УДК 004.522

ЧЕЛОВЕКО-МАШИННЫЙ ИНТЕРФЕЙС ИНТЕЛЛЕКТУАЛЬНЫХ РОБОТОВ

Никольская М.А., Романов М.П., Ал-Хафаджи исра М. Абдаламир

МИРЭА - Российский технологический университет, 119454, Россия, г. Москва, проспект Вернадского, 78, e-mail: nikolskaya@mirea.ru, m_romanov@mirea.ru, misnew6@gmail.com

В статье изложены примеры современных больших лингвистических моделей для реализации человеко-машинного интерфейса на естественном языке. Рассмотрены возможные решения, а также проблемы реализации человеко-машинного интерфейса с интеллектуальным роботом. Предложены алгоритмы реализации человеко-машинного интерфейса с интеллектуальным роботом на естественном языке.

Ключевые слова: ChatGPT, GPT-3, GigaChatPro, YandexGPT, человеко-машинный интерфейс, интеллектуальные роботы, генеративный искусственный интеллект, автономная робототехника.

HUMAN-MACHINE INTERFACE OF INTELLIGENT ROBOTS

Nikolskaya M.A., Romanov M.P., AL-Khafaji Israa M. Abdalameer

MIREA - Russian Technological University, 119454, Moscow, 78 Vernadsky Avenue, *Russia, e-mail: nikolskaya@mirea.ru, m_romanov@mirea.ru, misnew6@gmail.com*

The paper presents examples of modern large-scale linguistic models. The implementation problems of intelligent robot human-machine interface as well as their solutions are discussed. Finally, the algorithms of natural language human-machine interface intelligent robot in natural language are proposed.

Keywords: ChatGPT, GPT-3, GigaChatPro, YandexGPT, human-machine interface, intelligent robots, generative artificial intelligence, autonomous robotics.

Введение

В настоящее время в мире бурными темпами развивается интеллектуальная автономная робототехника. Интеллектуальные мобильные роботы предназначены для работы без участия человека - оператора в заранее неизвестных условиях. Области применения очень широки: от бытового до специального функционала применения, при этом они могут относиться к робототехнике сервисной и предназначенной для экстремальных условий функционирования, а среды работы робота бывают космического, воздушного, наземного, надводного и подводного базирования.

На первых этапах создания автономных роботов для планирования движения и поведения применяли строгую стратегию Planning (классическое планирование). Особенностью такой стратегии являлось четкое описание окружающей среды. Заранее должна быть полная информация для составления модели этой среды и формирования поведения в ней, строгое планирование взаимодействия между объектами. Такой метод имеет ряд недостатков, обусловленных тем, что даже незначительное изменение внешней среды может привести к ошибкам поведения объекта [1].

Интеллектуальные роботы, как правило, отличаются большим набором и разнообразием устройств очувствления, средствами навигации, обладают возможностью работы с глобальной цифровой картой и способны строить локальную карту рабочей сцены. Они умеют различать и обходить препятствия, работать в группе с другими интеллектуальными роботами, выполняя совместно целесообразные действия.

Системы управления таких роботов строятся по многоуровнему иерархическому принципу и включают исполнительный, тактический, стратегический и понятийный уровни. Каждый уровень управления связан с информационно-измерительной системой, на основе информации которой интеллектуальная система управления определяет наличие возмущений и неопределенностей и реализует адаптивные свойства при выполнении целесообразных действий по обеспечению возложенного на робота функционала.

Для решения задач интеллектуальной автономной робототехники используют машинное обучение с подкреплением. То есть модель приспосабливается и старается найти оптимальную среду, путем формирования и оценки матрицы перехода из одного состояния в другое состояние. Самые популярными алгоритмами такого рода являются: Q-learning, SARSA, DDPG, DQN [2]:

- Q-learning – алгоритм где интеллектуальный агент (робот) выполняет такое действие в динамичной среде, чтобы получить максимальное вознаграждение Q, при этом данный алгоритм не оценивает саму функцию вознаграждения, которая связана с Марковским принятием решения – случайным процессом управления с дискретным временем.
- SARSA (State-Action-Reward-State-Action), алгоритм, который используется для решения задачи управления Марковским процессом принятия решений, в целом очень похож на алгоритм Q-learning, разница заключается в том, что каждый шаг состоит из строгой последовательности: 1 - выбор действия; 2 - переход в новое состояние; 3 - получение награды; 4 - выбор действия и обновление оценки действия, то есть агент на каждой такой последовательности «состояние-действие», проверяет и оценивает значение оценки действия. То есть агент получает награды/штрафы за свои действия, взаимодействуя с окружающей средой. Цель алгоритма SARSA - научить агента выполнять оптимальные действия в разных состояниях среды, чтобы максимизировать общую полученную награду. При этом Марковский процесс принятия решений - это математическая модель, используемая для формализации задачи управления в условиях неопределенности. SARSA имеет ограничение связанное с тем, что действие формируется на основе текущего состояния, в то время как Q- learning не имеет таких ограничений в выборе действия при переходе в следующее состояние. Это дает ему возможность выбирать стратегию поведения не случайным образом, а учитывая опыт предыдущего взаимодействия со средой.
- DQN (Deep Q Network) – этот алгоритм также основывается на алгоритме Q-learning. Данный алгоритм использует нейронную сеть для оценки новых состояний, с которыми агент не встречался в пространстве «действие-состояние», и не знает какое действие предпринять.
- DDPG (Deep Deterministic Policy Gradient) – алгоритм, который также дополняет DQN, при высоких размерностях, а также с дискретным пространством действий.

Обучение с подкреплением разделяется на Model-Based, которое сводится к алгоритму MCTS – поиск по дереву Монте-Карло и применяется к средам с конечным количеством ходов и Model-free, который делится на on-policy – алгоритм SARSA (обучение происходит на основе действий от текущей стратегии поведения) и off-policy - Q-learning (обучение происходит на основе действий вне стратегии. Так как новое действие выбирается только для максимизации Q-значения в следующих состояниях).

Постановка задачи такому интеллектуальному роботу и общение с ним должно осуществляться на естественном языке. Человеко-машинный интерфейс реализуется на понятийном уровне управления. Понятно, что робот не человек и его мир существенно уже человеческого. Тем не менее, общение должно носить интерактивный характер и ставить конечной целью формирование требований к последовательности целесообразных действий по выполнению поставленных оператором целей и задач.

В настоящее время интеллектуальные роботы догоняют человека в его когнитивных способностях, однако ограниченные возможности бортовых вычислителей и трудности формализации ситуаций, с которыми сталкивается робот не оставляют шансов для робота в этом соревновании.

Одним из путей развития когнитивных способностей интеллектуальных роботов является самообучение при общении с человеком посредством естественного языка. В настоящее время существует достаточно большое количество лингвистических моделей, позволяющих общаться человеку с компьютером, однако для реализации этих подходов мощности бортовых вычислителей интеллектуальных роботов недостаточно. Поэтому актуальной задачей является анализ возможностей и принципа действия больших лингвистических моделей с целью упрощения их функционала для реализации общения человека с интеллектуальным роботом на естественном языке с учетом ограничений, накладываемых окружающим миром робота, который существенно уже по сравнению с миром человека.

Большие лингвистические модели

В марте 2022 года авторами статьи [3] была предложена возможность привязки задач высокого уровня, выраженных на естественном языке, к выбранному набору практических шагов с помощью больших языковых моделей. Большим преимуществом данного метода является – пополняемая гибкая модель об окружающем мире, соответственно, и еще большее количество вариантов взаимодействия с объектами. А также взаимодействие и обучение с робота на естественном языке. Авторы статьи использовали авторегрессионную генерацию: предобученная модель на вход принимает часть незавершенного запроса/задачи и модель на выходе должна завершить этот запрос, а также сформировать план выполнения поставленной задачи. В статье авторы провели ряд экспериментов на различных больших лингвистических моделях и вывели зависимость исполнимости действий от семантической корректности сгенерированных планов. Большие модели могут создавать планы действий, неотличимые от планов действий людей, но зачастую они не могут быть корректно выполнены в окружающей среде. Большие лингвистические модели при помощи, которых были проведены исследования это GPT.

Рассмотрим особенности функционирования наиболее развитых больших лингвистических моделей:

Языковая модель GPT

GPT-3 это генеративная текстовая обновленная модель, по сравнению с GPT-1 и GPT-2. GPT-3 имеет 175 миллиардов правил. Обучение модели происходило на информации из сети Интернета: сайты, форумы, книги, датасеты. По технологии обучения нейронной сети с учителем и подкреплением был задействован объем информации более 570ГБ [4]. Обучение с учителем (supervised learning)- подразумевает обучение нейросети на уже подготовленном наборе структурированных, проверенных, экспертных данных. Система обучения с подкреплением (Reinforcement Learning) включает понимание, что агенты обучаются на подходе: вознаграждение за правильные действия и наказание за неправильные. Обучение происходило на суперкомпьютере Microsoft Azure AI [5], было задействовано: 285 000 процессорных ядер (CPU Core), 10 000 графических процессоров (GPU) и обеспечена скорость сетевой передачи информации 400 гигабит в секунду для каждого сервера. Следует отметить, что модель учится отслеживать причинно-следственные связи, законы и роли слов в предложениях. Сама нейронная сеть GPT-3 построена на архитектуре Transformer, архитектуре глубоких нейронных сетей, с применением моделей внимания, но не использующая рекуррентные нейронные сети [6]. Данная архитектура имеет большую производительность за счет параллельной обработки информации [7].

Архитектура Transformer состоит из нескольких ключевых компонентов:

1. Механизм внимания: Transformer использует механизм внимания для учета контекста при обработке входных данных. Механизм внимания позволяет модели сосредоточиться на наиболее релевантных частях входных данных.
2. Multi-head attention (многократный параллельный проход через модуль внимания): Transformer использует Multi-head аутентификацию для обработки входных данных. Multi-head attention позволяет модели обрабатывать данные с разных точек зрения, что улучшает ее способность к обобщению.
3. Position-wise feed-forward network (позиционная сеть прямой связи): Transformer использует сеть прямого распространения для обработки входных данных. Position-wise feed-forward network позволяет модели учитывать позицию каждого токена в последовательности.
4. Residual connections (остаточные связи): Transformer использует остаточные связи для улучшения обучения модели. Остаточные связи позволяют модели передавать информацию между различными слоями трансформера.
5. Layer normalization (нормализация слоев): Transformer использует нормализацию слоев для улучшения стабильности обучения модели. Layer normalization позволяет модели нормализовать входные данные перед их обработкой.

В целом, архитектура Transformer представляет собой комбинацию механизма внимания, Multi-head attention аутентификации, позиционно-зависимой сети прямого распространения, остаточных связей и нормализации слоев, что позволяет модели обрабатывать входные данные последовательно и учитывать контекст.

По умолчанию, модель не имеет памяти, модель не хранит информацию о предыдущем запросе, по сравнению с человеком, который помнит, о чем говорил собеседник ранее.

Для реализации специальных задач, например, человеко-машинного интерфейса, при помощи тонкой настройки (fine tuning) можно создать свою нейронную сеть на основе GPT-3, обучить на своих примерах и использовать для управления интеллектуальным роботом.

Краткое описание архитектуры GPT:

В процессе генерации продолжения текста с помощью GPT происходит следующее: входной текст преобразуется в последовательность чисел (токенов), которые затем проходят через Embedding Layer и превращаются в список эмбеддингов (вложений, в нашем случае набор векторов). К каждому эмбеддингу добавляется positional embedding, после чего список эмбеддингов проходит через несколько одинаковых блоков (Transformer Decoder Block). После прохождения последнего блока, эмбеддинг, соответствующий последнему токену, умножается на входной Embedding Layer, транспонированный предварительно, и после применения алгоритма SoftMax получается распределение вероятностей следующего токена. Из этого распределения выбирается следующий токен, который добавляется к входному тексту, и процесс повторяет

Алгоритм работы GPT можно описать следующим образом:

1. Входной текст: Сначала входной текст токенизируется в последовательность чисел (токенов). В качестве токена как правило, используется номер слова в словаре.
2. Embedding Layer: Затем список токенов проходит через Embedding Layer (линейный слой) и превращается в список эмбеддингов (очень похоже на word2vec). Результатом этого действия является набор векторов.
3. Positional Embedding: К каждому эмбеддингу прибавляется positional embedding.

4. Transformer Decoder Block: Далее список эмбеддингов начинает своё движение через несколько одинаковых блоков (Transformer Decoder Block).

5. Матричное умножение: После того как список эмбеддингов пройдёт через последний блок, эмбеддинг, соответствующий последнему токену матрично умножается на всё тот же входной, но уже транспонированный Embedding Layer.

6. SoftMax: После применения SoftMax получается распределение вероятностей следующего токена.

7. Выбор следующего токена: Из этого распределения выбираем следующий токен (например, с помощью функции argmax).

8. Добавление токена к входному тексту: Добавляем этот токен к входному тексту и повторяем шаги 1-7.

Таким образом, алгоритм работы GPT заключается в токенизации входного текста, преобразовании его в список эмбеддингов, прохождении через несколько блоков трансформера, матричном умножении, применении SoftMax и выборе следующего токена.

Одна из трудностей при работе с большими данными и лингвистическими моделями, это предварительная подготовка данных для обучения и не структурированность их – неподготовленный обработкой текст, фотографии.

Так как мы говорим о методе встраивания, векторном пространстве и измерение схожести, в качестве которой используют косинус угла между векторным представлением сущностей, то диапазон принимаемых значений элементов векторов варьируется от 0 до 1. Косинусное сходство применяется при вычислении оценки сходства между запросами, преобразованными в вектора.

Также одной из задач в создании интеллектуального человеко-машинного интерфейса – является предсказание запросов: то есть при интеллектуальном взаимодействии с человеком робот должен уметь предсказывать запрос, на основе предыдущей информации или уже заведомо известной информации (если, например, картина мира объекта, ограничена). С этой задачей справляется также GPT-3, используя вышеупомянутый метод встраивания, но с более сложной архитектурой [6].

Если рассматривать применение предсказаний в робототехнике: предсказание следующего действия и его предложение как возможное действия мобильного робота, при автономном управлении роботом, также имеется возможность реализации интеллектуального интерфейса – обобщение сказанного, категоризации команд роботу по темам, формирование задач на естественном языке.

Важным аспектом в данной задаче является корректность и грамотность вводимого запроса: пользователь (оператор, например) может вводить запрос с ошибкой, опечаткой, некорректный запрос роботу. В случае стандартного подхода – рекомендаций и предсказаний не будет, либо при частичном вводе запроса, ответ будет очень обширным, что тоже не удовлетворит пользователя. Например, при вводе «Audi» будут рекомендоваться как сферы продажи, технические центры поддержки авто, так и магазины аксессуаров. В этом случае стоит применить нечеткий поиск [6], а именно метод расстояния Левенштейна [8]: это метрика, измеряющая по модулю, какая разность между двумя последовательностями символов [9]. То есть измеряется количество операций таких как: удаление, замена, вставка, для получения «схожести» найденного запроса в исходный первоначальный запрос. Данный алгоритм реализован в библиотеке textdistance [10]. С решением данной задачи справляется и косинусный поиск, в частных решениях можно комбинировать эти методы.

После определения корректности запроса, для ряда задач важно определить тему запроса, то есть определение классификатора запроса [6]. С этим справляется GPT-3 успешно, применяя функции встраивания, вычисления сходства предложений и категорий.

В [6] автор приводит пример Ассистента в аптеке, для предоставления советов по медицинским лекарствам. Общая схема выглядит так: на вход от пользователя подается строка с названием препарата, на выход от Ассистента – строка с рекомендацией по нему. В алгоритме программы применены функции токенизации, стемминга, тонкой настройки, для получения более точных ответов. При вводе пользователем посторонних вопросов, не относящихся к медицинским препаратам, Ассистент выдает однотипный ответ о некомпетентности в данном вопросе. Данное ограничение связано с ограниченной картиной мира, позволяет сэкономить на объемах обучении, вычислительных мощностях, памяти Ассистента. Данный подход вполне подходит для реализации человеко-машинного интерфейса с интеллектуальным роботом на естественном языке.

Так как GPT это генеративная модель, то по умолчанию, при каждом обращении к ней, она создает новый запрос и новый ответ на него, вне зависимости от истории предыдущего запроса и ответа, также, не вникая в контекст ситуации, полученной в результате нескольких обращений. С одной стороны, это хорошая возможность для модели: формирование не однотипных ответов (высокая креативность модели), с другой стороны – отсутствие возможности ведения полноценного диалога как с человеком.

Для решения данной задачи авторы [6] предложили использовать предыдущий ответ пользователя и его

учитывать – вставлять в новый запрос пользователя предыдущий запрос в начало. Такой диалог будет в конце концов окончен ошибкой в связи с переполнением токенов при запросе, а также такая модель будет очень дорогой в связи с большой длиной запроса на входе. Второй метод решения основан на работе с памятью «последний пришел-первым вышел», но тут тоже есть недостаток: возникает трудность при переключении тем в диалоге и возвращении к более древней теме, не предыдущей. Третий вариант решения предполагает разбиение ответов пользователя по файлам и по классификации этих файлов, затем анализ по косинусному сходству с новым запросом для определения к какому файлу необходимо обратиться.

Возможность модели GPT-3 по нескольким, малым количествам примеров (до пяти примеров) решать новые задачи [6] появилась за счет того, что она предварительно обучена на огромном объеме данных: миллиард слов. Такая возможность, few-shot learning (метод машинного обучения, целью которого является обучение моделей изучению новых задач или распознаванию новых классов объектов, используя лишь небольшой объем помеченных данных), в условиях различных ограничений бывает очень необходима, для экономии ресурсов времени, трудовых ресурсов. Также в GPT-3 предусмотрена возможность тонкой настройки, для достижения более точных результатов: требуется чуть больше примеров, но после обучения не требуется больше приводить обучающие примеры при каждом следующем запросе.

Технология GPT по своим возможностям вполне подходит для общения человека с интеллектуальным роботом, но по техническим требованиям к бортовому вычислителю совершенно не приемлема. Единственный способ на момент общения, при постановке задачи, подключать робот через интернет к ChatGPT для GOOGLE, но и это не приемлемо в рамках обеспечения Технологической независимости.

GigaChat

На российском рынке альтернатива модели GPT, является разработка нейросети команды Сбербанка [11] – GigaChat. GigaChat построен на архитектуре NEural Omnimodal Network with Knowledge-Awareness, объем данных задействованных при обучении 300ГБ: от книг до новостных форумов в сети Интернет. Поддерживает русский и английский языки [11].

В [12] разработчики SberIDP предложили свое решение для автоматизации в банковском секторе: автоматизация выдачи кредита – его одобрения или отклонения. Это сэкономило время клиента банка до 7 минут и уменьшило вероятность ошибки сотрудника банка при анализе документов клиента.

Если рассматривать технологию Сбербанка, основанную также на глубинном обучении, в области распознания именованных сущностей [12], сводящаяся к классификации слов. Изначально, авторы предлагали решение на основе архитектуры CharCNN-BLSTM-CRF, но наилучшие результаты в последнее время были достигнуты нейронной сетью с архитектурой BERT.

При подаче текста на вход сети, сначала происходит его токенизация. Токенами являются слова, которые присутствуют в словаре, или их составные части, если слово отсутствует в словаре. Словарь является частью модели, например, в BERT-Base используется словарь, содержащий около 30,000 слов. В самой нейронной сети токены кодируются своими векторными представлениями, которые включают представления самого токена, его номера предложения и позиции внутри предложения. Входные данные обрабатываются сетью параллельно, а не последовательно, при этом информация о взаимном расположении слов в исходном предложении сохраняется. Выходной слой основной сети имеет следующий вид: поле, отвечающее за ответ в задаче предсказания следующего предложения, а также токены в количестве, равном входному набору слов. Обратное преобразование токенов в вероятностное распределение слов осуществляется полносвязным слоем с количеством нейронов, равным числу токенов в исходном словаре.

Минусом данной архитектуры является необходимость в большом количестве размеченных данных в специализированных областях: это долго, дорого и сложно [12]. Из положительных характеристик можно отметить возможность стоить контекстную зависимость токенов. Для умения предсказания классов модель разбивает текст на токены, объединяет их в соответствующие классы, а затем по NER-разметки BIO-схеме каждом токену добавляется качество (префикс), затем токены объединяются в спаны. Спан представляет собой фрагмент непрерывного текста. Named-Entity Recognition (NER) -извлечение именнованных сущностей (например, дата/название/время, то есть определенные категории) путем выделения спанов (объединенных токенов) в тексте. Решение данной задачи помогает в структуризации данных: деление их по категориям, разделение по базам. Выделение сущностей помогает в построении вопросно-ответной системы. Схема BIO подразумевает, что в метке сущности будет добавляться определенный префикс, который будет означать позицию токена в спане данной сущности.

GigaChatPro имеет 29 миллиардов параметров с улучшениями для ускорения сходимости обучения. Сложность обучении данной модели: значительно меньшее количество набора качественных данных в сети Интернет на русском языке, чем на английском языках [13]. При обучении требуется большой объем памяти, в

связи с большими моделями, поэтому Сбербанк реализовал технику шардирования между вычислительными узлами. GigaChat позволяет загружать запрос размерностью до 8 тысяч токенов (ориентировочно до 12 страниц).

YandexGPT

YandexGPT [14] это большая лингвистическая модель компании Яндекс, последняя модель имеет 100 миллиардов правил. YandexGPT также как и выше обсужденные нейросети решает задачи с прогнозированием ответов, поддержанием диалога, памяти контекста предыдущего текста. Модель обучена частям речи, членам в предложении и пунктуации. Процесс обучения разделен на два этапа: Pretraining - обучение на данных Интернета по аналогии с вышеуказанными нейронными сетями: на основе общедоступных книг, сайтов, форумов и P-tuning - автоматическая подборка правильной подводки [15]. Качество данных для обучения является актуальной проблемой: кроме стандартных решений, команда Яндекса самостоятельно написала для обучения дополнительный датасет с эталонными моделями на 36 000 ответов.

В [16] автор провел эксперименты, 5 декабря 2023г., по оценке качества ответов моделей и сравнил метрику Side-By-Side с результатами: GigaChatPro (50%) и ChatGPT(50%), YandexGPT2 большая (43%)и GigaChatPro (57%), YandexGPT General (Lite) (38%)и GigaChatPro (62%). По результатам экспериментов на 5 декабря 2023г было выявлено, что GigaChat хорошо справляется с рядом задач, а при работе с документацией лучше, чем ChatGPT.

Алгоритмы реализации языковых моделей человеко-машинного интерфейса с интеллектуальным роботом

Воспользовавшись рассмотренными технологиями можно предложить следующий алгоритм обработки естественного языка на бортовом вычислителе интеллектуального робота.

При обработке естественного языка следует учитывать ограниченность мира робота, который сводится к перемещению робота по траекториям, выбранным с учетом предложенных оператором критериев, связанных с максимальным быстродействием, минимальным количествам поворотов, минимальным потреблением энергии и.п., а также с захватом и перемещением грузов различной формы, например, куб или шар, которые могут иметь различную окраску. Обход препятствий, движение по цифровой карте, построение локальной карты, распознавание образов и т.п. относится к внутреннему функционалу интеллектуального робота, а человеко-машинный интерфейс на естественном языке нужен только для постановки задачи.

Учитывая вышесказанное, из всех возможных диалогов формируем словарь токенов. Словарь будем формировать по следующему правилу: выбираем только уникальные слова, причем заносим только корень слова, а затем в качестве токенов вводим уникальные суффиксы и приставки. Ожидаемое количество токенов около тысячи. Сортируем словарь в алфавитном порядке.

Для определения наиболее подходящего ответа на запрос можно воспользоваться алгоритмом ***TF-IDF***, который позволяет учесть частотные свойства слов в тексте, при этом менее часто употребляющиеся слова являются более важными.

TF отражают нормализованную частоту

$$\boldsymbol{TF} = \frac{\boldsymbol{n_t}}{\sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \boldsymbol{n_i}},$$

где n_t - количество повторений слова в документе,

$\sum_{i=1}^{k} n_i$- количество всех слов в документе.

IDF -инверсия частоты определяется по формуле:

$$\boldsymbol{IDF} = \frac{\boldsymbol{n_c}}{\sum_{\boldsymbol{j=1}}^{\boldsymbol{k1}} \boldsymbol{n_j}}$$

где n_c - количество документов всего,

$\sum_{j=1}^{k1} n_{ij}$- количество документов, в которых встречается слово.

Формула для определения показателя ***TF−IDF*** имеет следующий вид:

$$\boldsymbol{TF-IDF=TF*IDF,}$$

Чем больше значение данного показателя, тем важнее данное слово в документе. Упорядочиваем словарь по значениям ***TF−IDF*** в порядке убывания и кодируем слова по новому словарю.

При реализации подхода ***TF-IDF*** каждое слово обрабатывается как отдельная сущность и не учитывается контекст, необходимый для правильной интерпретации запроса.

Использование технологии ***Word2Vec***, позволяет учесть контекст. ***Word2Vec*** реализуется в виде двух алгоритмов ***skip-gram*** и ***CBOW***. Согласно публикации [17] ***skip-gram*** хорошо работает с небольшими наборами данных и может лучше представлять редкие слова, а ***CBOW*** обучается быстрее, чем ***skip-gram***, и может лучше представлять часто встречающиеся слова. Поэтому выбор алгоритма зависит от решаемой задачи.

Алгоритмы ***Word2Vec*** за счет использования оконного кодирования слова учитывает только локальную статистику по близь лежащим словам. Алгоритм ***GloVe*** учитывает глобальную статистику, извлекает смысловое содержание из матрицы совместной встречаемости слов.

Пусть мы имеем два документа:

Возьми красный кубик и отнеси к шарику.

Шарик имеет красный цвет.

В этом случае вектора строятся из совместного рассмотрения двух и более документов (предложений), причем при размере окна равного 1 мы получаем матрицу совместимости, представленную в таблице 1.

Таблица 1. Матрица совместимости.

	Start	*Возьми*	*и*	имеет	к	красный	кубик	отнеси	цвет	шарику	*End*
Start	0	1	0	0	0	0	0	0	0	1	0
Возьми	1	0	0	0	0	1	0	0	0	0	0
и	0	0	0	0	0	0	1	1	0	0	0
имеет	0	0	0	0	0	1	0	0	0	1	0
к	0	0	0	0	0	0	0	1	0	1	0
красный	0	1	0	1	0	0	1	0	1	0	0
кубик	0	0	1	0	0	1	0	0	0	0	0
отнеси	0	0	1	0	1	0	0	0	0	0	0
цвет	0	0	0	0	0	1	0	0	0	0	1
шарику	1	0	0	1	1	0	0	0	0	0	1
End	0	0	0	0	0	0	0	0	1	1	0

Строки и столбцы составлены из словаря этих двух предложений, из токенизированных слов обоих предложений. *Start* и *End* - метки начала и конца предложений.

Алгоритм ***GloVe*** основан на методе наименьших квадратов, поэтому при обучении нейросети *функция потерь (cost function)* строится по алгоритму ***TF-IDF*** для каждой совместно встречаемой пары слов, что позволяет устанавливать более низкие веса для часто встречаемых пар слов, тем самым понижая их важность.

Заключение

Анализ языковых моделей показал существенные достижения в области общения компьютера с человеком на естественном языке. Однако за достигнутые результаты пришлось заплатить большими вычислительными мощностями.

Бурное развитие методов обработки естественного языка позволяет реализовать человеко-машинный интерфейс с интеллектуальным роботом на его бортовых вычислительных средствах, однако выбор алгоритмов зависит от решаемой задачи. Это и обусловило комбинированного применения алгоритмов ***TF-IDF, Word2Vec*** и ***GloVe.***

Список литературы

1. Веб сайт [Электронный ресурс]-https://habr.com/ru/companies/airi/articles/764102/
2. Веб сайт [Электронный ресурс]- https://habr.com/ru/articles/561746/
3. Wenlong Huang, Pieter Abbeel, Deepak Pathak, Igor Mordatch Language Models as Zero-Shot Planners:Extracting Actionable Knowledge for Embodied Agents // International Conference on Machine Learning 18 January 2022
4. Веб сайт [Электронный ресурс]-https://ru.wikipedia.org/wiki/GPT-3
5. Веб сайт [Электронный ресурс]-https://news.microsoft.com/ru-ru/build-2020-supercomputer/
6. OpenAI GPT For Python Developers-Aymen El Amri
7. Веб сайт [Электронный ресурс]-https://neerc.ifmo.ru/wiki/
8. Веб сайт [Электронный ресурс]-https://habr.com/ru/articles/117063/
9. Веб сайт [Электронный ресурс]-https://ru.wikipedia.org/wiki/ Расстояние_Левенштейна
10. Веб сайт [Электронный ресурс]-https://pypi.org/project/textdistance/
11. Веб сайт [Электронный ресурс]-https://news.microsoft.com/ru-ru/build-2020-supercomputer/
12. Веб сайт [Электронный ресурс]-https://habr.com/ru/companies/sberbank/articles/649609/
13. Веб сайт [Электронный ресурс]-https://habr.com/ru/companies/sberdevices/articles/780334/

14. Веб сайт [Электронный ресурс]-https://habr.com/ru/companies/yandex/articles/739626/
15. Веб сайт [Электронный ресурс]-https://habr.com/ru/companies/yandex/articles/588214/
16. Веб сайт [Электронный ресурс]- https://clck.ru/38eMLu
17. Веб сайт [Электронный ресурс]-https://neptune.ai/blog/vectorization-techniques-in-nlp-guide

References

1. Website [Electronic resource]-https://habr.com/ru/companies/airi/articles/764102/
2. Website [Electronic resource]-https://habr.com/ru/articles/561746/
3. Wenlong Huang, Pieter Abbeel, Deepak Pathak, Igor Mordatch Language Models as Zero-Shot Planners:Extracting Actionable Knowledge for Embodied Agents // International Conference on Machine Learning 18 January 2022
4. Website [Electronic resource]-https://ru.wikipedia.org/wiki/GPT-3
5. Website [Electronic resource]-https://news.microsoft.com/ru-ru/build-2020-supercomputer/
6. OpenAI GPT For Python Developers-Aymen El Amri
7. Website [Electronic resource]-https://neerc.ifmo.ru/wiki/
8. Website [Electronic resource]-https://habr.com/ru/articles/117063/
9. Website [Electronic resource]-https://ru.wikipedia.org/wiki/Levenshtein_distance
10. Website [Electronic resource]-https://pypi.org/project/textdistance/
11. Website [Electronic resource]-https://news.microsoft.com/ru-ru/build-2020-supercomputer/
12. Website [Electronic resource]-https://habr.com/ru/companies/sberbank/articles/649609/
13. Website [Electronic resource]-https://habr.com/ru/companies/sberdevices/articles/780334/
14. Website [Electronic resource]-https://habr.com/ru/companies/yandex/articles/739626/
15. Website [Electronic resource]-https://habr.com/ru/companies/yandex/articles/588214/
16. Website [Electronic resource]-https://clck.ru/38eMLu
17. Website [Electronic resource]-https://neptune.ai/blog/vectorization-techniques-in-nlp-guide