A hierarchical co-clustering approach for entity exploration over Linked Data

Ко-кластеризация для исследования сущностей в LD

Высоцкая Полина, Маринина Дарья, Прошин Антон, Мурманцева Злата, Давыдов Ален

Аннотация

С увеличением объёма связанных данных (Linked Data) в сети Интернет пользователям становится всё труднее быстро находить интересующие их сущности для дальнейшего анализа и изучения.
Одним из фундаментальных методов решения этой проблемы является кластеризация, которая позволяет организовывать сущности в осмысленные группы.

Как правило, ссылки (links) и классы сущностей (entity classes) обладают семантическими метками и могут использоваться для группировки связанных объектов.
Однако каждая сущность обычно связана со многими различными ссылками и классами, что приводит к информационной перегрузке.

Аннотация

В данной работе предлагается новый иерархический подход совместной кластеризации, который позволяет одновременно группировать как ссылки, так и классы сущностей. В рамках подхода вводятся меры внутрисходства между ссылками (intra-link similarity) и внутрисходства между классами (intra-class similarity), которые затем интегрируются в единый процесс совместной кластеризации. Предложенный метод реализован в разработанном авторами браузере связанных данных CoClus. Для оценки эффективности подхода проведено прикладное исследование с участием пользователей, в котором CoClus сравнивался с тремя другими браузерами связанных данных.

Результаты экспериментов показали, что предложенный метод обеспечивает пользователю более эффективную поддержку при исследовании сущностей.

Введение

Связанные данные могут быть представлены в виде набора троек, построенных по модели

Resource Description Framework (RDF)

На рис. показано RDF-описание Стивена Спилберга в базе ***DBpedia***

dbr: указывает на реальные объекты (resources) – статьи/сущности DBpedia

dbo: указывает на понятия онтологии (ontology) – типы свойств и классов

The description of **Steven Spielberg** in DBpedia

```
@prefix dbr: <http://dbpedia.org/resource/>
@prefix dbo: <http://dbpedia.org/ontology/>
<dbr: Steven_Spielberg, dbo: birthDate, 1946-12-18>
<dbr: Steven_Spielberg, dbo: birthPlace, dbr: Cincinnati >
<dbr: Steven_Spielberg, dbo: residence, dbr: Los_Angeles >
<dbr: Steven_Spielberg, dbo: spouse, dbr: Amy_Irving >
<dbr: Jurassic_Park, dbo: director, dbr: Steven_Spielberg >
<dbr: A.I., dbo: director, dbr: Steven_Spielberg >
<dbr: A.I., dbo: producer, dbr: Steven_Spielberg >
<dbr: The_Pacific, dbo: producer, dbr: Steven_Spielberg >
...
```

(a)

Введение

Сущность *Steven Spielberg* в DBpedia имеет **117** связанных объектов

Чтобы найти нужные из них, пользователю пришлось бы просмотреть все эти связи и вручную выбрать релевантные

Одним из основных методов организации большого объёма данных является ***кластеризация*** – процесс группировки объектов данных в несколько осмысленных групп на основе различных мер сходства, таких как текстовое сходство, совместное цитирование или SimRank

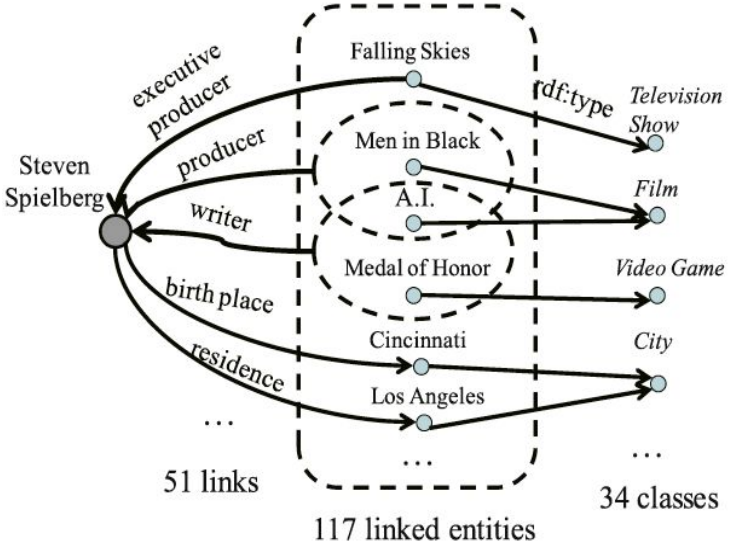
Ссылки (links) и классы сущностей (entity classes) являются двумя ключевыми аспектами описания сущностей, они часто используются для группировки связанных объектов, обеспечивают семантические метки, которые можно применять как подписи для создаваемых кластеров. Сущности часто связаны со множеством разных ссылок и классов, что создает когнитивную перегрузку

Введение

На рисунке показан пример просмотра сущности Steven Spielberg

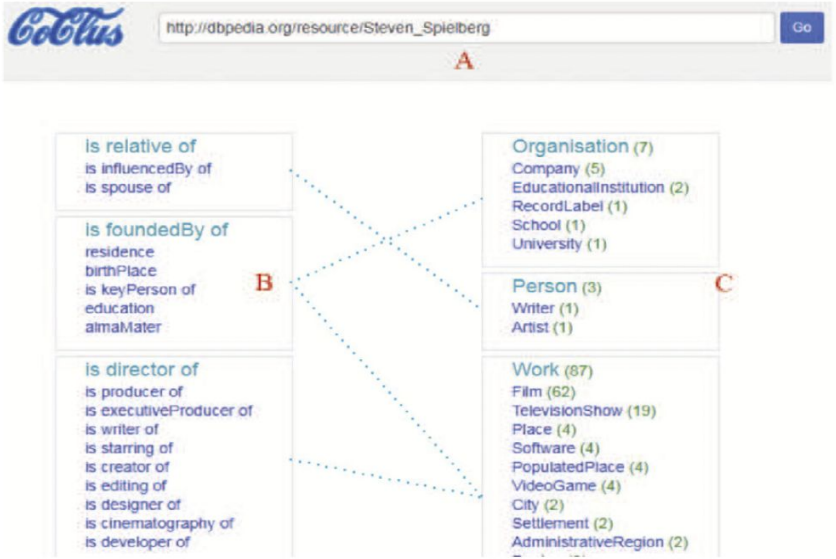
Чтобы сократить информационную перегрузку, ранее предпринимались попытки ранжировать ссылки и классы по различным критериям (например, по частоте использования или энтропии) или организовывать их иерархически

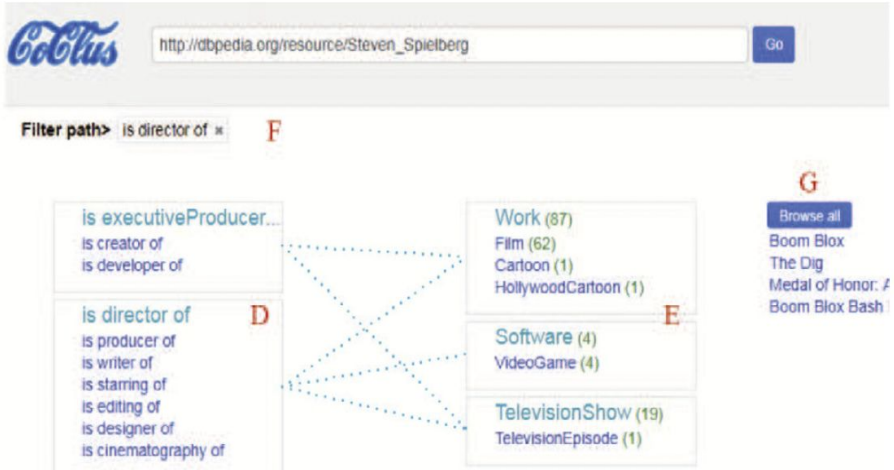
Однако существующие подходы учитывают только базовые элементы знаний (сами ссылки и классы) и игнорируют внутреннее сходство между ними, возможные взаимосвязи между ссылками и классами



Введение

Чтобы облегчить исследование сущностей, предлагается новый иерархический метод совместной кластеризации ***(hierarchical co-clustering)***, позволяющий одновременно группировать ссылки и классы сущностей





Основные результаты и вклад работы:

- Проведён обзор существующих мер сходства и алгоритмов совместной кластеризации. Предложен новый иерархический метод совместной кластеризации, который сочетает идеи иерархической кластеризации с информационно-теоретическим подходом. *В отличие от предыдущих методов, он избегает проблемы попадания в локальные минимумы при случайной инициализации, не требует пересчёта распределений по всему набору данных на каждой итерации*
- Проведена эмпирическая оценка эффективности метода, реализованного в системе CoClus. С помощью пользовательского эксперимента CoClus был сопоставлен с ограниченными версиями системы и традиционным браузером связанных данных. *Результаты показали статистически значимое превосходство по удобству и эффективности навигации*
- Предложенный алгоритм сравнивался с тремя базовыми алгоритмами совместной кластеризации, и по метрике Clustering Index показал лучшие результаты

Обзор литературы

Поскольку предлагаемая работа опирается на метод иерархической совместной кластеризации (hierarchical co-clustering) для исследования сущностей, в этом разделе мы отдельно рассмотрим предыдущие исследования в областях:

- кластеризации
- совместной кластеризации
- иерархической кластеризации
- кластеризации в контексте связанных данных (Linked Data)

Обзор литературы. Кластеризация и меры сходства

Кластеризация является одной из ключевых техник обучения без учителя, предназначенной для группировки схожих объектов и предоставляющей эффективный инструмент исследования и структурирования информации

В области поиска и извлечения документов, авторы предложили кластерный подход, позволяющий пользователю просматривать крупные коллекции документов по тематическим группам

В сфере веб-поиска кластеризация запросов и документов используется для идентификации наиболее популярных тем, поиска синонимичных выражений и повышения точности извлечения информации

Кластеризация помогает уточнять размытые поисковые запросы, визуализируя пользователю наиболее релевантные темы среди найденных результатов

Обзор литературы. Кластеризация и меры сходства

Задача: определение адекватных функций сходства и информативных признаков, обеспечивающих корректное объединение объектов в группы

Существующие меры сходства можно классифицировать на 2 типа:

- основанные на содержимом (content-based) оценивают степень близости объектов, сравнивая их внутренние характеристики – тексты, изображения или другие мультимедийные данные. На практике чаще всего используются текстовые данные, поскольку они проще поддаются анализу
- основанные на связях (link-based) моделируют отношения между объектами в виде графовых связей и вычисляют сходство, исходя из общих соседей. Чем больше общих соседних узлов имеют два объекта, тем выше считается их степень сходства

Обзор литературы. Кластеризация и меры сходства

К мерам, основанным на тексте *(text-based similarity measures)*, относятся:

- косинусное сходство
- методы, основанные на сингулярном разложении (SVD)
- латентно-семантический анализ (LSI)
- латентное распределение Дирихле (LDA)

К мерам, основанным на связях *(link-based measures)* относятся:

- совместное цитирование (Co-Citation)
- SimRank

В гетерогенных сетях для измерения сходства разработаны различные метрики

SimFusion использует URM для представления объектов и связей. Сходство вычисляется итеративно на основе URM, что позволяет выявить скрытые взаимосвязи и уменьшить разреженность

PathSim применяет концепцию мета-пути, учитывая семантику связывающих путей, но его выбор затруднителен при неизвестной схеме сети.

Авторы статьи выбрали комбинацию косинусного, лексического и семантического сходства

Обзор литературы. Совместная кластеризация

Большинство традиционных алгоритмов кластеризации выполняют одностороннее разбиение – группируют объекты по одному измерению, используя данные другого измерения в качестве признаков

В отличие от них, совместная кластеризация *(co-clustering)* позволяет одновременно группировать несколько типов объектов и делить их на подмножества согласованным образом

Обзор литературы. Совместная кластеризация

Предложено два классических подхода к совместной кластеризации:

- ***Bipartite Spectral Graph Partitioning (BSGP):*** метод разбиения двудольного графа, сводящий задачу совместной кластеризации к спектральному разбиению графа (документы и слова — две доли)
- ***Information-Theoretic Co-Clustering (ITCC):*** информационно-теоретический метод, рассматривающий таблицу совместных встречаемостей как эмпирическое совместное распределение. Оптимальная кластеризация максимизирует взаимную информацию между кластеризованными переменными

В данной работе используется ITCC, дополненная внутрисходством между ссылками и классами, что повышает точность группировки данных

Обзор литературы. Иерархическая кластеризация

Иерархическая кластеризация – метод, который объединяет объекты данных в иерархическую структуру (дерево кластеров)

Эта структура облегчает навигацию и анализ данных на разных уровнях обобщения

Существует делящая иерархическая модификация ***K-means*** (bisecting K-means), которая рекурсивно делит множество данных

Иерархическая совместная кластеризация сочетает преимущества иерархической и совместной кластеризации – она одновременно строит иерархии для 2 типов данных и выявляет скрытые взаимосвязи между ними

Обзор литературы. Иерархическая кластеризация

Были разработаны иерархические алгоритмы совместной кластеризации для анализа запросов и URL-адресов, для формирования иерархии тематических кластеров и сопоставления тем веб-страницам

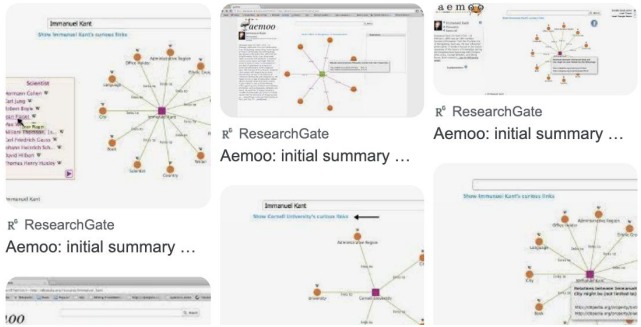
Вдохновленные этими идеями, предлагаем собственный иерархический алгоритм совместной кластеризации, который повышает точность и интерпретируемость результатов. Каждое разбиение узла трактуется как частный процесс совместной кластеризации, обеспечивая согласованность структуры между уровнями иерархии

Обзор литературы. Кластеризация в контексте Linked Data

Методы кластеризации нашли широкое применение при обработке RDF-данных, позволяя группировать сущности в осмысленные категории, чтобы облегчить пользователю навигацию и понимание структуры данных

Система Aemoo является примером исследовательского поиска по Семантической паутине, где соседние сущности группируются на основе типов DBpedia

Стандартные RDF-браузеры, такие как Tabulator, агрегируют связанные сущности через общие семантические ссылки и представляют их пользователю в виде списка



abulator

ID	Name	Progress	Gender	Rating
1	Madisyn Huel	75	male	4
2	Maudie Corkery	53	male	3
3	Ozella Marvin	68	female	3
4	Tatum Dach	4	female	3
5	Abel McLaughlin	6	female	4
6	Bailee Botsford	41	female	1
7	Cordie Crona	50	female	5
8	Aurore Schaefer	86	female	1

Обзор литературы. Кластеризация в контексте Linked Data

В отличие от перечисленных подходов, наш метод учитывает два взаимосвязанных аспекта сущностей:

- класс сущности
- семантическую ссылку

Мы выполняем их одновременную группировку с использованием иерархической совместной кластеризации.

Это первое исследование, в котором совместная кластеризация используется в качестве механизма интерактивного исследования сущностей в контексте связанных данных (Linked Data)

Иерархическая ко-кластеризация связей и классов

Навигация по Linked Data дает множество ссылок и классов, что сильно перегружает информацией

Что можно сделать:

- Строим навигационный граф
- Одновременно группируем ссылки и классы
- Повторяем рекурсивно и получаем иерархию кластеров

Resource Description Framework (RDF) тройки превращаем в граф навигации:
⟨subject, predicate, object⟩ ⇒ G = (L', R, C, E)

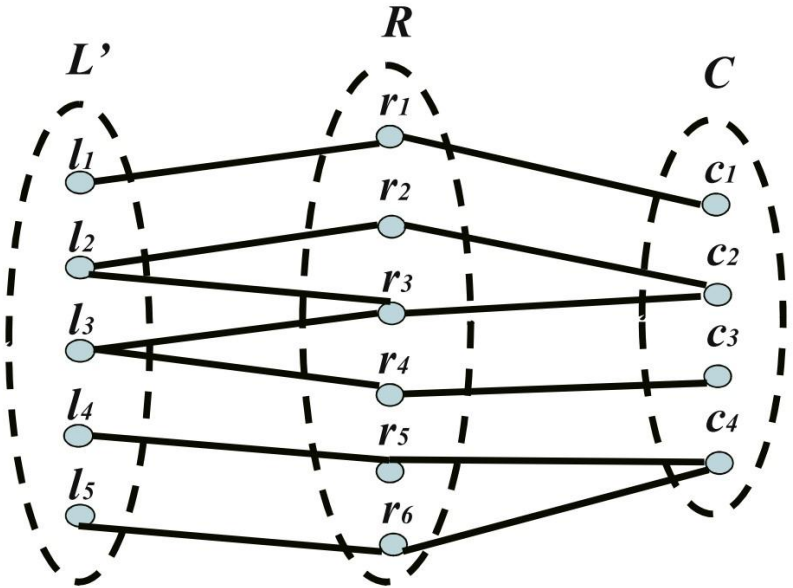


Fig. 4. An example of navigation graph.

Построение навигационного графа

Resource Description Framework (RDF) тройки превращаем в граф навигации:
⟨*subject, predicate, object*⟩ ⇒ *G = (L', R, C, E)*

G = <H, E> ***H = L' U R U C***

L' - типы связей *(dbo:director, dbo:birthPlace)*
R - связанные сущности *(Jurassic_Park, Cincinnati)*
C - типы сущностей *(Film, City)*
E - связи между ними

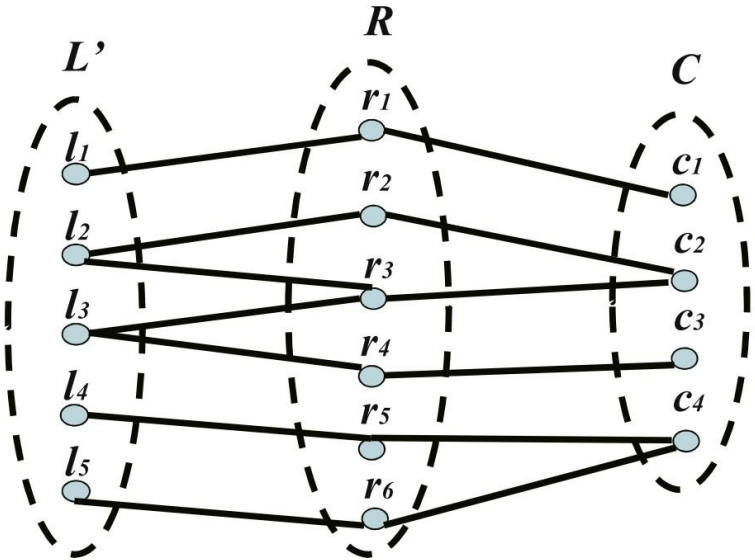


Fig. 4. An example of navigation graph.

$$E = \{(l,\ r) \mid \exists\ s \in S,\ (s,\ l,\ r)\ \in T\} \bigcup \{(r,\ c) \mid (r,\ rdf{:}\ type,\ c)\ \in T\}$$

(l, r) - ссылка, соединяет сущность с объектом *(subject u object)*

(r, c) - связь объекта и класса

Кластеризация

Задача – найти такие разбиения $\{\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_k\}$ для ссылок и $\{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_l\}$ для классов чтобы суммарное сходство внутри кластеров было максимальным

$$\max \left(\sum_{i=1}^{k} \sum_{x,x' \in \hat{x}_i} sim(x, x') + \sum_{j=1}^{l} \sum_{y,y' \in \hat{y}_j} sim(y, y') \right)$$

Три типа сходства:

1. **Cosine similarity** – на основе связанных сущностей
2. **Lexical similarity** – на основе текстовых названий
3. **Semantic similarity** – на основе иерархии

- Cosine similarity - два предиката или класса похожи, если они часто соединяются с одними сущностями

$$sim_{cos}(l_i, l_j) = \frac{l_i \cdot l_j}{||l_i|| \cdot ||l_j||}$$

- Lexical similarity - два элемента похожи, если они состоят из одинаковых слов

$$sim_{edit}(l_i, l_j) = \frac{1}{1 + editDist(s_{l_i}, s_{l_j})}$$

- Semantic similarity - если два класса или ссылки находятся близко в дереве, они схожи

$$sim_{sem}(c_i, c_j) = \frac{2 \cdot depth(LCA)}{depth(c_i) + depth(c_j)}$$

Комбинирование мер сходства

Функция оценки сходства классов c_i и c_j

$$sim(c_i, c_j) = \alpha \cdot sim_{cos}(c_i, c_j) + \beta \cdot sim_{edit}(c_i, c_j) + \gamma \cdot sim_{sem}(c_i, c_j)$$

$$\alpha,\ \beta,\ \gamma \in [0,\ 1]$$

$$\alpha + \beta + \gamma = 1$$

Полученная метрика сходства встраивается в Модифицированную целевую функцию ITCC в виде компонент, которые показывают сходство между кластерами

Модификация стандартного ITCC

Стандартная цель ITCC – минимизировать **потерю взаимной информации** при кластеризации

Потеря измеряется как дивергенция Кульбака-Лейблера (KL):

$$I(X;Y) - I(\hat{X};\hat{Y}) = D(p(X,Y) \mid\mid q(X,Y)) \equiv \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x,y) \log \frac{p(x,y)}{q(x,y)}$$

$I(X;Y)$ – взаимная информация между исходными связями (X) и классами (Y)

$I(\hat{X};\hat{Y})$ – взаимная информация между кластеризованными связями и классами

$p(X,Y)$ – функция распределения

$q(X,Y)$ – значение аппроксимирующего распределения

$$q(X,Y) = p(\hat{X},\hat{Y}) \frac{p(X|\hat{X})p(Y|\hat{Y})}{p(\hat{X})p(\hat{Y})}$$

Модификация стандартного ITCC

Чтобы использовать ITCC, определяем совместную вероятность *p(x,y)* для связи *x* и класса *y* на основе того, как часто они связаны с одними и теми же сущностями *r* в навигационном графе:

$$p(x,y) = \frac{|\{r \mid \ldots (x,r) \in E\} \cap \{r \mid \ldots (r,y) \in E\}|}{|R|}$$

|*R*| – общее число связанных сущностей

Стандартный ITCC игнорирует **внутреннее сходство** между самими связями или между самими классами

Решение – дополнить функцию ITCC, чтобы учесть это сходство
Новая целевая функция включает три компонента

$$\Phi = \underbrace{(I(X;Y) - I(\hat{X};\hat{Y}))}_{\text{Потеря взаимной информации}} + \underbrace{\lambda LCS}_{\text{Межкластерное сходство связей}} + \underbrace{\mu CCS}_{\text{Межкластерное сходство классов}}$$

$$LCS = \frac{2}{|\hat{X}|(|\hat{X}|-1)} \sum_{i>j} sim(\hat{x}_i, \hat{x}_j)$$

$$CCS = \frac{2}{|\hat{Y}|(|\hat{Y}|-1)} \sum_{i>j} sim(\hat{y}_i, \hat{y}_j)$$

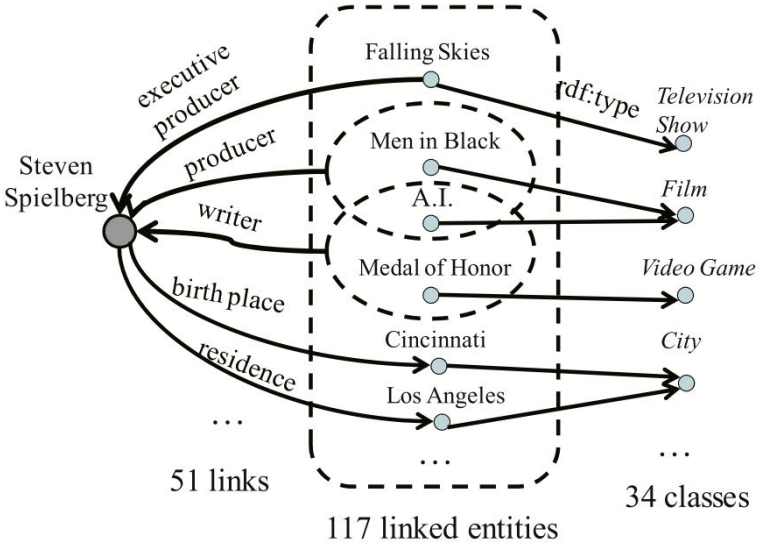
Co-Clustering

1. Случайно назначаем каждую ссылку x_i и класс y_j начальному кластеру
2. Вычисляем *q(x, y)*
3. Для каждого x_i и y_j выбираем кластер, при котором функция потерь минимальна
4. Повторяем, пока функция потерь не перестанет уменьшаться

Иерархический Co-Clustering

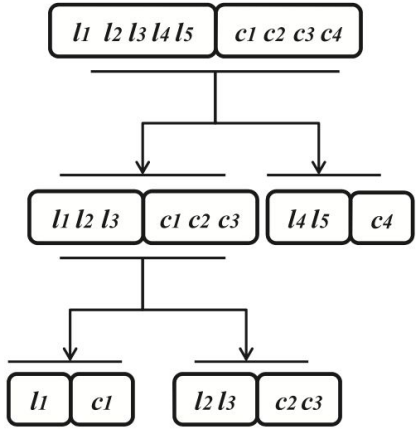
1. Применяем Co-Clustering ко всему графу – получаем крупные кластеры ссылки-классы
2. Для каждого кластера:
   a. применяем Co-Clustering
   b. если KL не уменьшается – останавливаемся
   c. повторяем
3. Повторяем, пока не $\Delta I(\hat{X};\hat{Y}) < \varepsilon$

Иерархический подход



l_1	executive producer $^{-1}$
l_2	producer $^{-1}$
l_3	writer $^{-1}$
l_4	birth place
l_5	residence

c_1	Television Show
c_2	Film
c_3	Video Game
c_4	City



Исходные данные представлены как сложная, неструктурированная сеть

Дерево показывает, как исходные элементы были упорядочены и сгруппированы на разных уровнях обобщения
Узлы на верхнем уровне представляют широкие темы, а узлы на нижнем уровне – узкие подгруппы

Выбор представителя класса

Выбирается класс и ссылка, которые имеют самое высокое среднее сходство со всеми остальными классами в этом же кластере $\hat{y}$

$$y^* = \arg\max_{y \in \hat{y}} \left(\sum_{y' \in \hat{y}, y \neq y'} \text{sim}(y, y') \right)$$

Итоговая метка для узла в иерархии формируется на основе этой представительной метки y*. Эта метка используется для навигации и отображения, облегчая пользователю просмотр и исследование сгруппированных сущностей

Код. Что реализовано и почему

Реализовано: ITCC (итеративная ко-кластеризация) + HCC (иерархическая обёртка).

HCC — основной вклад статьи: строит дерево кластеров links и classes, уменьшает когнитивную нагрузку.

BSGP/DiagCoClus — бейзлайны для сравнения; выбрали HCC для доклада из-за интерпретируемости.

Наши дополнения: семантические/лексические сходства, лог итераций, экспорт отчёта с ASCII-деревом.

Данные: DBpedia (dbo-слой) и подготовка

Источник: онлайн DBpedia, SPARQL POST (без длинных URL), батч-запросы.

Фильтры: только dbo: предикаты и dbo: классы (mapping-based слой, как в статье).

Гранулярность: 1-хоп окрестность центральной сущности (пример: dbr:Steven_Spielberg).

Типы: опционально оставляем наиболее специфичный класс (без предков по rdfs:subClassOf).

Скрипт: tools/fetch_dbpedia.py → экспорт в data/dbpedia_graph.ttl (локальные имена, онтологии классов/свойств).

Архитектура и структура проекта

tools/fetch_dbpedia.py — скачивание и фильтрация данных (dbo:), запись TTL.

src/hcc.py — DemoGraph (парсинг TTL, автоцентр, матрица p(x,y)); ITCC_HCC; ascii_tree().

run_hcc_dbpedia.py — прогон HCC: прогресс итераций, CI-метрики, отчёт в results/.

notebooks/ — Demo.ipynb (синтетика), DBpedia.ipynb (онлайн прогон).

setup.sh / setup.ps1 — установка зависимостей, ядро Jupyter.

Гиперпараметры и воспроизводимость

k, l — число кластеров для links и classes; max_iter — итерации ITCC.

Веса сходств (α,β,γ): косинус/лексика/семантика; штрафы λ, μ — схожесть кластеров (LCS/CCS).

Пресет «как в статье»: k=l=3, max_iter=20, λ=μ=0.5; веса t5=(0.6, 0.1, 0.3).

Команды: source .venv/bin/activate → fetch_dbpedia.py --entity ... → run_hcc_dbpedia.py.

Ускорение: исключить шумные линки (wikiPageWikiLink, externalLink, thumbnail), уменьшить l/max_iter, ранний стоп.

Результаты (Steven Spielberg, текущая DBpedia)

Размеры входа: links ≈ 10, classes ≈ 95 (dbo-фильтр; современная DBpedia).

Метрики: CI(links)=0.5206 — умеренно; CI(classes)=0.8047 — хорошо.

Время: ~942 сек (дороги шаги по classes при больших l и количестве типов).

Иерархия: 3 кластера links (биографические / описательные / «вики-ссылки»).

Кластера classes: центры Person / Artist / Settlement и родственные им подклассы.

Оценка и эксперименты

В данном разделе авторы с помощью прототипа системы CoClus оценивают качество своего подхода на следующих экспериментах:

1) Пользовательское исследование, в котором сравнивают CoClus с тремя системами для работы с Linked Data.

2) Оценка качества кластеризации, сравнение результатов работы CoClus с тремя алгоритмами совместной кластеризации

3) Далее исследуется масштабируемость системы - измеряется среднее время выполнения алгоритма при разных объемах данных.

Пользовательское исследование

- Гипотеза 1 - Иерархическая совместная кластеризация связей и классов сущностей обеспечивает иерархическую организацию данных с разной степенью детализации. Использование такой структуры может эффективно помочь пользователям при исследовании сущностей.

- Гипотеза 2 - Кластеры связей и кластеры классов являются взаимодополняющими в процессе исследования сущностей, поэтому предоставление пользователю обоих типов кластеров (как в CoClus) дает более удовлетворяющий результат, чем использование только одного из них.

Пользовательское исследование

Как оценивался эксперимент?

1) Время выполнения

2) Количество действий пользователя

3) Удовлетворенность пользователя

Table 1
Post-task questionnaire.

	Questions
Q1:	System had an overview of all the information.
Q2:	System had appropriate size of navigation options.
Q3:	It was easy to reorient myself in the exploration.
Q4:	System was easy-to-use with no prior knowledge.
Q5:	System offered helpful support for this task.

Исследование на людях

Результаты

Table 2
Results of post-task questionnaire.

	Response: Mean (SD)				$F(3, 92)$	LSD post-hoc
	Rhizomer	LinkClus	ClassClus	CoClus	(p-value)	($p < 0.05$)
Q1:	3.56 (0.85)	3.8 (1.056)	3.93 (0.95)	**4.18** (1.095)	11.543 (0.000)	CoClus > LinkClus, ClassClus > Rhizomer
Q2:	**3.67** (1.052)	3.45 (1.014)	3.34 (1.32)	3.02 (0.887)	9.56 (0.000)	Rhizomer, LinkClus, ClassClus > CoClus
Q3:	**3.83** (0.514)	3.667 (0.778)	3.78 (1.031)	3.5 (0.937)	11.67 (0.000)	Rhizomer > CoClus, LinkClus, ClassClus
Q4:	3.25 (0.793)	3.98 (0.855)	4.18 (1.055)	**4.33** (0.778)	10.36 (0.000)	CoClus, LinkClus, ClassClus > Rhizomer
Q5:	3.74 (0.717)	3.51 (0.582)	3.42 (1.02)	**4.02** (0.62)	13.65 (0.000)	CoClus > LinkClus, ClassClus > Rhizomer

Пользовательское исследование

Результаты

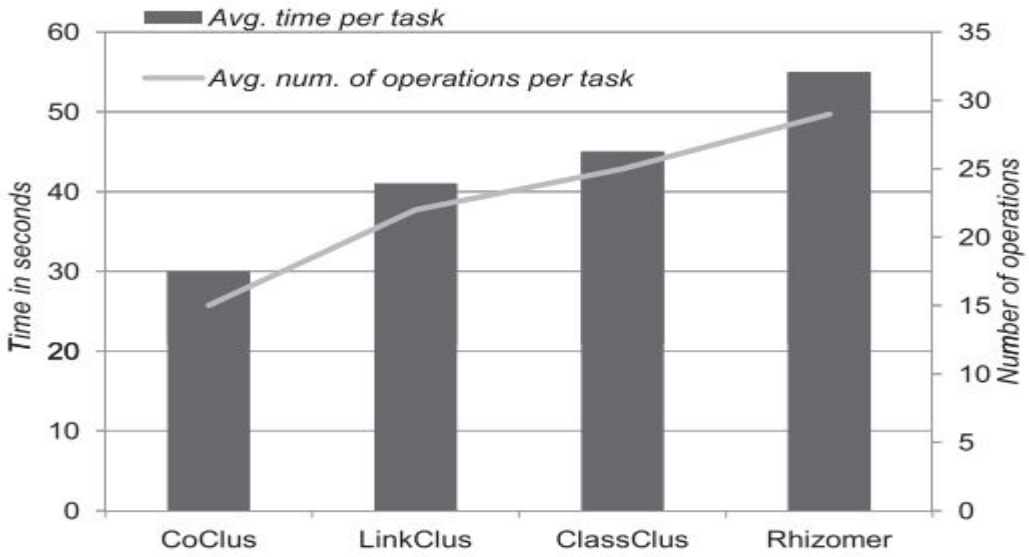


Fig. 6. Average time and number of user actions in exploration tasks for four systems.

Гипотезы подтверждены

Оценка качества кластеризации

Было использовано два датасета

- HCC
- ITCC
- bipartite spectral graph partition(BSGP) - находит разбиение вершин двудольного графа с минимальным разрезом
- DiagCoClus - делает матрицу совместных встречаемостей максимально близкой к диагональной блочной структуре

Table 3
Statistics of experimental datasets.

	DBpedia	YAGO
Entities	18,329	15,035
Average number of linked entities per entity	217.5	103.2
Average number of links per entity	25.8	10.1
Average number of classes per linked entity	9.6	38.7

Оценка качества кластеризации

Для определения эффективности каждого алгоритма кластеризации авторы использовали критерий индекса кластеризации (CI).

$$CI = \frac{\overline{\sigma}^2}{\overline{\sigma} + \overline{\delta}}$$

$$\overline{\sigma} = \frac{1}{k}\sum_{i=1}^{k}\sigma(O_i),\ \ \sigma(O_i) = \frac{\sum_{o\in O_i, o'\in O_i} sim(o, o')}{|O_i| \cdot (|O_i| - 1)}$$

$$\overline{\delta} = \frac{2}{k(k-1)}\sum_{i>j}\delta(O_i, O_j),\ \ \delta(O_i, O_j) = \frac{\sum_{o\in O_i, o'\in O_j} sim(o, o')}{|O_i| \cdot |O_j|}$$

t1 = (1, 0, 0)
t2 = (0, 1, 0)
t3 = (0, 0, 1)
t4 = (0.33, 0.33, 0.33)
t5 = (0.6, 0.1, 0.3)
t6 = (0.3, 0.1, 0.6)

Оценка качества кластеризации

Table 4
The *CI* scores with different k and (α, β, γ) on DBpedia.

		t_1	t_2	t_3	t_4	t_5	t_6
k=3	HCC	0.69	0.62	0.45	0.54	**0.77**	0.65
		0.54	0.58	0.62	0.48	0.68	0.72
	ITCC	0.65	0.57	0.4	0.23	0.7	0.63
		0.52	0.41	0.49	0.33	0.62	**0.73**
	BSGP	0.56	0.56	0.37	0.28	0.46	0.43
		0.29	0.26	0.43	0.35	0.58	0.63
	DiagCoClus			0.61			
				0.65			
k=5	HCC	0.65	0.69	0.47	0.53	**0.76**	0.64
		0.51	0.54	0.59	0.41	**0.74**	0.73
	ITCC	0.57	0.48	0.42	0.37	0.71	0.47
		0.43	0.34	0.47	0.32	0.69	0.51
	BSGP	0.49	0.27	0.21	0.21	0.52	0.38
		0.39	0.39	0.57	0.19	0.51	0.42
	DiagCoClus			0.72			
				0.68			
k=8	HCC	0.67	0.66	0.57	0.63	**0.83**	0.78
		0.58	0.59	0.65	0.67	0.81	**0.86**
	ITCC	0.63	0.56	0.51	0.34	0.78	0.71
		0.52	0.49	0.59	0.42	0.76	0.82
	BSGP	0.58	0.38	0.38	0.27	0.59	0.56
		0.54	0.34	0.61	0.33	0.61	0.66
	DiagCoClus			0.74			
				0.78			

Table 5
The *CI* scores with different k and (α, β, γ) on YAGO.

		t_1	t_2	t_3	t_4	t_5	t_6
k=3	HCC	0.66	0.65	0.64	0.71	0.78	**0.79**
		0.54	0.51	0.57	0.67	**0.73**	0.64
	ITCC	0.61	0.61	0.59	0.69	0.73	0.71
		0.51	0.55	0.64	0.63	0.61	0.56
	BSGP	0.62	0.59	0.37	0.58	0.68	0.63
		0.51	0.48	0.28	0.55	0.51	0.55
	DiagCoClus			0.75			
				0.69			
k=5	HCC	0.69	0.62	0.63	0.75	**0.81**	0.7
		0.53	0.53	0.57	0.67	**0.76**	0.66
	ITCC	0.64	0.61	0.54	0.69	0.77	0.67
		0.51	0.52	0.53	0.55	0.71	0.59
	BSGP	0.53	0.55	0.51	0.55	0.69	0.56
		0.52	0.34	0.56	0.32	0.5	0.49
	DiagCoClus			0.71			
				0.72			
k=8	HCC	0.7	0.61	0.68	0.79	**0.87**	0.73
		0.55	0.67	0.61	0.71	**0.79**	0.66
	ITCC	0.67	0.57	0.62	0.73	0.71	0.69
		0.55	0.61	0.57	0.62	0.74	0.61
	BSGP	0.69	0.53	0.58	0.67	0.61	0.61
		0.54	0.54	0.45	0.53	0.51	0.58
	DiagCoClus			0.78			
				0.71			

Оценка качества кластеризации

(1) В большинстве случаев HCC показывает лучшие результаты по сравнению с тремя другими алгоритмами.

(2) Что касается влияния трех параметров, то можно видеть, что косинусное и лексическая сходство лучше подходят для кластеризации связей, а семантическое сходство лучше подходит для кластеризации классов в экспериментальных наборах данных.

В дальнейшем планируется протестировать больше наборов данных.

Оценка масштабируемости

В оценке эффективности показано, что DiagCoClus работает быстрее всего. HCC самый медленный, но даже при росте данных время работы HCC остается приемлемым

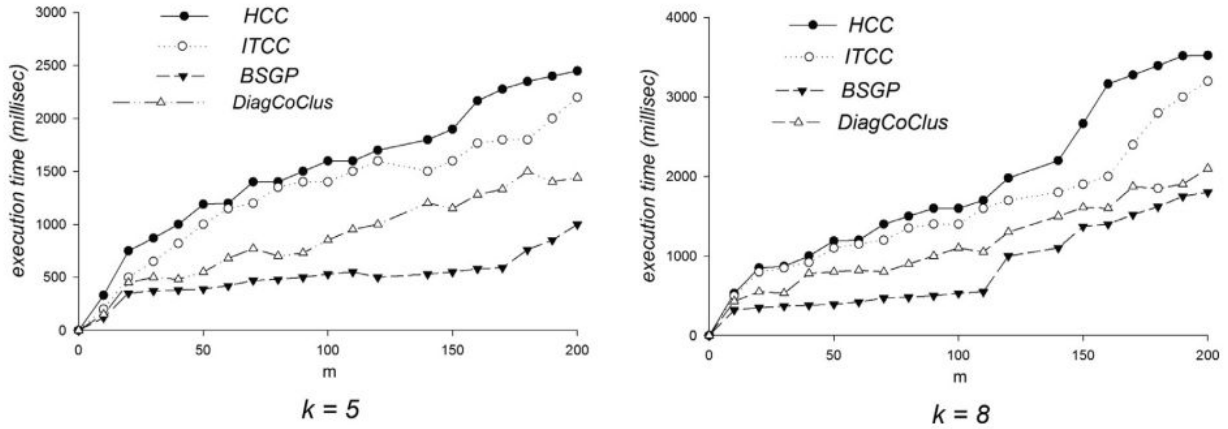


Fig. 7. Execution time of four algorithms with varying k and m.

Заключение

В работе предложен иерархический метод совместной кластеризации для улучшения исследования сущностей в Linked Data. Подход учитывает сходство между типами связей и классами, и интегрирует это в процесс co-clustering. Реализация выполнена в прототипе CoClus. Экспериментальная оценка показала, что метод помогает пользователям быстрее ориентироваться и находить нужные сущности. В дальнейшем планируется улучшать способы автоматической маркировки кластеров и учитывать предпочтения пользователей для более умного ранжирования результатов. Также авторы будут дальше экспериментировать с вышеперечисленными тестами при большей конфигурации параметров.

Спасибо за внимание!