

Графы знаний

Введение в графы знаний

Уважаемые слушатели, в этой лекции мы рассмотрим способ представления и хранения знаний, который стал стандартом де-факто в современных интеллектуальных системах. По сравнению с традиционными системами бизнес-правил, продукций или семантических сетей, отличительной особенностью графов знаний, является их логическая строгость, развитые возможности автоматического пополнения базы знаний, интеграция с хранилищами данных и масштабируемость на большие объемы информации, а также поддержка выполнения запросов подобно тому, как это делается в базах данных. Все это делает графы знаний очень удобным инструментом для работы со знаниями в сети Интернет.

Прежде чем говорить о знаниях в сети Интернет, давайте вспомним, насколько быстро растет сеть. Тот факт, что 90% информации в этом мире было сгенерировано за два последних года уже никому не кажется удивительным, но давайте попробуем взглянуть на эти числа. Перед нами годовая инфографика [Data never sleeps 6.0](#), иллюстрирующая производство информации и онлайн-поведение людей и разработанная с помощью платформы Domo, которая предоставляет инструменты для бизнес-аналитики и визуализации данных.

Как мы видим, темпы роста информации весьма и весьма внушительные. Только в США в среднем выкладывают в Интернет более 2,5 миллионов гигабайт данных каждую минуту. Но в контексте нашей лекции интересны даже не эти внушительные объемы информации, а их структура и всё более тесная взаимосвязь в данных. Для того, чтобы превратить разрозненную сеть веб-документов в самую большую в истории человечества коллекцию знаний, необходимо научиться извлекать эти знания и представлять их в форме, понятной человеку с одной стороны, но и подходящей для эффективной машинной обработки и анализа с другой.

Возможно, у вас уже возник вопрос, зачем мы подчеркиваем важность машиночитаемых и машинопонимаемых знаний? Ответ очень прост: чтобы автоматизировать рутинные процессы, которые отнимают у людей много времени, и

упростить более сложные процессы путем анализа исходных данных и возможных результатов.

Современной формой представления и хранения знаний в сети являются графы знаний — структуры, состоящие из уникальных сущностей (узлов графа) и связей между ними (ребер графа). Сущностями могут быть как материальные вещи, так и абстрактные концепции. Например, «*Университет ИТМО*» как материальный концепт и «*Университет*» как собирательное описание всех университетов. Связи (ребра графа) описывают отношения между сущностями и их атрибуты (также называемые свойствами). Например, «*дата основания*» — это атрибут для сущности «*Университет ИТМО*» и может содержать значение «*26.03.1900*», тогда как для абстрактного концепта «*Университет*» атрибут «*дата основания*» может содержать «*некоторую условную дату*» и создать высказывание, что каждый университет имеет дату основания. Таким образом, графы знаний позволяют моделировать как абстрактные логические высказывания и схемы, так и наполнять эти схемы конкретными объектами реального мира. Более того, графы знаний позволяют машинам производить рассуждения и выводить новые знания, ранее не описанные в графе. Формальная полуструктурированность и мощный логический аппарат отличают графы знаний от традиционных реляционных баз данных, которые являются структурированными, то есть обладают установленными связями и отношениями.

Исследование графов знаний стоит на стыке многих областей компьютерных наук:

- информационный поиск (information retrieval), позволяющий ускорить наполнение графа из различных источников;
- обработка естественных языков (natural language processing) и семантические технологии (semantic technologies), которые позволяют описывать и использовать при анализе смысл хранящихся знаний;
- системы управления базами данных (data management), обеспечивающие эффективное хранение графов;
- машинное обучение (machine learning), необходимое для анализа содержащихся в исходных данных знаний и генерации новых знаний;

- искусственный интеллект в целом (artificial intelligence) как общее теоретико-методологическое обобщение, изучающее возможности машинного интеллекта.

Графы знаний в их современном представлении имеют более чем 10-летнюю историю развития. Вероятно, наиболее ценными являются графы знаний, специфические для отдельных предметных областей. Но мы для простоты понимания их сути рассмотрим графы общего назначения. Основы и концепция открытых для пользователей графов знаний были впервые реализованы в 2007 в [базе знаний DBpedia](#), созданной в результате семантической обработки инфобоксов статей в [Wikipedia](#). Тогда как сам термин «граф знаний» ввела в обращение компания Google с ее [Google Knowledge Graph](#). Со временем в DBpedia добавилась подробная схема данных (онтология), географические данные и связи с другими графами. В настоящее время DBpedia считается одним из стандартов графов знаний и содержит более 6 миллиардов связанных фактов.

В 2008 году был разработан [граф YAGO](#). Его отличительная особенность в использовании семантического [тезауруса WordNet](#) и очень детальной иерархии классов сущностей. В настоящее время YAGO содержит около 120 миллионов фактов.

В 2010 году была запущена система Never-Ending Language Learner ([NELL](#)), которая «читает» веб-страницы и автоматически перемещается между ними, пытаясь выделять факты из текста веб-страниц в граф знаний. В настоящее время NELL содержит около 50 миллионов фактов, включая два миллиона фактов, в правдивости которых NELL полностью уверен.

Запущенный в 2007 году граф знаний Freebase использует отличный от трех предыдущих графов подход к моделированию фактов. Вместо использования заранее созданных схем данных (онтологий) Freebase позволяет пользователям самим назначать категорию описываемой сущности, что напоминает скорее облако тэгов, чем дерево классов. В 2014 году Freebase содержал около двух миллиардов фактов и был приобретен компанией Google, а затем преобразован в Google Knowledge Graph, предоставляющий единообразные знания всем сервисам компании, от поиска и почты до голосовых

помощников. Граф знаний Google значительно повысил интерес академического и бизнес-сообщества к задаче представления знаний, задав тренд на следующие годы.

Если DBpedia использует данные Wikipedia для наполнения графа, то разработанный и запущенный в 2012 году [граф Wikidata](#) предназначен для хранения знаний, которые будут использованы уже в Wikipedia (чаще всего в виде заполнения инфобоксов и таблиц на странице) на многих доступных языках. Wikidata использует усовершенствованный подход к моделированию знаний, позволяющий более детально описывать сущности и отношения. В настоящее время Wikidata содержит около 7 миллиардов фактов о более чем 50 миллионах сущностей. Большинство публикуемых в последнее время графов знаний старается использовать модель Wikidata или связывать свои сущности с имеющимися в Wikidata.

Доступные в Интернете графы знаний образуют облако связанных данных — Linked Open Data Cloud ([LOD Cloud](#)), семантически объединяя опубликованные графы в одну гигантскую сеть. И если в 2007 году это облако состояло всего из 12 графов, то в 2018 оно выросло до 1234 графов в девяти разных доменах.

Сценарии использования графов знаний

Рассмотрим различные сценарии использования графов знаний. Пожалуй, самый массовый сценарий на данный момент — это обогащение результатов поиска Google с помощью Google Knowledge Graph. Например, при поиске человека общая информация о нем (при наличии в графе) выводится в отдельном инфобоксе. При поиске ресторанов выводится место, ожидаемая кухня и примерная цена ужина. В дополнение, поисковая система умеет отвечать на простые фактологические вопросы: когда родился определенный человек или где находится определенное место.

Ответы на более сложные вопросы, заданные на естественном языке, могут быть решены с помощью графов DBpedia или Wikidata, когда исходный запрос анализируется на предмет нахождения сущностей, описанных в графе. Полученная комбинация сущностей и связей образует множество подграфов, некоторые из которых могут содержать ответ. Эти

подграфы переписываются в запрос к графу на формальном языке (например, на языке SPARQL, который мы рассмотрим позднее), и оценивается верность результата.

Интеграция данных из разнородных источников, особенно, когда источники представлены в разных форматах (CSV, XML, JSON, реляционные базы) и используют разные схемы данных. В этом случае графы знаний могут служить универсальным средством связи и интеграции этих источников (датасетов), как физически (когда данные и схема преобразуются в единое RDF представление), так и виртуально (когда данные остаются в исходных форматах, но граф знаний на уровне абстрактных схем интегрирует исходные датасеты). В таком случае запрос к графу знаний будет проанализирован и переписан для отправки в нативном формате конкретного источника. Другими словами, один SPARQL-запрос может быть разбит на подзапросы и переписан для исполнения в CSV, XML и JSON источниках.

Активно развивающаяся сейчас в сфере производства [концепция Индустрии 4.0](#) постулирует необходимость как можно большей автоматизации кибер-физических систем (например, роборука на сборочном производстве) и их взаимодействия между собой без участия человека. Для этого сценария использование графов знаний позволяет описывать технологический процесс производства конкретных изделий во взаимосвязи с конкретными технологическими операциями на конкретных машинах. Это, в свою очередь, позволяет роботам самостоятельно договариваться о необходимых действиях по обработке каждого изделия, обеспечивая массовое производство высоко кастомизированных продуктов, что и является одним из видений Индустрии 4.0. На сегодняшний момент рабочими группами, стандартизирующими эту концепцию, предусматривается использование стандарта RDF и подхода linked data, которые являются одними из основ графов знаний.

Онтологическое представление знаний в графах

Графы знаний характеризуются детальным описанием содержащихся в них сущностей и машинопонимаемыми логическими связями между ними. Давайте разберем два способа представления графов знаний. Первый способ, онтологическое представление, основан на формальной логике и семантике. Второй способ векторных представлений использует статистические механизмы для минимизации расстояний между близкими сущностями в многомерных пространствах.

Формальная логика и семантика для онтологического представления графов знаний опирается на такие стандарты консорциума WWW как [RDF](#) и [OWL](#). Стандарт RDF (Resource Description Framework) определяет модель триплета как:

- «*субъект - предикат – субъект*»;
- «*субъект - предикат – объект*».

То есть сущность — «*субъект*» может быть связана с другой «*сущностью*» или «*объектом*» через некоторое свойство — «*предикат*». При этом описываемые сущности и предикаты имеют уникальные идентификаторы. Таким образом, RDF высказывания образуют ориентированный граф. Например, триплет «*Университет ИТМО - находится в - Санкт-Петербург*» связывает именованные сущности «*Университет ИТМО*» и «*Санкт-Петербург*» посредством предиката «*находится в*». А триплет «*Университет ИТМО - основан - 26.03.1900*» связывает атрибут «*основан*» сущности «*Университет ИТМО*» со значением даты «*26 марта 1900 года*».

Логический базис RDF расширяется родственными стандартами [RDFS](#) (RDF Schema) и OWL (Web Ontology Language), которые вводят базовые аксиомы формальной логики, в том числе дескрипционной логики, что позволяет осуществлять логический вывод на основе триплетов. А это, в свою очередь, позволяет генерировать новые знания на основе уже имеющихся. Например, если «*находится в*» определить как транзитивный предикат, или отношение, обладающее свойством транзитивности, то при добавлении триплета «*Санкт-Петербург - находится в - Россия*» можно получить триплет (факт) «*Университет ИТМО - находится в - Россия*».

Выразительные и логические возможности стандартов настолько широки, что с их помощью можно моделировать высказывания практически любой сложности. Для выражения абстрактных концепций стандарты вводят понятия классов, иерархий классов и предикатов, экземпляров, области определения, области значений. Например, *«Университет ИТМО - является экземпляром класса - Университет», «Университет - является подклассом - Учебное заведение»*.

Для построения иерархии свойств можно определить: *«в городе - является подсвойством - находится в», «количество учащихся - имеет область определения - Учебное заведение», «количество учащихся - имеет область значений – число»*. Все они могут быть составными (сложными) и расширены с помощью аксиом формальной логики. То, что у нас получилось с добавлением новых триплетов, и есть базовый граф знаний, который мы рассматриваем в этой лекции.

Как же граф знаний в этом случае отличается от реляционных баз данных? Во-первых, семантическая модель RDF по своей природе полуструктурированная, то есть, если в схеме данных существует триплет *«количество учащихся - область определения - Учебное заведение»*, то это НЕ значит, что каждый экземпляр *«Учебного заведения»* ОБЯЗАН иметь значение свойства *«количество учащихся»*. Граф корректен и без такого предиката, тогда как в реляционных базах данных если введено такое отношение, то ему обязательно должно быть присвоено некоторое значение, пускай даже N/A, если точное число неизвестно. Из этого вытекает вторая отличительная особенность графов знаний RDF: они используют модель открытого мира вместо модели закрытого мира, присущей классическим базам данных. Другими словами, если некоторый факт явно не присутствует в графе, то на соответствующий вопрос о существовании данного факта можно ответить «мы не знаем», тогда как в реляционных моделях ответ всегда будет «нет».

Часть графа знаний, описывающая абстрактные концепции и связи между ними на высоком уровне, иначе еще называется онтологией. Существует множество определений понятия онтология, но все они сходны в том, что онтология — формализованная модель некоторой области знаний, согласованная с экспертами этой области. В процессе наполнения такой модели экземплярами реальных данных и получается граф знаний.

Онтологии могут быть разной степени выразительности — например, «*Университет - является подклассом - Учебное заведение*» представляет собой довольно простое высказывание, а, например, «*Университет - является подклассом - Учебное заведение И обеспечивает - Высшее образование И проводит - Научные Исследования*» дает более детальное описание концепции университета. Таким образом, чем выразительнее онтология, тем больше применений она имеет, например, для вывода новых знаний или проверки корректности поступающих данных для записи в граф знаний. Создание онтологий (и графов знаний в целом) — тема отдельной лекции. Здесь же мы скажем, что это чаще всего итеративный процесс, включающий в себя взаимодействие с экспертами определенной области знаний и инженерами по знаниям, которые способны эти знания записать на формальном языке онтологий. Активно развивающаяся область исследований — автоматическое создание онтологий без участия человека из слабоструктурированных данных (например, текста или таблиц).

Для хранения графов знаний в онтологическом представлении (в RDF) используются графовые базы данных или близкие к ним решения, например, колоночные базы или JSON-базы (если преобразовывать граф в JSON представление). Стандартным языком запросов для RDF-баз является язык [SPARQL](#). Этот акроним означает SPARQL Protocol and RDF Query Language. Обратите внимание, что это рекурсивный акроним, наподобие GNU — GNU's Not Unix. SPARQL способен эффективно использовать графовую модель представления знаний и имеет базовые механизмы логического вывода. Если язык SQL подразумевает табличную организацию баз данных, то SPARQL изначально создавался именно для графовых данных. Основной строительный блок SPARQL-запроса — образ или шаблон подграфа графа (graph pattern), который задает топологию перемещения по узлам и ребрам графа.

Пусть у нас есть исходный граф, состоящий из следующих триплетов:

ITMO_University	rdf:type	University .
ITMO_University	rdf:type	Research_Organization .
ITMO_University	locatedIn	Saint_Petersburg .
University	rdfs:subClassOf	Educational_Institution .

Рассмотрим следующий запрос получения всех семантических типов Университета ИТМО в некотором графе:

```
SELECT ?type WHERE {  
  ITMO_University rdf:type ?type .  
}
```

Образ подграфа содержит один триплет с одной неизвестной переменной *?type*. Система исполнения запросов совмещает образ графа в запросе с подграфом, связанным с «*ITMO University*», и фильтрует предикаты по названию (*rdf:type*). В результате остаются только исходящие ребра предиката *rdf:type*, а конечные узлы этих ребер и являются ответом на данный запрос — в нашем случае это будут сущности *University* и *Research_Organization*.

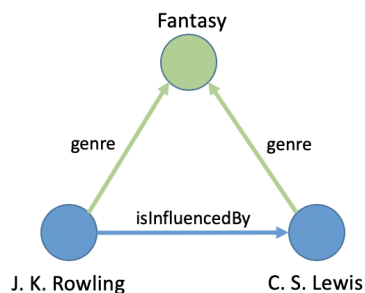
Статистическое представление знаний в графах

Статистическое представление графов знаний опирается на аппарат теории вероятностей. Одним из видов такого представления является назначение каждому факту, записанному в общем случае в виде логической функции предикат, имеющей аргументы (объект1, объект2), некоторого значения в интервале $[0, 1]$, которое говорит, насколько правдиво это утверждение. Более распространение в машинном обучении получил способ статистического представления графов знаний в многомерном векторном пространстве, которое мы и рассмотрим в этой лекции более подробно. Известная модель представления слов в векторном пространстве (word embeddings) word2vec, а также основанные на ней другие модели, показали, насколько эффективными могут быть векторные представления в задачах обработки естественного языка, полученные на большом массиве исходных данных. Такие представления слов в виде чисел сохраняют статистические отношения слов, встречающиеся в тексте, и даже некоторую скрытую семантику, располагая близкие по смыслу слова поблизости в векторном пространстве. Например, векторы слов «*Москва*» и «*Берлин*» будут располагаться ближе, чем «*Москва*» и «*Сахара*». Более того, вектор «*Москва*» будет относиться к вектору «*Россия*» примерно, как вектор «*Берлин*» к вектору «*Германия*».

Векторные представления слов позволяют применять более комплексные методы машинного обучения, например, нейронные сети, для решения большого количества задач — от распознавания имен собственных до систем машинного перевода и диалоговых менеджеров.

Мы будем недалеко от истины, если скажем, что векторные представления слов (word2vec, GloVe, FastText) произвели революцию в области обработки естественных языков. Аналогичный подход можно применить и для усовершенствования обработки графов знаний. Для этого необходимо представить элементы графа в некотором высокоразмерном пространстве, только вместо слов — триплеты, а обучающий массив — граф знаний. Чем больше граф, больше триплетов и связей в нем, тем точнее можно восстановить векторные представления утверждений и их отношений между собой.

Рассмотрим, как можно построить векторные представления для графов знаний. Если [word2vec](#) основан на частоте встречи и упоминания одного слова среди множества других, то для графов знаний основной критерий — минимизация функции расстояния между верными и схожими утверждениями, и максимизация между ложными (некорректными).



Например, на данном графе знаний имеются следующие верные утверждения:

- «Джоан Роулинг - находилась под влиянием - Клайва Льюиса»;
- «Джоан Роулинг - пишет в жанре - Фэнтези»;
- «Клайв Льюис - пишет в жанре - Фэнтези».

Тогда как, следующие утверждения являются ложными для указанного графа:

- «Джоан Роулинг - находилась под влиянием - Фэнтези»;
- «Клайв Льюис - пишет в жанре - Джоан Роулинг».

Возвращаясь к вопросу минимизации и максимизации расстояния, обратимся к следующему примеру. Вектор представления факта «*Университет ИТМО - экземпляр класса - Университет*» будет ближе к факту «*СПб Государственный Университет - экземпляр класса – Университет*» и дальше от «*Университет ИТМО - экземпляр класса – Библиотека*», а сущности «*Университет ИТМО*» и «*СПб Государственный Университет*» будут ближе друг к другу, нежели чем «*Университет ИТМО*» и «*Публичная библиотека*».

Разные методы подсчета векторных представлений используют различные метрики или функции расстояния. В результате каждой сущности и каждому отношению ставится в соответствие числовой вектор некоторой размерности. Большинство методов используют 50-75-мерные пространства ввиду того, что чем больше граф знаний, тем сложнее выполнить условия минимизации и максимизации расстояний в привычных маломерных пространствах. То есть, если объекты и связи между ними наделить всего 2-мя координатами на плоскости или даже 3-мя в пространстве - представление большого графа будет крайне грубым. Чем выше размерность, тем точнее представление. Такие векторы содержат скрытую или неявную семантику, которая не может быть непосредственно интерпретирована человеком, то есть мы не можем прямо сказать, что описывает первое измерение, а что 75-е измерение. Как правило, это сложная комбинация вероятностных параметров, обрабатываемая алгоритмом за множество итераций. Чем больше размер исходного графа, тем точнее векторные представления описывают скрытые связи между сущностями и отношениями. Однако, с другой стороны, с ростом графа растут и вычислительные требования, и векторные представления для больших графов могут потребовать слишком больших объемов оперативной памяти.

Векторное представление знаний в графах

Рассмотрим [алгоритм TransE](#) для получения векторного представления графов знаний. Представим триплет «субъект - предикат - объект» как кортеж (h, r, t) , где h (head) — субъект, r (relation) — предикат, t (tail) — объект. При этом будем считать, что эти кортежи формируют некоторое множество S , т.е. кортеж (h, r, t) принадлежит множеству S

$$(h, r, t) \in S,$$

которое называют тренировочным множеством. Субъекты и объекты формируют множество сущностей E , т.е. пары (h, t) принадлежат множеству E

$$(h, t) \in E,$$

а r принадлежащее множеству M образует множество связей между сущностями

$$r \in M.$$

Алгоритм TransE основан на предположении, что в векторном пространстве сумма векторов h и r будет примерно равна t . Другими словами, TransE вводит некую меру различия d , а затем сэмплирует некорректные факты из графа знаний, заменяя у данного триплета субъект h или объект t другой случайной сущностью h' или t' из графа. Задача TransE — минимизировать расстояние между схожими корректными утверждениями (триплетами) и максимизировать его между некорректными. Для этого вводится минимизируемая функция потерь L с параметром $\gamma > 0$, обозначающим минимальный зазор между областями корректных и некорректных фактов.

$$L = \sum_{(h,r,t) \in S} \sum_{(h',r,t') \in S'_{(h,r,t)}} \max(0, (\gamma + d(h + r, t) - d(h' + r, t')))$$

Это означает, что суммировать расстояния необходимо только в том случае, если выражение положительно, и имеется следующее множество

$$S'_{(h,r,t)} = \{(h', r, t) | h' \in E\} \cup \{(h, r, t') | t' \in E\}.$$

Процесс обучения начинается с инициализации координат векторов h, r, t случайным образом в некотором пространстве, а гиперпараметр γ как правило задается заранее, определяя баланс между точностью модели и скоростью ее построения.

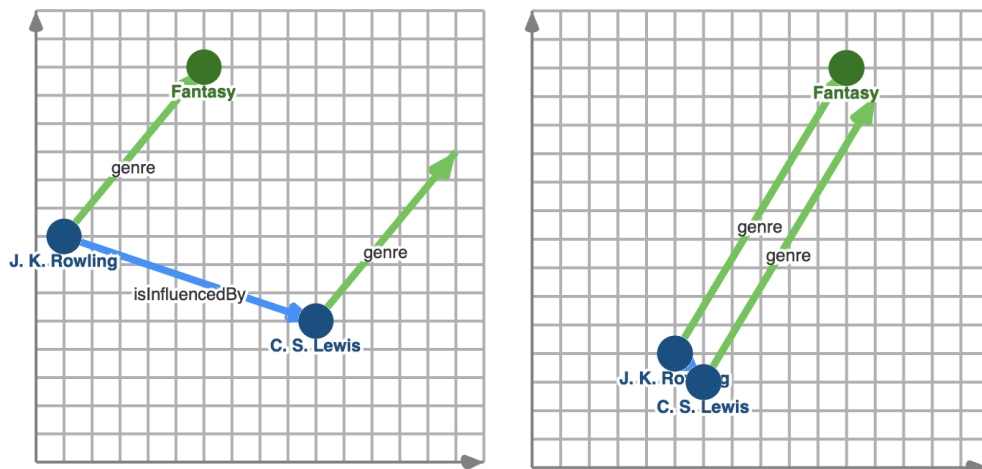
Если качестве функции расстояния $d(h + r, t)$ использовать квадрат евклидовой длины, который определен следующим образом

$$d(p, q) = \sum_{k=1}^n (p_k - q_k)^2,$$

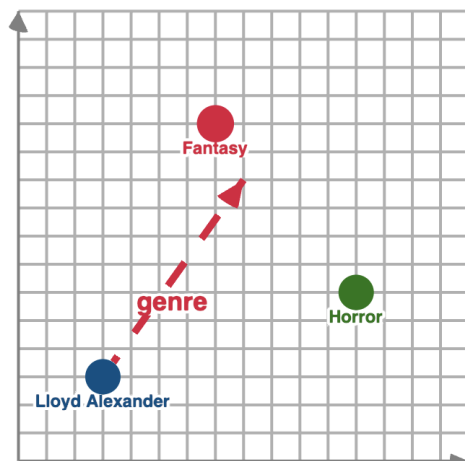
а также наложить ограничение так, чтобы суммы квадратов координат векторов h и t были равны 1, то выражение для функции потерь L можно переписать в таком виде:

$$L = \sum_{(h,r,t) \in S} \sum_{(h',r',t') \in S'_{(h,r,t)}} \max(0, (\gamma - 2(h^T t + r^T (t - h)) + 2(h'^T t' + r'^T (t' - h'))))$$

Обучение производится, как правило методом стохастического градиентного спуска и занимает до тысячи итераций, называемых также эпохами. [Визуализация такого процесса](#) на плоскости будет выглядеть следующим образом: от эпохи 1 до эпохи 3 расстояние между схожими сущностями уменьшилось («писатели»), а сущность «Фэнтези» удалилась на большее расстояние от них. При этом можно наблюдать некое несоответствие, вектор «жанр» от «Клайва Льюис» не попадает в сущность «Фэнтези». За это и отвечает гиперпараметр гамма.



В результате такого обучения модели, и отображения всего графа знаний в некотором пространстве, мы можем предсказать новые связи между сущностями. Например, если мы хотим предсказать в каком жанре писал «Ллойд Александер», используя полученные значения для вектора «жанр» мы с большей долей вероятности выберем жанр «Фэнтези», так как он ближе, чем жанр «Хоррор».



Как и с word2vec, для популярных в исследовании графов, таких как Freebase 15K или WordNet, существуют уже готовые подсчитанные TransE векторные представления.

Модель TransE предназначена для довольно простых графов. В частности, моделью не поддерживаются отношения «один ко многим» или «многие ко многим». Позднее появилось семейство моделей, называющееся [TransX](#), решающее те или иные концептуальные проблемы своего прародителя. Например, в [TransH](#) поддерживаются отношения «один ко многим» и «многие ко многим» с помощью проекции векторов сущностей и предикатов на специальную гиперплоскость с новой функцией, а [TransR](#) проецирует векторы сущностей в пространство векторов свойств. Новые подходы используют, например, вращение векторов в пространстве ([rotational embeddings](#)) или сверточные графовые нейронные сети ([ConvE](#)).

Векторные представления, как правило, хранятся в виде многомерных числовых массивов (тензоров). Для использования их нужно сперва загрузить в память. С ростом графа за счет роста количества сущностей и связей или за счет увеличения размерности векторных представлений растут и требования к памяти. Из-за этого подсчитать и использовать векторные представления самых больших графов знаний (Wikidata, DBpedia) в настоящий момент очень сложно.

Применение графов знаний

Итак, представим, что из множества исходных данных в определенной области мы построили полноценный граф знаний. Что же с ним делать и как графы знаний в принципе применяют? Прежде всего, графы знаний получили широкое применение в бизнесе.

Они способствуют расчету комплексной бизнес-аналитики. Например, одна из самых дорогих частных компаний [Palantir](#) использует динамические OWL онтологии и строит графы знаний для анализа контрагентов и цепочек поставок (так называемый corporate intelligence), поиска несоответствий в финансовых документах (financial compliance). Некоторые другие области включают здравоохранение, кибербезопасность, производство. Facebook поддерживает свой [Social Graph](#) и использует его среди прочего для анализа связей между людьми и рекомендательными системами по рекламе.

Графы знаний упрощают интеграцию данных из разнородных источников, например, корпоративных информационных систем, представленных во множестве форматов в data warehouse или data lake. Такие решения как [PoolParty Semantic Suite](#) от Semantic Web Company или [Ultrawrap](#) от Capsenta создают и управляют корпоративными графами знаний, собираемыми из реляционных баз данных или слабоструктурированных источников.

Собственные графы знаний лежат в основе чат-ботов Amazon Alexa, Google Assistant, Apple Siri, которые могут ответить на самый широкий спектр вопросов от определения геолокации, например для поиска ближайших ресторанов или исторической справки о каком-либо местонахождении, до запросов о недвижимости и медицинских услугах.

Также создаются открытые графы знаний для множества предметных областей: медицина, лингвистика, технические и социальные науки. Такие графы аккумулируют знания не только в человекочитаемой форме как Wikipedia, но и в машиночитаемой, создавая базис для обучения машин решению интеллектуальных задач в этих областях. Так, например, портал [Bio2RDF](#) является крупнейшим графом знаний медико-биологических наук. Граф объединяет 35 датасетов из 11 миллиардов триплетов, включая в том числе научные публикации из архива PubMed, который ведется с середины XX века, а также историю клинических испытаний лекарств, базы лекарственных средств, датасеты-

описания белков, катализаторов, генов, их реакций, и многое-многое другое. Таким образом, в машиночитаемом виде, пригодном для обучения, представлена практически вся история достижения человечества в области наук о жизни.

Наконец, помимо бизнес-задач графы знаний активно применяются в исследовании технологий обработки естественных языков и искусственного интеллекта в целом. А именно:

Дополнение графов знаний и автоматический вывод новых фактов (knowledge graph completion, link prediction), которые позволяют решить задачу о том, как имея набор фактов в символьном (онтологическом) или векторном представлениях, вывести на их основе новые знания об описываемых сущностях. Соответственно, в онтологическом представлении чаще всего используются механизмы логического вывода, а в статистическом представлении — механизмы линейной алгебры, теории вероятностей и статистики.

Определение семантического сходства сущностей, которые в общем случае могут быть записаны синтаксически по-разному. Например, два лекарственных средства, выпускаемые под разными брендами, но имеющие одно и то же действующее вещество в основе. В онтологическом представлении графа знаний о лекарственных средствах сходство можно обнаружить, проследив, например, классовую иерархию или топологию подграфов, узлы которых могут ссылаться на одни и те же узлы большего графа. В векторном представлении графов может использоваться косинусный коэффициент сходства векторов. Найденное сходство, однако, может быть неинтерпретируемо, или, иначе говоря, невозможно будет определить каким образом это значение сходства получено из-за высокой размерности векторного пространства.

Проверка корректности утверждений (fact-check) может применяться как для рассуждений об абстрактных вещах, например “Во всех ли университетах есть студенты”, так и для проверки высказываний из новостных заметок, которые часто страдают от фактологических ошибок или заведомо ложных утверждений.

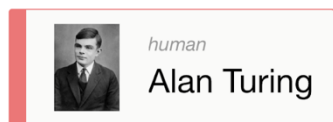
Распознавание именованных сущностей и извлечения отношений между ними (Named entity recognition и relation linking) на основе графов знаний. Как определить,







является ли слово в тексте именованной сущностью некоторого графа знаний? Например, в предложении «Никола Тесла изучал явления электромагнетизма», Тесла — человек, физик и ученый, а в предложении «Тесла — американская компания по производству автомобилей и батарей», Тесла уже компания, а в предложении «На экваторе магнитная индукция примерно равна 30 микротесла», тесла — единица измерения.

Рассмотрим, как можно строить графы знаний при анализе текстов на естественном языке. Пусть на вход подается первый параграф статьи из Wikipedia, например, об Алане Тьюринге:

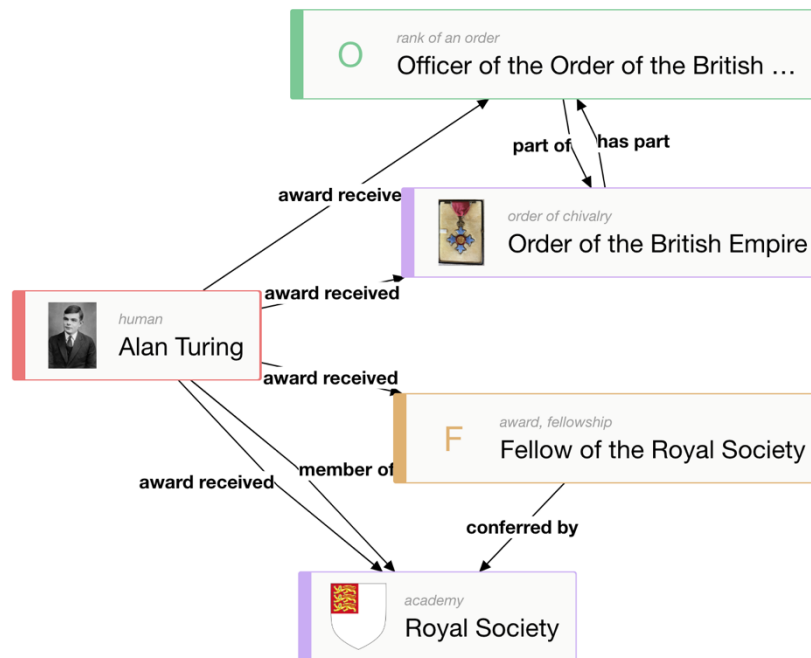
Alan Mathison Turing OBE FRS (23 June 1912 – 7 June 1954) was an English mathematician, computer scientist, logician, cryptanalyst, philosopher and theoretical biologist. Turing was highly influential in the development of theoretical computer science, providing a formalisation of the concepts of algorithm and computation with the Turing machine, which can be considered a model of a general-purpose computer. Turing is widely considered to be the father of theoretical computer science and artificial intelligence.

Наша задача здесь — выделить именованные сущности и связи между ними, объединив получившиеся факты в граф. Для этого мы используем платформу Metaphacts, подключенную к большому графу Wikidata. Прочитав текст, как правило, можно выделить несколько опорных точек, с которых можно начинать построение графа. Например, в случае Алана Тьюринга, OBE и FRS — аббревиатуры, свидетельствующие о наградах.

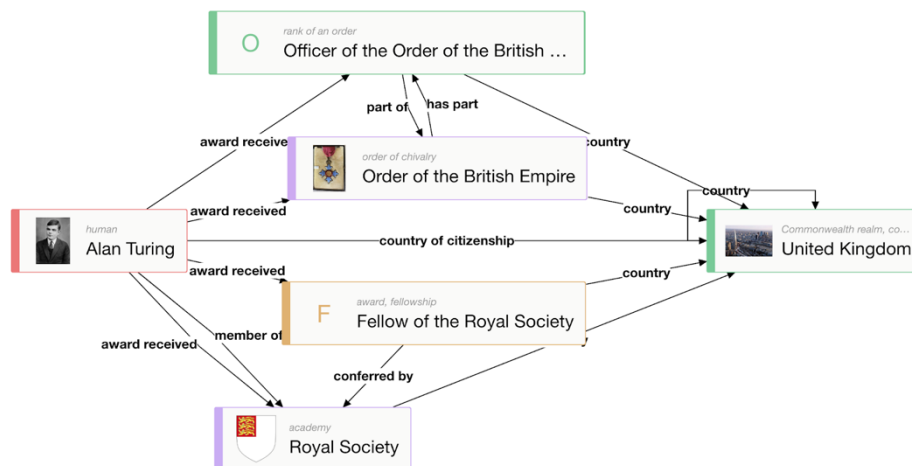


Connections		
<input type="text" value="Search for..."/>		
All	100+	+>
	author	15 +>
	award received	4 +>
	category's main topic	1 +>
	cause of death	1 +>
	child	2 +>
	commemorates	2 +>

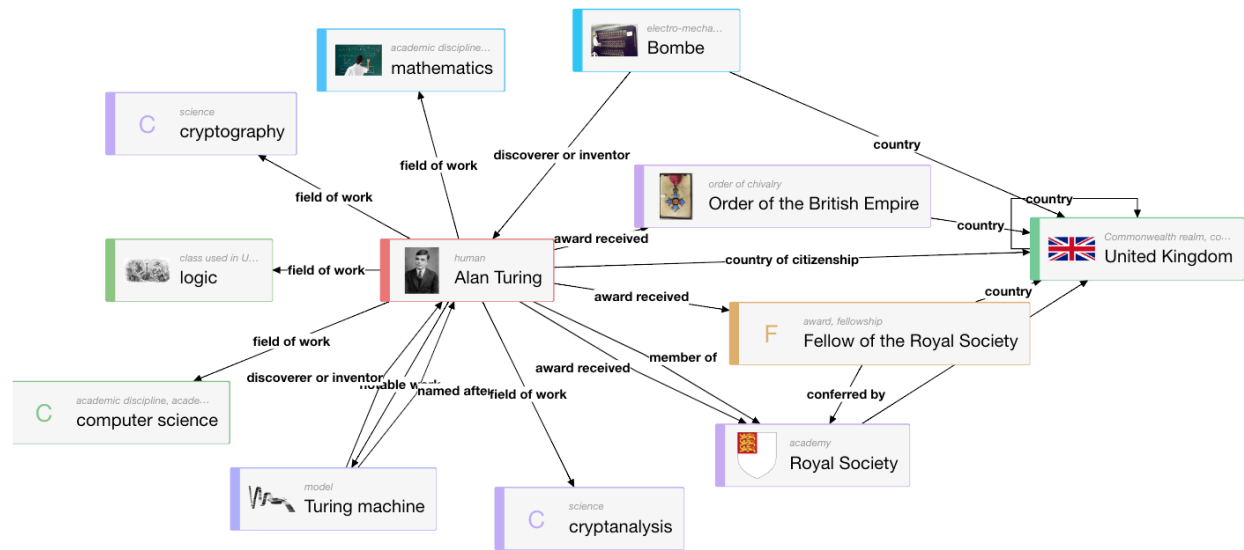
Им соответствует предикат «award received», объекты которого можно поместить на граф.



Далее можно заметить, что в тексте говорится о гражданстве Тьюринга. Добавим предикат «country of citizenship». При этом, граф автоматически пополнится связями между наградами и страной (в случае наличия этих связей).



Продолжая анализировать текст, найдем предикат «field of work», который связан с математикой, логикой, криптографией и computer science. Значительные достижения можно обнаружить в предикатах «notable work» или «discoverer/inventor», где мы найдем концепцию машины Тьюринга. Выбрав эти предикаты и объекты и расположив их на полотне, получим примерно следующий граф:



В этой лекции мы рассмотрели теоретические и практические аспекты области графов знаний. Появившись сравнительно недавно, графы знаний быстро завоевали популярность в технологичных отраслях за счет мощного теоретического базиса, широких аналитических возможностей и высокой масштабируемости. Несомненно, это направление исследований открывает новую эру в компьютерном представлении знаний, и мы станем свидетелями новых достижений в этой области, которые сделают следующий шаг на пути к построению сильного искусственного интеллекта.