DOI: 10.15514/ISPRAS-2022-34(4)-13

Методы и подходы к автоматическому связыванию сущностей на русском языке

[1,2] *А.А. Мезенцева, ORCID: 0000-0003-2159-5771 <anastmezentseva@gmail.com>*
[1,2] *Е.П. Бручес, ORCID: 0000-0002-1055-5339 <bruches@bk.ru>*
[1] *Т.В. Батура, ORCID: 0000-0003-4333-7888 <tatiana.v.batura@gmail.com>*

[1] *Институт систем информатики им. А. П. Ершова СО РАН,*
630090, Россия, г. Новосибирск, пр. Лаврентьева, д. 6
[2] *Новосибирский государственный университет,*
630090, Россия, г. Новосибирск, ул. Пирогова, д. 1

Аннотация. На сегодняшний день большое внимание уделяется решению задач обработки текстов с использованием информации об окружающем нас мире, например, в информационном поиске, построении вопросно-ответных и диалоговых систем. Поэтому важно установить соответствие между сущностями в обрабатываемом тексте и базой знаний. Данная статья посвящена автоматическому связыванию сущностей с Вики-данными. В качестве сущностей рассматриваются научные термины на русском языке. Традиционно система связывания сущностей состоит из трёх этапов: распознавание сущностей, генерация кандидатов и ранжирование кандидатов. Наша система принимает на вход текст, в котором уже выделены термины. Для генерации кандидатов мы используем построковое совпадение терминов и сущностей в базе знаний. Этап ранжирования кандидатов является наиболее сложным, так как требует использования семантической информации. Проведены эксперименты с различными подходами к решению этой задачей: с использованием косинусной близости, классическими методами машинного обучения и нейронными сетями. Также мы расширили корпус RUSERRC, добавив вручную размеченные данные для обучения моделей. Полученные результаты показали, что использование метода, основанного на косинусной близости, позволяет получить не только более высокие результаты, по сравнению с другими подходами, но и решать эту задачу без вручную размеченных данных. Набор данных и код находятся в открытом доступе и доступны для других исследователей.

Ключевые слова: связывание сущностей; база знаний; научные термины

Для цитирования: Мезенцева А.А., Бручес Е.П., Батура Т.В. Методы и подходы к автоматическому связыванию сущностей на русском языке. Труды ИСП РАН, том 34, вып. 4, 2022 г., стр. 187-200. DOI: 10.15514/ISPRAS-2022-34(4)-13

Methods and techniques to automatic entity linking in Russian

[1,2] *A.A. Mezentseva, ORCID: 0000-0003-2159-5771 <anastmezentseva@gmail.com>*
[1,2] *E.P. Bruches, ORCID: 0000-0002-1055-5339 <bruches@bk.ru>*
[1] *T.V. Batura, ORCID: 0000-0003-4333-7888 <tatiana.v.batura@gmail.com>*

[1] *A. P. Ershov Institute of Informatics Systems,*
6, Acad. Lavrentjev pr., Novosibirsk 630090, Russia
[2] *Novosibirsk State University,*
st. Pirogova, ð. 1, Novosibirsk, 630090, Russia

Abstract. Nowadays, there is a growing interest in solving NLP tasks using external knowledge storage, for example, in information retrieval, question-answering systems, dialogue systems, etc. Thus it is important to establish relations between entities in the processed text and a knowledge base. This article is devoted to entity

linking, where Wikidata is used as an external knowledge base. We consider scientific terms in Russian as entities. Traditional entity linking system has three stages: entity recognition, candidates (from knowledge base) generation, and candidate ranking. Our system takes raw text with the defined terms in it as input. To generate candidates we use string match between terms in the input text and entities from Wikidata. The candidate ranking stage is the most complicated one because it requires semantic information. Several experiments for the candidate ranking stage with different models were conducted, including the approach based on cosine similarity, classical machine learning algorithms, and neural networks. Also, we extended the RUSERRC dataset, adding manually annotated data for model training. The results showed that the approach based on cosine similarity leads to better results compared to others and doesn't require manually annotated data. The dataset and system are open-sourced and available for other researchers.

Keywords: entity linking; knowledge base; scientific terms

For citation: Mezentseva A.A., Bruches E.P., Batura T.V. Methods and techniques to automatic entity linking in Russian. Trudy ISP RAN/Proc. ISP RAS, vol. 34, issue 4, 2022. pp. 187-200 (in Russian). DOI: 10.15514/ISPRAS-2022-34(4)-13

1. Введение

Сегодня в области обработки естественного языка большое внимание уделяется подходам, позволяющим использовать внешние источники знаний для решения тех или иных практических задач: вопросно-ответные системы, автоматическое реферирование, машинный перевод и пр. В качестве таких источников чаще всего используются графы знаний, содержащие информацию о сущностях, а также их атрибутах и семантических отношений между сущностями. Примерами таких графов знаний являются Freebase [1], DBpedia [2], Вики-данные.

Для использования таких структур встаёт задача автоматического связывания сущностей (Entity Linking), в рамках которой нужно соотнести выделенную сущность во входном тексте с одной из сущностей в графе знаний. Сложность задачи заключается в том, что часто одна и та же сущность может быть по-разному выражена в тексте (например, "Python", "пайтон", "питон"), а также быть многозначной (например, извлечённая сущность из текста "NLP" может быть связана с сущностями из графов знаний "Natural Language Processing", "Non-linear programming", "Neuro-lingustic programming" и пр., в зависимости от контекста). Научная новизна данной работы заключается в следующем.

- Создан набор данных для решения задачи связывания сущностей, в котором термины из научных текстов на русском языке связаны с Вики-данными, если это возможно.
- Проведено сравнение нескольких подходов к связыванию сущностей, основанных на классическом машинном обучении и моделях глубокого обучения.

2. Обзор методов

При традиционном подходе решение данной задачи разбивается на два этапа: генерация множества кандидатов из сущностей из базы знаний, а затем ранжирование выбранных кандидатов [3, 4, 5].

С активным развитием систем конечного цикла (end-to-end), стали появляться решения этой задачи без разделения на два этапа. Так, в работе [6] описывается подобная система связывания сущностей.

Другим популярным направлением становится решение задачи в постановке zero-shot learning, т.е. создание системы без обучающих примеров. Такая идея описывается в работах [7, 8].

Кроме того, важным аспектом становится решение этой задачи без привязки к конкретному языку. Для подобных исследований в статье [9] предлагается корпус для оценки таких методов, содержащий данные на 100 языках, а также оценка подходов zero-shot learning и few-shot learning, применяемых ко всем языкам. В работе [10] показано, что межъязыковая

постановка задачи более сложная, чем мультиязыковая. Авторами описан опыт создания набора данных, в котором связаны статьи из Википедии с событиями из Вики-данных. Эксперименты проводились с базовым подходом для ранжирования BM25 и моделями на основе BERT.

Также появляются работы, решающие задачу связывания сущностей не только с использованием текста, но и изображений [11]. В статье описан размеченный авторами корпус новостных статей на английском языке с различными типами сущностей. Также было показано, что задача мультимодального связывания является значительно более сложной, чем связывания только текстовой или визуальной информации, к тому же, отсутствие текстового или визуального контекста приводит к снижению метрик.

3. Формальная постановка задачи

Данная работа посвящена задаче связывания сущностей, где в качестве сущностей рассматриваются термины.

Назовём *Entities* множество сущностей и *Properties* множество свойств. База знаний состоит из множества троек вида <*Subject Predicate Object*>, где *Subject* и *Object* являются элементами множества *Entities*, а *Predicate* – элементом множества *Properties*.

Назовём токеном x_i – слово или знак препинания в тексте. Рассмотрим последовательность токенов $X = \{x_1, x_2, \dots, x_n\}$. Сущностью *Ent* будет называться подпоследовательность таких токенов, которая представляет собой термин. Тогда мощность множества *E*, которое содержит в себе сущности *Ent*, всегда меньше либо равна мощности множества *X*, включая значение 0.

Задача автоматического связывания сущности состоит в построении такой функции *F*, которая для каждой сущности из множества *E* ставила бы в соответствие элемент из множества *Entities*, либо ε, где ε – отсутствие сущности в заданной базе знаний:

$$F: E \to Entities \cup \varepsilon.$$

4. Описание данных

4.1 Создание вручную размеченного корпуса

В открытом доступе существует корпус из 1680 научных статей в области информационных технологий на русском языке RuSERRC, в 80 текстах вручную размечены термины, отношения между ними, указаны ссылки на сущности из Викиданных [12]. Этот набор данных применялся только для тестирования системы и никак не использовался на этапе обучения моделей.

Табл. 1. Статистика по вручную размеченного корпуса
Table 1. Manually annotated set statistics

Параметр	Количество
Тексты	136
Токены	12809
Термины	2028
Ссылки на сущности из Вики-	938

Для обучения моделей нам потребовался дополнительный вручную размеченный корпус. Чтобы получить его, были выполнены следующие шаги:

1) выгружены тексты аннотаций научных статей по информационным технологиям на русском языке из журнала «Программные продукты и системы»;
2) произведена авторазметка имеющимся модулем [13], где ранжирование реализовано с помощью косинусного расстояния между векторами;
3) исправлены ошибки авторазметки, а именно, для каждой определенной системой из п.2 сущности определялось, подходит ли название и описание к термину, упомянутому в тексте, и если не подходит, то нужно предложить верную сущность из базы знаний.

Корпус находится в открытом доступе, его статистические параметры указаны в табл. 1.

4.2 Подготовка данных для проведения экспериментов

В данной статье задача выбора наиболее релевантной сущности рассматривается как бинарная классификация. Для обучения бинарного классификатора необходим набор данных, в котором содержались бы позитивные (1) и негативные (0) примеры. Позитивные примеры были взяты из размеченного корпуса, описанного выше. Неподходящие кандидаты (“нули”) получились в результате применения существующего модуля генерации (подробнее в разделе 4), исключением уже полученных кандидатов из корпуса. Различную статистическую информацию о полученном корпусе можно найти в табл. 2.

Табл. 2. Статистика по корпусу для обучения моделей
Table 2. Training set statistics

Параметр	Количество
Экземпляры класса 1	637
Экземпляры класса 0	23552
Уникальные термины	1645
Минимальная длина контекста	6
Максимальная длина контекста	21
Средняя длина контекста	13
Идентификаторы из Вики-данных	24189
Уникальные идентификаторы	6586

5. Описание системы

В ходе исследования была разработана система для связывания сущностей. В общем виде алгоритм состоит из следующих этапов.

1) На вход подается термин и его контекст (пять токенов до и после). Выполняется лемматизация без учёта контекста (все слова приводятся в нижний регистр).
2) Генерируются запросы, чтобы получить список униграмм, биграмм и триграмм из названия входного упоминания.
3) Генерируется список кандидатов для выполнения поиска по построковому совпадению с основным и альтернативными названиями сущности и в дампе Вики-данных.
4) Выполняется ранжирование кандидатов с использованием алгоритма получения коэффициентов для оценки того, насколько кандидат является подходящим. Также были проведены эксперименты с использованием дополнительных весов, для того чтобы учесть количество совпадающих токенов в кандидате и в упоминании сущности.

5) На выходе алгоритм выдает наиболее релевантный идентификатор сущности из Викиданных.

Иллюстрация взаимодействия модулей системы представлена на рис. 1.

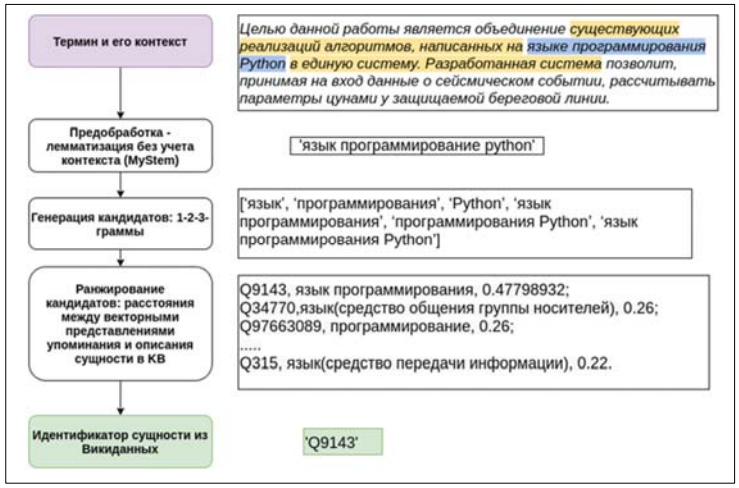

Рис. 1. Схема работы системы
Fig. 1. General architecture of the system

6. Эксперименты

6.1 Базовый подход

В качестве базового алгоритма ранжирования было использовано косинусное расстояние между векторами, полученными из предобученной модели DeepPavlov. Для кандидатов, не имеющих подходящей сущности в базе знаний, был введен порог, равный 0.35. Первый вектор – потокенно усредненный для контекста входного упоминания, состоящий из n токенов до и после, где $n = 5$. Второй вектор – так же усредненный для названия, описания и синонимов сущности-кандидата из базы знаний. Результаты для этого подхода представлены в первой строке табл. 5.

6.2 Генерация векторов признаков

Модели машинного обучения принимают на вход вектора признаков. Для разных моделей использовались разные способы получения этих векторов.

Для моделей классического машинного обучения и перцептрона использовалась конкатенация усреднённого вектора входного термина (и его контекста), а также названия (и описания) сущности-кандидата из базы знаний. На рис. 2 представлена общая схема получения векторов.

Пусть A – последовательность токенов;

$len(A)$ – количество элементов последовательности A;

$FastText(w)$ – вектор для токена w, взятый из предобученной модели FastText.

Тогда для последовательности A потокенно усредненный вектор будет вычисляться по формуле:

$$WordMean(A) = \sum_{i=1}^{len(A)} \frac{FastText(A[i])}{len(A)}.$$

В этом случае векторы признаков, подаваемые на вход модели машинного обучения, будут вычисляться по формуле:

$$features = WordMean(X) \cdot WordMean(Y), \text{ где}$$

$\cdot$ – операция конкатенации;

X – последовательность токенов термина и контекста;

Y – название и описание сущности-кандидата из базы знаний.

Затем к конкатенации были добавлены общий усреднённый вектор данных векторов или косинусное расстояние между данными векторами, т.е.

$$features_{mean} = features \cdot \frac{WordMean(X) + WordMean(Y)}{2};$$

$$features_{cosine} = features \cdot cosine_{distance}\big(WordMean(X), WordMean(Y)\big), \text{ где}$$

$cosine_{distance}(X, Y)$ – функция определения косинусного расстояния между векторами.

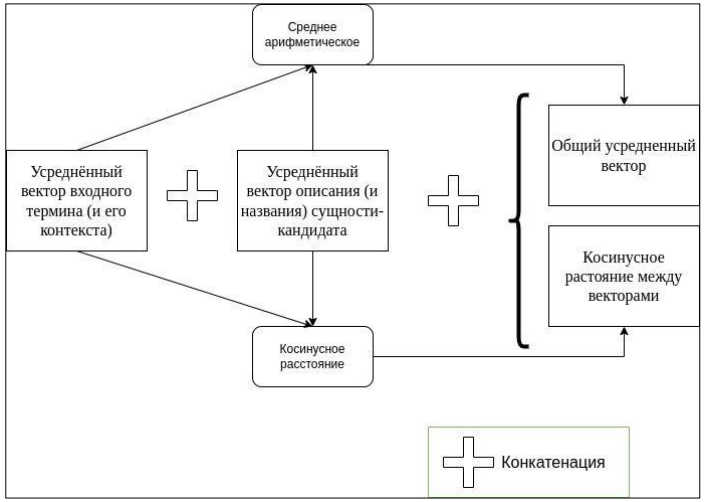

Рис. 2. Схема генерации векторов признаков
Fig. 2. Feature vector generation scheme

Для моделей со сверточными слоями на входе использовалась матрица фиксированного размера – была взята максимальная длина в 60 токенов. В векторы преобразовывалась следующая последовательность: термин, контекст, описание кандидата и альтернативные названия, т.е.

$M = [FatText(X[1]), FastText(X[2]), \dots, FastText(Y[i-1]), FastText(Y[i])], len(M) = 60.$

6.3 Описание процесса обучения и тестирование модели

Шаги для обучения модели выполнялись следующие.

1) Загрузка данных в следующий формат – список кортежей. Каждый кортеж состоит из двух элементов: первый – это контекст упоминания в тексте и описание сущности из базы знаний; второй – метки классов (0 или 1).
2) Получение векторов признаков одним из способов, описанных в разделе 6.2;

3) Разделение выборки на обучающую (80%) и валидационную (20%) с применением стратификации – статистический метод, который влияет на то, чтобы в каждой выборке содержался примерно одинаковый процент примеров каждого класса. Для стратификации использовался метод train_test_split (с параметром stratify) из библиотеки Scikit-Learn.
4) Классы в полученном наборе данных (который описан в разделе 4.2) несбалансированные. Поэтому в экспериментах применялись различные методы для решения этой проблемы:
   a) в случае с CatBoostClassifier указывался параметр auto_class_weights='Balanced';
   b) для моделей глубокого обучения функция class_weight из библиотеки Scikit-Learn.
5) Чтобы модели не переобучались, применялся ранний останов, т.е. максимум через 10 эпох обучение останавливалось, если значение функции потерь на валидационном множестве не уменьшалось.

Общая схема обучения модели представлена на рис. 3, а применение в модуле ранжирования – на рис. 4.

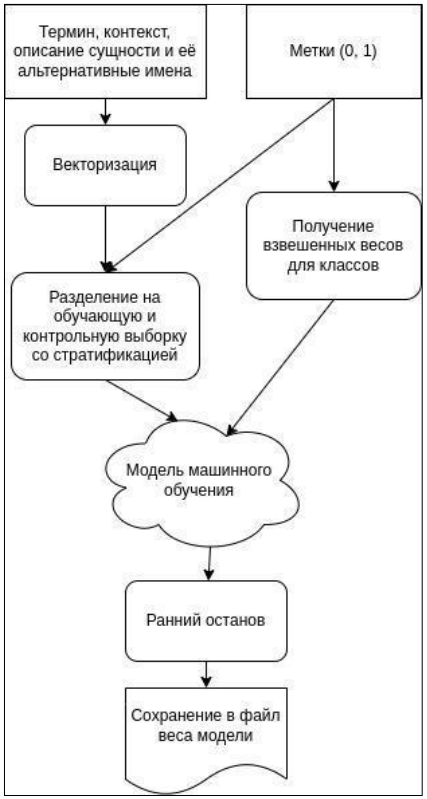

Рис. 3. Схема обучения модели
Fig. 3. Model training scheme

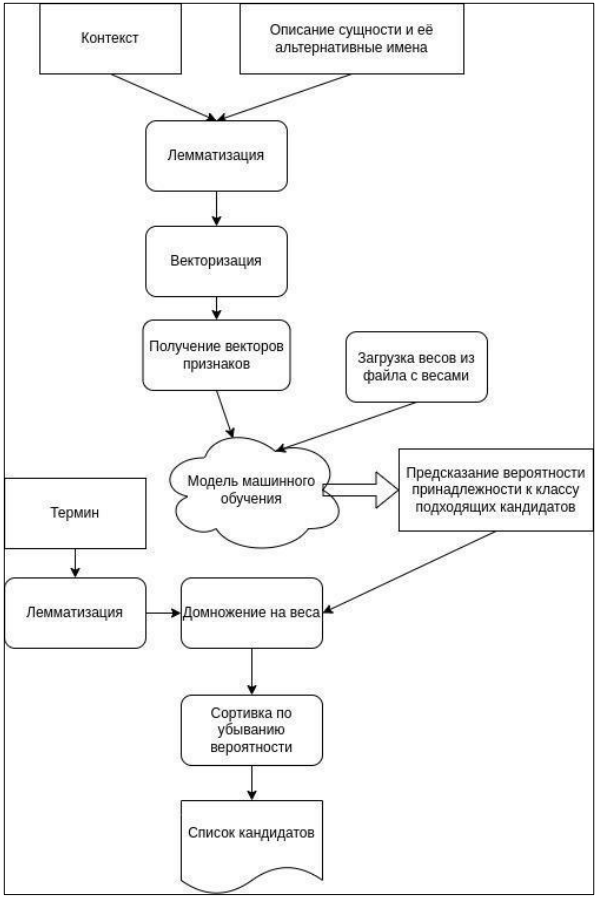

Рис. 4. Схема применения модели в модуле ранжирования кандидатов
Fig. 4. Scheme of applying the model in the candidate ranking module

6.4 Модели классического машинного обучения

Среди моделей классического машинного обучения нами были исследованы два базовых подхода: *логистическая регрессия* и *градиентный бустинг*. В качестве реализации для *логистической регрессии* использовался класс LogisticRegression из библиотеки scikit-learn. На вход модели подавались конкатенации векторов (более подробное описание приведено в разделе 6.2). К сожалению, метрика linked accuracy на тестовых данных составила чуть больше одного процента, поэтому в итоговые результаты данный эксперимент не вошёл. В качестве реализации для *градиентного бустинга* [14] был выбран CatBoostClassifier. Были проведены эксперименты с различными параметрами, но лучше всего подошли те, которые были выставлены по умолчанию. Существенные настройки следующие: ранний останов – 10, балансировка весов, валидационная выборка задана, максимальное количество деревьев – 500, глубина дерева – 6, все деревья одного размера.

6.5 Модели глубокого обучения

В качестве векторов признаков в ниже описанных моделях использовалась только конкатенация векторов без каких-либо модификаций. Для реализации использовался модуль Keras из фреймворка Tensorflow. Эксперименты проводились с несколькими архитектурами:

I. Многослойный персептрон в нескольких модификациях, гиперпараметры которых указаны в табл. 3.

Табл. 3. Эксперименты с архитектурой многослойного перцептрона
Table 3. Experiments with the architecture of a multilayer perceptron

Гиперпараметры	**Эксперимент 1**	**Эксперимент 2**	**Эксперимент 3**
Архитектура сети	два полносвязных слоя – 600 и 300; выходной – один нейрон	два полносвязных слоя – 600 и 200; выходной – два нейрона	три полносвязных слоя – 600, 400, 400; выходной – два нейрона
Функция активации	ReLu		
Функция активации на выходном слое	сигмоида	softmax	
Шаг обучения	0.00003		0.00002
Функция потерь	binary_crossentropy	categorical_crossentropy	
Размер батча	24		
patience	10		

Шаг обучения эмпирически подбирался на первом эксперименте и дальше изменялся, если модель не начинала обучаться. Для второго и третьего эксперимента метки в датасете преобразовывались следующим образом: 0 →[1,0]; 1→[0,1].

II. Сверточная нейронная сеть.

В качестве реализации использовался Conv1D слой из библиотеки Keras. Общие параметры для всех экспериментов: шаг обучения – 0.00003; размер пулинг слоя – 5; 2 нейрона на выходном слое с активационной функцией softmax.

Отличие трёх экспериментов отражено в табл. 4, а именно, количество и размерность окон для сверточных слоёв, входной размер которых (60, 300) с активационной функцией ReLu.

Табл. 4. Размерность карт признаков
Table 4. Feature maps' dimension

Параметр	**Эксперимент 1**	**Эксперимент 2**	**Эксперимент 3**
Размерность карты признаков	100, 5	100,5 - 100,5	200,5 - 200,5

III. Классификатор с двумя входами (упоминание, описание сущности).

На вход классификатору подаются два вектора: 1) упоминание и контекст; 2) описание сущности из базы знаний. Вектора признаков получены тем же способом, что и для сверточной сети. Затем каждый из векторов подается на вход модели следующей архитектуры: два сверточных слоя (карта признаков - (200,5)), после каждого из которых слой

пулинга. После этого выходы для каждого из входов конкатенируются друг с другом и подаются на вход следующей модели, которая уже состоит из двух полносвязных слоёв: 1) двадцать нейронов и relu активация, 2) один нейрон и сигмоида.

При проведении эксперимента модели подавались по 50 примеров за одну итерацию обучения.

IV. Классификатор с тремя входами (термин, контекст, описание сущности).

Основные отличия от классификатора с двумя входами в следующем:

1) вместо двух имеется три входа: 1) упоминание, 2) контекст упоминания, 3) описание сущности; вектор упоминания был вынесен в отдельный вход, исходя из предположения, что если его подавать вместе с контекстом, то он может стать менее четкой информация о том, какой именно термин необходимо связать с сущностью;
2) количество нейронов на предпоследнем слое берется равным 100.

При проведении эксперимента модели подавались по 100 примеров за одну итерацию обучения.

Для подходов, описанных выше, чтобы улучшить метрики на тестовых данных, был применен метод домножения на веса. Гипотеза состоит в том, что важно учитывать то, какое количество токенов в кандидате совпадает с теми, из которых состоит упоминание. Значения весов вычислялись по формуле:

$$weight = \frac{n_{matching}}{n_{all}},$$

$n_{matching}$ – количество совпадающих токенов;

n_{all} – количество токенов во входной сущности.

7. Результаты

Для оценки подходов к ранжированию использовалась точность, которая определяется как отношение количества верно связанных терминов ко всем терминам. Так как нам удалось связать не все термины в корпусе, информативнее будет разделить эту метрику на две: *accuracy* – принимает во внимание все сущности, и *linked_accuracy* – считается только на том наборе терминов, для которых нашлась сущность в графе знаний в корпусе. Таким образом, *accuracy* вычисляется по формуле:

$$accuracy = n\_correct\_entities/n\_all\_entities, \text{ где}$$

n_correct_entities – количество верно связанных терминов;

n_all_entities – количество всех терминов в корпусе.

Обозначим через *n_all_linked_entities* количество всех терминов в корпусе, которые имеют связь с сущностью в Викиданных. Тогда *linked_accuracy* вычисляется по формуле:

$$linked\_accuracy = n\_correct\_linked\_entities \;/\; n\_all\_linked\_entities, \text{ где}$$

n_correct_linked_entities – количество верно связанных терминов среди всех связанных терминов.

Итоговые результаты для основных подходов перечислены в табл. 5.

Значение метрик приведено в двух вариантах – с весами и без них. Не применяя подхода к взвешиванию, только CatBoostClassifier, где признаки подсчитывались по формуле для $features_{cosine}$ (см. подраздел 6.2), удалось добиться лучшего значения 0.28 метрики *linked_accuracy*. Также можно заметить, что классические подходы и нейросетевые находятся примерно в одном диапазоне значений: *accuracy* – от 15 до 17, *linked_accuracy* – от 22 до 28 процентов.

Табл. 5. Результаты тестирования методов
Table 5. Experimental results

Подход		Метрики без весов		Метрики с весами	
		accuracy	linked_accuracy	accuracy	linked_accuracy
Косинусное расстояние		0.38	0.22	0.55	0.54
Catboost + конкатенация		0.16	0.25	0.20	0.35
Catboost конкатенация косинусное		0.17	0.28	0.23	0.43
Catboost конкатенация среднее		0.15	0.23	0.20	0.36
Перцептрон	Эксперимент 1	0.14	0.20	0.16	0.24
	Эксперимент 2	0.16	0.24	0.17	0.28
	Эксперимент 3	0.15	0.22	0.16	0.25
CNN	Эксперимент 1	0.17	0.28	0.19	0.32
	Эксперимент 2	0.16	0.25	0.18	0.29
	Эксперимент 3	0.17	0.28	0.19	0.32
Классификатор с двумя входами		0.15	0.22	0.15	0.23
Классификатор с тремя входами		0.15	0.22	0.15	0.23

Домножение на веса показало себя неплохо, так как позволило увеличить значения для каждого из подходов. Но в то же время ни один из методов не смог превзойти базовый метод, основанный на косинусном расстоянии между векторами. Полученные результаты показали, что подход, учитывающий косинусное расстояние и количество совпадающих токенов, дает лучшую точность по сравнению с другими.

Мы предполагаем, что это может быть связано с несколькими причинами. Во-первых, сущности в базе знаний не всегда содержат полную и точную информацию, что может привести к неверным предсказаниям моделей. Во-вторых, в данной работе используются статические векторы слов, которые плохо справляются с проблемами омонимии, а в данной задаче эта проблема является основной. Мы предполагаем, что контекстуальные векторы (такие, как ELMo [15] или BERT [16]) справятся с данной задачей лучше. Кроме того, кажется важным включить информацию не только о локальном контексте вокруг термина, но и о более глобальном, например, общей тематике статьи и пр.

В качестве ориентира для сравнения полученных метрик применялись данные из статьи [17], где авторы рассматривают задачу связывание сущностей в мультиязычном аспекте. В частности, они связывали элементы Вики-данных, используя дополнительно имена соответствующих статей из Википедии на разных языках. На датасете Wikinews авторы получили точность, равную 0.66 для текстов с общеупотребимой лексикой на русском языке.

8. Заключение

В данной работе представлена система автоматического связывания сущностей с внешней базой знаний, где в качестве сущностей выступают научные термины, и связываются они с сущностями из Викиданных.

Особое внимание уделено этапу ранжирования как наиболее сложному во всей системе, т.к. необходимо установить семантическую связь между двумя сущностями, которые могут быть представлены в тексте в разном виде, быть синонимичными или омонимичными. Задача ранжирования решалась как задача бинарной классификации: требовалось установить соответствие между термином из текста и сущностью из базы знаний. Были проведены эксперименты с различными архитектурами машинного обучения: логистической регрессией, градиентным бустингом, разными видами нейронных сетей.

Для генерации признаков для моделей были использованы статические векторы FastText, но мы предполагаем, что для данной задачи лучше подойдут контекстные векторные представления. Эксперименты такого типа являются одним из наиболее приоритетных направлений дальнейшей работы.

Список литературы / References

[1]. Bollacker K., Evans C. et al. Freebase: A collaboratively created graph database for structuring human knowledge. In Proc. of the 2008 ACM SIGMOD International Conference on Management of Data, 2008, pp. 1247-1249.

[2]. Auer S., Bizer C. et al. DBpedia: A Nucleus for a Web of Open Data. Proceedings of the 6th International Semantic Web Conference (ISWC). Lecture Notes in Computer Science, vol. 4825, 2007, pp. 722-735.

[3]. Bunescu R., Pąsca M. Using encyclopedic knowledge for named entity disambiguation. In Proc. of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL), 2006, pp. 9-16.

[4]. Cucerzan S. Large-scale named entity disambiguation based on Wikipedia data. In Proc. of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), 2007. pp. 708-716.

[5]. Ratinov L., Roth D. et al. Local and global algorithms for disambiguation to Wikipedia. In Proc. of the 49th Annual Meeting of the Association for Computational Linguistics, 2011. pp. 1375-1384.

[6]. Kolitsas N., Ganea O., Hofmann T. End-to-end neural entity linking. In Proc. of the 22nd Conference on Computational Natural Language Learning, 2018, pp. 519-529.

[7]. Logeswaran L., Chang M. et al. Zero-Shot Entity Linking by Reading Entity Descriptions. In Proc. of the 57th Annual Meeting of the Association for Computational Linguistics, 2019, pp. 3449-3460.

[8]. Zhang S., Cheng H. et al. Knowledge-rich self-supervision for biomedical entity linking. arXiv:2112.07887, 2021, 13 p.

[9]. Botha J., Shan Z., Gillick D. Entity linking in 100 languages. In Proc. of the 2020 Conference on Empirical Methods in Natural Language Processing, 2020, pp. 7833-7845.

[10]. Pratapa A., Gupta R., Mitamura T. Multilingual event linking to wikidata. In Proc. of the Workshop on Multilingual Information Access (MIA), 2022, pp. 37-58.

[11]. Wang X., Tian J. et al. WikiDiverse: A Multimodal Entity Linking Dataset with Diversified Contextual Topics and Entity Types. In Proc. of the 60th Annual Meeting of the Association for Computational Linguistics, vol. 1, 2022, pp. 4785-4797.

[12]. Bruches E., Pauls A. et al. Entity Recognition and Relation Extraction from Scientific and Technical Texts in Russian. In Proc. of the Science and Artificial Intelligence Conference (S.A.I.ence 2020), 2020, pp. 41-45.

[13]. Bruches E., Mezentseva A., Batura T. A System for Information Extraction from Scientific Texts in Russian. Communications in Computer and Information Science, vol. 1620, 2022, pp. 234-245.

[14]. Dorogush A., Gulin A. et al. Fighting biases with dynamic boosting. arXiv:2011.09817, 2017, 5 p.

[15]. Peters M., Neumann M. et al. Deep contextualized word representations. In Proc. of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics, vol. 1, 2018, pp. 2227-2237.

[16]. Devlin J., Chang M. et al. Bert: Pre-training of deep bidirectional transformers for language understanding. In Proc. of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL), vol. 1, 2019. pp. 4171-4186.

[17]. De Cao N., Wu L. et al. Multilingual autoregressive entity linking. Transactions of the Association for Computational Linguistics (TACL), 2022. vol. 10, pp. 274-290.

Информация об авторах / Information about authors

Анастасия Алексеевна МЕЗЕНЦЕВА – студентка НГУ, программист 1-ой категории ИСИ СО РАН. Сфера научных интересов: связывание сущностей, вопросно-ответные системы.

Anastasia Alekseevna MEZENTSEVA – student, NSU, first grade programmer IIS SB RAS. Research interests: entity linking, question-answering systems.

Елена Павловна БРУЧЕС – кандидат технических наук, младший научный сотрудник ИСИ СО РАН, старший преподаватель НГУ. Сфера научных интересов: извлечение информации, распознавание именованных сущностей, создание онтологий.

Elena Pavlovna BRUCHES – PhD in Technical Sciences, Junior Researcher, IIS SB RAS, Senior Lecturer at NSU. Research interests: information extraction, named entity recognition, ontology creation.

Татьяна Викторона БАТУРА – кандидат физико-математических наук, доцент, старший научный сотрудник ИСИ СО РАН. Сфера научных интересов: компьютерная лингвистика, автореферирование, извлечение информации, пополнение графов знаний.

Tatiana Viktorovna BATURA – PhD in Physics and Mathematics, Associate Professor, Senior Researcher, IIS SB RAS. Research interests: computational linguistics, summarization, information extraction, knowledge graph completion.