Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Системы обработки информации и управления

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОМУ ПРОЕКТУ

НА ТЕМУ:

«Исследование методов машинного мультимодального обучения на текстовых данных и графах знаний»

Студент ИУ5-33М _____ Д.Р. Громоздов
(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель _____ Ю.Е. Гапанюк
(Подпись, дата) (И.О.Фамилия)

Консультант _____ _____
(Подпись, дата) (И.О.Фамилия)

2022 г.

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ

Заведующий кафедрой _____
(Индекс)

_____ _____
(И.О.Фамилия)

« _____ » _____ 20 ____ г.

З А Д А Н И Е
на выполнение курсового проекта

по теме « Исследование методов машинного мультимодального обучения на текстовых данных и графах знаний» _____

_____

Студент группы __ИУ5-33М__

_____Громоздов Данила Романович_____
(Фамилия, имя, отчество)

Направленность НИР (учебная, исследовательская, практическая, производственная, др.)

__исследовательская_____

Источник тематики (кафедра, предприятие, НИР) _____НИР_____

График выполнения НИР: 25% к _4_ нед., 50% к _8_ нед., 75% к 12 нед., 100% к 16_ нед.

Техническое задание: Изучить литературу, содержащую информацию по теме ВКРМ. Составить на её основе краткий обзор методов, применяемых при решении задач мультимодального обучения на графах знаний._____

Оформление научно-исследовательской работы:

Расчетно-пояснительная записка на _ _ листах формата А4.
Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)

_____

_____

Дата выдачи задания « _01_ » __сентября_____ 2022_ г.

Руководитель НИР _____ __Ю.Е. Гапанюк_
(Подпись, дата) (И.О.Фамилия)

Студент _____ __Д.Р. Громоздов
(Подпись, дата) (И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

Оглавление

Методы работы с графами знаний

Графы знаний – это база знаний, использующая топологию графа для отображения взаимосвязей между сущностями и явлениями. При этом связанные элементы представляют собой вершины графа, а поименованные рёбра отвечают за описание семантики этой взаимосвязи [6]. В таких графах в качестве вершин могут выступать данные различных модальностей (т. е. разных типов представления информации, или способов получения данных от сенсоров: текст, изображения (в частности, цвет, яркость, глубина), дата-время, координаты и другие) [4] [10]. Таким образом мы потенциально можем описать явления и их связи с разных сторон, в зависимости от имеющихся в распоряжении данных и выявленных в них закономерностей. Мультимодальное обучение – разновидность машинного обучения, в которой в качестве входных данных могут использоваться данные, представленные разными модальностями. При хорошем построении архитектуры нейронной сети использование графов знаний может позволить выявить больше характерных признаков, а следовательно, возможно улучшить её работу и возможности. Возможна обработка не только конкретного объекта, но и контекста, в котором он находится.

Для обучения на графовых представлениях используется вложение (отображение, embedding) его элементов (вершин и рёбер) в векторы или матрицы. Такой подход называется Вложением графа знаний (Knowledge graph embedding / KGE). На имеющихся данных изучаются возможные связи между сущностями, находящимися в вершинах этого графа, что позволяет достраивать неполные графы знаний, а также производить классификацию объектов, основываясь на доступных в графе признаках объектов и их взаимосвязях. Мультимодальное вложение графов знаний позволяет производить извлечение связей из графа, основываясь на информации, представленной несколькими модальностями одновременно. Данный подход, однако, требует приведения

признаков всех модальностей (а для некоторых моделей и связей между ними) к общему пространству представлений для графа.

Графы знаний позволяют не просто использовать для анализа несколько модальностей, но также и отражать семантические взаимосвязи между ними. Если обычные модели мультимодального обучения получают на вход информацию разных типов раздельно, то в случае использования графового представления данных информация будет передаваться в виде векторов, содержащих семантику взаимосвязи между объектами различной модальности.

Представление используемой для обучения информации в виде графа знаний можно рассматривать как один из этапов предобработки данных. Выявляются семантические зависимости между элементами, принадлежащими разным типам представления информации, таким образом эффективно расширяется признаковое пространство и упорядочиваются данные.

Области применения моделей

- Предсказание связи между данной сущностью и имеющейся в графовом представлении данных.
- Задачи классификации. В том числе тройная классификация: предсказание правдивости (имеет ли место) данной направленной связи между двумя сущностями. Их часто используют в медицинской сфере, так же сейчас развивается использование данного подхода для классификации реальных новостей и фейковых.
- Разрешение двоякого представления узла графа. Определение одинаковых по сути сущностей, которые могут быть записаны в немного разном виде, что исключает повторы данных.
- Задачи кластеризации так же входят в возможные области применения.

Данные модели могут быть полезны и для фундаментальной науки: развивается идея создания графа знаний, содержащего информацию об имеющихся статьях. Такой граф можно использовать для решения задачи более

персонализированного подбора статей, использующего не только заголовки и ключевые слова, но и семантику компонентов непосредственного содержания статьи.

Общая схема эмбеддинга мультимодального графа знаний

В данном разделе будет рассмотрена общая схема обработки мультимодальных графов знаний в ходе процесса обучения (рис. 1).

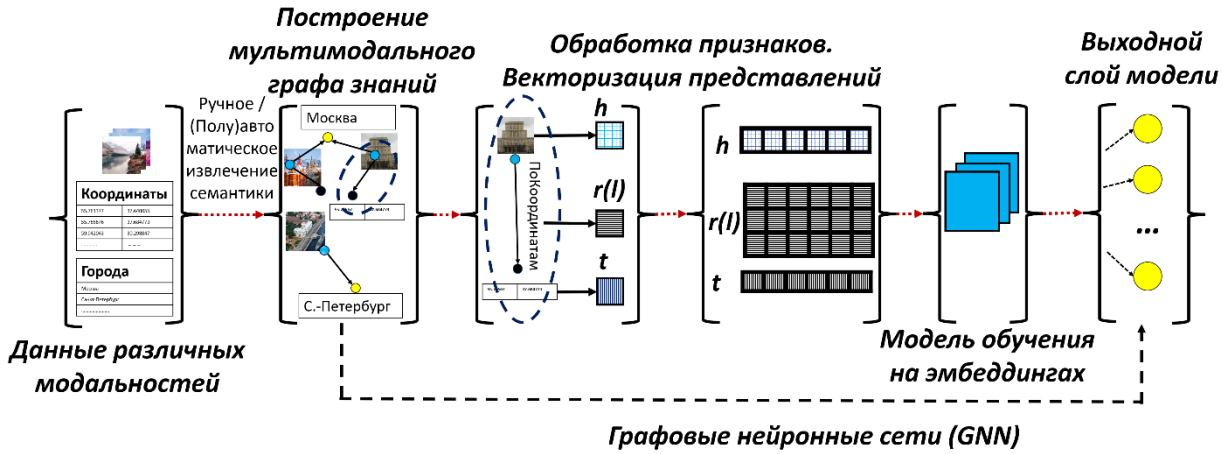


Рис. 1. Общая схема работы с мультимодальными графами знаний.

В начале берётся набор данных (датасет), содержащий информацию различных типов. На основе него извлекаются семантические связи между входящими элементами. Извлечение может осуществляться как вручную – связи полностью определяются человеком на основе знаний о предметной области, так и автоматически или полуавтоматически с использованием алгоритмов анализа и правил. Сущности и семантические связи между ними формируют граф знаний. При этом, по большей части рёбра в таких графах имеют направленный характер из-за особенностей работы с семантическими связями (как видно на примерах *«МГТУ им. Баумана – НаходитсяВ – Москва»* и *«Москва – НаходитсяВ – МГТУ им. Баумана»*, они обозначают не одно и то же!).

Далее связи внутри графа знаний представляются в виде триплетов вида (*h*, *r*, *t*), встречается так же вариант (*h*, *l*, *t*), где *h* и *t* – вершины графа, а *r* (или *l*) –

семантическая связь между ними. При этом учитывается направленность связи поскольку *h* – это «головная» (head) сущность, а *t* – «хвостовая» (tail) сущность. Субъектом связи является вершина *h*, а её объектом, соответственно, вершина графа *t*. Возвращаясь к предыдущему примеру, *«МГТУ им. Баумана»* было бы вложено в представление *h*, *«НаходитсяВ»* вкладывалось бы в связь *r*, *«Москва»*, объект связи, вложилась бы в представление *h*.

На данном этапе как раз критична разная модальность данных в графе, поскольку все вершины для работы модели должны находиться в одном пространстве представлений (в некоторых моделях также и связи) . Соответственно, для них применяются различного рода методы предобработки данных. Для обработки изображений может быть предварительно использована, например, свёрточная нейронная сеть , извлекающая признаки из изображения, которые затем могут быть вложены в пространство представлений сущности h или t.

Далее данные, представленные в виде (*h, r, t*) передаются на вход в модель обучения. При этом для каждого факта (*h, r, t*) вычисляется его оценочная (score) функция. Она представляет собой оценку вероятности того, что данный факт имеет место . Процесс обучение же заключается в максимизации общего значения этой функции (для верных фактов значение выше, для ложных утверждений ниже). Для перевода взаимодействия сущностей в связи используются разные модели, в том числе линейные архитектуры, рекуррентные нейронные сети и другие.

Выход будет соответствовать решаемой задаче: соответствующий второй элемент графа для предсказания связи; верность или неверность утверждения о том, что данный тип связи может быть установлен для взятых двух сущностей и отношение элемента к конкретной семантической категории для задач классификации; удаление или не удаление возможного дубликата узла графа.

Оценочные функции

Как говорилось выше, оценочная (score) функция используется в моделях вложения графов знаний для оценки вероятности существования связи между некоторыми двумя сущностями. Такие функции можно разделить на два типа: основанные на оценке расстояний и на оценке сходства.

Первые оценивают расстояние между вершинами, сущностями, в графе, и на основе него предоставляют вероятность существования факта. К таковым относится, например, широко используемая модель TransE, KG2E, ManifoldE и другие.

Вторые дают оценку вероятности правильности факта на основании семантического сходства. К моделям, использующим оценочные функции данного семейства относят HolE, DistMult.

Модели обучения

Для работы с графовыми представлениями данных могут использоваться различные модели, переводящие взаимосвязи между сущностями в связи (r). Это могут быть линейные модели, нейронные сети, графовые нейронные сети. Графовые нейронные сети особенно интересны, поскольку они используются конкретно в работе с графами, а также часто встречаются в работах по мультимодальному обучению на графах знаний. На выходе такие модели дают векторизованные представления вершин и рёбер графа.

Для задач, решаемых на мультимодальных данных, могут использоваться как уже имеющиеся модели обучения с соответствующей предобработкой данных разных модальностей для вложения их в векторное пространство сущностей, так и новые модели, которые используют разные архитектуры для разных модальностей. Например, модель REMAP, предложенная в статье [6] отдельно работает со связями в графах и с семантикой текста (граф и текст представляют собой две модальности), используя разные модели обучения и общую функцию потерь, представляющую собой линейную вариацию функций потерь каждой из моделей. А в статье [10] модель вложения графов в векторное

представление – TransE, адаптируется под использование на мультимодальном графе знаний.

Модели обычно либо расширяют число возможных модальностей для сетей, используемых на графах (например, использую различные методы предобработки данных на модальностях, для приведения их к общему векторному пространству), либо могут дополнять принципы архитектуры мультимодального обучения использованием данных, представленных графом знаний.

Модели с ранним слиянием модальностей (early fusion).

Раннее слияние модальностей происходит до их входа в мультимодальную модель. В таком случае после объединения массивов данных в алгоритм обучения будет поступать один вектор характеристик, включающий в себя связанные между собой записи из разных модальностей.

В наиболее простом случае слияние происходит за счёт конкатенации матриц с данными модальностей. Такой тип объединения подходит не для всех задач и может быть чувствителен к нехватке данных в модальностях.

Часто для задач, связанных с происходящими во времени процессами, используют предварительное извлечение признаков из сырых данных, содержащихся в модальностях. Уже сами полученные признаки объединяются в общий вектор и становятся входными данными. Таким образом можно добиться большей синхронизации поступления данных в мультимодальную модель, а также несколько уменьшить размер входного вектора. Так же уменьшается чувствительность слияния к нехватке данных.

Этот тип слияния используется во многих областях. Достаточно часто используется в моделях, связанных с распознаванием движений и в системах автопилота. При использовании извлечения признаков, раннее слияние позволяет эффективно и быстро обрабатывать информацию, поступающую от сенсоров в виде одного вектора признаков. Модель мультимодального обучения работает одновременно со всеми модальностями.

Позднее слияние (late fusion).

Объединение данных из модальностей происходит только после их изучения собственными моделями (могут быть одинаковыми для всех модальностей, а могут быть и разными, чаще всего архитектура подбирается в соответствии с типом данных и необходимым результатом обучения). Слияние происходит уже между представлениями данных модальностей.

Данный тип слияния позволяет работать с сильно не коррелирующими между собой модальностями. В таком случае легче провести их независимый анализ, а затем объединить получившиеся представления.

В различных исследованиях, для разных задач позднее слияние может как показывать более эффективную работу, так и быть менее оптимальным вариантом или быть более-менее одинаково производящим решение задачи, по сравнению с ранним слиянием. Из этого можно сделать вывод, что эффективность применения позднего слияния в достаточной степени зависит от того, к какой задаче он применяется, для лучшего понимания, подходит ли он для данной задачи стоит изучить литературу по использованию раннего и позднего слияния со схожими модальностями и архитектурой.

Позднее слияние также часто используется при мультимодальном обучении. Одна из особенностей, которая делает его привлекательным для построения архитектуры – это способность снижать количество полученных из-за неоднозначности исходных данных ошибок, которые часто имеют малую корреляционную зависимость между модальностями. При этом типе слияния в модели мультимодального обучения идёт параллельный анализ отдельных модальностей в их собственных алгоритмах.

Промежуточное слияние (intermediate fusion).

Слияние может происходить на разных слоях у разных модальностей, каждая имеет свой канал из слоёв, в котором происходит обработка по алгоритму. При объединении двух или нескольких модальностей одновременно выделяется общий слой для объединённого представления. Они в свою очередь могут сливаться с представлениями других модальностей. В

итоге, имеется один общий слой, в котором заканчиваются каналы обработки всех массивов данных, он подаётся на выход.

Этот тип слияния наиболее гибкий, есть возможность выбирать после обработки на скольких слоях можно слить определённые модальности. Другие при этом могут продолжать обрабатываться на большем числе слоёв, может происходить дальнейший анализ слитых ранее представлений на слоях, расположенных далее.

Такое устройство слияния даёт достаточно большие возможности для оптимизации процесса обучения. Каждая модальность может проходить через столько промежуточных слоёв искусственной нейронной сети, сколько ей не обходимо для оптимального анализа и выявления необходимых признаков. Таким образом в перспективе есть возможность сэкономить ресурсы системы и избежать возможного переобучения модели от слишком долгого пребывания данных модальностей в нейронах.

Промежуточное слияние позволяет последовательно сливать более и менее корреляционно зависимые модальности между собой. В таком случае удаётся добиться высоких показателей эффективности для используемой в решении задачи модели.

Этот тип слияния может демонстрировать высокую эффективность, однако требует достаточно точного подбора архитектуры. Необходимо тщательно подходить к выбору структуры каналов для модальностей, не стоит проводить слияние слишком рано или слишком поздно, однако при правильном подборе количества слоёв показатели получившейся модели могут быть крайне высоки.

Возможно, в связи с таким сложным процессом проектирования архитектуры эти модели пользуются чуть меньшей популярностью по сравнению с двумя другими, рассмотренными выше.

Исследование методов представления графов

Для того, чтобы оценить результат работы с графовыми моделями, можно воспользоваться средствами визуализации графов. Через графическое представление можно исследовать характер связей внутри графа, наличие или отсутствие кластеров и хабов, их примерное количество. Также можно узнать, является ли итоговый граф связным и/или плоским. Наличие различных сгруппированных структур в графе может свидетельствовать о наличии в нём разделения по тематикам связей или вершин. Это может помочь в формировании, например, гипер- или метаграфовых представлений данных.

Возможности отображения графов были оценены на наборе данных $ATOMIC_{20}^{20}$, представляющем из себя базу знаний в виде условий если-то, отражающих логический вывод для ситуаций реального мира. Часто этот набор данных используется для задач commonsense reasoning.

Этот набор устроен как набор триплетов, характеризующийся объектом, субъектом и типом их логической связи. В качестве объектов и субъектов могут выступать люди, события или эмоциональное состояние. Такое устройство датасета делает его очень удобным для представления его в виде графа.

Для исследования визуализации графов была выбрана Python библиотека igraph. Результат работы функций этого модуля представлен на

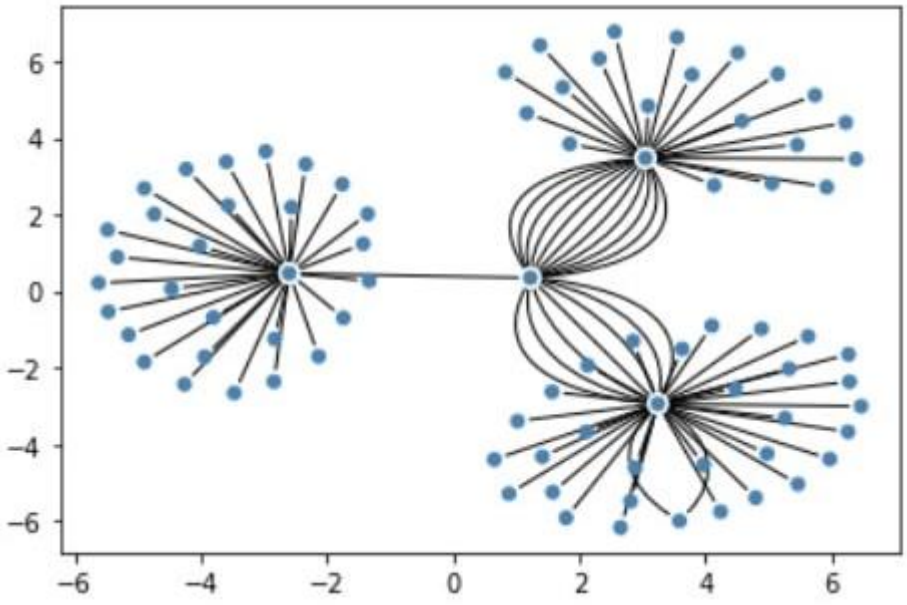


Рис. 2. Визуализация выборки данных из $ATOMIC_{20}^{20}$ с помощью *igraph*.

Уже на небольшой выборке данных мы видим, что в данной системе скорее всего можно будет наблюдать хабы, имеющие множество связей с другими вершинами. Также можно выделить следующую особенность: между двумя вершинами может существовать сразу несколько связей.

Эта библиотека предоставляет инструменты и для вычисления различных характеристик, например, матрицы смежности для вершин, которые могут быть использованы в алгоритмах работы с графовыми данными, в том числе и для машинного обучения.

Для более точного анализа следует провести визуализацию сразу всего набора данных, однако это процесс, затрачивающий много ресурсов. Следующим этапом может быть использование мощностей среды, например, Google Colab для обработки большего количества вершин и рёбер. Дополнительная проблема, для которой важно разработать решение – это читаемость графов с большим количеством вершин и рёбер, особенно, если они содержат текстовые данные.

Методы обработки текстовых данных

Один из способов работы с мультимодальными данными – это предварительно обработать их и привести к общему виду с другими данными. Для этого подходит алгоритм эмбеддинга. В литературе есть примеры, где предварительно данные приводят к векторному виду. Одни из передовых моделей в векторизации – это представители семейства моделей word2vec, получающие на вход массив токенов, извлечённых из текста, а на выходе дающий вектора для каждого токена в n-мерном пространстве.

Актуальным вопросом является степень эргономичности машинного обучения. Для её оценки воспользуемся моделью информационной системы содержащей модуль МО и модуль, в котором работу с данными осуществляет человек.

Описание исследуемой информационной системы

Рассмотрим информационную следующего обобщённого вида, представленного на рисунке.

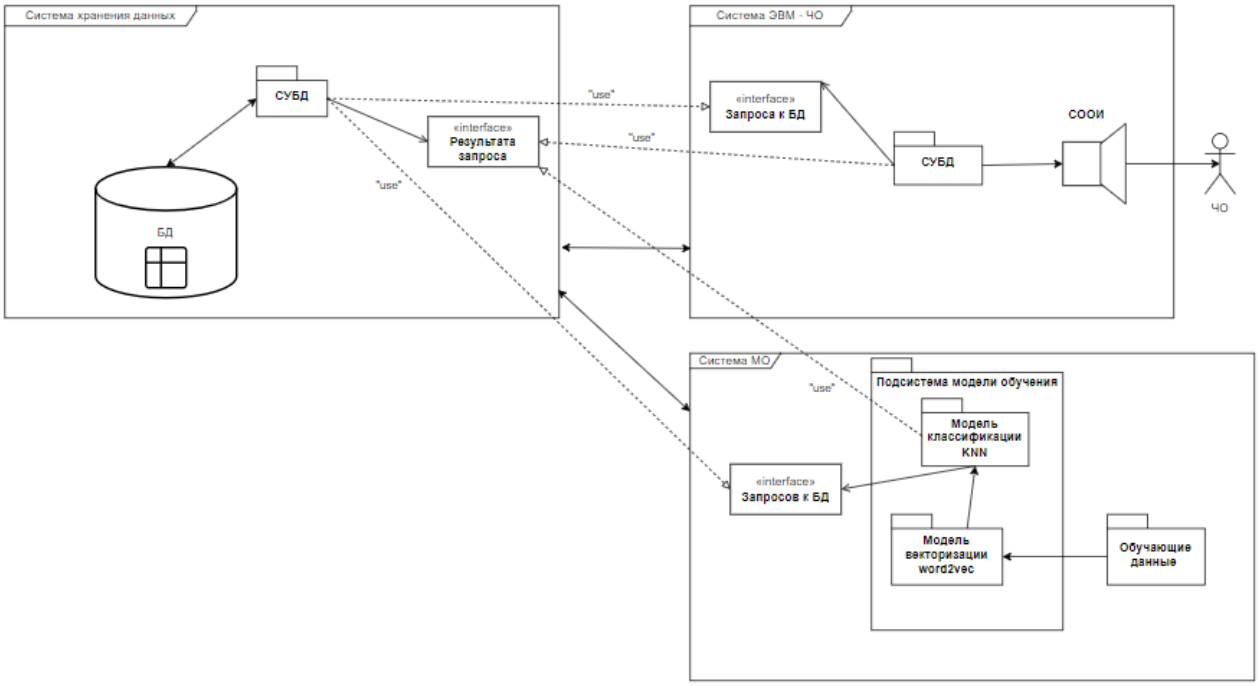


Рисунок 1. Описание исследуемой информационной системы в нотации UML

В целом здесь присутствуют две отдельных информационных системы, связанные между собой системой хранения данных, к которой можно обращаться с помощью запросов и получать их результаты.

Первая система

Первая система – это система ЧО – ЭВМ – СОИ. Сначала к системе хранения данных посылается запрос, который передаётся по каналу связи с помощью радиосигналов по сети Wi-Fi. Получив этот запрос, СУБД обрабатывает его и производит реализацию в базе данных. Системой управления формируется ответ на запрос, который по тому же каналу передаётся в систему ЧО – ЭВМ.

Далее полученная информация средствами СУБД в ЭВМ человека-оператора отображается в виде экранной формы на СОИ человеку-оператору, который её обрабатывает.

Вторая система

Вторая система представляет собой систему машинного обучения. В ней сначала происходит обучение с учителем модели на обучающей выборке.

Представим цепочки добавленной стоимости для процесса обучения данной модели. Сначала происходит импорт необходимых библиотек обработки данных. Они содержат способы представления данных, функции и классы, описывающие работу алгоритмов машинного обучения. Далее в пространство, в котором будет происходить обучение происходит загрузка в переменную значений данных в читаемом для алгоритма МО представлении. На следующем этапе происходит токенизация. Это процесс преобразования с помощью алгоритма-токенизатора данные типа строка в отдельные *токены*, которые будут восприниматься последующими моделями как единица текста. В данном случае используется алгоритм WordPunctTokenizer, который разбивает строку на отдельные слова на основе табуляции и пунктуации. Например строка `'Lorem ipsum dolor sit amet, consectetur adipiscing elit.'` будет преобразована в массив `['Lorem', 'ipsum', 'dolor', 'sit', 'amet', 'consectetur', 'adipiscing', 'elit']`.

Далее данные в таком виде поступают на модель векторизации, в качестве которой здесь используется модель word2vec. Данное семейство моделей представляет собой комбинацию двух типов архитектур нейронных сетей: CBoW (Continuous Bag of Words) и Skip-gram. Первая модель предсказывает следующий токен исходя из имеющейся последовательности токенов (прогнозирует следующее слово в предложении) на основе «мешка слов» (Bag of Words), т.е. на основе всего набора токенов, встречающихся в наборе данных, и их частоты встречаемости, CBoW добавляет «непрерывность», то есть ещё учитывает контекст. Вторая модель позволяет предсказывать слово по его контексту (отсюда skip, поскольку модель работает с пропусками в наборе токенов). Она основана на концепции n-грамм, т.е. разбиении слов на последовательности по n-символов. Особенность word2vec в том, что она с помощью этих архитектур может формировать

векторные представления слов. Каждый токен в процессе обучения присваивается некоторому вектору в n-мерном пространстве. На основе взаимного расположения этих векторов можно делать выводы о близости значений слов и вероятности их появления рядом в контексте.

Используемая в исследуемой ИС модель word2vec имеет в основе архитектуру CBoW, поэтому используется алгоритм, разбивающий на токены, соответствующие словам. Обучение модели проходит в течение 5 эпох (проходов по всему обучающему набору данных), она игнорирует слова, встречающиеся реже 10 раз, максимальное количество слов контекста = 10. Не обязательно использовать возможность предсказания токенов по контексту предшествующих ему, в контексте исследуемой ИС более важно получить от этой модели векторные представления. Это многомерное пространство векторов можно делить различными плоскостями для решения целевой задачи: классификации.

Следующей в цепи процессов располагается, собственно, модель классификации. В исследуемой системе это алгоритм KNN (k-Nearest Neighbors) или k-ближайших соседей. Эта модель относит выборочное значение к тому или иному классу на основании того, к какому классу принадлежат его ближайшие соседние значения (их количество и обозначает параметр k). На практике удовлетворительные результаты классификации получаются при $k \approx 3$. При k больше 10 часто не происходит улучшения, но временные затраты на алгоритм классификации увеличиваются. Путём анализа работы алгоритма при разных значениях для исследуемой системы параметр k приняли равным 5.

Рис. 3. ARIS-диаграмма цепочки добавленной стоимости для модели МО

Далее к системе хранения данных также посылается запрос на данные для считывания по аналогичному каналу передачи данных. Результат запроса от БД передаётся обратно по данному каналу. Результат запроса подаётся на

обученную модель машинного обучения (в ассоциации с моделью векторизации), которая производит обработку полученных данных.

Важно отметить, что этап непосредственно обучения для модели машинного обучения происходит в общем случае один раз, соответственно, во время первой итерации обработки данных. Интересно поведение функции времени обработки в области второй итерации, поскольку далее будет необходимо только использовать обученную модель для поставленной задачи обработки данных, что будет занимать меньшее количество времени, чем на итерации с обучением.

Оценка времени человека-оператора:

$$t_{\text{ЧО}} = t_{\text{канала}} + t_{\text{запроса}} + t_{\text{отобр.СОИ}} + t_{\text{ИП}} \quad (1)$$

Оценка времени машинного обучения:

$$t_{\text{МО}} = t_{\text{обучения}} + t_{\text{кода}} + t_{\text{счит.}} \quad (2)$$

Теоретическая оценка значений времени

Теоретическая оценка времени работы системы ЧО

Оценим параметр восприятия ЧО информации с экранной формы воспользуемся формулой (1). Для реальных масштабов времени значение времени информационного поиска лежит в пределах 0,8 – 1,2 с., время зрительной фиксации равно примерно 0,65 с. На экранной форме PostgreSQL располагаются элементы для работы с СУБД, Окно данных, в котором располагается таблица с двумя элементами заголовка, в которой выдаются 15 записей из запроса.

$$t_{\text{ИП}} = \frac{E + a}{a \cdot \left(1 + M(H_\alpha)\right)} * t_{\text{фик}} \quad (3)$$

Где, $t_{\text{ИП}}$ – время информационного поиска для человека-оператора;

E – общий объём элементов информационной модели;

a – объём зрительного восприятия (характеристика зрительного анализатора);

$M(H_\alpha)$ – математическое ожидание числа элементов ИМ с заданным для поиска набором характерных параметров α;

$t_{фик}$ – продолжительность зрительной фиксации.

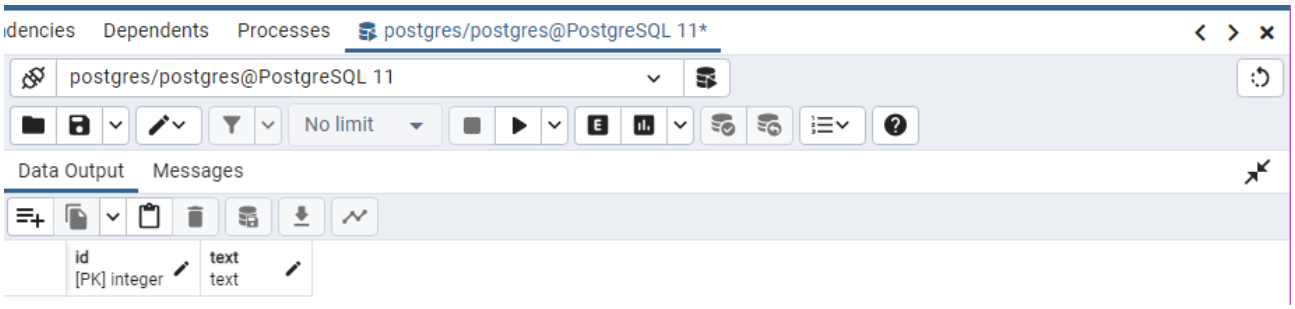


Рисунок 2. Общий вид экранной формы в PostgreSQL.

На экранной форме присутствуют 43 элемента, необходимые для работы внутри СУБД. В заголовке таблицы присутствует 23 элемента обозначающих характер и тип данных внутри (4 из них входят в целевой набор параметров – обозначение атрибута text, который ЧО будет классифицировать). В записях колонки id присутствует в среднем 2 символа (однако они не входят в целевой набор параметров), всего: 15*2 = 30 элементов на одной экранной форме.

Целью поиска является найти и воспринять текстовые данные писем. В среднем, как было рассчитано выше, в письме содержится в среднем 1082,92 символьных элемента, на 15 символов 15*1082,92 = 16243,8 элемента. Итого всего элементов на экранной форме: E = 16340 элементов, $M(H_\alpha)$ = 16248 элементов. Объём зрительного восприятия человека $a \approx 3$. Так же информационный поиск происходит на двух экранных формах, а значит нужно удвоить время:

$$t_{ИП} = 2 * \left(\frac{16340 + 3}{3 \cdot (1 + 16248)}\right) \times 0{,}65 = 2 * 0{,}2179 = 0{,}4358 \text{ с.}$$

Параметр времени	Полученное значение (с)
$t_{канала_1}$	0,00001
$t_{канала_2}$	0,0027
$t_{запроса}$	0,0873
$t_{отобр.СОИ}$	0,0450
$t_{ИП}$	0,4358

Общее время:	0,57081

Теоретическая оценка времени работы системы МО

К сожалению данные о сочетании архитектуры word2vec и KNN в литературе не описаны, но для алгоритма векторизации word2vec мы имеем данные о его обучении совместно с архитектурой TCN (Temporal Convolutional Network), которая в данном случае отвечает за задачу классификации. Поскольку алгоритм классификации k-ближайших соседей проходит только один раз, то возьмём в качестве аналога одну эпоху (проход по всей обучающей выборке) обучения word2vec+TCN.

Приблизительная оценка, которая может быть дана времени обучения в исследуемой системе тогда: $t_{\text{обучения}} \approx 17{,}4$ (с)

Время работы кода можно примерно определить, зная скорость загрузки данных в среду и количество информации в файлах импортируемых библиотек и обучающих данных. Скорость загрузки данных $v = 45$ (МБ/с). Файл csv с обучающими данными имеет размер $I_{learn} = 5.5$ МБ, файлы импортируемых библиотек суммарно имеют размер около $I_{lib} = 62.5$ МБ.

Воспользовавшись формулой (5) передачи данных по каналу (но в данном случае канал внутренний по шине) и формулой времени выполнения кода (16) получаем следующую формулу времени:

$$t_{\text{кода}} = \frac{I_{learn} + I_{lib}}{v} \tag{23}$$

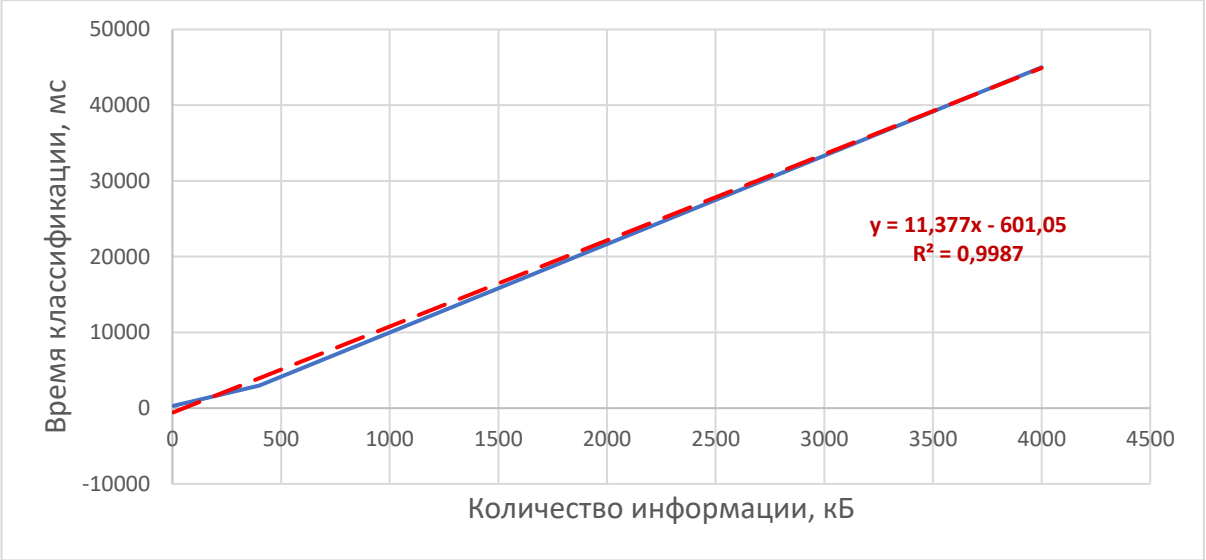
Подставив значения, получим:

$$t_{\text{кода}} = \frac{5.5 + 62.5}{45} = 1{,}511(\text{с})$$

Для алгоритма классификации k-ближайших соседей, KNN, в литературе приводятся следующие данные:

Количество информации (кБ)	Скорость классификации (мс)
4,00	300,00
400,00	3000,00
4000,00	45000,00

Построим по этим данным график и попробуем аппроксимировать его линейной функцией:



Насколько можно судить по трём измерениям, приближение (обозначенное красным пунктиром) можно назвать удачным. Воспользуемся полученным уравнением приближения. В данном случае выражение для времени классификации будет таким:

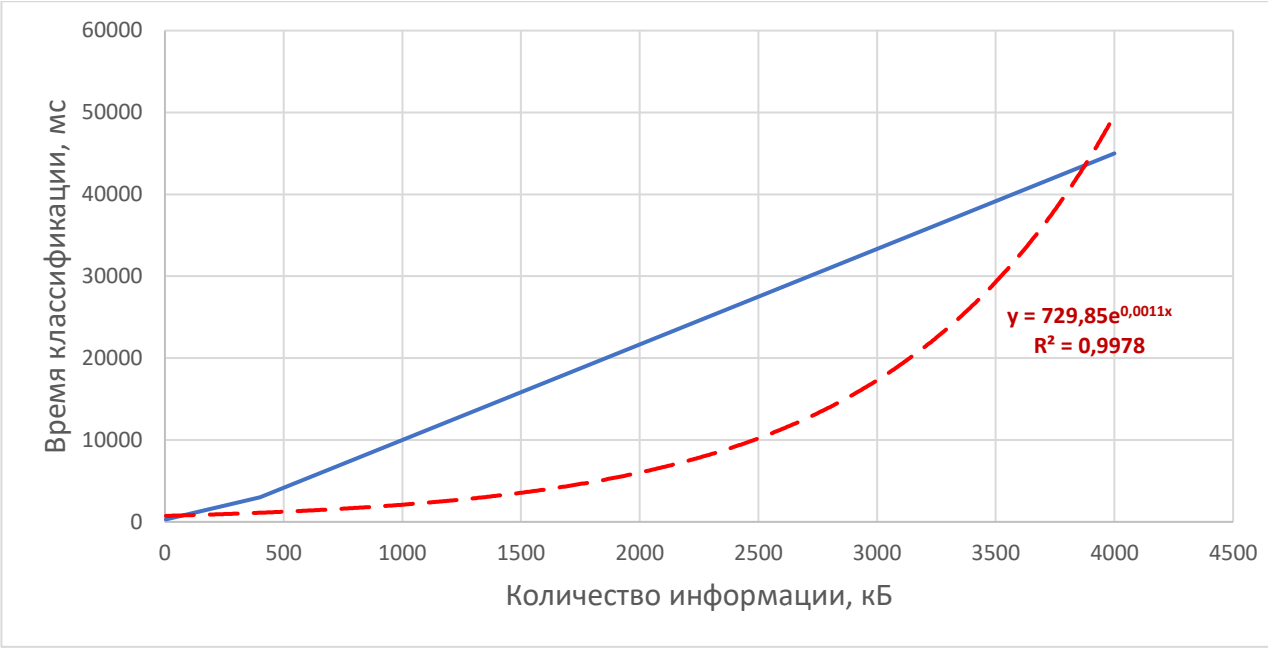
$$t_{\text{классиф.}}(k, m_a) = 11{,}377 \cdot k \cdot \frac{(Avg(m_1) \cdot \log_2 N_1)}{8000 \text{ бит}} - 601{,}05$$

Мы уже оценивали данные параметры ранее: $k = 30$; $Avg(m_1) = 1082{,}92$; $\log_2 N_1 = 8$.

$$t_{\text{классиф.}} = 11{,}377 \cdot 30 \cdot \frac{1082{,}92 \cdot 8}{8000} - 601{,}05 = -231.44(\text{мс})$$

Тут мы попали в неблагоприятную для наших значений область (количество информации в сообщениях около 32 кБ), где тренд заходит в отрицательную область.

Воспользуемся экспоненциальной аппроксимацией:



Экспоненциальная оценка тоже хорошо описывает данные. Тогда:

$$t_{\text{классиф.}}(k, m_a) = 729{,}85 \cdot e^{\left(0{,}0011 \cdot k \cdot \frac{(Avg(m_1) \cdot \log_2 N_1)}{8000 \text{ бит}}\right)}$$

Вычисляем аналогично:

$$t_{\text{классиф.}} = 729{,}85 \cdot e^{\left(0{,}0011 \cdot 30 \cdot \frac{(1082{,}92 \cdot 8)}{8000 \text{ бит}}\right)} \approx 756(\text{мс})$$

Параметр времени	Полученное значение (с.)
$t_{\text{обучения}}$	17,4
$t_{\text{кода}}$	1,511
$t_{\text{запроса}}$	0,0873
$t_{\text{канала}_1}$	0,00001
$t_{\text{канала}_2}$	0,0027
$t_{\text{классиф.}}$	0,756
Общее время:	19,75701

Результаты экспериментального измерения времени

Результаты для системы с человеком-оператором

После запуска сценария первой исследуемой системы ЧО получаем следующие результаты для временных характеристик в соответствии с формулой (19):

Параметр времени	Полученное значение (с)
$t_{\text{канала}_1}$	0,00001

$t_{\text{канала}_2}$	0,0028
$t_{\text{запроса}}$	0,083
$t_{\text{отобр.СОИ}}$	0,045
$t_{\text{ИП}}$	0,4358
Общее время:	0,56661

Результат для системы машинного обучения

После запуска сценария второй исследуемой системы МО получаем следующие результаты для временных характеристик в соответствии с формулой (21):

Параметр времени	Полученное значение (с.)
$t_{\text{обучения}}$	7,06
$t_{\text{кода}}$	1,515
$t_{\text{запроса}}$	0,017
$t_{\text{канала}_1}$	0,000009
$t_{\text{канала}_2}$	0,00275
$t_{\text{классиф.}}$	0,0625
Общее время:	8,657259

Получено значение точности accuracy для первого класса: 0.9741379310344828; и для класса «спам»: 0.9

Как видно, модель достаточно неплохо справляется с задачей, а значит мы хорошо её обучили.

Анализ результатов.

Сравнение теоретических и практических данных

Для системы с человеком оператором:

Параметр времени	Теоретические данные	Экспериментальные данные
$t_{\text{канала}_1}$	0,00001	0,00001
$t_{\text{канала}_2}$	0,0027	0,0028
$t_{\text{запроса}}$	0,0873	0,083
$t_{\text{отобр.СОИ}}$	0,045	0,045
$t_{\text{ИП}}$	0,4358	0,4358
Общее время:	0,57081	0,56661

Как можно видеть из сравнительной таблицы, полученная нами формула имеет достаточно большую точность отображения реальной исследуемой системы. Самое большое отклонение наблюдается во времени запроса. Данная характеристика имеет большую амплитуду колебаний в реальной СУБД, но даже с учётом таких колебаний отклонение в данном параметре составляет лишь несколько миллисекунд.

Колебание во времени канала передачи данных обусловлено меняющейся скоростью передачи по каналу связи, но даже с учётом таких факторов отклонение составляет лишь десятую долю миллисекунды.

В целом, можно сказать, что разработанная здесь временная оценка адекватно описывает поведение системы ЧО–ЭВМ.

Для системы машинного обучения:

Параметр времени	Теоретические данные	Экспериментальные данные
$t_{\text{обучения}}$	17,4	7,06
$t_{\text{кода}}$	1,511	1,515
$t_{\text{запроса}}$	0,0873	0,0170
$t_{\text{канала}_1}$	0,00001	0,000009
$t_{\text{канала}_2}$	0,0027	0,00275
$t_{\text{классиф.}}$	0,756	0,0625
Общее время:	19,75701	8,657259

Здесь мы наблюдаем куда большие колебания в показателях. Тем не менее, стоит отметить, что оценка времени передачи по каналу и времени выполнения кода имеют высокую точность оценки.

Колебание во времени обучения, скорее всего, обусловлено необходимостью проводить приблизительную оценку по аналогам, которые имеют не точное соответствие исследуемой системе. Эта характеристика имеет зависимость от количества данных и выбранной модели, но эту связь трудно адекватно описать по аналогам. Тем не менее, учитывая разброс времени обучения в разных моделях, мы можем получить примерную оценку порядка параметра времени обучения модели, экспериментальные и теоретические данные имеют схожий порядок около 10^1 с.

Получено меньшее время выполнения запроса на чтение для кода на Python. Возможно, это связано с тем, что СУБД тратит значительное время на процесс создания данных в нужном виде для дальнейшего отображения экранных форм ЧО. В коде Python такой необходимости нет. Соответственно, они отличаются на 70 мс.

Для оценки времени классификации получены экспериментальные данные, отличающиеся на порядок друг от друга. Отчасти причиной этого

может быть проблема в наличии всего трёх временных точек для приближения, а также различия в моделях: в случае теоретической оценки мы имеем 4 класса, а в исследуемой системе их всего 2. К тому же на вход KNN подаётся предварительно обработанная информация в виде векторов, что может упрощать процесс классификации.

В целом, в разработанной оценке имеются характеристики, которые дают не совсем адекватное представление о модели системы, однако, если имеет место возможность ошибки в пределах одного порядка, то данная оценка может быть эффективна. Для сложных систем машинного обучения, где время обучения составляет гораздо большие значения, ошибка на порядок может оказаться не столь существенна.

Сравнение систем с человеком-оператором и машинного обучения

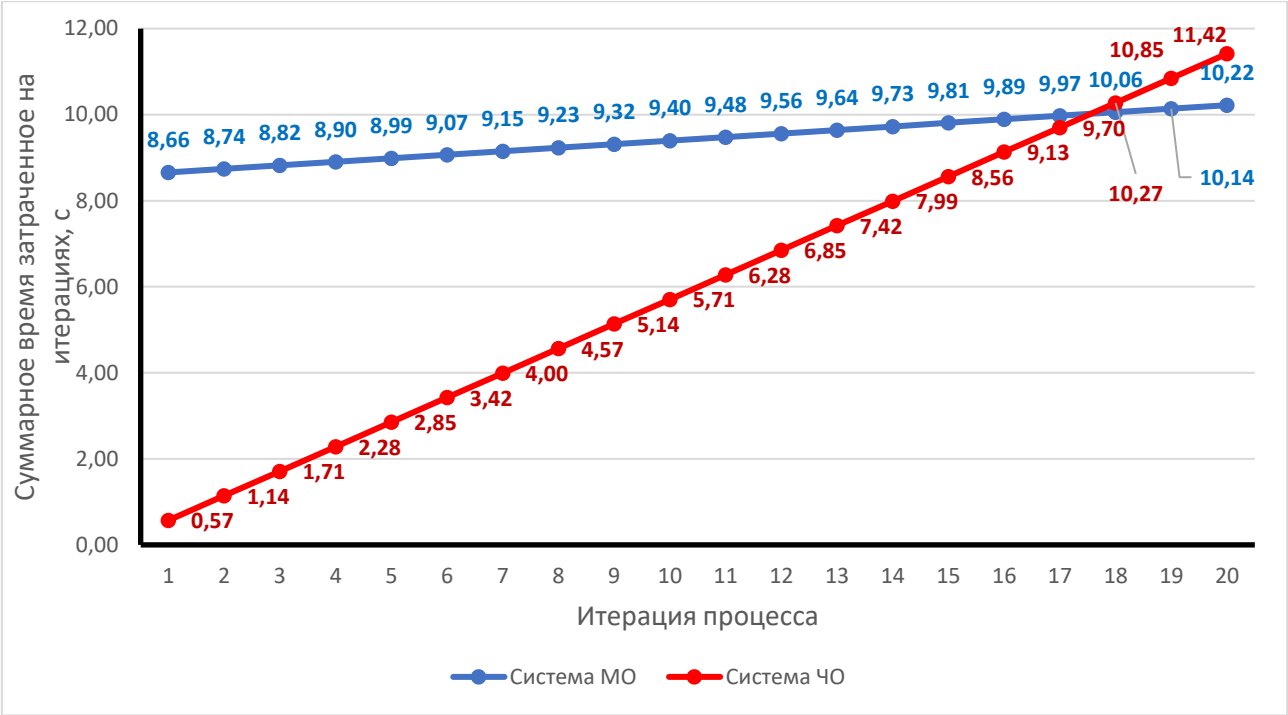
	Система ЧО	Система МО
Общее время (с.)	0,56661	8,657259

Первое сравнение системы с человеком-оператором и системы машинного обучения показывает, что система ЧО справляется с задачей быстрее, чем система МО. В данном случае эффективнее использовать устройство процесса так, чтобы работа выполнялось исключительно человеком-оператором.

Однако мы здесь видим срез только первой итерации решения задачи. Для более точной оценки стоит оценить время рабочего процесса в динамике. Посмотрим, как будет вести себя затрачиваемое время на протяжении 10 итераций.

Тут стоит отметить, что процесс обучения ($t_{\text{обучения}}$) и загрузки библиотек и данных ($t_{\text{кода}}$) нужен только на первой итерации, далее мы получаем уже обученную модель, которая и производит классификацию. Прирост времени с каждой итерацией для МО тогда получается: 8,657259 – 7,06 – 1,515 = 0,082259 с. При этом для ЧО прирост всё ещё будет равен времени его работы.

Тогда получается следующая динамика времени, затраченного на протяжении итераций:



На данной диаграмме видно, что на 18 итерации скорость работы системы машинного обучения начинает превышать скорость человека-оператора (или затрачиваемое время МО становится меньше, чем у ЧО). Таким образом, система МО не эффективна в краткосрочной перспективе, но показывает большую эффективность по сравнению с первой системой в долгосрочной при увеличении количества итераций.

Заключение.

Исследованы методы работы с мультимодальными данными, графами и текстом. Проведён анализ визуализации данных в графовом видес помощью библиотеки igraph. Проведено исследование модели векторизации на текстовых данных и эргономичность данной системы машинного обучения.

Система использующая машинное обучение не является эффективной в краткосрочной перспективе, поскольку время обучения значительно превышает время процесса для человека-оператора. Однако, в долгосрочной перспективе прирост времени в итерацию у машинного обучения сильно ниже, чем для ЧО (у которого он равен времени процесса), что обусловлено единичностью процесса обучения. Таким образом, системы МО оказываются

более эргономичными для задачи классификации, чем системы с человеком-оператором.

Для человека-оператора разработанная в этой работе система оценки показала свою адекватность реальной модели. Таким образом, её можно использовать в моделях эргономического анализа систем обработки данных человеком-оператором. Система оценки для машинного обучения показала меньшую эффективность, тем не менее её можно использовать для приблизительной оценки эргономичности сложных моделей, где отличие значения теоретического и практического на порядок могут быть не столь критичны.

Список литературы

1. Biswas R. Embedding based link prediction for knowledge graph completion [Text] – In Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020 – pp. 3221-3224.
2. Bordes A. Translating embeddings for modeling multi-relational data [Text]/ Bordes A., Usunier N., Garcia-Duran A., Weston J., Yakhnenko O. – Advances in neural information processing systems, 2013. – p. 26.
3. De Vries G. K. D. A Fast and Simple Graph Kernel for RDF [Text]/ De Vries G. K. D., De Rooij S. – DmoLD, 2013 – p. 1082.
4. Ji S. A survey on knowledge graphs: Representation, acquisition, and applications [Text]/ Ji S., Pan S., Cambria E., Marttinen P., Philip S. Y. – IEEE Transactions on Neural Networks and Learning Systems, 2021.
5. Kannan A. V. Multimodal knowledge graph for deep learning papers and code [Text]/ Kannan A. V., Fradkin D., Akrotirianakis I., Kulahcioglu T., Canedo A., Roy A., Al Faruque M. A. – In Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, October. – pp. 3417-3420.
6. Lin Y. Multimodal Learning on Graphs for Disease Relation Extraction [Text]: preprint arXiv:2203.08893/ Lin Y., Lu K., Yu S., Cai T., Zitnik M. – arXiv, 2022.
7. Ramachandram D. Deep multimodal learning: A survey on recent advances and trends [Text]/ Ramachandram D., Taylor G. W. – IEEE signal processing magazine, 34(6), 2017 – pp. 96-108.
8. Sap M. et al. Atomic: An atlas of machine commonsense for if-then reasoning //Proceedings of the AAAI conference on artificial intelligence. – 2019. – Т. 33. – №. 01. – С. 3027-3035.
9. Sun J. et al. Categorizing malware via a Word2Vec-based temporal convolutional network scheme //Journal of Cloud Computing. – 2020. – Т. 9. – №. 1. – С. 1-14.
10. Trstenjak B., Mikac S., Donko D. KNN with TF-IDF based framework for text categorization //Procedia Engineering. – 2014. – Т. 69. – С. 1356-1364.

11. Wang Q. Knowledge graph embedding: A survey of approaches and applications [Text]/ Wang Q., Mao Z., Wang B., Guo L. – IEEE Transactions on Knowledge and Data Engineering, 29(12), 2017. – pp. 2724-2743.

12. Wang Y. Fake news detection via knowledge-driven multimodal graph convolutional networks [Text]/ Wang Y., Qian S., Hu J., Fang Q., Xu C.// In Proceedings of the 2020 International Conference on Multimedia Retrieval. – 2020. – pp. 540-547.

13. Wang Z. Multimodal data enhanced representation learning for knowledge graphs [Text]/ Wang Z., Li L., Li Q., Zeng D.//In 2019 International Joint Conference on Neural Networks (IJCNN) – IEEE, 2019, June. – pp. 1-8.

14. Wilcke W. X. (2020). End-to-End Entity Classification on Multimodal Knowledge Graphs [Text]/ Wilcke W. X., Bloem P., de Boer V., van't Veer R. H., van Harmelen F. A. H. – arXiv, 2020.

15. Wu X. Multimodal news story clustering with pairwise visual near-duplicate constraint [Text]/ Wu X., Ngo C. W., Hauptmann A. G. – IEEE Transactions on Multimedia, 10(2), 2008. – pp.188-199.

16. Горячкин Б. С. Эргономический анализ систем обработки информации и управления //Вестник евразийской науки. – 2017. – Т. 9. – №. 3 (40). – С. 72.

17. Елисеева Е. А., Горячкин Б. С., Виноградова М. В. ИССЛЕДОВАНИЕ ПРОИЗВОДИТЕЛЬНОСТИ СУБД ПРИ РАБОТЕ С КЛАСТЕРНЫМИ БАЗАМИ ДАННЫХ НА ОСНОВЕ ЭРГОНОМИЧЕСКОГО АНАЛИЗА //StudNet. – 2022. – Т. 5. – №. 4. – С. 2888-2910.

18. [Электронный ресурс] igrhttps://igraph.org/aph – Network analysis software – документация по библиотеке визуализации графов igraph.