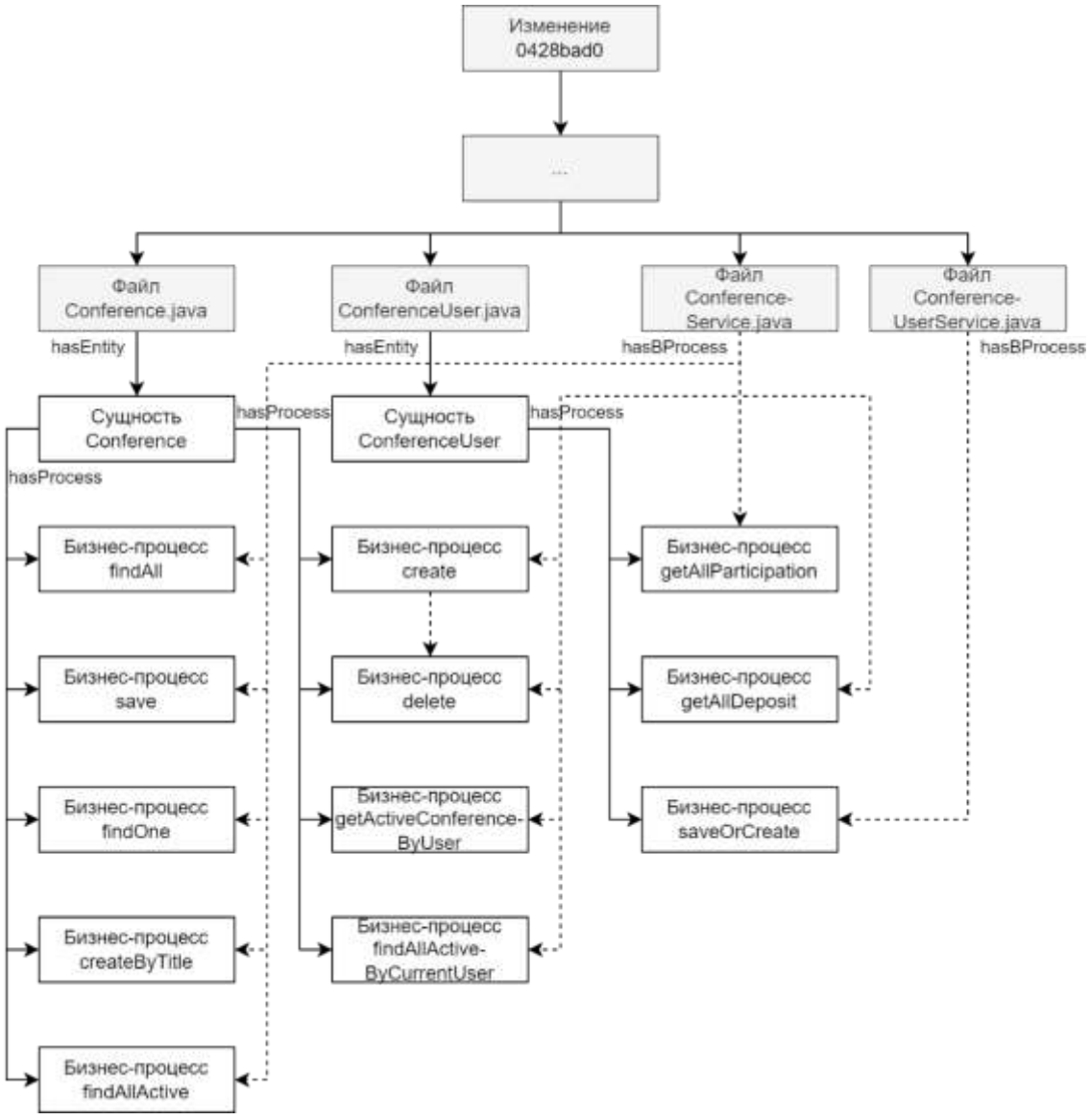

Рис.6.6. Пример фрагмента базы знаний для представления D

Шаг 5. Извлечение терминологического окружения проекта W^B

Терминологическое окружение проекта W^B содержит множество понятий и терминов, которые описывают эти понятия. Термины

представляют собой синонимы для обозначения сущностей и бизнес-процессов. Синонимы позволяют сопоставлять сущности, которые имеют одинаковую семантику и разное написание. Фрагмент терминологического окружения проекта W^B изображен на рисунке 6.7.

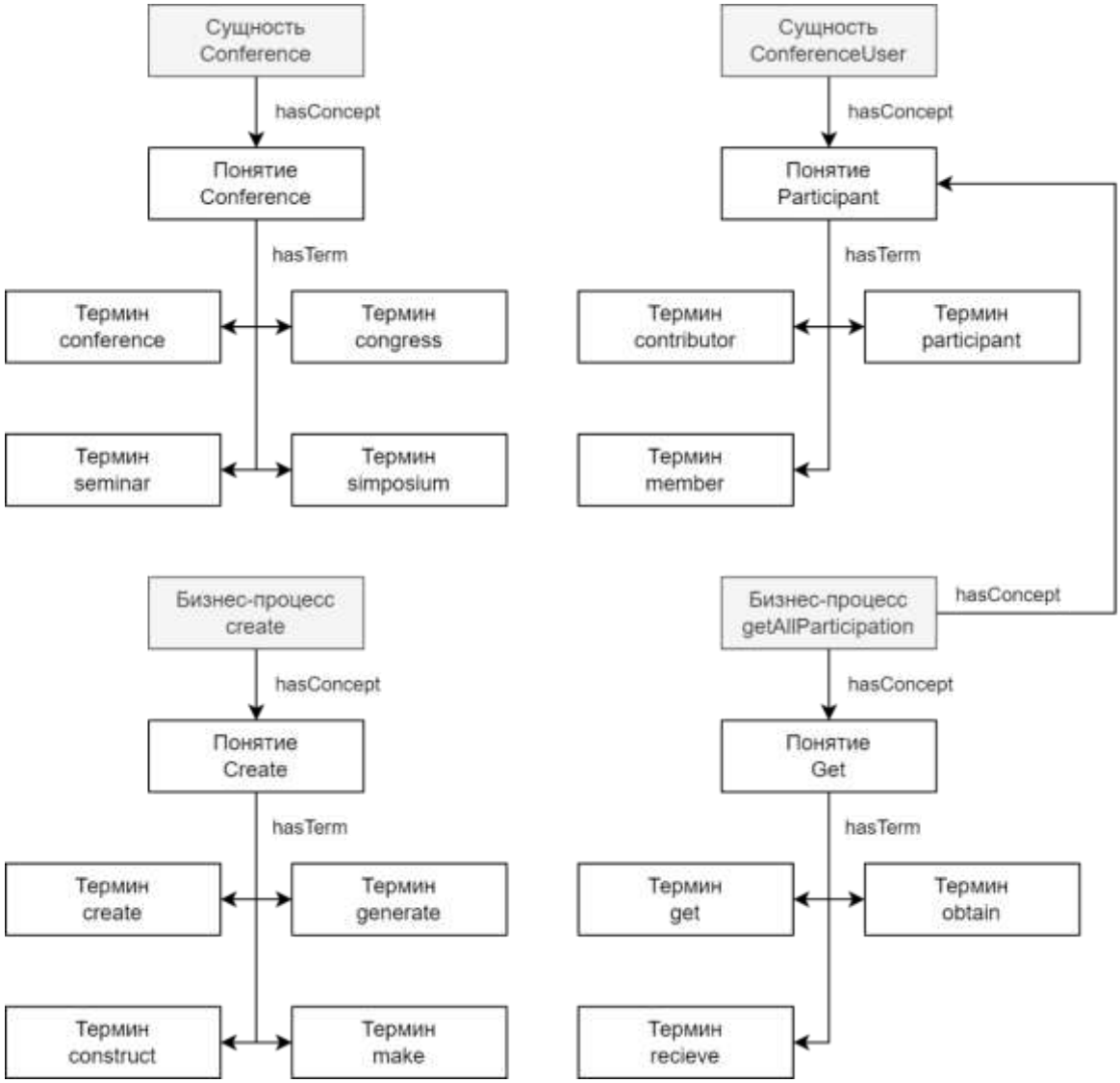

Рис. 6.7. Фрагмент терминологического окружения проекта W

Как видно из рисунка 6.7, одной сущности или бизнес-процессу представления D^B может соответствовать несколько понятий терминологического окружения. Терминологическое окружение используется в модуле информационного поиска для расширения

поискового запроса синонимами для повышения показателей качества поиска.

6.1.4 Метод диагностической аналитики для поддержки процесса разработки проекта

Использование методов инженерии знаний в процессе моделирования и анализа временных рядом позволяет учесть ограничения предметной области при выборе вида модели и ее параметров для повышения качества [98].

Источником данных предложенного метода диагностической аналитики для поддержки процесса разработки проекта является база знаний, модель которой представлена в предыдущем разделе. В рамках данной задачи центральной сущностью базы знаний является «Изменение», с которой связаны как сущности представления процесса разработки проекта L (рис. 6.1), так и сущности остальных представлений (рис. 6.8).

Рис. 6.8. ER-диаграмма связи сущности «Изменение» с сущностями базы знаний других представлений

Как видно из рисунка 6.8, на основе множества изменений проекта появляется возможность извлекать из базы знаний временные ряды

различных показателей с необходимой периодичностью и дискретностью с применением функции агрегации.

Представим функцию получения временных рядов из базы знаний с помощью выражения

$$F^{TS}: B_i \times Period \rightarrow TS,$$

в котором B_i – фрагмент базы знаний для *i*-го проекта;

$Period$ – настройки периода и дискретности формируемых временных рядов;

$TS = \{TS_1, TS_2, \dots, TS_i, \dots, TS_n\}, \forall |TS_i| = m$ – множество извлеченных из базы знаний временных рядов показателей в количестве n длиной m;

TS_i – временной ряд *i*-го показателя.

Далее на основе множества экспертных правил производится классификация полученных временных рядов на ряды, имеющие положительное и отрицательное влияние на качество процесса разработки, а именно

$$F^{Cl}: TS \rightarrow Dyn,$$

где F^{Cl}– функция для классификации временных рядов;

$Dyn = \{TS_1^{pos}, TS_2^{neg}, \dots, TS_i^{pos}, \dots, TS_n^{pos}\}$ – множество временных рядов, представляющих положительную или отрицательную динамику развития проекта.

Затем к полученным показателям динамики проекта применяется функция для извлечения знаний из временных рядов, которая имеет вид

$$F^{St}: Dyn \rightarrow St,$$

где F^{St} – функция извлечения знаний из временных рядов, включающая:

1. Оценку значения индикатора с помощью множества экспертных правил вида «если-то». Каждое правило определяет интервал значений, при попадании в который индикатору присваивается некоторое лингвистическое значение.

2. Корректировка значений состояний на основе взаимного влияния индикаторов друг на друга. Например, при увеличении количества разработчиков оценка количества изменений должна быть скорректирована в сторону уменьшения.

$St = \{\langle i, \{state_1^i, state_2^i, \dots, state_m^i\}\rangle\}$ – множество оценок состояния тенденций анализируемых временных рядов. Набор состояний представляется лингвистическими значениями, например: мало, средне, много.

Полученные оценки могут быть использованы для формирования рекомендаций для этапа планирования при начале новой итерации процесса разработки.

6.1.5 Диагностическая аналитика проектов

Рассмотрим пример работы подсистемы диагностической аналитики на примере следующих репозиториев:

– https://github.com/killjoy1221/tabbychat – плагин для добавления чата в игру Minecraft [100];

– https://gitlab.com/romanov73/ng-tracker – веб-сервис для планирования работы [101].

Данные репозитории выбраны, так как написаны на языке Java и имеют сопоставимые по значениям индикаторы.

В таблице 6.1 представлены временные ряды, извлеченные из данных проектов.

После применения метода диагностической аналитики для поддержки процесса разработки проекта, представленного в подразделе 2.2, были получены оценки состояния анализируемых репозиториев, представленные в таблице 6.2. Диапазоны значений правил для оценки размера команды были взяты из рекомендаций по организации разработки [102]. Диапазоны значений правил для оценки количества изменений были

аппроксимированы на временной интервал 1 месяц исходя из данных исследования [103]. Правила для оценки количества сущностей и бизнес-процессов были подобраны эмпирически.

Таблица 6.1. Временные ряды, извлеченные из анализируемых репозиториев

Проект	**Индикатор (количество)**	**Месяц**				
		1	**2**	**3**	**4**	**5**
tabbychat	Изменения	57	64	76	117	130
	Разработчики	3	4	6	6	6
	Сущности	7	7	7	7	7
	Бизнес-процессы	158	164	164	164	164
ng-tracker	Изменения	4	6	95	230	251
	Разработчики	1	1	8	8	8
	Сущности	5	5	18	18	19
	Бизнес-процессы	115	129	206	271	273

Таблица 6.2. Оценки состояния анализируемых репозиториев

Проект	**Индикатор (количество)**	**Месяц**				
		1	**2**	**3**	**4**	**5**
tabbychat	Изменения	средне	средне	мало	мало	мало
	Разработчики	средне	средне	мало	мало	мало
	Сущности	мало	мало	мало	мало	мало
	Бизнес-процессы	средне	средне	средне	средне	средне
ng-tracker	Изменения	мало	мало	мало	мало	мало
	Разработчики	мало	мало	средне	средне	средне
	Сущности	мало	мало	средне	средне	средне
	Бизнес-процессы	средне	средне	мало	мало	мало

На основе анализа полученных состояний приглашенный эксперт сформулировал следующую рекомендацию для проекта ng-tracker:

Рост количества сущностей при снижении количества реализованных бизнес-процессов показывает отсутствие прогресса в разработке новой функциональности проекта. Следует уделить внимание разработке бизнес-методов.

Эксперт сделал данный вывод, так как в проекте tabbychat наблюдалась стабильность показателя «количество сущностей», в то время как в проекте ng-tracker этот показатель рос. При этом в проекте ng-tracker, в отличие от проекта tabbychat, наблюдается снижение количества реализованных бизнес-методов, отражающих сложность реализованной бизнес-логики.

В дальнейшем планируется добавить в подсистему диагностической аналитики модуль поддержки принятия решений [104], который позволит автоматизировать процесс формирования рекомендаций за счет формализации экспертных знаний.

6.2 Извлечение сведений о динамике показателей объектов некоторой предметной области

Изучение динамики показателей объектов проводится с целью установления закономерностей поведения такого объекта при функционировании во внешней среде. Особенностью такого функционирования является то, что кроме знания внутреннего устройства и, часто, детерминированных алгоритмов поведения самого объекта на него оказывают влияние связанные объекты, а также может менять само поведение объекта из-за того, что в новых условиях начинают работать скрытые ветви алгоритма функционирования объекта. В таких условиях наличие детерминированной модели объекта исследования является недостаточным. Появляется необходимость в создании интеллектуальных моделей, обладающих свойствами обучаемости, отражения нелинейных зависимостей, отражения тенденций изменения показателей.

Динамика изменения показателя исследуемого объекта в общем случае представляется как числовой вектор, компонентами которого являются значения (уровни) показателя, измеренные в определенные интервалы времени: $< v_1, \ldots, v_n >$, где v_i – значение показателя, n – число измерений. Для изучения динамики применяются различные методы. Такие методы различаются по своим возможностям, сгруппируем их для пояснения.

1. Статистические методы рассматривают вектор значений показателя как выборку, проводя исследования которой можно сделать заключения о различных параметрах, таких как математическое ожидание, стационарность, трендовая, сезонная и периодическая компонента и т. д. Выделяются следующие методы: метод выделения тренда (временного сглаживания), регрессионный, автокорреляционный, адаптивный (скользящих средних).

2. Нейросетевые методы. Аппроксимация математической функции при помощи обучения нейросетевой модели на определенном наборе данных позволяет получить преимущество по сравнению со статистическими методами за счет возможности дообучения, переобучения модели на новых данных. Такая гибкость позволяет строить системы поддержки принятия решений в различных прикладных областях с меньшими затратами как на разработку программного обеспечения (снижается необходимость постоянного изменения алгоритмов) так и на проведение дополнительных аналитических работ, потому что остается только типовой цикл «обучение модели» - «использование модели».

3. Нечеткие модели. Моделирование динамики показателей при помощи нечеткого подхода заключается либо в приведении исходных числовых данных в нечеткую форму или использование изначально нечетких сведений. Этот факт является первым преимуществом с точки

зрения расширения возможностей моделирования неподготовленных данных, в том числе и неструктурированных. Другим преимуществом нечеткой аппроксимации является то, что полученная нечеткая модель может быть представлена как база знаний, обладающая при это возможностью экспертной корректировки и высокой степенью интерпретируемости результатов моделирования лицом, принимающим решения. Это достигается за счет возможности представления уровней показателей исследуемого объекта в лингвистической форме и использования таких меток при составлении базы знаний, например в форме правил «если-то».

4. Гибридный подход. В нем используется сочетание моделей или их отдельных элементов для решения задач исследования динамики. Например, может быть выбран один базисный подход, а для решения задачи оптимизации параметров или предобработки выбирается метод, относящийся к другому классу. Множество таких подходов в настоящее время активно развиваются благодаря развитию методов искусственного интеллекта. В результате проведения таких исследований получаются модели, преодолевающие недостатки известных или классических подходов.

Гибридные решения, преодолевающие ограничения существующих отдельных моделей, позволяют получать как новые результаты. Однако разработка методов и алгоритмов, основанных на искусственном интеллекте, требует большой экспериментальной части для достижения требуемой конфигурации и точности моделей. Еще одна особенность вытекает из преимуществ использования методов искусственного интеллекта, а именно аналитике данных. Если качество самих данных низкое, наблюдаются неполнота, зашумленность, пропуски, то и результаты работы интеллектуальных моделей будут некачественными.

Однако стоит выделить виды систем управления. К первому классу относятся системы автоматического управления, в которых роль лица, принимающего решения сведена к минимуму. Чаще всего в таких системах реализуются ПИД-регуляторы на основе различных типов моделей. Другим классом систем является класс систем поддержки принятия решений, в которых управление нераздельно связаны с вмешательством оператора в цепь управления. При этом возможен иерархический подход к декомпозиции задач анализа данных и принятия решений.

Направлениями развития методов ИИ являются расширение совокупности анализируемых данных, усложнение применяемых моделей, повышение интерпретируемости результатов и объяснение способа получения выводов.

До текущего момента мы рассматривали динамику только как вектор значений некоторого параметра исследуемого объекта. Необходимо ввести несколько значимых условий для моделирования динамики, которые помогут повысить точность моделирования и расширят возможности применения методов моделирования. Это позволит применять различные гибридные подходы для более гибкого решения задач.

1. Причинно-следственные связи между последовательными значениями показателя. Данное условие означает, что выборка сведений должна быть рассмотрена и промоделирована в упорядоченном режиме. Чаще всего такое ограничение приводит к необходимости представления показателя исследуемого объекта в форме временного ряда: $ts = \{ts_i\}, ts_i = (t_i, v_i), i = [1, \ldots, n], i \in N$. Данное условие приводит к необходимости учета второго условия – временного интервала между последовательными измерениями.

2. Временной интервал между измерениями значения показателя важен с той точки зрения, что влияет на тип получаемого временного ряда

(равномерного или с неравномерными временными интервалами), а также на то, что время становится параметром, который необходимо учитывать в модели. Особенно это важно для моделей, в которых выделяются отдельные компоненты временного ряда (тренд, сезонная, периодическая составляющая), рис. 6.9.

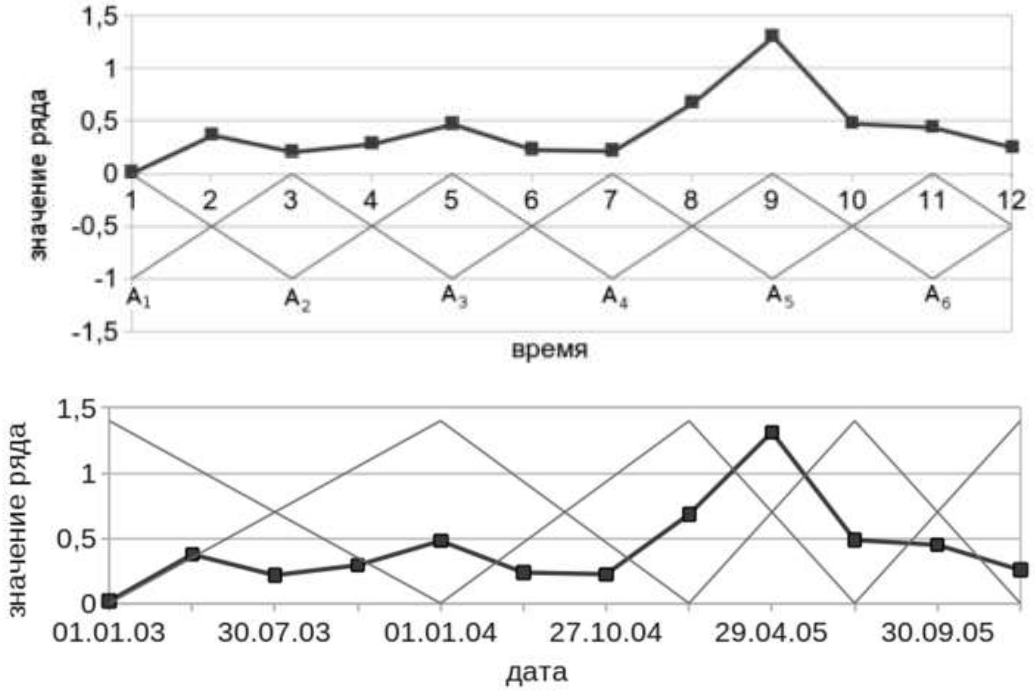

Рис. 6.9. Пример временного ряда с выделением областей для кусочно-линейного тренда

Извлечение сведений о динамике показателя *indicator* в общем виде можно представить как функцию генерирования числовых последовательностей с отметками времени: $e(indicator, t_i) = (t_i, v_i)$, т. е. по существу это выражение – обоснование использования временных рядов для отображения динамики изменения значений показателя. Основными проблемами извлечения значений показателя в динамике являются:

– Дискретность. Многие значения являются непрерывными, и поэтому при формировании вектора значений с привязкой ко времени

информация будет преобразована. Важно при этом сохранить необходимую достаточную для анализа исследуемого объекта частоту дискретизации.

– Сведения о значении показателя без отметки времени. Такие сведения не сохраняются и не предоставляются самим объектом. В этом случае отметка времени может быть задана самой функцией извлечения значения показателя.

Обобщим задачу извлечения сведений о динамике. Для этого формализуем функцию извлечения числового представления сведений об анализируемом объекте *O*. Анализируемый объект рассматривается как совокупность атрибутов *attr:* $O = \{attr_i\}$. Все множество атрибутов делится на следующие виды:

1. Числовые, имеющие отметку времени c^1.
2. Числовые без отметки времени c^2.
3. Нечисловые с отметкой времени c^3.
4. Нечисловые без отметки времени c^4.

Тогда общая функция извлечения сведений о динамике пример вид: $e(O) = \{e(attr^{c^1}), e(attr^{c^2}), e(attr^{c^3})\, e(attr^{c^4})\}$. Для атрибутов без отметки времени необходимо сформировать функцию задания временной метки. Функция извлечения сведений о динамике порождает многомерный временной ряд. Этот подход однозначно свидетельствует о том, что анализируемый объект представляется совокупностью числовых показателей и для того, что не терять смысловую информацию о значениях атрибутов необходимо применять подход с контекстным моделированием.

Рассмотрим пример системы, для которой характерно наличие сведений о значении показателей, как с отметкой времени, так и без нее. Предметной областью является разработка программного обеспечения, а именно рассматриваются процессы управления разработкой с

использованием сведений из веб-сервисов для хостинга IT-проектов. В настоящие время такие сервисы (github, gitlab и др.) применяются для размещения репозиториев программных проектов. Эти сервисы позволяют анализировать состояние проекта на основе изучения различных артефактов разработки: исходного кода, списка задач, элементов самого репозитория, отражающих активность разработчиков. Извлечение и анализ динамики значений показателей таких репозиториев и сервисов позволяют выявлять значимые для управления проекта тенденции. В соответствии с вышеописанной процедурой извлечения сведений о динамике можно выделить два типа извлекаемых временных рядов: $TS = \{TS^r, TS^s\}$, где TS – все множество временных рядов, TS^r – множество временных рядов, показателей репозитория программного кода, TS^s – множество временных рядов показателей веб-сервиса программного проекта.

$$TS = \{TS^r, TS^s\},$$

где TS – все множество временных рядов, TS^r – множество временных рядов, показателей репозитория программного кода, TS^s- множество временных рядов показателей веб-сервиса программного проекта.

Для каждой группы временных рядов необходимо определить порядок формирования точек временного ряда. Из условия задачи ясно, что точки временного ряда должны содержать отметку времени. Следовательно, для группы временных рядов TS^r это условие выполняется из-за того, что все фиксации в репозитории имеют отметку времени. Для групп временных рядов TS^s способ формирования необходимо определить по-другому. Особенностью это процесса является то, что в самом веб-сервисе записи могут создаваться и удаляться без сохранения истории. А поскольку анализ ведется на основе количественных значений объектов, то необходимо задать временную отметку. Был выбран подход, заключающийся в извлечении точек временных рядов показателей по

расписанию. В итоге состав множеств временных рядов показателей имеет вид

$$TS^r = \{TS^{commit}, TS^{commit\_author}, TS^{dependency}, TS^{entity},$$
$$TS^{file}, TS^{class}, TS^{interface}, TS^{process}\},$$

где TS^{commit} – временной ряд числа коммитов, TS^{commit_author} – временной ряд количества коммитов авторов, $TS^{dependency}$ – временной ряд количества зависимостей проекта, TS^{entity}- временной ряд количества сущностных классов проекта, TS^{file} – временной ряд количества файлов проекта, TS^{class} – временной ряд количества классов проекта, $TS^{interface}$ – временной ряда количества интерфейсов проекта, $TS^{process}$ – временной ряд количества бизнес-процессов, описанных в классах проекта.

$$TS^s = \{TS^{branch}, TS^{issue}, TS^{star}\},$$

где TS^{branch} – временной ряд количества ветвей кода проекта, TS^{issue} – временной ряд количества открытых задач по проекту, TS^{star} – временной ряд количества звезд проекта.

Функция извлечения i-й точки временного ряда для данного примера имеет вид: $e(TS^j, t_i)$.

Передача сведений об анализируемом показателе важна для использования контекста работы объекта исследования.

6.3 Разработка подхода для использования контекста при анализе данных динамики показателей объектов

Использование сведений контекста при моделировании временных рядов используется для повышения качества моделирования (выбор адекватной модели), повышения гибкости и расширения возможностей используемых подходов.

Если будем рассматривать систему управления для некоторой проблемной области, то можно выделить вектор входных данных, который

известен для системы управления, $X = \{x_1, \ldots, x_n\}$ подается на вход объекта управления, на который также воздействуют входы $W = \{w_1, \ldots, w_r\}$, значения которых невозможно учесть в системе управления. Выходом объекта управления является вектор $V = \{v_1, \ldots, v_n\}$ Вектор обратной связи от системы управления $U = \{u_1, \ldots, u_k\}$ тоже поступает на вход объекта управления.

Анализ временных рядов в теоретическом и практическом аспектах является важнейшей частью изучения сложных объектов. Чаще всего можно встретить исследования, основанные на анализе единого независимого временного ряда. В сложных организационно-технических системах часто наблюдаются взаимосвязанные изменения, представляющие интерес для анализа. Кроме процесса идентификации и параметризации модели временного ряда часто используется корреляционный анализ. Также рассматривается проблема создания меры для установления корреляции между временными рядами. Отмечено, что анализ интенсивности и направления тенденций является важным инструментом для выявления зависимостей.

В нашем исследовании мы используем контекстные сведения об объекте анализа в форме закономерностей, ограничений, связей. В отличие от рассмотренных материалов вместо поиска зависимостей через анализ временных рядов мы будем использовать априорную информацию, заданную через контекст описания объектов организационно-технической системы.

Для использования предлагаемого подхода нужно изменить традиционную схему управления. В нее добавляется новый блок, отвечающий за моделирование динамических параметров и использование контекстных данных. Добавляемый блок не является частью системы управления, потому что в равной степени предоставляет информацию для

лица, принимающего решения, и для системы автоматического контроля, если требуется. Данное решение не рассматривается как полноценная система поддержки принятия решения, а является средством повышения гибкости при создании систем управления. В дальнейших исследованиях планируется продолжить совершенствование контекстного анализа динамики показателей.

Отличием в схеме будет являться:

– наличие внешней информации о контексте управления объектом *D*;

– накопление и использование динамики выходных показателей объекта управления;

– использование рекомендаций по управлению лицом, принимающим решения и в контуре управления.

Примером проблемной области, для которой можно применить предлагаемый подход, будет являться разработка программного обеспечения на основе гибких методологий. В указанной области своевременная реакция на возникающие события является особенно важной, поскольку позволяет реализовывать такие аспекты как инспекция и адаптация в условиях экстремальных значений параметров разработки.

Основой для моделирования процесса управления станут наборы данных, отражающие динамику формирования новой функциональности при разработке программных проектов, а также организационные мероприятия.

Проектирование архитектуры программной системы для анализа динамики показателей предполагает учет всех вышеописанных особенностей гибридных моделей, использующих как существующие программные компоненты для хранения, анализа данных и получения

информации для лица, принимающего решения. Основными программными компонентами при этом являются:

1. Модуль преобразования записей в базе данных во временные ряды.
2. Модуль формирования точек временных рядов для динамических (создаваемых и удаляемых) объектах, которые не обладают историческими сведениями.
3. Сервис моделирования и прогнозирования временных рядов.
4. Модуль работы с базой знаний.
5. Модуль для контекстного анализа временных рядов.

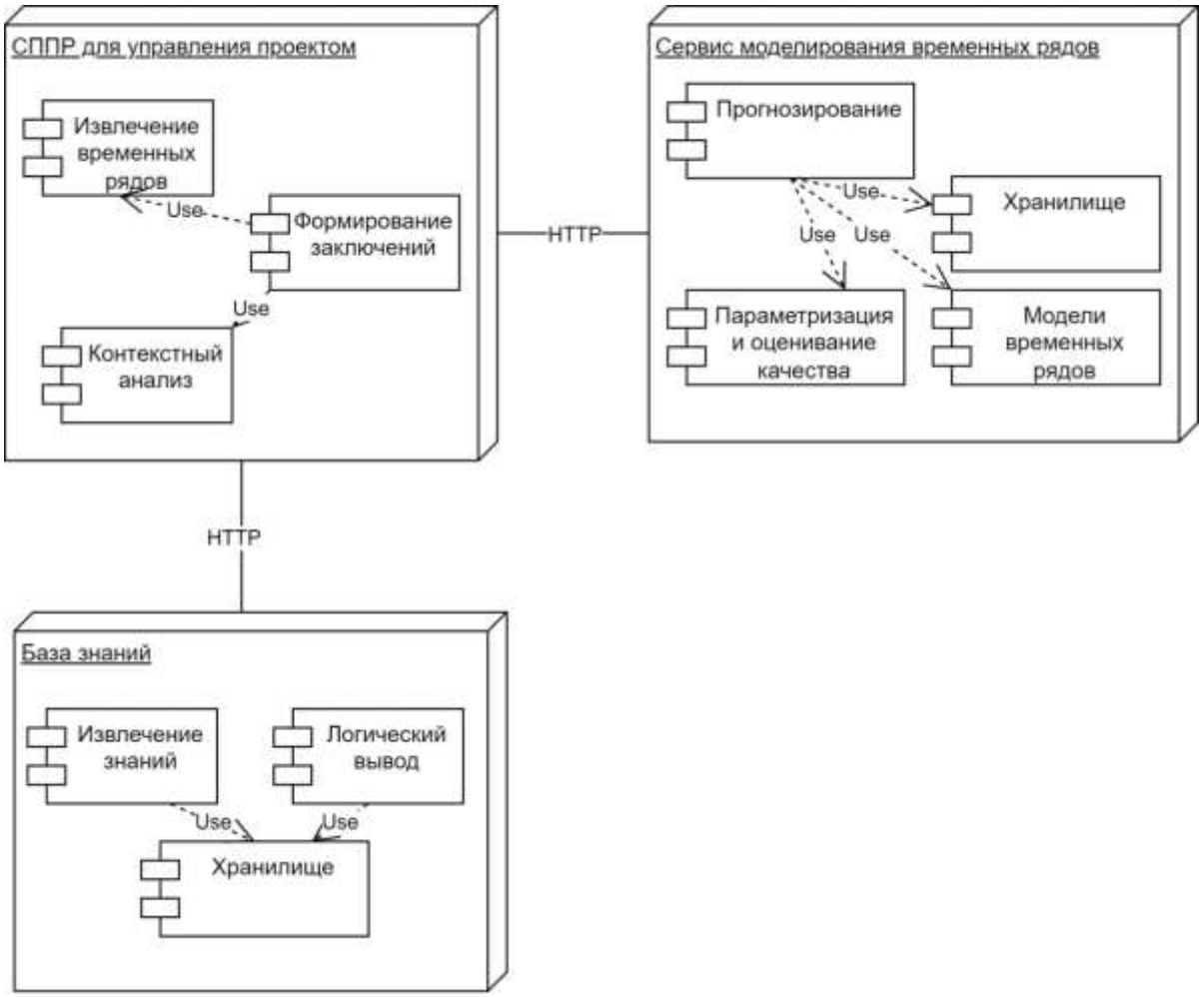

Рис. 6.10. Диаграмма компонент интеллектуальной системы

На представленной диаграмме (рис. 6.10) представлены все вышеперечисленные модули. Особенностью является то, что указанная

система включает в семя несколько отдельных сервисов, решающих задачи в рамках отдельных проблемных областей, а именно: моделирование и прогнозирование временных рядов, работа с базой знаний. Поскольку такие сервисы не позволяют работать с моделированием контекста динамики показателей, то необходимо иметь объединяющую систему, а именно саму систему поддержки принятия решений, отвечающих за решение задач передачи данных между сервисами, подготовки исходных данных, работу алгоритма контекстного моделирования временных рядов.

В свою очередь каждый сервис, базы знаний и моделирования временных рядов, предоставляет программных интерфейс для выполнения задач выбора необходимых моделей, задания требуемых параметров, формирования результата моделирования и прогнозирования временных рядов, выполнения логического вывода.

В результате спроектированная система позволяет решать задачи контекстного моделирования динамики показателей процессов при управлении некоторой системой. В то же время предложенный подход позволяет наращивать функциональность системы поддержки принятия решений, адаптируя ее под новые задачи, при этом остальные сервисы могут быть использованы без доработок.

6.4 Разработка интеллектуальной программной системы для анализа динамики процессов разработки программного обеспечения

Разработанная программная система позволяет просмотреть список проиндексированных репозиториев (рисунок 6.11)

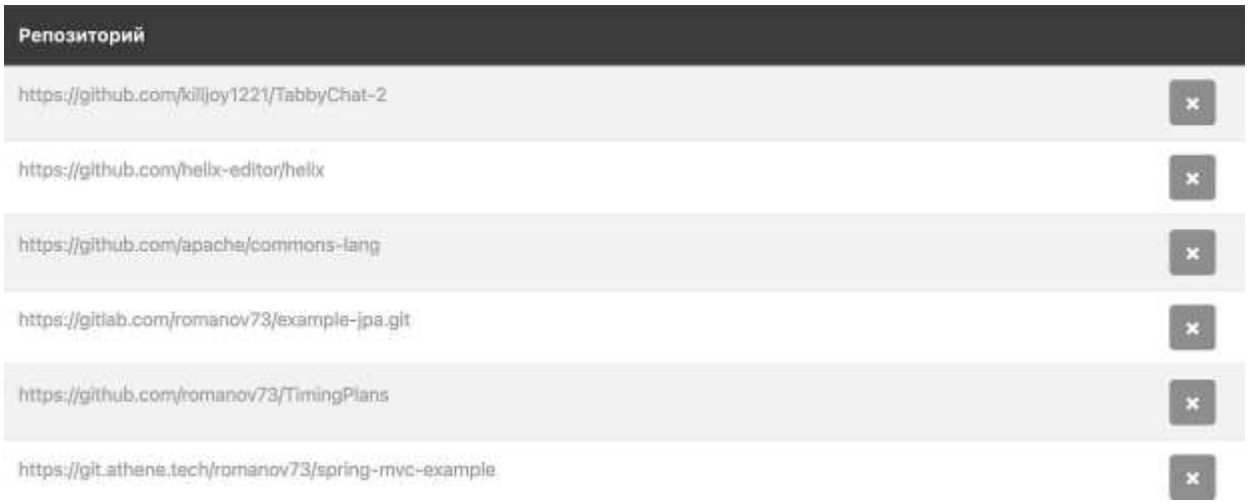

Рис. 6.11. Пример отображения списка проанализированных репозиториев программных проектов

Целью анализа в данной системе является отражение состояния программного проекта на основе анализа похожих тенденций показателей других проектов. Как было описано выше, такой анализ производится на основе анализа временных рядов. Пример временного ряда для одного из проанализированных репозиториев приведен на рисунке 6.12.

Рис. 6.12. Пример временного ряда «количество коммитов»

Кроме исходных данных репозитория программного проекта система позволяет производить обучение базы знаний путем описания экспертом отдельных характерных тенденций в многомерных временных рядах, рисунок 6.13. Для того чтобы эксперт смог наполнить базу знаний, необходимо предусмотреть возможности снижения трудоемкости заполнения, а именно сокращение числа возможных ситуаций развития репозитория. Для этого применяется предварительное сглаживание временных рядов и их сжатие, выделяются временные интервалы временных рядов и ищутся совпадающие в других временных рядах, рис. 6.14. В результате остаются только коррелирующие между собой участки с ярко выраженными тенденциями, которые сможет охарактеризовать эксперт. Для этого ему нужно заполнить форму, как показано на примере для одного из репозиториев, рис. 6.15.

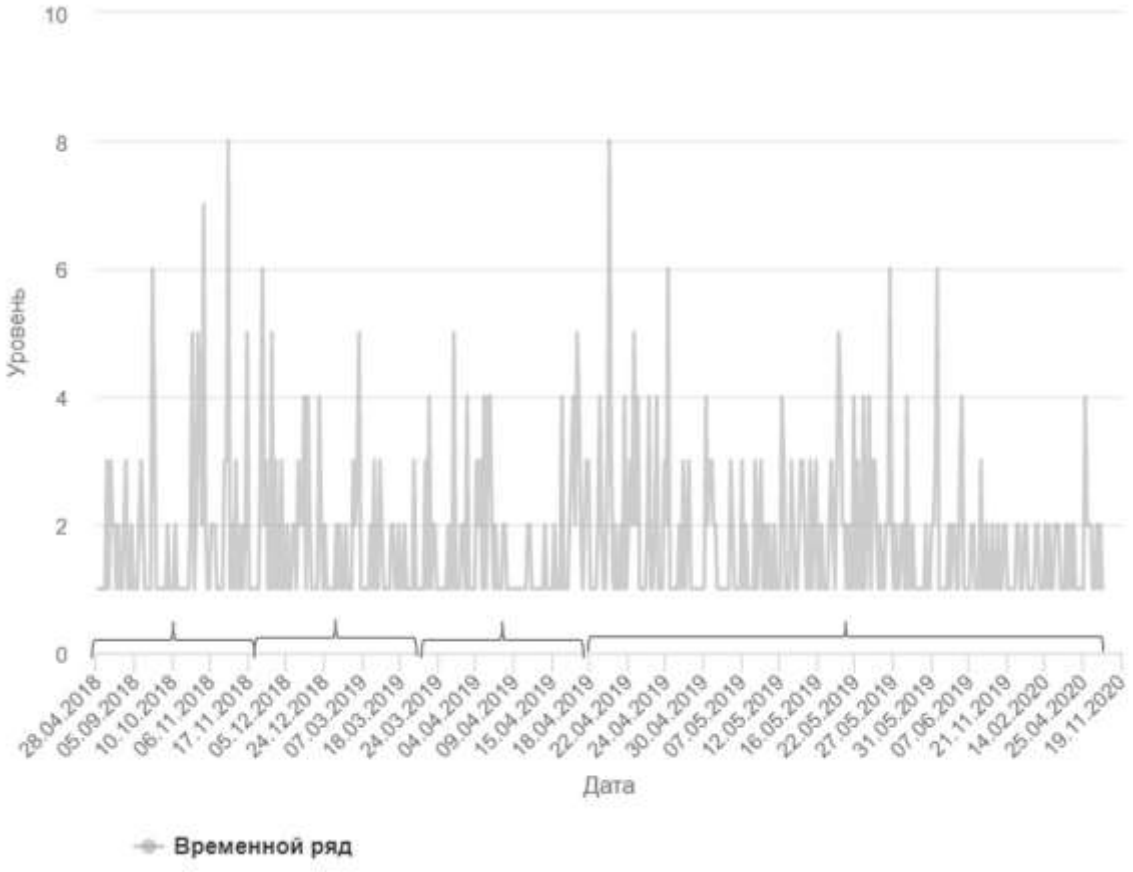

Рис. 6.14. Временной ряд коммитов проекта с выделенными временными интервалами

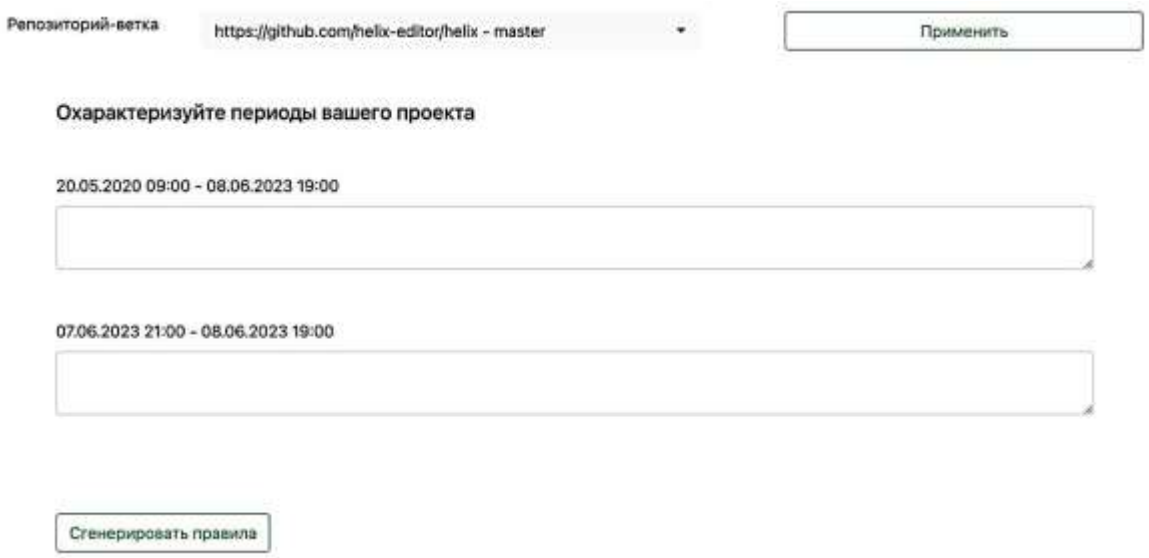

Рис. 6.15. Пример экранной формы для заполнения экспертом описания характерных этапов проекта

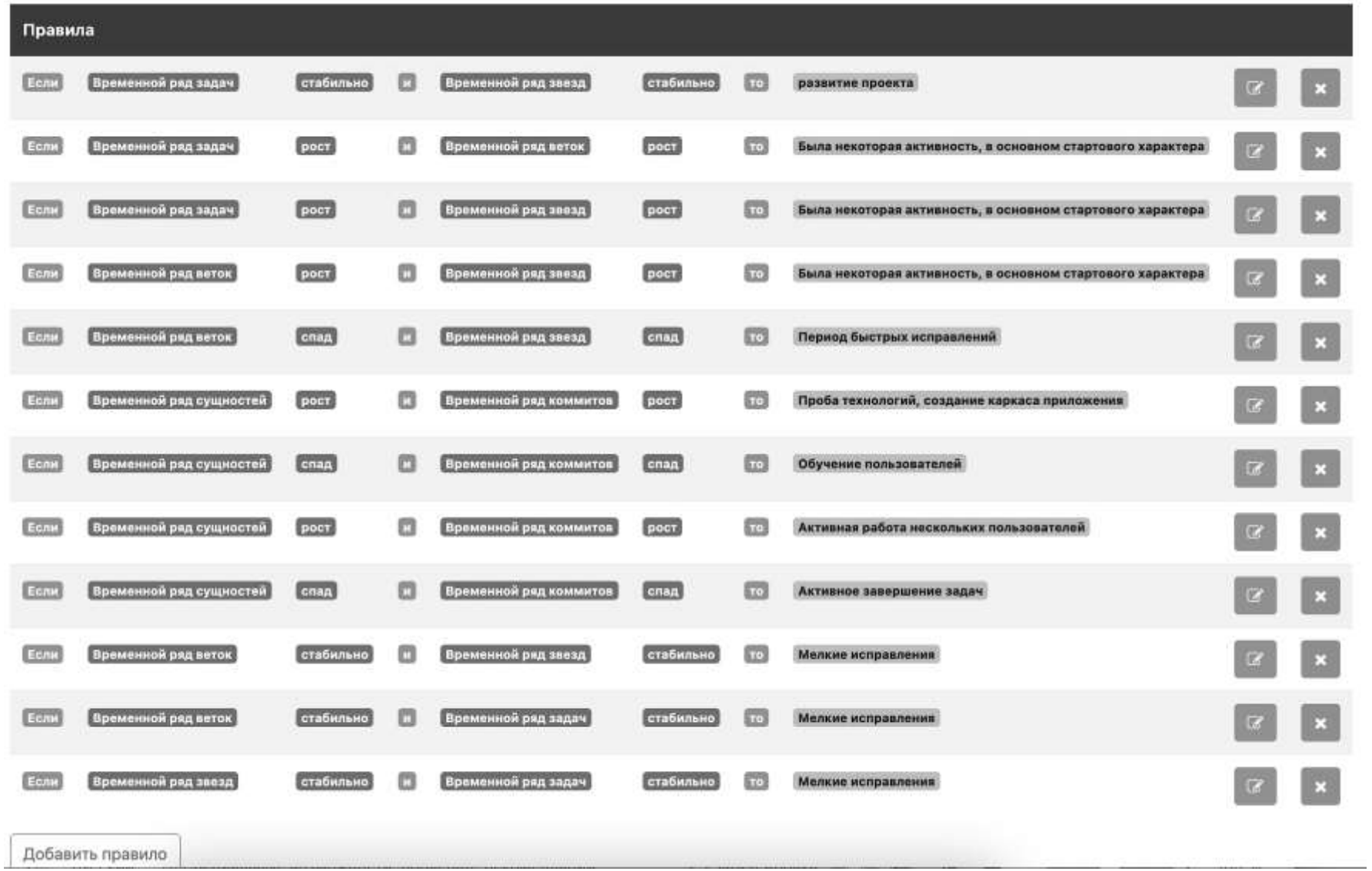

Рис. 6.13 Пример экранной формы с отображением списка правил оценивания репозитория

После заполнения экспертом характеристик временных интервалов можно сгенерировать нечеткие правила. Для известного эксперту проекта https://git.athene.tech/romanov73/ng-tracker были созданы следующие правила, таблица 6.1.

Таблица 6.1. Примеры сгенерированных нечетких правил

	Показатель	**тенденция**		**Показатель**	**тенденция**		**Заключение**
Если	Временной ряд задач	рост	и	Временной ряд веток	рост	то	Была некоторая активность, в основном стартового характера
Если	Временной ряд задач	рост	и	Временной ряд звезд	рост	то	Была некоторая активность, в основном стартового характера
Если	Временной ряд веток	рост	и	Временной ряд звезд	рост	то	Была некоторая активность, в основном стартового характера
Если	Временной ряд веток	спад	и	Временной ряд звезд	спад	то	Период быстрых исправлений
Если	Временной ряд сущностей	рост	и	Временной ряд коммитов	рост	то	Проба технологий, создание каркаса приложения
Если	Временной ряд сущностей	спад	и	Временной ряд коммитов	спад	то	Обучение пользователей
Если	Временной ряд сущностей	рост	и	Временной ряд коммитов	рост	то	Активная работа нескольких пользователей
Если	Временной ряд сущностей	спад	и	Временной ряд коммитов	спад	то	Активное завершение задач
Если	Временной ряд веток	стабильно	и	Временной ряд звезд	стабильно	то	Мелкие исправления
Если	Временной ряд веток	стабильно	и	Временной ряд задач	стабильно	то	Мелкие исправления

Для оценки адекватности предложенного способа генерации правил был проведен анализ этого же проекта на соответствующих правилам временных интервалах. Было получено 100% совпадения заключений, предложенных системой, с заключениями эксперта на этапе разметки.

7 ВЫЧИСЛИТЕЛЬНЫЕ ЭКСПЕРИМЕНТЫ ПО ФОРМИРОВАНИЮ КОНТЕКСТА АНАЛИЗА

7.1 Применение генетического алгоритма в филогенетическом анализе нуклеотидных последовательностей ДНК

В решении фундаментальных и практических задач анализа биологических объектов филогенетический анализ позволяет реконструировать исторический ход эволюции, предсказать дальнейшие вероятности отдельных мутаций и функцию получившихся нуклеотидных последовательностей (аннотация генов), осуществить поиск генов в геноме и провести тест на гомологию (определить уровень дивергенции видов).

Предметом исследования является филогенетическое дерево – графическое изображение хода исторического развития групп организмов с общим происхождением. Объект исследования – генетический материал водных растений. Исходным данными для проведения филогенетического анализа выступают нуклеотидные последовательности участков анализируемых генов.

Нуклеотидная последовательность – это порядок следования нуклеотидных оснований (A (аденин), T (тимин), C (цитозин) и G (гуанин)) в фрагменте ДНК.

Генетическая изменчивость биологических объектов в итоге генерируется мутациями. Такие генные мутации, как инсерции (вставка одного или группы нуклеотидов), делеции (потеря одного или группы нуклеотидов) и замена оснований (точечные однонуклеотидные замены (SNP)) являются самыми распространенными и оказывают влияние на архитектуру генов и в конечном счете на видообразование [105].

Систематизация исходных нуклеотидных последовательностей, как и большинство молекулярно-генетических исследований растений, была осуществлена за счет использования небольшого количества биологического материала с помощью полимеразной цепной реакции

ПЦР [106]. Для анализа популяционной изменчивости небольших фрагментов генома и уточнения типа мутаций отдельных генов растений использовалось секвенирование по методу Сэнгера [107].

Для описания дивергенции видов у гибридных таксонов традиционные модели эволюции, предполагающие постепенное накопление мутаций, сопровождаемое дихотомическим ветвлением филогенетических ветвей, подходят плохо, поэтому используют комбинированные алгоритмы [108].

Необходимо провести филогенетический анализ нуклеотидных последовательностей ДНК выбранных образцов растений с использованием разных методов построения филогенетического дерева.

Применить новый метод для построения филогенетического дерева – генетический алгоритм.

В ходе построения и анализа филогенетического исследования требуется выполнить определенный набор процедур. В данном исследовании выделены следующие этапы:

1. Выбор последовательности генов для анализа.

2. Выравнивание нуклеотидных последовательностей.

3. Подбор модели эволюции.

4. Построение филогенетического дерева и оценка полученного результата.

7.2 Выбор последовательности генов для анализа

В рамках первого этапа выбран участок анализируемых генов – маркер trnL-F, который кодирует транспортную РНК лейцина пластомного генома [109]. В ряде работ (Fennell et al. 1998; Bakker et al. 1999; Eldenäs and Linder 2000) подтверждается, что маркер trnL-F полезен на низких уровнях иерархии, особенно между родами и видами. Данный маркер состоит из интрона хлоропластного гена trnL и межгенного спейсера trnL-

trnF. Некодирующий участок (интрон) содержит больше инсерций и делеций, чем замен, что может затруднить выравнивание последовательностей и определение гомологичных оснований, но при этом содержит филогенетически значимую информацию.

Для проведения эксперимента отобраны пять образцов растений с территории России (обозначенные как H1.1, H1.2, HR2, HR3, HR4), что позволит на малой группе оценить результаты работы разных методов построения филогенетического дерева. Длина исходных нуклеотидных последовательностей оказалась в рамках от 1113 до 1178 нуклеотидов с большим содержанием аденина (в рамках 37%) и тимина (от 29 до 30%), меньшим - цитозина (в рамках 17%) и гуанина (в рамках 16%). Разница в нуклеотидных последовательностях составила до 4%. Большинство генных мутации оказалось участками с делециями (в два раза больше, чем участков со вставками, и в три раза больше одиночных мутаций). Наличие генных мутаций в анализируемых последовательностях гена позволяет проводить дальнейшие исследования.

7.3 Выравнивание нуклеотидных последовательностей

В рамках второго этапа проводится выравнивание полученных на первом этапе последовательностей. Под выравниванием последовательностей понимают такое размещение последовательностей друг относительно друга, при котором визуализируются сходные участки, при этом мало уделяется внимания участкам с изменениями, хотя именно генетическая изменчивость является предпосылкой эволюционных изменений.

В исследовании выбран метод множественного выравнивания MAFFT, который расширяет возможности парного выравнивания, позволяя включать в анализ одновременно пять анализируемых последовательностей. Алгоритм MAFFT (Multiple Alignment using Fast

Fourier Transform) [110] – основан на прогрессивном выравнивании, в котором последовательности были сгруппированы с помощью быстрого преобразования Фурье, что позволяет быстро находить участки гомологии. Данный алгоритм предлагает упрощенную систему подсчета параметров, которая хорошо работает для увеличения скорости и повышения точности выравнивания, даже для последовательностей, имеющих большие инсерции или делеции. После ручной корректировки матрица расстояний множественного выравнивания методом простого сходства показала проценты сходства до 98%.

7.4 Подбор модели эволюции

Третьим этапом в проводимом исследовании является подбор модели эволюции (замещения ДНК). Модели эволюции - это набор предположений о процессе нуклеотидных замен. Они описывают различные вероятности перехода от одного нуклеотида к другому вдоль филогенетического дерева, позволяя выбирать между различными филогенетическими гипотезами для объяснения имеющихся данных.

Основные параметрические модели эволюции нуклеотидных последовательностей работают согласно матрице нуклеотидных замен [111]. Частота замен в параметрической модели определяется как функция от некоторых параметров, оцененных для каждой группы анализируемых данных. Выделяют два класса параметров: параметры вероятности замен и параметры скорости мутаций. Рассматривается разные гипотезы частоты транзиций (замены одного пуринового основания на другое (аденина на гуанин, или наоборот) или одного пиримидинового основания на другое (тимина на цитозин, или наоборот)) и трансверсий (замены пуринового основания (аденин, гуанин) на пиримидиновое (тимин, цитозин) и наоборот).

Подбор модели эволюции является отдельной научно-исследовательской задачей. Наиболее обширные обзоры выбора моделей в филогенетике предоставлены в статьях Sullivan and Joyce [112] и Johnson and Omland [113].

Рассмотрим модели эволюции, используемые на следующем шаге проводимого исследования:

1. По модели эволюции модель Джукса и Кантора (JC) [114] – предполагается, что все четыре нуклеотида (А, Ц, Т, Г) присутствуют в ДНК в одинаковых пропорциях, а вероятность замены одного нуклеотида на другой одинакова для любой пары нуклеотидов.

2. По модели Кимура (K80) [115] – применяются одинаковые частоты встречаемости нуклеотидов и и учитывает разницу между переходами и трансверсиями с одним параметром.

3. Самая часто применяемая модель эволюции GTR (General Time Reversible Model) [116] - учитывает различные частоты нуклеотидов (4 параметра) и все типы замен между ними (6 параметров).

7.5 Построение филогенетического дерева и оценка полученного результата

В рамках последнего этапа выбирается метод построения филогенетического дерева, осуществляется настройка параметров и формируется филогенетическое дерево сходства и эволюционного развития биологических объектов на основе анализируемых участков генов.

Результаты филогенетической реконструкции напрямую зависят от методов и алгоритмов построения, которые показывают разные результаты для различных выборок данных и имеют ряд сложностей при применении.

Чаще всего выделяют две группы методов построения филогенетических деревьев:

– дистанционно-матричные методы (UPGMA, метод минимальной эволюции, метод ближайших соседей), которые использую оценки генетических дистанций, т. е. предположения о том, насколько велико генетическое сходство между двумя анализируемыми сущностями и, таким образом, демонстрируют конечную степень дивергенции таксонов, но полученное дерево не отражает исторический процесс эволюции;

– дискретные (символьные) методы (метод максимальной экономии, метод максимального правдоподобия, метод Байеса) строятся на основании анализа дискретных признаков – апоморфий (характерных признаков только оформившегося таксона) и основное предположение заключается в том, что члены группы имеют общую эволюционную историю.

Следовательно, результаты реконструкции методами из разных групп могут дать различную иерархию дерева.

В рамках данного исследования проведены эксперименты по построению дерева следующими стандартными методами:

– метод ближайших соседей;

– метод максимального правдоподобия;

– метод Байеса;

Предложено провести эксперимент с применением нового метода для построения филогенетического дерева - генетический алгоритм.

Для статистического подтверждения корректности создаваемых разными методами филогенетических деревьев используются методы на основе алгоритма бутстрэп (bootstrap) [117]. Метод заключается в создании набора выравниваний, которые состоят из нуклеотидов, взятых из случайных позиций исходного выравнивания. Для достоверности считается необходимым построение от 100 до 1000 таких наборов. По

полученным выравниваниям строятся филогенетические деревья таким же методом, как анализируемое дерево. Количество одинаковых узлов, обнаруженных при сравнении полученных деревьев с анализируемым, показывает бутстрэп-поддержку расхождения двух ветвей. Достоверным значением поддержки считается 90% совпадающих топологий.

7.6 Построение филогенетического дерева методом ближайших соседей

Алгоритм метода ближайших соседей (Neighbor-joining) [118] – дистанционный, начинает построение древа с полностью неразрешенной топологии, используя входящую матрицу дистанций. Два ближайших различных таксона соединяются в новый узел, который после присоединяется к центральному узлу. Затем подсчитывается новая матрица дистанций по полученной топологии и этот процесс повторяется до полного построения древа. Данный метод можно рассматривать как алгоритм для оптимизации дерева в соответствии с критерием «сбалансированной минимальной эволюции» [119], хорошо применим с методом бутстрэпа [120] и часто выдает дерево с правильной топологией [121]. Достоинствами данного метода являются простота и скорость, что снижает нагрузку на вычислительные ресурсы. Недостатки заключаются в невысокой точности построения в сравнении с более сложными методами [122].

В рамках исследования было построено филогенетическое дерево для пяти нуклеотидных последовательностей участка гена trnL-F образцов растения одного вида с территории России. Использовалось программное обеспечение – метод Neighbor-joining в пакете PHYLIP, для подсчета матрицы дистанций использовалась модель Кимура и индекс бутстреп-поддержки при 10 000 репликаций.

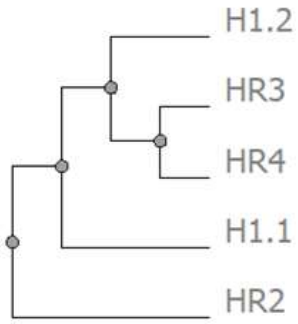

Рис. 7.1. Филогенетическое дерево, построенное методом ближайших соседей

Как видно, по результатам работы метода ближайших соседей выдвигается гипотеза (рис. 7.1), что ветви образцов HR3 и HR4 самые близкие и объединены в один узел, далее находятся ветви образцов H1.2 и H1.2, самым дальним в корневом узле оказалась ветвь с образцом HR2. Поскольку данный метод относится к группе методов на основании генетической дистанции, полученное дерево может не отображать исторический процесс эволюции, а только демонстрировать конечную степень дивергенции таксонов.

7.7 Построение филогенетического дерева методом максимального правдоподобия

Метод максимального правдоподобия (Maximum-likelihood) [123] относится к методам, анализирующим дискретные признаки. Он использует модели замен для построения древа, основываясь на оценке правдоподобия присутствия определенного нуклеотида в конкретной позиции. Применяя функцию правдоподобия, можно определить вероятности нахождения нуклеотидов во всех анализируемых последовательностях и, исходя из них, построить древо являющееся максимально правдоподобным. Метод максимального правдоподобия рассматривает отдельно каждый нуклеотид, а не дистанции между последовательностями, что позволяет повысить достоверность

полученного древа, но в то же время это негативно сказывается на скорости подсчета.

Для реконструкции исторической реколонизации вида используем метод максимального правдоподобия PhyML [124] – модификацию базового алгоритма максимального правдоподобия, который заключается в оптимизации параметров с использованием числового метода «золотое сечение» и многократном улучшении дерева (изменении ветвления) [125], а также с «greedy»-алгоритмом [126].

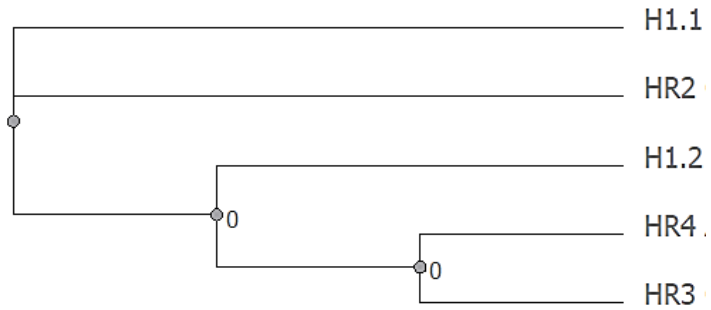

Рис. 7.2. Филогенетическое дерево, построенное методом максимального правдоподобия

В результате получим дерево (рис. 7.2), построенное методом максимального правдоподобия с использованием модели замен Кимура в течении 100 000 репликаций до достижения требуемого значения индекса бутстреп-поддержки. Метод максимального правдоподобия получает схожую топологию и выдвигается гипотеза, что ветви с образцами HR3 и HR4 самые поздние по хронологии и объединены в один узел. Порядок следования остальных ветвей: сначала ветвь с образцами H1.2, далее в корневом узле находятся ветви с образцами HR2 и H1.1. При этом, поскольку данный метод отображает исторический процесс эволюции, можно говорить о том, что образцы H1.1 и HR2 появились исторически быстрее, чем другие образцы.

7.8 Построение филогенетического дерева методом Байеса

Байесовский метод (Bayes) использует функцию правдоподобия для подсчета апостериорной вероятности получения той или иной топологии древа, основываясь на модели замен нуклеотид. Достоинства метода заключаются в том, что он учитывает филогенетическую неопределенность, а также использует сложные модели эволюции нуклеотидных последовательностей. Недостатком можно считать высокую чувствительность к неверно выбранной модели замен, а также низкую скорость вычисления [127].

Для реконструкции используем инструмент MrBayes [128], в котором для снижения вычислительной сложности алгоритма расчет апостериорной вероятности реализуется методом Монте-Карло для марковских цепей [129].

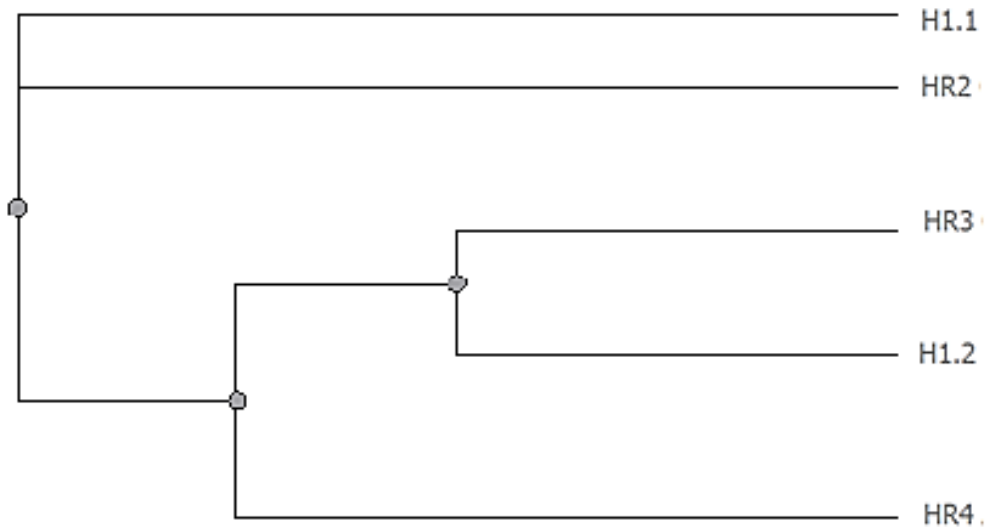

Рис. 7.3. Филогенетическое дерево, построенное методом Байеса

Дерево, построенное методом Байеса с использованием модели замен GTR (General Time Reversible Model) в течение 100 000 репликаций, предоставлено на рис.6.3. По результатам работы метода Байеса выдвигается гипотеза, по которой следует, что образцы H1.1 и HR2 определяются как самые ранние в видообразовании растения, затем

отделяется ветвь с образцом HR4 и ветви с образцами HR3 и H1.2 объединяются в один узел.

7.9 Построение филогенетического дерева методом генетического алгоритма

В рамках исследования предложен новый метод для филогенетического анализа – генетический алгоритм, который является методом оптимизации и относится к области мягких вычислений. Данный алгоритм основан на эволюционных принципах естественного отбора и генетики и довольно часто используется для оптимизации различных областей данных.

Каждая особь в генетическом алгоритме представляет собой потенциальное решение, таким образом, популяция в рассматриваемом случае будет содержать множество топологий (набор филогенетических реконструкций).

В рамках данного эксперимента особь кодируется хромосомой, состоящей из последовательности ген, которые должны хранить информацию об узлах и ветвях дерева с нуклеотидными последовательностями.

На эффективность алгоритма существенно влияет задание начальной популяции. В данном эксперименте предложено за точку старта (нулевой узел) принять самый информативный образец нуклеотидных последовательностей, определенный как максимальным количеством нуклеотидов в матрице данных после множественного выравнивания.

В начальной популяции родительская ветвь задается рандомно. Если брать в качестве родительской ветви разные нуклеотидные последовательности, то можно получить до n различных хромосом.

В качестве фитнесс-функции используется функция вида

$$S = \sum_{i,j \epsilon M} S(g_i, g_j),$$

где g – множество ветвей в матрице топологии. За минимизируемый параметр принимается расхождение между нуклеотидными последовательностями, которое задает порядок следования и объединения ветвей в филогенетическом дереве.

Для поиска наибольшего сходства между ветвями филогенетического дерева предложено использовать матрицу дистанций. Для определения матрицы дистанций выбрана модель эволюции Джукса и Кантора (JC). Дистанция рассчитывается по формуле

$$d = -\frac{3}{4}\ln(1 - \frac{4p}{3}),$$

где p – количество несовпадающих нуклеотидов в попарно сравниваемых нуклеотидных последовательностях. При p>0,75 выражение не имеет смысла, это является недостатком метода, так как ситуации с более 75% различающихся нуклеотидов принципиально не исключены. Поскольку в данном исследовании процент различающихся нуклеотидов значительно меньше, данная модель эволюции можно использовать в эксперименте.

Для филогенетической реконструкции выбран оператор рекомбинации - одноточечный кроссинговер, когда для родителей случайным образом определяется точка внутри хромосомы, в которой обе хромосомы делятся на две части и обмениваются ими. В исследовании для каждой особи в качестве оператора мутации случайно выбирается позиция и с малой вероятностью выполняется инвертирование значения переменной в выбранной позиции.

При формировании матрицы топологии обязательно необходимо учитывать, что связи ветвей могут быть только иерархическими, таким образом, в алгоритме проверяется, чтобы ветвь с нуклеотидной последовательностью входила только в один узел дерева.

В результате проводимого исследования генетический алгоритм был приспособлен для задачи построения филогенетического дерева.

Программное обеспечение для проведения эксперимента было реализовано на Phyton.

Генетический алгоритм для пяти нуклеотидных последовательностей участка гена trnL-F образцов растения одного вида был настроен следующим образом:

– количество особей в популяции было задано 50;

– для расчета матрицы дистанций использовалась эволюционная модель Джукса и Кантора;

– были заданы вероятности для оператора скрещивания (значение 0,9) и оператора мутации (значение 0,1);

– максимальное количество поколений установлено 50.

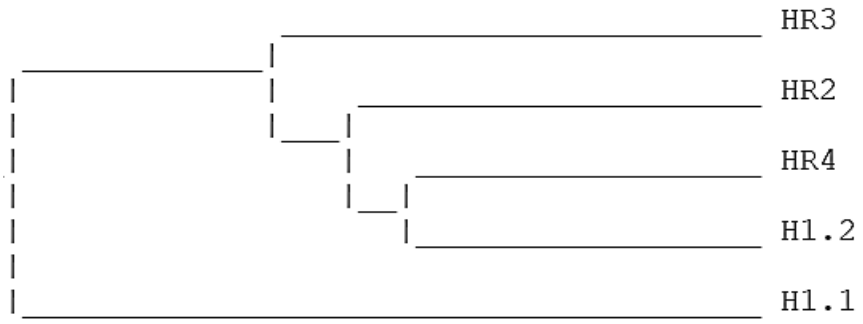

Рис.7.4. Филогенетическое дерево, построенное методом генетического алгоритма

Дерево, построенное методом генетического алгоритма (рис.7.4) является гипотезой филогенетической реконструкции вида растения на основании нуклеотидных последовательностей ДНК. Как видно, в данной гипотезе образец H1.1 определен как исторически самый ранний, затем отделяется ветвь с образцом HR3, потом с HR2, и самими близкими и поздними по данной гипотезе оказались ветви с образцами H1.2 и HR4, объединенные в один узел.

Разница в топологиях, полученных генетическим алгоритмом и другими методами может быть обусловлена применением разных моделей эволюции. В методе Байеса выбрана более сложная модель эволюции.

В дальнейшем в исследовании планируется добавить подбор модели эволюции в зависимости от исходных данных и оптимизировать работу операторов генетического алгоритма.

Предложенный метод генетического алгоритма для построения филогенетического дерева по пяти нуклеотидным последовательностям участка гена trnL-F выдвинул гипотезу с топологией филогении образцов растения одного вида, которая схожа с результатами работы метода ближайших соседей, метода максимального правдоподобия, метода Байеса. Это свидетельствует о том, что конечное дерево описывает реальные эволюционные события и генетический алгоритм может и в дальнейшем применяться для построения филогенетического дерева.

7.10 Анализ зависимости времени прогнозирования временного ряда

В рамках диссертационного исследования [130] был предложен алгоритм прогнозирования временных рядов и использованием энтропийных временных рядов. Прогнозное значение получалось как среднее арифметическое от прогноза, полученного каким-либо методом прогнозирования и прогноза, полученного через энтропийный временной ряд.

$$\boldsymbol{Forecast} = \frac{ForecastOrigMethod + ForecastByEntropy}{\mathbf{2}}$$

Прогнозное значение энтропийного временного ряда получалось из значения энтропии в последней точки ряда, а также прогноза значения динамики энтропии. При этом прогноз значения динамики получался с использованием того же метода прогнозирования (по временному ряду динамки энтропии).

В рамках эксперимента планируется определить, какие методы прогнозирования будут показывать лучшее время выполнения и точность

получаемого прогнозного значения. Будет использовано 4 метода прогнозирования с разными подходами:

– первый метод прогнозирования – метод экспоненциального сглаживания [131]. Данный метод является простым в реализации, однако имеет низкое качество прогноза. Данный метод выбран как один из самых быстрых, но имеющий наихудшую точность прогнозного значения;

– второй метод прогнозирования – метод экспоненциального сглаживания с трендом и сезонностью Хольта-Уинтерса [132]. Более сложный метод прогнозирования, чем метод экспоненциального сглаживания. Метод Хольта-Уинтерса позволяет учитывать сезонность и общий тренд временного ряда. Метод является расширением метода Хольта до трехпараметрического экспоненциального сглаживания. Для получения прогноза необходимо определить эти три параметра. Определение параметров может производиться, например, путем простого перебора;

– в качестве третьего метода выбран метод прогнозирования на основе нейронной сети. Данный метод имеет высокую точность, однако его отличает использование иных алгоритмах прогнозирования, нежели рассмотренные ранее. В качестве нейронной сети была выбрана сеть, обучаемая с помощью метода обратного распространения ошибки [133];

– последним методом был выбран метод прогнозирования тенденции временных рядов на основе нечеткого моделирования с использованием системы нечетких правил для различных гипотез прогноза: гипотезы сохранения тенденции, гипотеза смены тенденции и гипотезы смены тенденции на заданный период. Данный метод отличает использование нечеткого подхода при получении прогнозных значений;

Таким образом, в эксперименте будут задействованы методы прогнозирования, основанные на разных подходах, что позволит оценить

как временные затраты разных подходов в прогнозировании, так и точности прогнозируемых значений.

В описываемом эксперименте для сравнения временных затрат и точности прогнозируемых значений были взяты ВР с соревнований CIF 2015 [134]. В рамках соревнования применялись различные методы прогнозирования ВР.

ВР, разработанные для CIF 2015, были рассчитаны для методов вычислительного интеллекта, к которым относят: нечеткий метод, искусственные нейронные сети, эволюционные алгоритмы, решения и регрессионная структура, поддерживающие векторные машины, гибридные подходы и т. д., используемые во всех областях прогнозирования, прогнозирования и анализа временных рядов и т. д. Также допускаются комбинированные методы, использующие любой метод CI.

Так как ВР, на основе которых будут проводится эксперименты, не имеют предметной области, для формирования НВР из ВР будет применяться простая FCM-кластеризация.

По результатам проведенных экспериментов получились следующие значения (таблица 7.1).

Таблица 7.1. Результаты эксперимента

Метод	**Среднее время выполнения**	**Средняя ошибка прогнозного значения**
Метод Хольта-Уинтерса	6,86	0,12
Метод экспоненциального сглаживания	0,188	0,58
Метод прогнозирования на основе нейронной сети	6,67	0,14
Метод прогнозирования тенденции ВР	5,12	0,21

По результатам проведенного эксперимента получается, что методы Хольта-Уинтерса и метод прогнозирования на основе нейронной сети показывают схожие результаты, как по времени выполнения, так и по точности. С другой стороны, метод прогнозирования тенденции BP показывает более лучше время выполнения с не сильно отличающейся разницы точности прогноза (более 1 секунды при менее 10% разницы точности).

В итоге, если при прогнозировании значения BP ключевым фактором будет является время выполнения, то можно использовать методы, наподобие метода прогнозирования тенденции BP, который позволяет получить прогноз быстрее других методов, хоть и с меньше точностью. Однако если важна точность прогнозного значения, то тут нужно использовать методы, которые требует большего времени выполнения.

ЗАКЛЮЧЕНИЕ

Методы машинного обучения активно используются для решения широкого круга задач различных предметных областей. Для получения качественной модели машинного обучения требуется наличие обучающей выборки достаточного объема, а также определение (конструирование) признаков, на основе которых модель машинного обучения сможет выполнить обобщение и выявить закономерности. Полученные закономерности представляют собой нетривиальные знания об особенностях сущностей и/или процессов и могут быть использованы для поддержки принятия управленческих решений.

В представленной книге авторы изложили основные подходы, методы и алгоритмы в области получения, подготовки и синтеза данных, конструирования признаков с учетом особенностей контекста и в условиях неопределенности:

– формирования контекста анализа динамики показателей процессов;

– конструирование признаков с учетом особенностей контекста и в условиях неопределенности;

– доверие и интерпретацию результатов предиктивной аналитики;

– извлечение признаков объектов из больших неструктурированных текстовых данных;

– извлечения атомов продукционных правил базы экспертных знаний из неструктурированных текстов.

Теоретические разработки поддержаны иллюстративными примерами, а также практическими программными разработками, которые прошли экспериментальную апробацию. Авторы надеются, что настоящая книга будет полезна не только научным работникам, но и прикладным пользователям и будут благодарны за отклик.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Интеллектуальный анализ временных рядов [Текст] : учебное пособие для студентов высших учебных заведений, обучающихся по направлению «Прикладная информатика» / Н. Г. Ярушкина, Т. В. Афанасьева, И. Г. Перфильева. – Москва : Форум : ИНФРА-М, 2012. – 159 с. : ил., табл.; 22 см. - (Высшее образование).

2. A new improved forecasting method integrated fuzzy time series with exponential smoothing method / Peng Ge, Jun Wang, Peiyu Ren, Huafeng Gao and Yuyan Luo // Int. J. Environment and Pollution, Vol. 51, 2013, P. 206–221.

3. Viertl R. Statistical Methods for Fuzzy Data. John Wiley & Sons, Ltd, 2011, 270 P.

4. How to Do Machine Learning with Small Data? A Review from an Industrial Perspective / I. Kraljevski, Y. C. Ju, D. Ivanov, C. Tschöpe, M. Wolff // arXiv preprint arXiv:2311.07126. 2023.

5. Conditional synthetic data generation for robust machine learning applications with limited pandemic data / H. P. Das, R. Tran, J. Singh, X. Yue, G. Tison, A. Sangiovanni-Vincentelli, C. J. Spanos // Proceedings of the AAAI Conference on Artificial Intelligence. 2022. Vol. 36 (11). P. 11792–11800.

6. Narayanan A., Hardy T. Synthetic data generation for machine learning model training for energy theft scenarios using cosimulation // IET Generation, Transmission & Distribution. 2023. Vol. 17 (5). P. 1035–1046.

7. Dahmen J., Cook D. SynSys: A synthetic data generation system for healthcare applications // Sensors. 2019. Vol. 19 (5). P. 1181.

8. Generating synthetic time series to augment sparse datasets / G. Forestier, F. Petitjean, H. A. Dau, G. I. Webb, E. Keog // 2017 IEEE international conference on data mining (ICDM). IEEE, 2017. P. 865–870.

9. Brain D., Webb G. I. On the effect of data set size on bias and variance in classification learning // Proceedings of the Fourth Australian Knowledge Acquisition Workshop, University of New South Wales. 1999. P. 117–128.

10. Nonnemaker J., Baird H. S. Using synthetic data safely in classification // Document recognition and retrieval XVI. SPIE, 2009. Vol. 7247. P. 134–144.

11. Инженерия машинного обучения / пер. с англ. А. А. Слинкина. – М.: ДМК Пресс, 2022. – 306 с.: ил.

12. Featuretools – An open source python framework for automated feature engineering. URL: https://www.featuretools.com (дата обращения: 01.12.2023).

13. Van Dyk D. A., Meng X. L. The art of data augmentation //Journal of Computational and Graphical Statistics. 2001. Vol. 10 (1). P. 1–50.

14. Augmented ultrasonic data for machine learning / I. Virkkunen, T. Koskinen, O. Jessen-Juhler, J. Rinta-Aho // Journal of Nondestructive Evaluation. 2021. Vol. 40. P. 1–11.

15. Shorten C., Khoshgoftaar T. M. A survey on image data augmentation for deep learning //Journal of big data. 2019. Vol. 6 (1). P. 1–48.

16. Bansal M., Krizhevsky A., Ogale A. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst //arXiv preprint arXiv:1812.03079. 2018.

17. Nikolenko S. I. Synthetic data for deep learning // arXiv preprint arXiv:1909.11512. 2019.

18. Gupta A., Vedaldi A., Zisserman A. Synthetic data for text localisation in natural images // Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. P. 2315–2324.

19. Synthetic data and artificial neural networks for natural scene text recognition / M. Jaderberg, K. Simonyan, A. Vedaldi, A. Zisserman // arXiv preprint arXiv:1406.2227. 2014.

20. Synthetic Data Generation for Improved covid-19 Epidemic Forecasting / N. Bannur, V. Shah, A. Raval, J. White // medRxiv. 2020. Pp. 2020.12. 04.20243956.

21. Dermgan: Synthetic generation of clinical skin images with pathology / A. Ghorbani, V. Natarajan, D. Coz, Y. Liu // Machine learning for health workshop. PMLR, 2020. P. 155–170.

22. Helal A., Mendez-Vazquez A., Hossain S. Specification and synthesis of sensory datasets in pervasive spaces // 2009 IEEE Symposium on Computers and Communications. IEEE, 2009. P. 920–925.

23. Persim-Simulator for human activities in pervasive spaces / S. Helal, J. W. Lee, S. Hossain, E. Kim, H. Hagras, D. Cook // 2011 Seventh International Conference on Intelligent Environments. IEEE, 2011. P. 192–199.

24. Synthetic data augmentation using GAN for improved liver lesion classification / M. Frid-Adar, E. Klang, M. Amitai, J. Goldberger, H. Greenspan // 2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018). IEEE, 2018. P. 289–293.

25. Zero-shot learning – a comprehensive evaluation of the good, the bad and the ugly / Y. Xian, C. H. Lampert, B. Schiele, Z. Akata //IEEE transactions on pattern analysis and machine intelligence. 2018. Vol. 41 (9). P. 2251–2265.

26. Pan S. J., Yang Q. A survey on transfer learning // IEEE Transactions on knowledge and data engineering. 2009. Vol. 22 (10). P. 1345–1359.

27. A comprehensive survey on transfer learning / F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, Q. He // Proceedings of the IEEE. 2020. Vol. 109 (1). P. 43–76.

28. A survey on deep transfer learning / C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, C. Liu // Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27. Springer International Publishing, 2018. P. 270–279.

29. Weiss K., Khoshgoftaar T. M., Wang D. D. A survey of transfer learning //Journal of Big data. 2016. Vol. 3 (1). P. 1–40.

30. Dropout: a simple way to prevent neural networks from overfitting / N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov // The journal of machine learning research. 2014. Vol. 15 (1). P. 1929–1958.

31. Das H. P., Abbeel P., Spanos C. J. Likelihood Contribution based Multi-scale Architecture for Generative Flows //arXiv preprint arXiv:1908.01686. 2019.

32. Ioffe S., Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift //International conference on machine learning. pmlr, 2015. P. 448–456.

33. An anomaly detection framework for identifying energy theft and defective meters in smart grids / S. C. Yip, W. N. Tan, C. Tan, M. T. Gan, K. Wong //International Journal of Electrical Power & Energy Systems. 2018. Vol. 101. P. 189–203.

34. SMOTE: synthetic minority over-sampling technique / N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer // Journal of artificial intelligence research. 2002. Vol. 16. P. 321–357.

35. Sun Y., Wong A. K. C., Kamel M. S. Classification of imbalanced data: A review // International journal of pattern recognition and artificial intelligence. 2009. Vol. 23 (4). P. 687–719.

36. Kaggle. URL: https://www.kaggle.com (дата обращения: 01.12.2023).

37. Hugging Face. URL: https://huggingface.co (дата обращения: 01.12.2023).

38. Интеграция информационных систем на основе метода слияния онтологий при расчете баланса производственных мощностей / Н. Г. Ярушкина, А. А. Романов, А. А. Филиппов, А. Ю. Долгановская, М. С. Григоричева // Известия Самарского научного центра Российской академии наук. 2018. 20 (4-3). С. 468–476.

39. Rudolph S. Foundations of description logics // Reasoning Web International Summer School. Berlin, Heidelberg : Springer Berlin Heidelberg, 2011. Pp. 76–136.

40. OWL 2 Web Ontology Language. URL: https://www.w3.org/TR/owl2-overview/ (дата обращения: 01.12.2023).

41. Dorri A., Kanhere S. S., Jurdak R. Multi-agent systems: A survey // IEEE Access. 2018. Vol. 6. P. 28573–28593.

42. Recent advances in consensus of multi-agent systems: A brief survey / J. Qin, Q. Ma, Y. Shi, L. Wang // IEEE Transactions on Industrial Electronics. 2016. Vol. 64 (6). P. 4972–4983.

43. Naciri N., Tkiouat M. Multi-agent systems: theory and applications survey // International Journal of Intelligent Systems Technologies and Applications. 2015. Vol. 14 (2). P. 145–167.

44. An approach to building decision support systems based on an ontology service / A. Romanov, J. Stroeva, A. Filippov, N. Yarushkina //Mathematics. 2021. Vol. 9 (22). P. 2946.

45. Yarushkina, N. Soft computing and complex system analysis // International Journal of General Systems, 2001 EID: 2-s2.0-0035058896

46. Time series analysis using soft computing methods / I. Perfilieva; N. Yarushkina, T. Afanasieva, A. Romanov // International Journal of General Systems. 2013. DOI: 10.1080/03081079.2013.798911)

47. Geographic Variations in Cutaneous Melanoma Distribution in the Russian Federation / A. Muntyanu, E. Savin, F. Ghazawi, A. Alakel, A. Zubarev, I. Litvinov // Dermatology. 2020. Vol 236(6). P. 500–507.

48. Каприн А. Д., Старинский В. В., Петрова Г. В. Состояние онкологической помощи населению России в 2021 году. М.: МНИОИ им. П.А. Герцена, 2022.

49. Каприн А. Д., Старинский В. В., Петрова Г. В. Злокачественные новообразования в России в 2017 году (заболеваемость и смертность). М.: МНИОИ им. П.А. Герцена, 2018.

50. Global Cancer Statistics 2020: GLOBOCAN Estimates of Incidence and Mortality Worldwide for 36 Cancers in 185 Countries / H. Sung, J . Ferlay, R.L. Siegel, M. Laversanne, I. Soerjomataram, A. Jemal, F. Bray // CA: a cancer journal for clinicians.2021. Vol. 71(3). P. 209–249.

51. El-Khatib H., Popescu D., Ichim L. Deep learning–based methods for automatic diagnosis of skin lesions // Sensors. 2020. Vol. 20 (6). P. 1753.

52. Melanoma and nevus skin lesion classification using handcraft and deep learning feature fusion via mutual information measures / J. A. Almaraz-Damian, V. Ponomaryov, S. Sadovnychiy, H. Castillejos-Fernandez // Entropy. 2020. Vol. 22 (4). P. 484.

53. Computer-aided diagnosis of skin diseases using deep neural networks / M. N. Bajwa, K. Muta, M. I. Malik, S. A. Siddiqui, S. A. Braun, B. Homey, S. Ahmed // Applied Sciences. 2020. Vol. 10 (7). P. 2488.

54. Computational methods for the image segmentation of pigmented skin lesions: a review / R. B. Oliveira, E. Mercedes Filho, Z. Ma, J. P. Papa, A. S. Pereira, J. M. R. Tavares // Computer methods and programs in biomedicine. 2016. Vol. 131. P. 127–141.

55. Almeida M. A. M., Santos I. A. X. Classification models for skin tumor detection using texture analysis in medical images //Journal of Imaging. 2020. Vol. 6 (6). P. 51.

56. Oliver A., Vàzquez M. TBXTools: a free, fast and flexible tool for automatic terminology extraction // Proceedings of the international conference recent advances in natural language processing. 2015. P. 473–479.

57. Simon N. I., Kešelj V. Automatic term extraction in technical domain using part-of-speech and common-word features // Proceedings of the ACM symposium on document engineering 2018. 2018. P. 1–4.

58. Linders G. M., Louwerse M. M. Lingualyzer: A computational linguistic tool for multilingual and multidimensional text analysis // Behavior research methods. 2023. P. 1–28.

59. Gurbanova A. M. Comparative analysis of methods of automatic term extraction from texts // Problems of Information Technology. 2021. P. 55–69

60. Андреев, И. А. Построение унифицированной базы знаний на основе данных профилей социальных сетей / И. А. Андреев, В. С. Мошкин // Гибридные и синергетические интеллектуальные системы : Материалы VI Всероссийской Поспеловской конференции с международным участием, Калининград, 27 июня – 01 2022 года. – Калининград: Балтийский федеральный университет имени Иммануила Канта, 2022. – С. 324-331.

61. Dai Z., Callan J. Deeper text understanding for IR with contextual neural language modeling // Proceedings of the 42nd international ACM SIGIR

conference on research and development in information retrieval. 2019. P. 985–988.

62. Zhang L., Tian R., Chen J. Text Mining for US Pension De-Risking Analysis // Risks. 2022. Vol. 10 (2). P. 41.

63. Creating Russian WordNet by Conversion / N. V. Loukachevitch, G. Lashevich, A. A. Gerasimova, V. V. Ivanov, B. V. Dobrov // In Proceedings of Conference on Computatilnal linguistics and Intellectual technologies Dialog-2016, 2016. P. 405–415.

64. Создание системы мониторинга интернет-текстов по теме «Социально-политическая жизнь регионов Российской Федерации» / Е. Б. Козеренко, Ю. И. Морозова, К. И. Кузнецов, М. М. Шарнин, С. А. Ступников, Д. О. Брюхов, А. Е. Вовченко // Материалы III Международной научно-практической конференции (Москва, 18-19 сентября 2014): Сборник статей и тезисов. М.: МГГУ им. МА Шолохова. 2014. С. 51–55.

65. Дмитриев А. С., Соловьев И. С., Заболеева-Зотова А. В. Извлечение взаимосвязей между объектами и терминами в текстах на экономическую тематику // Известия Волгоградского государственного технического университета. 2015. №. 13. С. 55–60.

66. Наместников А. М., Филиппов А. А., Шигабутдинов И. М. Подход к извлечению многословных терминов из текстов на естественном языке с применением синтаксических шаблонов // Автоматизация процессов управления. 2021. №. 3. С. 87–95.

67. Semantic web-based diagnosis and treatment of vector-borne diseases using SWRL rules / R. Chandra, S. Tiwari, S. Agarwal, N. Singh //Knowledge-Based Systems. 2023. Vol. 274. P. 110645.

68. SWRL reasoning on ontology-based clinical dengue knowledge base / R. Devi, D. Mehrotra, H. B. Zghal, G. Besbes // International Journal of Metadata, Semantics and Ontologies. – 2020. Vol. 14 (1). P. 39–53.

69. Мошкин В. С., Ярушкина Н. Г. Особенности интеграции механизмов логического вывода в онтологическую модель представления знаний с помощью SWRL-правил // Четырнадцатая национальная конференция по искусственному интеллекту с международным участием КИИ-2014: Труды конференции. 2014. Т. 1. С. 173.

70. Воронина И.Е., Пигалкова Е.А. Интеграция знаний продукционного характера в правовую онтологическую модель с помощью SWRL-правил // Вестник ВГУ, серия «Системный анализ и информационные технологии». 2010. № 2. С. 139–144

71. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. URL: https://www.w3.org/Submission/SWRL (дата обращения: 01.12.2023).

72. Semantic rules representation in controlled natural language in FluentEditor / A. Wróblewska, P. Kapłański, P. Zarzycki, I. Ługowska // 2013 6th International Conference on Human System Interactions (HSI). IEEE, 2013. P. 90-96.

73. HermiT: an OWL 2 reasoner / B. Glimm, I. Horrocks, B. Motik, G. Stoilos, Z. Wang // Journal of automated reasoning. 2014. T. 53. P. 245–269.

74. Expressing and executing informed consent permissions using SWRL: the all of us use case / M. Amith, M. R. Harris, C. Stansbury, K. Ford, F. J. Manion, C. Tao // AMIA Annual Symposium Proceedings. American Medical Informatics Association, 2021. Vol. 2021. P. 197.

75. Moshkin V., Yarushkina N. Modified knowledge inference method based on fuzzy ontology and base of cases // Creativity in Intelligent Technologies and Data Science: Third Conference, CIT&DS 2019, Volgograd,

Russia, September 16–19, 2019, Proceedings, Part II 3. Springer International Publishing, 2019. P. 96–108.

76. SWRL Net: A spectral, residual deep learning model for improving short-term wave forecasts / J. Mooneyham, S. C. Crosby, N. Kumar, B. Hutchinson //Weather and Forecasting. 2020. Vol. 35 (6). P. 2445–2460.

77. Chaos reports 1994-2017. URL: http://www. standishgroup.com (дата обращения: 01.12.2023).

78. Ralph P. The sensemaking-coevolution-implementation theory of software design // Science of Computer Programming. 2015. Vol. 101. P. 21–41.

79. Sosnin P. Substantially evolutionary theorizing in designing software-intensive systems // Information. 2018. Vol. 9 (4). P. 91.

80. Design thinking: Challenges for software requirements elicitation / H. Ferreira Martins, A. Carvalho de Oliveira Junior, E. Dias Canedo, R.A. Dias Kosloski, R. Ávila Paldês, E. Costa Oliveira // Information. 2019. Vol. 10 (12). P. 371.

81. Modeling context with an architecture viewpoint / A. Bedjeti, P. Lago, G.A. Lewis, R.D. De Boer, R. Hilliard // 2017 IEEE International Conference on Software Architecture (ICSA). IEEE, 2017. P. 117–120.

82. Wongthongtham P., Pakdeetrakulwong U., Marzooq S.H. Ontology annotation for software engineering project management in multisite distributed software development environments // Software Project Management for Distributed Computing. Springer, Cham, 2017. P. 315–343.

83. Namestnikov A., Guskov G. Ontological mapping for conceptual models of software system // Proceedings of Conference Open Semantic Technologies for Intelligent Systems, Minsk, Republic of Belarus, 2017. P. 16–18.

84. Гуськов Г.Ю., Наместников А.М., Ярушкина Н.Г. Подход к поиску похожих по структуре проектов, основанный на онтологии языка UML // Радиотехника. 2017. №. 6. С. 122–127.

85. Bechberger L., Kühnberger K.U. A thorough formalization of conceptual spaces // Joint german/austrian conference on artificial intelligence (künstliche intelligenz). Springer, Cham, 2017. P. 58–71.

86. A fuzzy ontology-based approach for tool-supported decision making in architectural design / T. Di Noia, M. Mongiello, F. Nocera, U. Straccia // Knowledge and Information Systems. 2019. Vol. 58 (1). P. 83–112.

87. ГОСТ Р 57100-2016/ISO/IEC/IEEE 42010:2011. Системная и программная инженерия. Описание архитектуры. URL: https://docs.cntd.ru/document/1200139542 (дата обращения: 27.11.2022).

88. Analyzing the effects of test driven development in GitHub / N. C. Borle, M. Feghhi, E. Stroulia, R. Greiner, A. Hindle // Empirical Software Engineering. 2018. Vol. 23 (4). P. 1931–1958.

89. Learning from, understanding, and supporting devops artifacts for docker / J. Henkel, C. Bird, S.K. Lahiri, T. Reps // 2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE). IEEE, 2020. P. 38–49.

90. Schermann G., Zumberi S., Cito J. Structured information on state and evolution of dockerfiles on github // Proceedings of the 15th international conference on mining software repositories. 2018. P. 26–29.

91. Predicting health indicators for open source projects (using hyperparameter optimization) / T. Xia, W. Fu, R. Shu, R. Agrawal, T. Menzies // Empirical Software Engineering. 2022. Vol. 27 (6). P. 1–31.

92. Splicing community patterns and smells: A preliminary study / M. De Stefano, F. Pecorelli, D. A. Tamburri, F. Palomba, A. De Lucia //

Proceedings of the ieee/acm 42nd international conference on software engineering workshops. 2020. P. 703–710.

93. Discovering community patterns in open-source: a systematic approach and its evaluation / D. A. Tamburri, F. Palomba, A. Serebrenik, A. Zaidman // Empirical Software Engineering. 2019. Vol. 24 (3). P. 1369–1417.

94. Filippov A., Romanov A., Iastrebov D. An Approach to Data Mining of Software Repositories in Terms of Quantitative Indicators of the Development Process and Domain Features // International Conference on Intelligent Information Technologies for Industry. Springer, Cham, 2023. P. 346–357.

95. Печерских А.А., Романов А.А., Береснев Ю.И. Подход к поиску структурно-похожих проектов программных систем // Автоматизация процессов управления. 2022. №. 3 (69). С. 20–26.

96. Namestnikov A.M., Filippov A.A., Avvakumova V.S. An ontology based model of technical documentation fuzzy structuring // CEUR SCAKD Workshop Proceedings. 2016. P. 63–74.

97. Наместников А.М., Филиппов А.А., Шигабутдинов И.М. Подход к извлечению многословных терминов из текстов на естественном языке с применением синтаксических шаблонов // Автоматизация процессов управления. 2021. №. 3 (65). С. 87–95.

98. Modeling the Context of the Problem Domain of Time Series with Type-2 Fuzzy Sets / A.A. Romanov, A.A. Filippov, V.V. Voronina, G. Guskov, N.G. Yarushkina // Mathematics. 2021. Vol. 9, no. 22. p. 2947. DOI: 10.3390/math9222947.

99. A case study on the impact of similarity measure on information retrieval based software engineering tasks / M.M. Rahman, S. Chakraborty,

G. Kaiser, B. Ray // arXiv preprint arXiv:1808.02911. 2018. URL: https://arxiv.org/pdf/1808.02911.pdf (дата обращения: 27.11.2023).

100. TabbyChat. URL https://github.com/killjoy1221/tabbychat (дата обращения: 27.11.2023).

101. NG-Tracker. URL: https://gitlab.com/romanov73/ng-tracker (дата обращения: 27.11.2023).

102. The 2020 Scrum Guide: Scrum Team. URL: https://www.scrumguides.org/scrum-guide.html#scrum-team (дата обращения: 27.11.2023).

103. Grönlund M., Jefford-Baker J. Measuring correlation between commit frequency and popularity on GitHub. Stockholm, Sweden : School of Computer Science and Communication (CSC), 2017.

104. Romanov A., Yarushkina N., Filippov A. Application of time series analysis and forecasting methods for enterprise decision-management // International Conference on Artificial Intelligence and Soft Computing. Springer, Cham, 2020. P. 326–337.

105. Hershberg R. Mutation – the engine of evolution: studying mutation and its role in the evolution of bacteria // Cold Spring Harbor perspectives in biology. 2015. Vol. 7 (9). P. a018077.

106. Primer-directed enzymatic amplification of DNA with a thermostable DNA polymerase / R. K. Saiki, D. H. Gelfand, S. Stoffel, S. J. Scharf, R. Higuchi, G. T. Horn, K. B. Mullis, H. A. Erlich // Science. 1988. Vol. 239 (4839). P.487–491.

107. Slocombe P. M., Smith M. Nucleotide sequence of bacteriophage phi X174 DNA //Nature. 1977. Vol. 265 (5596). P. 687-695.

108. Huson D. H., Bryant D. Application of phylogenetic networks in evolutionary studies // Molecular biology and evolution. 2006. Vol. 23 (2). P. 254–267.

109. Thetortoise and the hare II: relative utility of 21 noncoding chloroplast DNA sequences for phylogenetic analysis / J. Shaw, E. Lickey, J. T. Beck, S. B. Farmer, W. Liu, J. Miller, K. C. Siripun, C. T. Winder, E. E. Schiling, R. L. Small // American Journal of Botany. 2005. Vol. 92 (1). P. 142–166.

110. MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform / K. Katoh, K. Misawa, K. I. Kuma, T. Miyata // Nucleic acids research. 2002. Vol. 30 (14). P. 3059–3066.

111. Bos D. H., Posada D. Using models of nucleotide evolution to build phylogenetic trees // Developmental & Comparative Immunology. 2005. Vol. 29 (3). P. 211–227.

112. Sullivan J., Joyce P. Model selection in phylogenetics // Annu. Rev. Ecol. Evol. Syst. 2005. Vol. 36. P. 445–466.

113. Johnson J. B., Omland K. S. Model selection in ecology and evolution // Trends in ecology & evolution. 2004. Vol. 19 (2). P. 101–108.

114. Jukes T. H., Cantor C. R. Evolution of protein molecules // Mammalian protein metabolism. 1969. Vol. 3. P. 21–132.

115. Kimura M. A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences // Journal of molecular evolution. 1980. Vol. 16. P. 111–120.

116. The general stochastic model of nucleotide substitution / F. J. L. O. J. Rodriguez, J. L. Oliver, A. Marín, J. R. Medina // Journal of theoretical biology. 1990. Vol. 142 (4). P. 485–501.

117. Felsenstein J. Confidence limits on phylogenies: an approach using the bootstrap // Evolution. 1985. Vol. 39 (4). P. 783–791.

118. Trees R. P. The Neighbor-joining Method: A New Method for // Mol. Biol. Evol. 1987. Vol. 4 (4). P. 406–425.

119. Gascuel O., Steel M. Neighbor-joining revealed //Molecular biology and evolution. 2006. Vol. 23 (11). P. 1997–2000.

120. Holmes S. Bootstrapping phylogenetic trees: theory and methods //Statistical Science. 2003. Vol. 18 (2). P. 241–255.

121. Mihaescu R., Levy D., Pachter L. Why neighbor-joining works // Algorithmica. 2009. Vol. 54. P. 1–24.

122. Felsenstein J. Distance methods for inferring phylogenies: a justification // Evolution. 1984. P. 16–24.

123. Felsenstein J. Evolutionary trees from DNA sequences: a maximum likelihood approach //Journal of molecular evolution. 1981. Vol. 17. P. 368–376.

124. Guindon S., Gascuel O. A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood //Systematic biology. 2003. – Vol. 52 (5). P. 696–704.

125. New algorithms and methods to estimate maximum-likelihood phylogenies: assessing the performance of PhyML 3.0 / S. Guindon, J. F. Dufayard, V. Lefort, M. Anisimova, W. Hordijk, O. Gascuel // Systematic biology. 2010. Vol. 59 (3). P. 307–321.

126. PartitionFinder: combined selection of partitioning schemes and substitution models for phylogenetic analyses / R. Lanfear, B. Calcott, S. Y. Ho, S. Guindon // Molecular biology and evolution. 2012. Vol. 29 (6). P. 1695–1701.

127. Suzuki Y., Glazko G. V., Nei M. Overcredibility of molecular phylogenies obtained by Bayesian phylogenetics // Proceedings of the National Academy of Sciences. 2002. Vol. 99 (25). P. 16138–16143.

128. Geneious Basic: an integrated and extendable desktop software platform for the organization and analysis of sequence data / M. Kearse, R. Moir, A. Wilson, S. Stones-Havas, M. Cheung, S. Sturrock, A. Drummond // Bioinformatics. 2012. Vol. 28 (12). P. 1647–1649.

129. Mau B., Newton M. A., Larget B. Bayesian phylogenetic inference via Markov chain Monte Carlo methods // Biometrics. 1999. Vol. 55(1). P. 1–12.

130. Мошкина И. А., Эгов Е. Н. Прогнозирование состояния проекта по временным рядам метрик и выявленным аномалиям // Автоматизация процессов управления. 2019. №. 2 (56). С. 67–74.

131. Прогнозирование методом экспоненциального сглаживания (ES, exponential smoothing). URL: https://fnow.ru/algorithm-comparison/jeksponencial-noe-sglazhivanie (дата обращения: 15.12.2023)

132. Метод Хольта-Винтерса. URL: https://fnow.ru/algorithm-comparison/metod-holta-wintersa (дата обращения: 15.12.2023)

133. Алгоритм обучения многослойной нейронной сети методом обратного распространения ошибки (Backpropagation). URL: https://habr.com/ru/post/198268/ (дата обращения: 15.12.2023).

134. Computational Intelligence in Forecasting (CIF). URL: http://irafm.osu.cz/cif/main.php (дата обращения: 15.12.2023).

Научное электронное издание

ЯРУШКИНА Надежда Глебовна
МОШКИН Вадим Сергеевич
ГУСЬКОВ Глеб Юрьевич
и др.

КОНСТРУИРОВАНИЕ ПРИЗНАКОВ ИЗ КОНТЕКСТА ДИНАМИКИ ПОКАЗАТЕЛЕЙ ПРОЦЕССОВ

ЛР № 020640 от 22.10.97
Дата подписания к использованию 25.12.2023.

ЭИ № 1877. Объем данных 3,4 Мб. Заказ № 22.

Ульяновский государственный технический университет

432027, Ульяновская обл., Ульяновск, Сев. Венец, 32.

ИПК «Венец» УлГТУ, 432027, Ульяновская обл., Ульяновск, Сев. Венец, 32.

Тел.: (8422) 778-513

e-mail: venec@ulstu.ru

venec.ulstu.ru