1. Анализ естественного языка. Мотивация. Направления.

Обработка естественного языка (Natural language processing - NLP) – область, находящаяся на пересечении computer science, искусственного интеллекта и лингвистики. Цель заключается в обработке естественного языка для перевода текста и ответа на вопросы.

Как правило, проводится морфологический анализ: установление для слов некоторых инвариантных форм псевдооснов или лемм, установление морфологических признаков, таких как часть речи, число, род, падеж. Для разных языков набор морфологических признаков может различаться. Далее проводится синтаксический анализ: выделение синтаксических групп или установление синтаксических зависимостей между словами в рамках предложения. Дальше может выполняться семантический анализ, дискурсивный анализ и иногда сюжетный анализ текста. Многие задачи обработки естественного языка могут решаться без привлечения знаний о том, что такое язык, но большинство задач решается эффективно с применением знаний о языке и с выполнением лингвистического анализа текста.

Одной из первых стояла задача машинного перевода — автоматического перевода текста с одного языка на другой с помощью компьютера или вычислительной машины.

Вторая задача — создание диалоговых систем, которые ведут с человеком диалог на естественном языке.

Одна из основных проблем анализа естественного языка — это многозначность, которая проявляется на всех уровнях и обычно снимается за счет контекста и учета некоторой регулярности в использовании конструкций в языке. Вторая проблема заключается в том, что методы анализа текстов сильно зависят от языка, жанра, предметной области: анализ художественного текста не то же самое, что анализ новости или текста из социальных сетей, поэтому всегда требуется некоторая настройка.

**Где применяется NL**

* • поиск (письменный или устный);
* автоматический (или при содействии) перевод;
* распознавание речи и чат-боты,
* голосовые помощники.

**Векторное представление (text embeddings)**

В NLP слова рассматриваются как дискретные символы, которые далее представляются в виде one-hot векторов. Проблема со словами — дискретными символами — отсутствие определения cхожести для one-hot векторов. Поэтому альтернатива — обучиться кодировать схожесть в сами векторы.

Векторное представление — метод представления строк, как векторов со значениями. Строится плотный вектор (dense vector) для каждого слова так, чтобы встречающиеся в схожих контекстах слова имели схожие вектора. Векторное представление считается стартовой точкой для большинства NLP задач и делает глубокое обучение эффективным на маленьких датасетах.

**Word2Vec**

Принимает большой корпус текста, в котором каждое слово в фиксированном словаре представлено в виде вектора. Далее алгоритм пробегает по каждой позиции в тексте, которая представляет собой центральное слово c и контекстное слово o. Далее используется схожесть векторов слов для и , чтобы рассчитать вероятность при заданном (или наоборот), и продолжается регулировка вектор слов для максимизации этой вероятности. Word2vec представлен в 2 вариациях моделей:

Skip-Gram: рассматривается контекстное окно, содержащее последовательных слов. Далее пропускается одно слово и обучается нейронная сеть, содержащая все слова, кроме пропущенного, которое алгоритм пытается предсказать. Следовательно, если 2 слова периодически делят cхожий контекст в корпусе, эти слова будут иметь близкие векторы.

Continuous Bag of Words: берется много предложений в корпусе. Каждый раз, когда алгоритм видим слово, берется соседнее слово. Далее на вход нейросети подается контекстные слова и предсказываем слово в центре этого контекста.

**GloVe**

Стремится решить эту проблему захватом значения одного word embedding со структурой всего обозримого корпуса. Чтобы сделать это, модель ищет глобальные совпадения числа слов и использует достаточно статистики, минимизирует среднеквадратичное отклонение, выдает пространство вектора слова с разумной субструктурой.

**Машинный перевод**

Преобразование текста на одном естественном языке в эквивалентный по содержанию текст на другом языке. Делает это программа или машина без участия человека. В машинном переводе использутся статистика использования слов по соседству.

Нейросетевой машинный перевод — подход к моделированию перевода с помощью рекуррентной нейронной сети (Recurrent Neural Network, RNN). RNN — нейросеть c зависимостью от предыдущих состояний, в которая имеет связи между проходами. Нейроны получают информацию из предыдущих слоев, а также из самих себя на предыдущем шаге.

Главная проблема RNN — проблема исчезновения градиента, когда информация теряется с течением времени. Если вес примет значение 0 или 100000, предыдущее состояние не будет слишком информативно. Сети краткосрочной-долгосрочной памяти (Long/short term memory, LSTM) пытаются бороться с этой проблемой вводя гейты (gates) и вводя ячейку памяти. Каждый нейрон представляет из себя ячейку памяти с тремя гейтами: на вход, на выход и забывания (forget). Эти затворы выполняют функцию телохранителей для информации, разрешая или запрещая её поток.

**Голосовые помощники**

Исследователи из Microsoft и Facebook создали нейросеть, способную создавать чувствительные к контексту ответы в разговоре. Эта система может тренироваться на большом количестве неструктурированных диалогов в Twitter.

Разработанная в Гонконге нейронная машина для ответов (далее NRM — Neural Responding Machine) — генератор ответов для коротких текстовых бесед. NRM использует общий кодер-декодер фреймворк.

Последняя модель — Google’s Neural Conversational Model предлагает простой подход к моделированию диалогов, используя sequence-to-sequence фреймворк. Модель поддерживает беседу благодаря предсказанию следующего предложения, используя предыдущие предложения из диалога

**Вопросно-ответные (QA) системы**

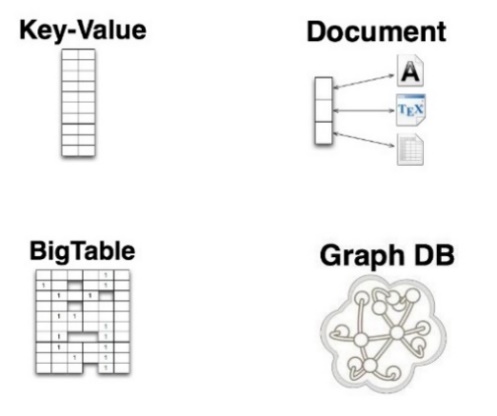
Идея систем заключается в извлечении информации непосредственно из документа, разговора, онлайн поиска или любого другого места, удовлетворяющего потребности пользователя. Вместо того, чтобы заставлять пользователя читать полный текст, QA системы предпочитают давать короткие и лаконичные ответы.

**Краткое изложение текста (Text Summarization)**

Извлечение краткого содержания (Text Summarization) – инструмент для помощи в интерпретации текстовой информации. Выделяют два фундаментальных подхода к сокращению текста: извлекательный и абстрактный. Первый извлекает слова и фразы из оригинального текста для создания резюме. Последний изучает внутреннее языковое представление, чтобы создать человекоподобное изложение, перефразируя оригинальный текст. Методы в извлекательном сокращении работают на основе выбора подмножества. LexRank и TextRank — хорошо известные представители этого подхода, которые используют вариации алгоритм сортировки страниц Google PageRank.

1. Классификация и различия NoSQL хранилищ

NoSQL использует разные хранилища данных для оптимизации для конкретных целей в зависимости от требований к системе, на рисунке 4 категории хранилищ данных NoSQL:



**1. key-value store - Berkeley DB, MemcacheDB, Redis, Riak, Amazon DynamoDB**

Сохраняет данные (значения) и применяет к ним метку (ключ) и сохраняет их либо в памяти, либо в системе хранения, оптимизированной для быстрого извлечения данных.

Значение обычно можно получить, только сославшись на его ключ, поэтому научиться запрашивать конкретную пару ключ-значение обычно просто. Базы данных типа "ключ-значение" отлично подходят для случаев, когда вам нужно хранить большие объемы данных, но вам не нужно выполнять сложные запросы для их получения.

**2. document store - CouchDB, Couchbase, MongoDB, Berkeley DB XML**

Предназначен для хранения, поиска и управления документальной информацией. Хранилища документов сохраняют всю информацию об объекте как экземпляр в базе данных, а не распределяют данные по разным таблицам.

Хранит данные в документах, подобных объектам JSON (JavaScript Object Notation). Каждый документ содержит пары полей и значений. Значения обычно могут быть различных типов, включая такие вещи, как строки, числа, логические значения, массивы или объекты, и их структуры обычно совпадают с объектами, с которыми разработчики работают в коде.

Может масштабироваться по горизонтали для размещения больших объемов данных.

**3. wide column database - Apache HBase, Apache Cassandra, Apache Accumulo, Amazon SimpleDB**

Использует таблицы, строки и динамические столбцы, отличные от реляционных баз данных, поскольку для каждой строки не требуется один и тот же столбец, что позволяет горизонтальное масштабирование базы данных. Пример использования - хранение профилей пользователей.

wide column database (Хранилища с широкими столбцами обеспечивают большую гибкость по сравнению с реляционными базами данных, поскольку не обязательно, чтобы каждая строка имела одинаковые столбцы. Хранилища с широкими столбцами отлично подходят, когда вам нужно хранить большие объемы данных и вы можете предсказать, какими будут ваши шаблоны запросов. Хранилища с широкими столбцами обычно используются для хранения данных Интернета вещей и данных профилей пользователей.

**4. graph database - Neo4j, MarkLogic, Ontotext, OrientDB**

Хранит данные в виде узлов и ребер. Узлы хранят данные объекта, а ребра хранят информацию о взаимосвязи узлов. Используется для просмотра взаимосвязей и закономерностей в данных. Варианты использования - социальные сети, искусственный интеллект и механизм рекомендаций.

Graph отлично подходят для тех случаев, когда вам нужно изучить отношения для поиска паттернов, таких как социальные сети, обнаружение мошенничества и механизмы рекомендаций.