УДК 001.894

Модель ТРИЗ-системы, предназначенная для создания ассистирующей программы

Марчук А.Г. (Институт систем информатики СО РАН, Новосибирский государственный университет)

В статье представлены некоторые наработки автора в области ТРИЗ – теории решения изобретательских задач. Наработки предполагается превратить в программу-ассистента, предназначенную для широкого круга пользователей и широкого круга задач. Предложена модель постановки задачи, в виде расширенного сценарного анализа, далее выявляется сложность, мешающая реализации этого сценария, предлагаются способы разрешения противоречий. В основном, предложенная методика постановки и решения задачи соответствует теории, разработанной и развитой Г.С. Альтшуллером е его учениками. Новое – это модель, объединяющая разные «веточки» теории и практики ТРИЗ.

Ключевые слова: *Теория решения изобретательских задач, ТРИЗ, программа-ассистент, постановка и решение творческих задач, информационные технологии.*

1. Введение

Под задачей здесь понимается самая общая постановка: Как сделать ИЗМЕНЕНИЕ? Причем нас будут интересовать не тривиальные или инженерные решения, а творческие, преодолевающие какие-трудности. С такими постановками имеют дело разные ветви эвристики, совместно с основной ветвью эвристики – мозговым штурмом. Не отвергая этот подход, в СССР усилиями Г.С. Альшуллера и его учеников, была разработана Теория решения изобретательских задач (ТРИЗ) [1, 2, 3, 10]. Теория была порождена и развивалась на материалах технического творчества и дает возможности создавать новые технические решения от рационализаторского уровня до самых крупных изобретений и открытий.

ТРИЗ хорошо показала себя и на других творческих задачах: задачах стратегического планирования [13], рекламы и PR, обучению школьников, вплоть до младших и детского сада и т.д. Делались попытки алгоритмизировать теорию [14], в первую очередь это АРИЗ [8, 9] (Алгоритм решения изобретательских задач), и реализовать в виде программы-ассистента. Известная мне успешная попытка выполнялась в 80-х годах в г. Минске, под руководством

В.М. Цурикова [6, 7]. Это так называемая Изобретающая машина. Наработки автора статьи в области ТРИЗ позволяют надеяться, что подобную систему можно идейно переосмыслить и реализовать на новом техническом уровне. Возможно быстрое развитие систем искусственного интеллекта, в дальнейшем сможет также внести вклад в науку, методология и практику решения творческих задач, обучения творчеству, нетривиальному мышлению. ТРИЗ применяется и в информационных технологиях [12].

2. Основная идея

Была разработана простая модель, охватывающая задачу в целом, ее постановку и направления ее решения.

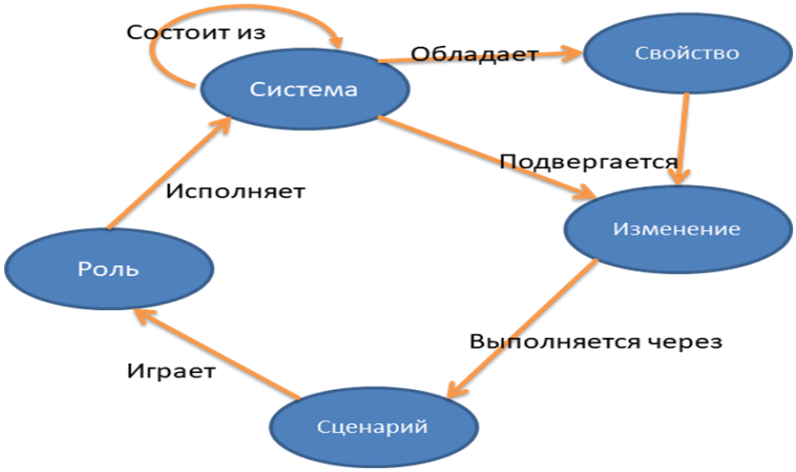

Рисунок 1. Сценарный анализ

Предполагается, что имеется набор систем (предметов), возможно выстроенных иерархически, обладающих некоторыми свойствами. Какую-то из систем или какое-то свойство требуется подвергнуть изменению. Изменение – это самый общий переход из состояния1 в состояние2. Утверждается, что изменение всегда осуществляется через сценарий (кто-то что-то делает в какой-то последовательности). Сценарий состоит из ролей, которые ответственны за структуру и динамику воздействий, роли исполняют какие-то системы. Типичные роли: изделие, инструмент. В этой стройной картине могут возникнуть трудности (в ТРИЗе они называются противоречиями), которые мешают выполнять изменение. Собственно преодоление этих трудностей и делается с помощью творческого акта (Эврика!), порождающего нетривиальное решение. Для примера, рассмотрим совсем простенькую задачу «Разбить вазу».

Рисунок 2. Иллюстрация к задаче «Разбить вазу»

Задача приведена намеренно простейшая. Это для того, что бы объяснить и как выполняется сценарный анализ и как формируются возможные решения.

Казалось бы, а в чем проблема? Действительно, если сценарий не содержит проблемы, то и задачи нет – надо просто его выполнять. Мы усложним задачу, обратите внимание на то, что молоток резиновый, т.е. мягкий и с помощью него, вероятно, разбить вазу не получится. Теперь построим модель задачи:

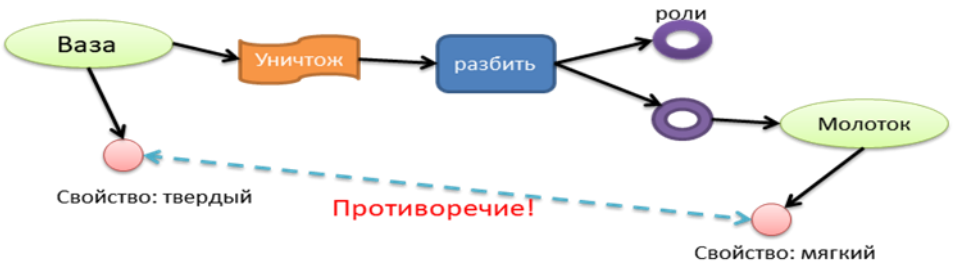

Рисунок 3. Модель задачи

Мы видим, что противоречие создало цикл в графе, хотя по построению граф должен быть деревом, т.е. без циклов. Мое предположение заключается в том, этот граф можно разрезать в любом месте и это создает варианты решения задачи. На этом «детском» примере, можно разрезать граф по пунктиру (по противоречию). Такой способ нахождения сильных решений фактически является основным в ТРИЗ. Глубокая идея Г.С.Альтшуллера заключается в том, что если свойство1 (твердый) одного предмета противоречит свойству2 (мягкий) другого предмета, то может быть АНТИсвойство1 первого предмета не противоречит свойству2. Это создает классическую задачу разрешения противоречия: обеспечить, чтобы, тогда когда надо и где надо, появлялось антисвойство1. В данном случае, надо «размягчить» вазу в прямом или переносном смысле. Например, если ваза из меди, то, наверное, ее можно нагреть и сделать «мягкой». Разрезание по основному противоречию, на самом деле, дает два направления поиска. Второй вариант, может свойство1 не противоречит АНТИсвойству2. Для этой задачи, надо поискать способ «упрочнения» резинового молотка на время, напр.

путем сильного охлаждения или в конкретной точке напр. вбив штырь в резину или через структурные преобразования, напр. привязав камень, надев жесткий футляр и др.

Но разрезание по противоречию не единственное направление поиска. Можно поискать ресурсы сначала в исполнителях, потом в сценарии. Заменить исполнителя (молоток), очевидно вполне интересное предложение, только надо подобрать его из подходящих ресурсов. Образно говоря, использовать микроскоп вместо молотка нехорошо, но если надо… Возможности предложить другой сценарий зависит от задачи, а задача ставиться как «произвести изменение». Мы остановились на каком-то решении, а решений может быть много. Если «произвести изменение» подразумевает что ваза должна исчезнуть, то наверное можно придумать несколько сценариев, которые его реализуют. Например, можно перенести вазу в другую комнату, задрапировать вазу и т.д. Новые сценарии могут потребовать новой модели и нового анализа. Очень может быть, в новом сценарии есть трудность, которая мешает ее решить, мы снова приходим к противоречию и т.д.

Следующий элемент анализа – анализ цепочки целей. Уничтожить вазу – это цель. А для чего ее уничтожать? Формулируется цель более высокого порядка, достижение которой требует нескольких ДРУГИХ сценариев. Новая цель сформулирована, а для чего ее достигать? Переходим на новый уровень. В общем, даже для небольших постановок, множество потенциальных решений будет расти. Главное при этом, мы не отрываемся от основы ТРИЗ: идеальный конечный результат – противоречие – разрешение противоречия.

3. Подробности и детали модели и ее использования

В принципе, процесс решения задачи по ТРИЗ един и заключается в поиске правильной постановки задачи с выходом на возможное решение. Для сложных задач, напр. задач стратегического планирования, результатом этого процесса может быть не один и не два, а много вариантов решения. Применение методики требует определенной квалификации и осваивания основ теории и практики ТРИЗ’а. Есть предположение о том, что создание программной поддержки методики позволит легче ее осваивать и использовать для решения практических задач. Для удобства восприятия детали модели и методики разбиты на отдельные части, обладающие некоторой замкнутостью понятий и средств.

Иногда решаемая задача имеет широкий спектр возможных решений. Это бывает напр. при стратегическом планировании, в рекламе. Также хочется предлагать варианты в режиме «мозгового штурма».

3.1 «Починить сценарий»

Для начала ограничимся только сценарием. Такой класс задач условно назовем «Починить сценарий». Он часто встречается в техническом творчестве. Когда перед изобретателем стоит задача улучшить конструкцию или агрегат, который перестал удовлетворять требованиям из-за изменившейся ситуации. Такая изолированная постановка актуальна в тех случаях, когда цель (изменение) не подвергается сомнению, а способ практически зафиксирован имеющимися традициями.

В этом случае, определяется, что с чем конфликтует в данном варианте сценария. Это выделяет противоречие. Основные варианты противоречия: 1) либо какой-то исполнитель не справляется с требуемой ролью, 2) либо выделяются два элемента сценария, конфликтующие между собой.

Первый вариант решается принципиально – отказ от негодного исполнителя. И подбор годного. Здесь возможны разные варианты такой подмены: отказ от собственно роли; переложение функций роли на другие роли; нахождение другого исполнителя за счет имеющихся ресурсов; введение «элемента-Х», годного для исполнения роли; разбиение роли на части и отказ исполнителю только в той части, с которой он не может справиться.

Второй вариант противоречия, т.е. конфликтующая пара, наиболее популярное противоречие, разрешению которого посвящена основная часть методологии ТРИЗ. Как уже пояснялось в предыдущем разделе, ситуацию можно довести до полярной – свойство-Антисвойство. В описанных методиках ТРИЗ или АРИЗ выход на полярную ситуацию производится часто через этапы «административное противоречие», «техническое противоречие», «физическое противоречие». Потом противоречие можно разрешать в пространстве, во времени, в структуре или в отношениях.

Важный информационный ресурс представляет собой качественно собранные Г.И.Альтшуллером две таблицы: Таблица приемов разрешения технических противоречий [4] и Таблица применения приемов разрешения технических противоречий [5]. Они позволяют быстрее выйти на вероятное решение-кандидат. То есть, вариантов разрешить противоречие несколько:

- Самый общий, т.е. думать о том как преодолеть данное противоречие в пространстве, во времени, в структуре или отношениях;
- Перебрать 40 приемов из «Таблицы приемов…» в поисках подходящего решения;

- На основе дополнительной информации о направлениях улучшения/ухудшения общих характеристик, выбрать наиболее часто встречающиеся приемы, среди которых не наверняка, но с увеличенной вероятностью, возможно решение задачи.

3.2 Поиск альтернативного сценария

Дальнейшее расширение постановки и формирование новых решений-кандидатов, выполняется в «обратном» направлении построения модели задачи, то есть, в сторону изменения. Это может «увести» от образа, заданного первичным сценарием, но создает богатое пространство для нетривиальных решений. Как уже констатировалось, сценарий – это способ реализации изменения. Но только один из способов. Если не закладывать в постановку задачи излишней конкретики, то довольно легко выйти на альтернативные способы. Обычно бытовая или техническая задача ставится слишком определенно, что мешает расширять круг вариантов постановки и вариантов ее решения. Например, в задаче «принести книгу», сразу фиксируется и «принести» и «книгу». Не всегда психологически просто выйти на формулирование изменения типа «надо, чтобы какой-то или определенный информационный носитель появился бы у меня». Приходится использовать такие «сказочные» термины и понятия как «появление», «исчезновение», спонтанное перемещение, самобранки, скороходы и т.д. Итогом уточненной формулировки изменения будет возможность придумать новый способ (сценарий) реализации изменения.

Породив другой сценарий, мы возвращаемся на начало анализа (роли, исполнители) и выходим на трудность, переводим ее в противоречие и приемы разрешения противоречий. Вся методика, в этом смысле, рекуррентна и легко «навешивается» на варианты постановок.

3.3 Парадоксальные вопросы

Иногда уместна методика, основанная на простых, но парадоксальных вариантах. Мне известен один: методика В.Г.Сибирякова и Л.Н.Семеновой (методика от авторов получена устно, публикация неизвестна). Среди порожденных этим творческим дуэтом методик есть методика, которая настолько парадоксальная, что выглядит невозможной. В моем переизложении методика заключается в следующем.

Пусть требуется произвести ИЗМЕНЕНИЕ. Задаются четыре следующих вопроса:

ИЗМЕНЕНИЕ производить не надо, в каком случае?

ИЗМЕНЕНИЕ произойдет само собой, в каких случаях?

ИЗМЕНЕНИЕ произвести за счет РЕСУРСА

ИЗМЕНЕНИЕ не производить, а цель достигнуть.

Анализ этих вопросов показывает, что они не так глупы, как кажутся. Более того, они полностью соответствуют ряду приемов, применяемых в ТРИЗе. Главное в этих вопросах – попытка коротко сформулировать идеальный конечный результат (ИКР) и нацелить решающих на применение ключевых способов его достижения. Действительно, первый вопрос можно пропустить, поскольку он фактически является комбинацией следующих двух. Второй вопрос формулирует один из вариантов достижения ИКР, что изменение произойдет само собой. Оказывается многое так устроено, это и снег, который «сам» сходит, это и многочисленные инфраструктурные потоки (почта, транспорт, детские сады, специализированные службы и т.д.). Главное – нацелить решающих на нужное направление поиска.

Третий вопрос, как правило, есть способ породить множество вариантов по схеме: предлагается подумать что или кто из окружающих предметов, сущностей, персонажей может существенно повлиять на решение поставленной задачи. Берется любое и «примеривается». Случайное блуждание мысли, характерное для обычного мозгового штурма здесь нивелируется за счет фокусировки не столько на методе, сколько на изменении. С точки зрения рассмотренных ранее моделей, это «игра» с текущим или гипотетическим сценарием, реализующим ИЗМЕНЕНИЕ. Вводятся новые роли, новые исполнители ролей, элементы-Х и т.д. Все по ТРИЗ.

Четвертый вопрос вообще непонятен. Тут присутствует и «не-решение» задачи и ее решение в одном вопросе. Однако, если разобраться, то вопрос весьма осмысленный. Вспомним про цепочку целей и примерим ее к постановке «сделать ИЗМЕНЕНИЕ». А для чего нужно делать оговоренное изменение? Для решения более общей задачи (цели). Вполне уместным является другое изменение, не совпадающее с первично сформулированным, но решающее достижение более общей цели. Опять – все по ТРИЗ!

Методика компактна и готова к употреблению, особенно под руководством опытного модератора. И вообще, осознание ТРИЗ каждым человеком идет медленно. Во многом это результат того, что ТРИЗ – это мирвозрение. Но использование технологий, решение практических задач, будет ускорять осваивание теории и переход на новое мышление.

3.4 Причинно-следственная цепочка

Методика обычно неявно присутствует в разных вариантах анализа по ТРИЗ’у. Как отдельная, хотя часто и промежуточная, методика сформулирована в материале [11]. Речь идет об определении вариантов трудностей или противоречий. Как уже неоднократно упоминалось, главное в ТРИЗ’е это выявление противоречия с дальнейшей попыткой его

преодоления. Здесь важным является уровень и формулировка противоречия. Правильная формулировка облегчает поиск решения, неправильная – усложняет.

Наиболее общая ситуация, это когда какой-то элемент системы не сочетается (конфликтует) с другим элементом системы. Мы будем рассматривать частный случай когда свойство1 элемента1 противоречит свойству2 элемента2. Выделим из этой пары «мешающее» свойство одного из элементов (напр. первого) и сведем противоречие к полярному: СВОЙСТВО мешает – анти-СВОЙСТВО не мешает.

Первая часть методики заключается в последовательном задании вопроса «по какой причине?». СВОЙСТВО мешает – по какой причине? Причина обязательно существует, иначе не было бы помехи. Причина – это также помеха и мы также можем попытаться сформулировать причину в виде мешающего свойства. Теперь имеется рекуррентная ситуация, когда можно сформировать цепочку свойств-причин и поискать среди них наиболее интересный вариант для преодоления.

Пример приведу по [11]:

Детали картины не видны посетителям ($Св_1$).

По какой причине? – Картина затемнена ($Св_2$).

По какой причине? – У картины нет добавочного освещения ($Св_3$).

По какой причине? – Торшер неустойчив ($Св_4$).

По какой причине? – Высоко расположен центр тяжести торшера ($Св_5$)

Причинно-следственная цепочки может быть преобразована в «полярные» вопросы, т.е. – требуется и свойство и антисвойство, напр.:

Пусть центр тяжести торшера расположен высоко, но он будет устойчив.

Пусть у картины нет добавочного освещения, но она не будет затемнена.

…

То есть, используется совмещение $Св_i$, НО $анти\text{-}Св_{i+1}$

3.5 Цепочка целей, дерево целей

В модели, изображенной на рисунке 1 не хватает очень важного направления развития модели. Указывается необходимое изменение, но не указывается для чего выполнять это изменение. Иногда это существенно. Таким образом, есть изменение как цель задачи. Но если задать и ответить на вопрос «А для чего делать это ИЗМЕНЕНИЕ?», то появляется цель более высокого уровня. И появляются элементы модели, связанные с этим более высоким уровнем: сценарий или сценарии, роли в сценариях, исполнители ролей. Набор ролей и

систем-исполнителей поменяется. К этой новой постановке можно и нужно применять ТРИЗ'овские приемы и получаемые решения могут быть более высокого уровня.

Процесс выявления целей более высокого уровня может быть продолжен. Концовка этой цепочки в теории называется миссией, а весь анализ часто встречается в стратегическом планировании и рекламе.

Интересной частью анализа цепочки целей является обратный анализ. Дело в том, что изменение, даже если это супер-цель, реализуется какими-то сценариями. Надо хотя бы обозначить сценарии, входящую в основную цепочку изменений и сценариев. В идеале, цепочка сценариев должна приходить к тому, с чего все началось. Но на этом пути могут быть существенные ответвления. Фокусируясь на первичной постановке задачи и выполняя анализ цепочек, все решения должны иметь отношение к тому, с чего начали.

Другое направление работы с целями и сценариями заключается в движении от более комплексных целей к более простым задачам, от сложных сценариев, к простым. Это – построение дерева целей. Смысл его в следующем:

Берем текущий сценарий, в нем есть роли. Роли, это изменения, которые в совокупности должны привести к выполнению сценария, а значит, к достижению цели. В науке, такое разделение закрепляется терминологически: есть цель и есть задачи. Разбиение цели на задачи, задач на подзадачи является обычным методологическим решением и к ТРИЗ'у прямого отношения не имеет. Но в наборе предоставляемых инструментов, работа с деревом целей имеет ясную позицию и не противоречит Теории решения изобретательских задач.

Заключение

ТРИЗ является уникальной теорией и методикой решения творческих задач. Ее преимущество по сравнению с разными вариантами «Мозгового штурма» в направленности поиска решения, заключающейся в том, что задача должна быть хорошо поставлена, должно быть правильно сформулирован идеальный конечный результат, выявлены противоречия, мешающие решению, найдены разрешения противоречий. Основная схема решения задачи по ТРИЗ дополнена сценарным анализом и формированием цепочек или деревьев целей.

Автор попробовал максимально упростить сущность и форму ТРИЗ'овского подхода к постановке и решению задач, формировать у пользователя прозрачную картину задачи и манипуляций с задачей.

В модели пока отсутствуют некоторые важные составляющие, такие как S-образное развитие технических и организационных систем, системный анализ.

Наличие модели, позволяет достаточно понятно выстраивать программный и визуальный интерфейс для программы-ассистента. Корнем построения является изменение, т.е. цель задачи. У изменения могут быть несколько вариантов сценарного решения, каждый сценарий расписывается как совокупность ролей, у ролей имеются исполнители. Некоторые существенные связи элементов графа будут нарушать древовидность. Например, это касается противоречий. Но в целом, пользователь может мыслить привычными формами иерархических построений.

Список литературы

1. Альтшуллер Г. С. Найти идею. Введение в ТРИЗ – теорию решения изобретательских задач / Г. С. Альтшуллер — «Альпина Диджитал», 1986
2. Альтшуллер Г.С., Злотин Б.Л., Зусман А.В., Филатов В.И. Поиск новых идей: от озарения к технологии (Теория и практика решения изобретательских задач) — Кишинев: Картя Молдовеняскэ, 1989. — 384 с.: ил. — ISBN: 5-362-00147-7.
3. Альшуллер Г.С., Селюцкий А.Б. Крылья для Икара: Как решать изобретательские задачи. - Петрозаводск: Карелия, 1980. -224 с.
4. Альтшуллер Г.С., Типовые приемы устранения технических противоречий, 1973 // https://www.altshuller.ru/triz/technique1.asp
5. Альшуллер Г.С. Таблица разрешения приемов разрешения технических противоречий, 1973 // https://www.altshuller.ru/triz/technique2.asp
6. В.М. Цуриков Проект "Изобретающая машина" / ТЕОРИЯ И ПРАКТИКА ОБУЧЕНИЯ ТЕХНИЧЕСКОМУ ТВОРЧЕСТВУ, Миасс, 23-27 мая 1988 г
7. В.М. Цуриков Проект "Изобретающая машина": интеллектуальная среда поддержки инженерной деятельности / Журнал ТРИЗ, 1991-1-2, сс. 17-34 / http://ratriz.ru/wp-content/uploads/2018/07/3-ZH-TRIZ-2-1.91.pdf
8. Петров В. Алгоритм решения изобретательских задач. Учебное пособие / Тель-Авив, 1999.
9. Орлов М.А. Основы классической ТРИЗ. Практическое руководство для изобретательного мышления. — 2-е изд., испр. и доп. — М.: СОЛОН-ПРЕСС. 2006. - 432 с. ISBN 5-98003-191-X
10. Генрих Альтшуллер Найти идею: Введение в ТРИЗ – теорию решения изобретательских задач / – 5-е изд. – М.: Альпина Паблишер, 2012. (Серия «Искусство думать»). 442 сс. ISBN 978-5-9614-2189-7
11. Гин А.А. Теория решения изобретательских задач. Учебное пособие I уровня : учебно-методическое пособие / А.А. Гин, А.В. Кудрявцев, В.Ю. Бубенцов, А. Серединский. – 3-е изд. – Томск : Изд-во Томского политехнического университета, 2017. – 64 с.
12. М.С. Рубин. Основы ТРИЗ. Применение ТРИЗ в программных и информационных системах: Учебное пособие. – Санкт-Петербург, СПбГУ, Математико-механический факультет,

Лаборатория системного программирования и информационных технологий (СПРИНТ), 2011. – 226 стр.

13. Кожемяко А.П. ТРИЗ. Пошаговое руководство для бизнеса в схемах - ИД Синергия, ISBN 978-5-6045849-4-1, 208 с.
14. М.Литвин, А.Любомирский Алгоритмы ТРИЗ в "Алгоритме": от искусства к ремеслу // Журнал ТРИЗ, №2(15) апрель 2006, с.10-12

УДК 81’33:004.822

Автоматизация построения терминологического ядра онтологии по компьютерной лингвистике на основе корпуса текстов

Овчинникова К. А. (Новосибирский национальный исследовательский государственный университет),

Иванов А. И. (Новосибирский национальный исследовательский государственный университет),

Сидорова Е. А. (Институт систем информатики СО РАН)

В работе предлагается подход к автоматическому построению терминологического ядра онтологии по компьютерной лингвистике. Рассматриваются вопросы создания онтологии верхнего уровня, определяющей возможные классы терминов для их дальнейшего поиска и систематизации. Предложен алгоритм генерации и начального пополнения предметного словаря, включающий два основных этапа. На первом шаге строится система лексико-семантических классов, основанных на классах онтологии. На втором шаге осуществляется наполнение словаря терминами и их соотнесение с классами словаря на основе имеющихся ресурсов: универсальной онтологии научного знания, тезауруса и портала по компьютерной лингвистике. Для проведения экспериментального исследования был собран корпус аналитических статей по компьютерной лингвистике с сайта Хабр и созданы наборы данных с разметкой терминов, включающие по 1065 предложений на русском языке. Проведены эксперименты для решения двух задач: обнаружение терминов и их классификация относительно классов онтологии. Для первой задачи были рассмотрены три нейросетевые модели: xlm-roberta-base, roberta-base-russian-v0 и ruRoberta-large. Лучшие результаты получены на последней модели: 0.91 F-меры. Проведен анализ ошибок классификатора, который показал, что высокую частотность ошибки неполного выделения термина. Для второй задачи была выбрана модель ruRoberta-large, показавшая лучшие результаты для первой задачи. Среднее значение F-меры для 12 используемых классов онтологии составило 0.89. Предложена общая архитектура системы создания и пополнения онтологий, интегрирующая лингвистические подходы и методы машинного обучения.

Ключевые слова: *терминологическое ядро онтологии; компьютерная лингвистика; онтология компьютерной лингвистики; извлечение терминов; классификация терминов.*

1. Введение

Систематизация знаний в активно развивающейся области, такой как компьютерная лингвистика (КЛ), является важной, но ресурсоемкой задачей. Большое количество информационных ресурсов, приложений, моделей требуют оперативного решения задач поиска и анализа информации о последних достижениях науки. Так, например, интерес к применению нейронных сетей для решения задачи автоматической обработки текстов вызвал появление огромного количества новых методов, наборов данных и языковых моделей, знания о которых необходимы специалистам. Для решения этой проблемы разрабатываются различные интернет-ресурсы: каталоги, вики-ресурсы, тематические форумы, порталы знаний [20]. Для описания и систематизации информации используют иерархические модели знаний, в первую очередь графы знаний и онтологии. На основе таких моделей далее создаются решения под управлением знаниями (knowledge-driven applications), которые решают проблему информационной совместимости и формализации и обеспечивают постоянную генерацию новых знаний, непрерывно анализируя множество разрозненных источников информации.

Согласно общепринятому определению в компьютерных науках, *онтология* — это способ формализации знаний, абстрактных или специфических, в какой-либо предметной области, реализованный на основе формального описания объектов, фактов и отношений между ними, и ориентированный на многократное использование для различных задач [27]. Для графов знаний онтология — это семантическая основа представления данных, базирующаяся на логике и включающая терминологический словарь и набор утверждений о моделируемых объектах.

Для разработки новых онтологий могут быть использованы так называемые *онтологии верхнего уровня (*или *базовые онтологии)*, которые включают основные или базовые понятия и отношения, используемые для описания и формализации знаний в целом классе предметных областях, объединенных общим назначением, например, научные предметные области. Так, в качестве основы для описания научной области может быть использована универсальная онтология научной области знаний, представленная в работе [15].

Для конкретизации онтологии на точную предметную область в первую очередь необходимо построить *терминологическое ядро онтологии* — терминологическую систему, описывающую концептуальный состав (множество понятий) онтологии предметной области.

Терминологическая система — это иерархически-структурированная совокупность терминов, принадлежащих к определенной предметной области. Под терминами понимаются слова и словосочетания, являющиеся наименованиями понятий моделируемой области знаний.

Разработка онтологий является сложным и трудоемким процессом, поэтому существует необходимость создания подходов, которые поддерживают и автоматизируют этот процесс. Одним из естественных способов выделить начальный набор понятий (терминов) — обеспечить извлечение названий сущностей предметной области из текстов. Данная задача сводится к двум подзадачам: извлечение ключевых терминов и их классификация относительно классов базовой онтологии. Эту задачу можно отнести к задаче *распознавания именованных сущностей* (named entity recognition, NER), когда в тексте необходимо выделять спаны (непрерывные фрагмента текста) сущностей из заранее заданного набора категорий. Большое количество исследований, посвященных NER, показывает актуальность этой задачи. При этом методы глубокого обучения получают наибольшее внимание, так как часто дают самые лучшие результаты. Однако соревнования, проводимые в данной области демонстрируют, что для разных задач одни и те же методы NER показывают различное качество работы [11].

Целью данного исследования является разработка методов автоматизации построения терминологического ядра онтологии по компьютерной лингвистике на основе базовой онтологии и корпуса текстов. Теоретическая часть работы включала конкретизацию базовой онтологии на область компьютерной лингвистики — требовалось выявить отсутствующие понятия и уточнить основные отношения. Практическая часть работы заключалась в апробации различных методов извлечения терминов и генерации предметных словарей.

Для достижения поставленной цели был собран русскоязычный корпус аналитических статей с веб-сайта Хабр (https://habr.com). С сайта были отобраны статьи, связанные с направлением Natural Language Processing за период с января 2010 года по сентябрь 2023 года (1681 статья).

2. Обзор существующих методов

На данный момент существует различные методы и подходы к построению онтологий. В основном применяются различные методики ручного проектирования онтологии и методы автоматизации отдельных этапов создания онтологии, например, извлечение терминов из текста, классификация терминов, извлечение объектов и отношений и др.

Первые методологии онтологического проектирования предложены еще в конце 90-х годов: Methontology [4], On-To-Knowledge [13], Enterprise Model [14]. Данные подходы были нацелены на создание онтологии предметной области для решения конкретных задач, например, извлечения знаний, систематизации, формализации, поиска информации. На протяжении последних пятнадцати лет развивается подход к разработке онтологий, базирующийся на применении паттернов онтологического проектирования (Ontology Design Patterns или ODP) [5], являющихся стандартизованными описаниями ранее созданных фрагментов онтологий. Суть предлагаемых методик сводится к описанию различных способов переиспользования паттернов при разработке новых онтологий.

Для разработки онтологии по компьютерной лингвистике полезно использовать схожие по тематике ресурсы. На настоящий момент ресурсов по компьютерной лингвистике немного. Так, например, существует русско-английский тезаурус по компьютерной лингвистике [26] и портал по компьютерной лингвистике [25], разработанные в 2010-х годах и не пополняющиеся на данный момент. Также существует онтология Машинного обучения [7], для которой авторы только собираются определить необходимые инструменты для пополнения.

Для автоматизации построения и пополнения онтологий на основе текстов на естественном языке применяются как лингвистические методы, так и методы на основе машинного обучения. К лингвистическим методам можно отнести методы на основе лексико-синтаксических паттернов онтологического проектирования [10, 3]. Они задают отображения языковых структур в онтологические структуры, с помощью шаблонов [19].

Для извлечения терминов и их классификации чаще всего используются методы машинного обучения. Одними из самых ранних подходов для распознавания именованных сущностей были подходы, основанные на классических методах машинного обучения, таких как метод опорных векторов (SVM) или метод случайного леса (Random Forests). Эти методы опираются на признаки слов, такие как принадлежность к синтаксической группе, семантический тип, чтобы в конечном счете определять расстояния между словами и разделить их на классы [16]. Лучшие результаты в данной категории показывают ансамблевые методы.

На сегодняшний день в большинстве случаев используют методы глубокого обучения или гибридные подходы на их основе, о чем свидетельствуют различные обзоры [23, 9]. Для задачи NER в основном используются таких нейросетевые архитектуры как рекуррентные нейронные сети (Recurrent Neural Networks, RNN), рекуррентные нейронные сети с долгой краткосрочной памятью (Long short-term memory, LSTM) или трансформеры типа BERT. Так,

в работе [6] используется разновидность рекуррентных нейронных сетей RNN-T, которая позволяет не только распознавать сущности, но и учитывать их вложенность друг в друга. NER в условиях ограниченных наборов обучающих данных можно рассматривать как отдельную разновидность задачи выделения именованных сущностей. Для решения данной проблемы в работе [17] авторы создают большой набора данных с размеченными именованными сущностями относительно хорошего качества, после чего обучают базовую модель с трансформерной архитектурой NER-BERT на данном наборе данных. После этого авторы обучают базовую модель для более специфичной задачи и более узкого домена сущностей. Этот метод дает лучшие результаты в сравнении с классическими подходами, в которых базовая модель обучается на более общей задаче, такой как маскирование. Созданный набор данных включает только тексты на английском языке, что не дает возможности использовать модель для задач анализа русскоязычного текста. Русскоязычные датасеты достаточного большого объема и качества существуют только для ограниченного набора сущностей, что также ограничивает их применения для более специфичной задачи извлечения терминов по компьютерной лингвистике.

3. Онтология верхнего уровня

В качестве онтологии верхнего уровня для создания онтологии по компьютерной лингвистике была взята онтология, предложенная в [15]. Данная онтология делится на онтологию научной деятельности и онтологию научного знания [21]. *Онтология научной деятельности* включает базовые понятия, относящиеся к организации научно-исследовательской деятельности, такие как *Персона, Организация, Деятельность, Публикация*. *Онтология научного знания* содержит метапонятия и отношения, задающие структуры для описания предметной области (научной дисциплины) портала знаний, например, *Раздел науки, Предмет исследования, Объект исследования, Метод исследования*, позволяющие выделить в данной науке значимые разделы и подразделы, задать типизацию предметов, объектов и методов исследования, описать результаты научной деятельности. Всего универсальная онтология содержит 11 классов и 92 отношения.

Для конкретизации онтологии на область компьютерной лингвистики использовались найденные систематизированные ресурсы: русско-английский тезаурус по компьютерной лингвистике [26] и портал по компьютерной лингвистике [25]. Информация о базовых сущностях предметной области была собрана на основе корпуса с помощью анализа употреблений сущностей в тексте, построения конкордансов и списка частотных терминов.

Полученную онтологию так же, как и базовую, можно условно разделить на две части: то, что относится к научному знанию (Рис. 1, Рис. 2) и то, что можно отнести к научной деятельности (Рис. 3). Ниже описаны особенности, полученные при конкретизации базовой онтологии.

A) Изменена структура онтологии верхнего уровня: убраны классы *Событие* и *Географическое место*, так как предполагается, что онтология будет в большей степени представлять научное знание, а не научную деятельность, несмотря на то, что одно трудно отделяется от другого. Класс *Предмет исследования* был выделен как подкласс класса *Объект исследования*, поскольку предмет исследования в одном исследовании может быть объектом исследования в другом.

B) *Метод исследования* в нашем исследовании является ключевым классом. Современные методы КЛ основаны на методах машинного обучения (МО), поэтому потребовалось ввести такое понятие как *Метод машинного обучения*, который был добавлен как подкласс *Метода исследования* (Рис. 1)

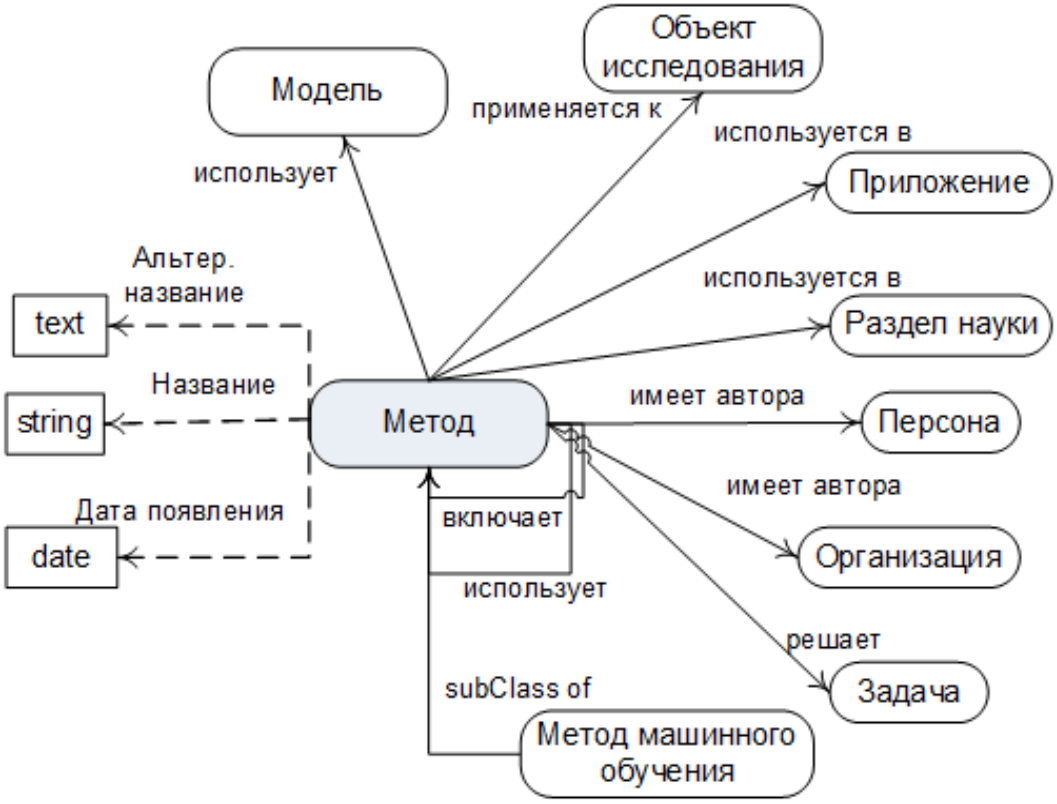

Рис. 1. Фрагмент онтологии, описывающий класс Метода исследования

C) Появление подкласса *Метод Машинного обучения* подразумевают наличие таких классов, как *Модель машинного обучения*, *Набор данных*, *Метрика* (Рис. 2). Выделение класса *Метрика* подтверждается вхождением одноименного термина в корпус текстов 510 раз.

D) Для класса *Модель машинного обучения*, была добавлена аксиома: <*Абстрактная модель* не может иметь связь с *Набором данных*>, т.к. это свойство есть только у конкретной модели.

E) Были добавлены такие классы как *Приложение* и *Окружение* (Рис. 3). Для объектов класса *Приложение* характерны текстовые фрагменты вида:

«в статье использовалась c++ библиотека openfst»

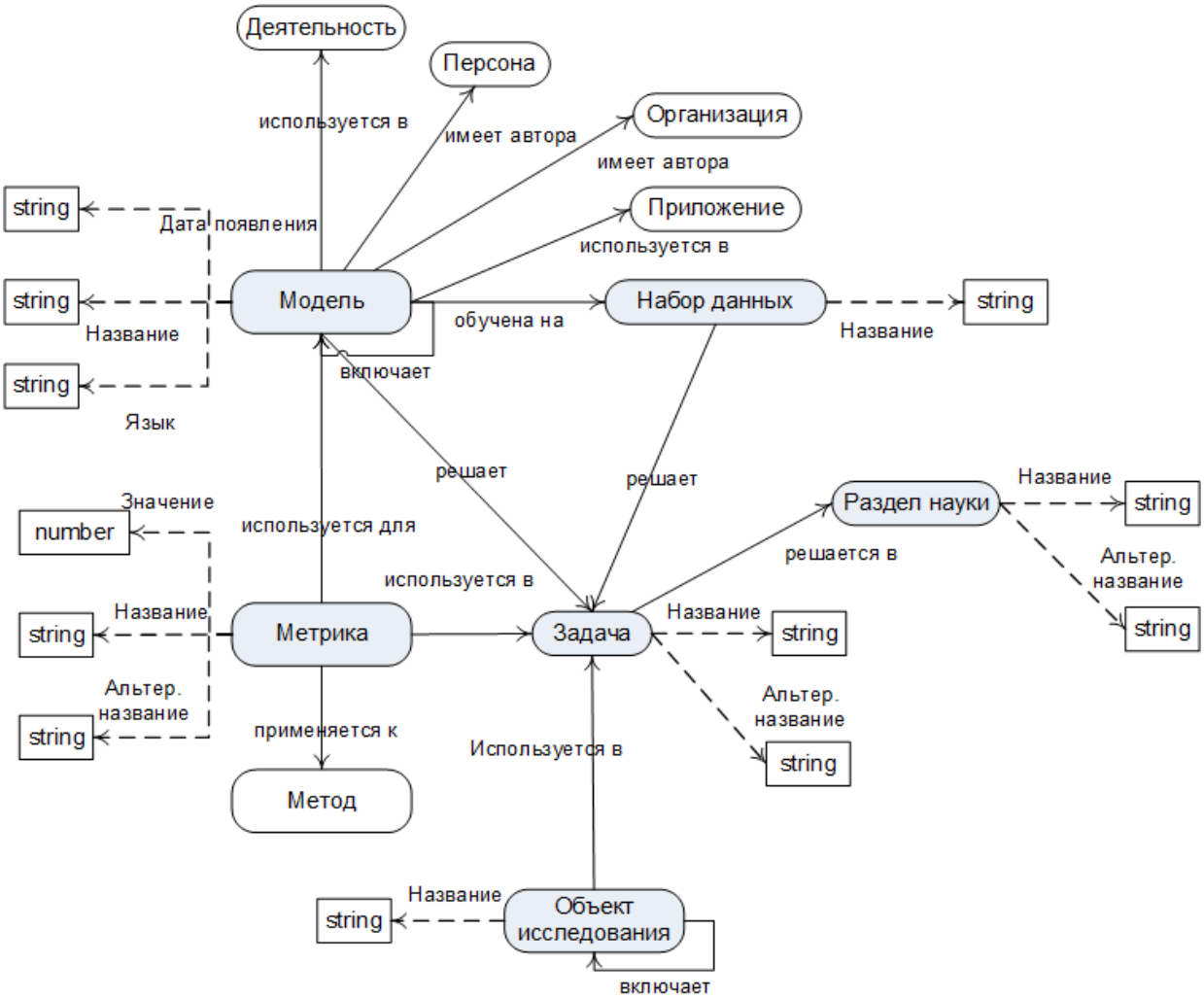

Рис. 2. Фрагмент онтологии, описывающий классы, связанные с научным знанием

F) Классу *Деятельность* было добавлено два важных подкласса, которых не было ранее: *Исследование* и *Эксперимент*. Для объектов класса *Исследование* характерны предложения следующего типа:

«Systran проводит исследование, в котором качество перевода оценивается путем <…>».

Необходимость выделения класса *Эксперимент* подтверждается предложениями типа:

«В рамках Джорджтаунского эксперимента демонстрировалась система, которая автоматически перевела 60 предложений с русского языка на французский».

Добавление класса *Эксперимент* как подкласса класса *Исследование* происходит на основе разделения исследований на два типа: теоретических и практических. Последние мы считаем экспериментами.

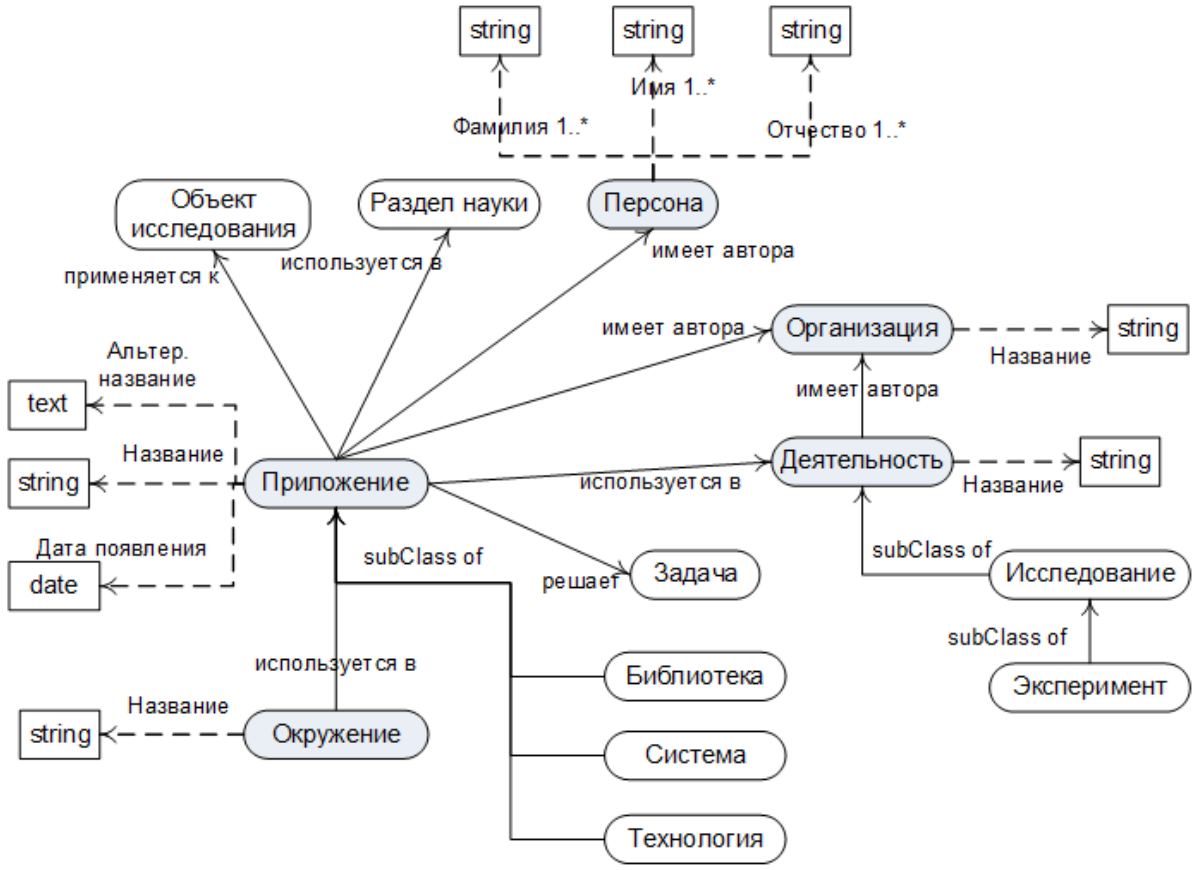

Рис. 3. Фрагмент онтологии, описывающий классы, связанные с научной деятельностью

G) В созданном наборе данных не оказалось представленным достаточного количества терминов и отношений, относящихся к классам *Информационный ресурс* (отдельно рассматривается его подкласс *Набор данных*), *Научный результат* и *Публикация*. Несмотря на это, было решено оставить данные классы в онтологии.

В итоге, созданная онтология по КЛ содержит 15 классов, 10 из которых уже содержались в базовой онтологии, и 111 отношений, 92 из которых присутствовали в онтологии до конкретизации. Для экспериментальных исследований классы, которые в текстах упоминались редко, были исключены из списка возможных категорий терминов.

4. Построение терминологического ядра онтологии

Построение терминологического ядра онтологии по компьютерной лингвистике включает в себя следующие этапы. Во-первых, необходимо составить систему классов предметного словаря, основанную на классах выбранной онтологии, и осуществить его начальное

наполнение базовыми терминами из онтологии и терминами из сопутствующих информационных ресурсов (тезаурусу и порталу по КЛ). На следующем этапе на основе составленного ранее корпуса текстов было необходимо разметить данные и создать датасеты для машинного обучения. На последующих этапах, можно непосредственно решать задачи извлечение и классификация терминов методами машинного обучения.

4.1. Генерация предметного словаря

Система классов в словаре генерируется на основе структуры онтологии, отражая иерархию ее объектов и отношений. Названия классов для терминов, обозначающих отношения, состоят из названий онтологических элементов в соответствии с шаблоном

<название_класса.название_отношения.название_класса>,

например: *Метод исследования.решает задачу.Задача*

Каждый термин словаря снабжается морфологической и семантической информацией, которые впоследствии используются при автоматической обработке текста.

Первоначальное наполнение предметного словаря велось в два этапа. На первом этапе был проанализирован русско-английский тезаурус по компьютерной лингвистике [26]. Из него были взяты подходящие термины классов: *Деятельность, Задача, Информационный ресурс, Метод исследования, Метрика, Объект исследования, Предмет исследования, Раздел науки* и *Результат*. Количество терминов составило 585.

Термины из тезауруса были сопоставлены с терминами на портале по компьютерной лингвистике [25], на котором расположена онтология научной деятельности, содержащая описанные выше классы. Терминам были добавлены подходящие онтологические классы, а тем, которых не оказалось на портале, были сопоставлены классы их синонимов или родовых понятий (более общее понятие для термина), если такие имелись. Оставшиеся термины были размечены вручную. Количество всех терминов, найденных на портале и в тезаурусе, составило 2640.

Для оценки качества автоматического сопоставления были просмотрены все термины на предмет неточного соотнесения с классом. Анализ показал, что неточности связаны с отсутствием класса в онтологии и неучтенной омонимии. Так, термин «*дискурс*» был отнесен к *Информационному ресурсу*, на основании существования *системы ДИСКУРС*. Это не является ошибкой, однако решено вручную добавить синонимичный термин и отнести его к классу *Деятельность*. Полнота автоматического соотнесения составила 75,9%, а точность – 99%. Высокая точность объясняется классификацией с опорой на синонимы и родовые

термины. При увеличении количества отношений в сопоставлении (не только синонимы и родовые понятия), полнота улучшалась, но точность сильно понижалась.

Поскольку тезаурус и портал не содержат терминов, появившихся в последние десять лет (в том числе относящихся к области машинного обучения), на следующем этапе необходимо было добавлять термины на основе корпуса современных текстов по КЛ.

4.2. Создание наборов данных

Для автоматического пополнения словаря терминами были составлены наборы данных на основе подобранных фрагментов текста из собранного корпуса по компьютерной лингвистике. Фрагменты текста подбирались так, чтобы получить репрезентативную выборку для всех классов терминов. На первом этапе для облегчения ручной разметки все тексты были автоматически размечены с помощью модели «ru_core_news_sm» из библиотеки Spacy. Использовалась BIO-разметка.

BIO-разметка текста (IOB-разметка) [8] – это способ разметки последовательных данных, таких как именованные сущности в тексте. Каждое слово в тексте помечается как "B" (начало сущности), "I" (продолжение сущности) или "O" (вне сущности). После пометки B, I или O через дефис указывается название класса. Так, для задачи извлечения терминов будет указываться B-TERM или I-TERM, а для задачи классификации для указания класса термина будут использоваться метки класса, например, к классу метод будут использоваться метки B-Method, I-Method.

Следующий этап заключался в отборе подходящих предложений, содержащих термины, относящиеся к нашей предметной области. Из них были составлены два набора данных: для задачи извлечения терминов и для задачи классификации терминов (размеченный по классам онтологии). Пример предложения в датасете для задачи извлечения терминов:

```
# sent_id = 744
# text = Создателем ORES является Wikimedia Foundation.
Создателем O
ORES B-TERM
является O
Wikimedia B-TERM
Foundation I-TERM
. O
```

Для второго набора данных для разметки терминов использовался список классов онтологии, описанный в предыдущих разделах, для которых метка имела вид B-Class для

начала термина, I-Class для остальных частей термина, а O указывало на часть вне термина, где Class - наименование онтологического класса. Пример предложения в датасете для задачи извлечения терминов:

```
# term_id = 5:
[TEXT]: <term>Извлечение именованных сущностей</term> из текста - одна из самых востребованных функций текстовой аналитики.
[TERM]: Извлечение именованных сущностей
[CLASS]: Task
```

Таким образом, в наших наборах было выделено 12 признаков для терминов классов, а объем каждого набора данных составил 1065 предложений (4180 терминов).

4.3. Извлечение терминов из корпуса

Для извлечения терминов из текста и отнесения их к соответствующим классам был применен нейросетевой подход на основе архитектуры трансформер. Задача извлечения терминов из текста относится к классу NER (Named Entity Recognition): каждое отдельное слово может быть либо началом термина, либо продолжением термина, либо не термином.

В качестве нейросетевой модели была рассмотрена XLM-RoBERTa, предобученная на 2.5 терабайтах данных из корпуса CommonCrawl на 100 различных языках. Как видно из названия, данная модель относится к семейству моделей RoBERTa, впервые она была представлена в [1]. В дополнение к слоям трансформера были добавлены линейный слой и softmax для осуществления классификации токенов (Рис. 4).

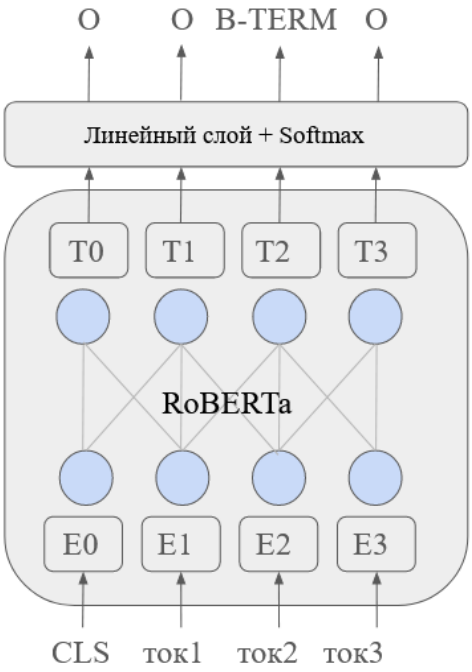

Рис. 4: Структура классификатора для извлечения терминов

Отдельно стоит отметить, что в общем случае токенизатор преобразует слово в несколько токенов, поэтому каждому токену присваивается метка класса исходного слова. Затем, когда классификатор установит метку каждого токена, метка слова определится как самая частая метка среди его токенов.

Помимо модели xlm-roberta-base в исследовательских целях рассматривались также следующие модели: roberta-base-russian-v0, предобученная на корпусе русскоязычных текстов «Тайга» [12], и ruRoberta-large, представленная среди семейства русскоязычных трансформеров в [18].

Обучающая и тестовая выборки получились путем разделения исходного набора данных в соотношении 9 к 1. Полученные результаты описаны в Таблице 1. Использование модели ruRoberta-large позволяет достичь показателя F1-меры в 91%.

Таблица 1. Значения метрик для задачи извлечения терминов

Модель	Полнота	Точность	F1-мера
roberta-base-russian-v0	0.81	0.76	0.80
xlm-roberta-base	0.86	0.83	0.85
ruRoberta-large	0.92	0.90	**0.91**

Была проведена классификация выявленных ошибок, допускаемых моделью. Примеры ошибок и их процент от совокупности всех ошибок представлены в Таблице 2. Несмотря на

то, что выделение середины термина является частным случаем ошибок, возникающих при извлечении вложенных сущностей, мы выделили ее отдельно. Это объясняется тем, что именно эта разновидность нуждается в корректировке и постобработке.

Таблица 2. Примеры ошибок разметки терминов моделью

Тип ошибки	% ошибок	Термин	Ожидаемое значение	Полученное значение
Не извлечено	41%	*классификация*	B-TERM	O
Новый термин	26,8 % (21,4%)	*модель сентиментного анализа*	O O O	B-TERM I-TERM I-TERM
Вложенная сущность	24,1%	*метод морфологического анализа*	B-TERM I-TERM I-TERM	O B-TERM I-TERM
Выделение середины термина	6,3%	*логистическая регрессия*	B-TERM I-TERM	O I-TERM
Выделение однородных членов	0,9%	*лексические и синтаксические признаки*	B-TERM O B-TERM I-TERM	B-TERM O B-TERM O
Пропуск середины термина	0,9%	*программу машинного перевода*	B-TERM I-TERM I-TERM	B-TERM O I-TERM

Анализ ошибок показал, что в 45,4%, несмотря на неточное извлечение термина моделью, эксперты отметили их как допустимые. К таким случаям относятся выделение вложенных сущностей и некоторых новых терминов (эксперты отметили их как допустимые в 21,4%случаев).

Для решения проблемы выделения середины термина в рамках постобработки возможно изменение метки I-TERM на B-TERM. Это мотивировано тем, что выделяется не вся сущность, а вложенная в нее другая сущность. Так, например, возможно выделения термина *регрессия*, поскольку *логистическая регрессия* является ее частным случаем.

Пример извлечения однородных групп “*лексические и синтаксические признаки*” показывает необходимость их постобработки и извлечения нескольких словосочетаний с общим словом(-ами): *лексические признаки* и *синтаксические признаки.*

Решение проблемы пропуска середины термина возможно объединением во время постобработки всего словосочетания от ближайшего начала (B-TERM) до последнего последовательного идущего указания на часть термина (I-TERM).

4.4. Классификация терминов

Следующим шагом была классификация извлеченных терминов. В качестве данных использовалась исходная выборка, однако вместо общих меток для терминов использовались метки конкретных классов.

Классифицировать термин в отрыве от контекста было бы некорректно, в связи с этим было принято решение передавать модели все предложение, в котором термин будет выделен специальным образом — это позволит модели сфокусировать основное внимание именно на нем. Таким образом, классифицироваться будет весь текст, который передается на вход модели. Структура классификатора представлена на Рис. 5.

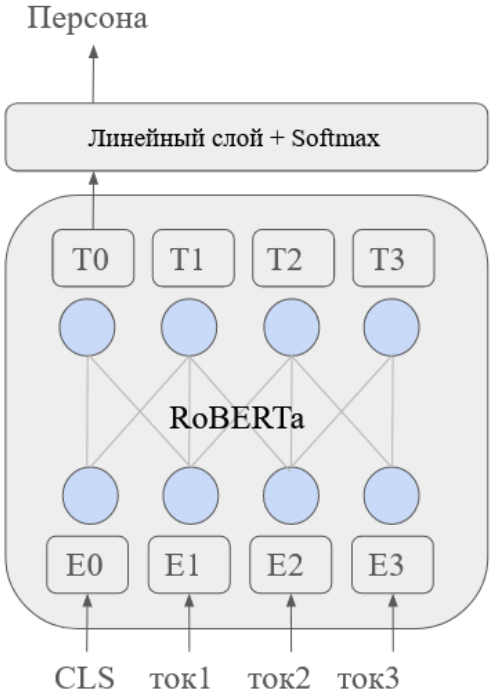

Рис. 5: Структура классификатора терминов

Для упрощения классификации некоторые похожие классы были объединены в один, так, например, классы *Приложение*, *Библиотека*, *Технология*, *Окружение* были объединены в один общий класс *R-Приложение*, поскольку они все так или иначе имеют семантику приложения. Аналогично *Набор данных* и *Корпус* были объединены в *R-Набор данных*. Итоговый список классов: *Метод, Деятельность, Объект, Персона, Задача, Организация, Модель, Метрика, Значение, Дата, Язык, R-Набор данных, R-Приложение, Раздел науки.*

В качестве модели для классификации была выбрана ruRoberta-large, показавшая лучшие результаты для задачи извлечения терминов. В результате среднее значение F1-меры составило 89%. Для каждого отдельного класса значение представлено на Рис. 6.

Для четырех классов F1-мера составила 100%, еще для шести F1-мера превысила 85%. Однако распознавание объектов класса *Деятельность* модели дается с трудом: значение составляет лишь 66%.

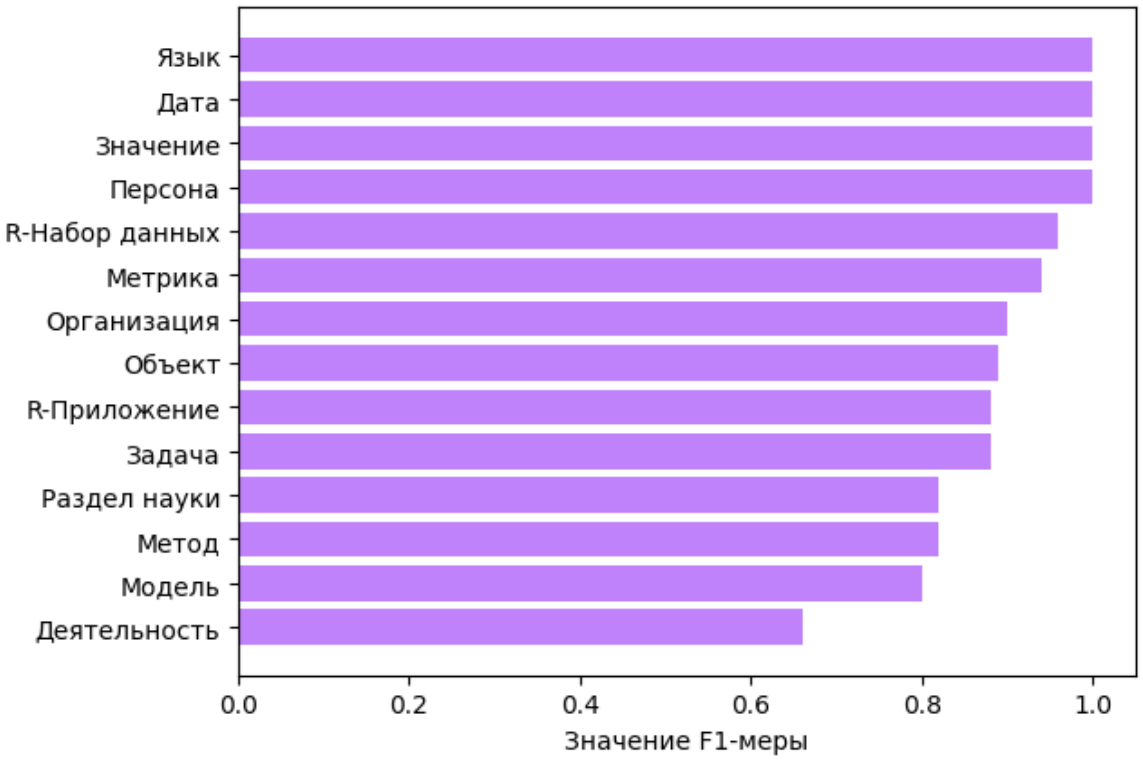

Рис. 6: Значения F1-меры для классов терминов

Анализ ошибок показал сложности для модели в различении названий организаций и названий приложений. Так, например, термин *Яндекс.Браузер* был определен как название организации, а не приложения, а термин *Майкрософт* модель напротив выделила как название приложения. Также иногда возникают сложности с тем, чтобы различать термины, которые относятся к классам Задача, Раздел науки и Деятельность. Так, например, в отрывке из текста "*Для задачи <term>моделирования</term> языка ULMFit использует <...>*" модель отнесла *моделирование* к классу Деятельность, а не классу Задача.

5. Архитектура системы пополнения онтологии

Переход от традиционных способов разработки онтологий, когда основным источникам знаний выступает эксперт, к интеллектуальным технологиям, обеспечивающих извлечение

знаний из неструктурированных источников данных, привел к возникновению технологий, в которых роль эксперта заключается в решении совсем других задач: проектировании концептуальных верхнеуровневых абстракций, разметке данных (для использования методов машинного обучения) и валидации полученных результатов. Процесс формирования онтологий на основе неструктурированного контента получил название обучение онтологий (ontology learning). Разрабатываемая система для обучения и дальнейшего пополнения онтологии включает ряд задач (Рис. 7).

Рис. 7: Основные этапы для решения задачи пополнения онтологии

На первом этапе необходимо определиться с онтологией верхнего уровня. Можно взять уже существующую [15], либо использовать паттерны онтологического проектирования для описания основных классов онтологий [2] научной области знаний: *Метод исследования, Задача, Модель обучения, Набор данных, Метрика, Приложение, Окружение, Персона,*

Научный результат, Раздел науки, Объект исследования, Предмет исследования, Публикация, Деятельность, Информационный ресурс.

Второй этап заключается в создании корпуса текстов по выбранной предметной области. На его основе проходит конкретизация онтологии на выбранную область: обобщаются и формируются новые классы. Также фиксируются фрагменты текстов, демонстрирующие примеры употребления для новых классов, отсутствующих в базовой онтологии, для которых впоследствии описываются паттерны содержания.

Следующий этап заключается в создании предметного словаря. На основе онтологических классов, атрибутов и отношений генерируется система лексико-семантических классов словаря. Для автоматического пополнения словаря предлагается использовать модели машинного обучения. В соответствии с этим необходимо создание набора данных на основе фрагмента собранного корпуса текстов. Извлеченные термины классифицируются в соответствии с лексико-семантическими классами и добавляются в словарь. Для пополнения словаря также можно использовать ресурсы близкой тематики (WikiData, тезаурусы, порталы и т.д.). Они сопоставляются друг с другом, после чего уникальные термины добавляются в словарь и снабжаются подходящими онтологическими классами.

Следующие этапы не рассматривались в рамках данной работы. На пятом этапе происходит генерация вопросов оценки компетентности (ВОК). Они могут быть использованы для извлечения отношений из текстов с использованием моделей машинного обучения, либо для извлечения грамматических ограничений для лексико-синтаксических паттернов [24].

На следующем этапе происходит генерация лексико-синтаксических паттернов для извлечения новых терминов и объектов предметной области из текста. Генерация может происходить в том числе с помощью мета-паттернов и онтологической информации [22].

На последнем этапе сгенерированные шаблоны, словарь и корпус используются для пополнения онтологии.

Заключение

Данная работа проводилась в рамках общего проекта по созданию автоматизированных методов построения онтологий [22, 14]. Особенностью рассматриваемого подхода является использование базовой онтологии, конкретизируемой на предметную область, и предметного словаря, а также пополнение словаря с использованием методов, основанных на глубоком обучении. Подход применялся для построения онтологии современных методов компьютерной лингвистики.

Применение корпусных методов анализа понятий базовой онтологии научной области знаний позволил выявить несоответствия и провести конкретизацию онтологии, в частности добавлены новые классы, связанные с методами машинного обучения. Для доказательства обоснованности изменений использовался подсчет встречаемости терминов в корпусе текстов. В результате данной работы создана онтология по КЛ, которая включает в себя 15 классов и 111 отношений. Онтология представлена в формате OWL и будет выложена в открытый доступ.

Другой особенностью является подход к созданию и пополнению предметного словаря. Система классов генерируется на основе структуры онтологии в соответствии со специализированным шаблоном. Для пополнения словаря были сопоставлены русско-английский тезаурус и портал по компьютерной лингвистике. Из их объединения были взяты уникальные термины, которым впоследствии были выведены классы. Точность и полнота автоматического соотнесения 99% и 75,9% соответственно. Количество терминов в словаре составило 2640.

В целях пополнения терминологического ядра онтологии был применен нейросетевой подход. С этой целью был создан размечен набор данных (датасет), включающий 1000 предложений из собранного корпуса текстов по компьютерной лингвистике с BIO-разметкой[1]. Рассматривались две подзадачи: извлечение терминов и их классификация. В задаче обнаружения терминов получилось достичь уровня 91% F1-меры на тестовой выборке. Анализ ошибок показал, что часто модель выделяет термин только частично, поэтому целесообразно будет проводить дальнейшие исследования с учетом этой особенности (например, провести постобработку на основе правил). Задача классификации терминов на 12 классов была осложнена небольшим размером набора данных для каждого класса, что является следствием достаточно трудоемкого процесса ручной разметки текстов. Тем не менее средний показатель F1-меры на тестовой выборке составил в среднем 89%, что является довольно неплохим результатом для начального исследования, однако в дальнейшем планируется рассмотреть различные способы улучшения качества результатов и для данного этапа.

Создаваемая терминологическая система является основой для дальнейшего построения и пополнения онтологии по компьютерной лингвистики, качество которой в значительной степени зависит от корректности выделения терминов. В качестве развития темы планируется также рассмотреть возможность извлечения названий отношений и значений атрибутов.

[1]Наборы данных доступен по ссылке: https://github.com/pasukka/NLP-Dataset.git

Список литературы

1. Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, Veselin Stoyanov. Unsupervised Cross-lingual Representation Learning at Scale, 2019. doi: https://doi.org/10.48550/arXiv.1911.02116.
2. Association for Ontology Design & Patterns. Режим доступа: http://ontologydesignpatterns.org (дата обращения: 18.09.2018).
3. Blomqvist E., Hammar K., Presutti V. Engineering Ontologies with Patterns: The eXtreme Design Methodology // Ontology Engineering with Ontology Design Patterns. Studies on the Semantic Web. IOS Press, 2016. P.23-50.
4. Fernández-López M., Gómez-Pérez A., Pazos A., Pazos J. Building a Chemical Ontology Using Methontology and the Ontology Design Environment. IEEE Intelligent Systems & their applications, 1999, 4(1). P. 37–46.
5. Gangemi A., Presutti V. Ontology Design Patterns // Handbookon Ontologies. Springer, 2009. P. 221-243.
6. Hagen Soltau, Izhak Shafran, Mingqiu Wang, Laurent El Shafey. RNN Transducers for Nested Named Entity Recognition with constraints on alignment for long sequences, 2022. doi: https://doi.org/10.48550/arXiv.2203.03543.
7. Juliao Braga, Joaquim L. R. Dias, Francisco Regateiro. A Machine Learning Ontology, 2023. doi: https://doi.org/10.31226/osf.io/rc954.
8. Lance A. Ramshaw, Mitchell P. Marcus. Text Chunking using Transformation-Based Learning, 1995. P. 82-94. arXiv:cmp-lg/9505040
9. Li J., Sun A., Han J., Li C. “A survey on deep learning for named entity recognition”, IEEE Transactions on Knowledge & Data Engineering. , 2022. Vol. 34, no. 1. P. 50-70.
10. Maynard D., Funk A., Peters W. Using lexico-syntactic ontology design patterns for ontology creation and population. In Proc. of WOP2009 collocated with ISWC2009, V. 516. pp. 39-52, CEUR-WS.org.
11. Piskorski J., Babych B., Kancheva Z., Kanishcheva O., Lebedeva M., Marcinczuk M., Nakov P., Osenova P., Pivovarova L., Pollak S., et al., “Slav-ner: The 3rd cross-lingual challenge on recognition, normalization, classification, and linking of named entities across slavic languages”, in Proceedings of the 8th Workshop on Balto-Slavic Natural Language Processing, 2021. P. 122-133.
12. Shavrina T., Shapovalova O. (2017) To the methodology of corpus construction for machine learning: «taiga» syntax tree corpus and parser. in proc. of “CORPORA2017”, international conference , Saint-Petersbourg, 2017.
13. Sure Y., Staab S., Studer R. On-To-Knowledge Methodology // Handbook on Ontologies. 2003. № 6 P.135–152.

14. Uschold M., King M. Towards a Methodology for Building Ontologies // Proceeding of the Workshop on Basic Ontological Issues in Knowledge Sharing. Montreal, Canada. 1995. P. 6.1–6.10.
15. Zagorulko Y. A. Using a System of Heterogeneous Ontology Design Patterns to Develop Ontologies of Scientific Subject Domains // Programming and Computer Software, 2020. P. 273-280.
16. Zhu F, Shen B. Combined SVM-CRFs for biological named entity recognition with maximal bidirectional squeezing, 2012. doi: https://doi.org/10.1371/journal.pone.0039230.
17. Zihan Liu, Feijun Jiang, Yuxiang Hu, Chen Shi, Pascale Fung. A Pre-trained Model for Low-Resource Entity Tagging, 2021. doi: https://doi.org/10.48550/arXiv.2112.00405.
18. Zmitrovich D., Abramov A., Kalmykov A., Tikhonova M., Taktasheva E., Astafurov D., Baushenko M., Snegirev A., Shavrina T., Markov S., Mikhailov V., Fenogenova A. A Family of Pretrained Transformer Language Models for Russian, 2023. doi: https://doi.org/10.48550/arXiv.2309.10931.
19. Большакова Е.И., Баева Н.В., Бордаченкова Е.А., Васильева Н.Э., Морозов С.С. Лексико-синтаксические шаблоны в задачах автоматической обработки текстов. Компьютерная лингвистика и интеллектуальные технологии: Труды Международной конференции Диалог'2007. М.: Издательский центр РГГУ, 2007. С. 70-75.
20. Загорулько Ю. А., Боровикова О. И., Кононенко И. С., Соколова Е. Г. Методологические аспекты разработки электронного русско-английского тезауруса по компьютерной лингвистике // Информатика и её применения. 2012. № 6(3). С. 22-31.
21. Загорулько Ю.А. Построение порталов научных знаний на основе онтологий // Вычислительные технологии, спецвыпуск 2. 2007. Т. 12.
22. Кононенко И.С., Сидорова Е.А. Методика разработки лексико-семантических паттернов для извлечения // Системная информатика. 2022. № 20. С. 25-46.
23. Лагутина Н.С., Васильев А.М., Зафиевский Д.Д. Задачи в области распознавания именованных сущностей: технологии и инструменты. Моделирование и анализ информационных систем. 2023. № 30(1). С. 64-85.
24. Овчинникова К. А. Автоматическая генерация лексико-синтаксических паттернов на основе онтологии для извлечения информации о научной деятельности // Литературоведение. Прикладная лингвистика. Языкознание: Материалы 60-й Междунар. науч. студ. конф. Новосибирск: ИПЦ НГУБ, 2022. С. 203-205.
25. Портал по компьютерной лингвистике. Режим доступа: https://uniserv.iis.nsk.su/cl (дата обращения: 07.04.23)
26. Русско-английский тезаурус по компьютерной лингвистике. Access mode: https://uniserv.iis.nsk.su/thes (дата обращения: 07.04.23)
27. Соловьев В. Д., Добров Б. В., Иванов В. В., Лукашевич Н. В. Онтологии и тезаурусы (учебное пособие). Казань, Москва, 2006. 157 с.