



UNIVERSIDADE FEDERAL DE PERNAMBUCO  
CENTRO DE INFORMÁTICA  
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

**EVERALDO COSTA SILVA NETO**

**Discovering a Domain-Specific Schema from General-Purpose Knowledge Base**

Recife  
2023

**EVERALDO COSTA SILVA NETO**

**Discovering a Domain-Specific Schema from General-Purpose Knowledge Base**

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, como requisito para obtenção do título de Doutor em Ciência da Computação.

**Área de Concentração:** Banco de Dados  
**Orientadora:** Ana Carolina Brandão Salgado  
**Coorientador:** Luciano de Andrade Barbosa

Recife  
2023

Catálogo na fonte  
Bibliotecária Nataly Soares Leite Moro, CRB4-1722

S586d Silva Neto, Everaldo Costa  
*Discovering a domain-specific schema from general-purpose knowledge base* / Everaldo Costa Silva Neto – 2023.  
118 f.: il., fig., tab.

Orientadora: Ana Carolina Brandão Salgado.  
Tese (Doutorado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2023.  
Inclui referências e apêndices.

1. Banco de dados. 2. Descoberta de esquema. 3. Descoberta do domínio.  
4. Representação de entidade. I. Salgado, Ana Carolina Brandão (orientadora).  
II. Título

025.04 CDD (23. ed.) UFPE - CCEN 2023 – 113

**Everaldo Costa Silva Neto**

**“Discovering a Domain-Specific Schema from General-Purpose Knowledge Base”**

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação. Área de Concentração: Banco de Dados

Aprovado em: 13/06/2023.

---

**Orientadora: Profa. Dra. Ana Carolina Brandão Salgado**

**BANCA EXAMINADORA**

---

Prof. Dr. Fernando da Fonseca de Souza  
Centro de Informática / UFPE

---

Prof. Dr. Altigran Soares da Silva  
Insitituto de Computação / UFAM

---

Profa. Dra. Damires Yluska Souza Fernandes  
Unidade Acadêmica de Informática / IFPB

---

Prof. Dr. André Câmara Alves do Nascimento  
Departamento de Computação / UFRPE

---

Prof. Dr. Carlos Eduardo Santos Pires  
Departamento de Sistemas e Computação / UFCG

*To my family, especially my nephews Liz and João.*

## ACKNOWLEDGEMENTS

*"...respirar, ah meu Deus, obrigado porque eu vivi para ver esse dia chegar...".* Finalmente! Foram (longos) 5 anos até a conclusão do doutorado, mas se estou escrevendo esta página da tese é porque deu tudo certo. Se cheguei até aqui é porque eu pude encontrar pessoas que me ajudaram (direta ou indiretamente) a conquistar esse feito. São a essas pessoas que eu quero dedicar esse espaço.

Primeiramente, agradeço a Deus pelo dom da vida, por Sua graça infinita que, mesmo sem eu merecer, tem me alcançado dia após dia.

Agradeço aos meus pais, Emerson e Geórgia, por todo apoio e incentivo. Obrigado por me ensinar a valorizar a educação e mostrar que ela é a chave que abre muitas portas. Estendo meus agradecimentos as minhas irmãs, Clara e Ana Cecília. Eu sei que vocês vibram com cada conquista minha. Tanta coisa mudou nesse tempo, inclusive com a chegada de Liz e João...

Agradeço à minha orientadora, Ana Carolina (Carol), por todos esses anos de trabalho. Obrigado por abrir as portas da pesquisa científica para mim (desde o mestrado). Sempre com uma palavra de ânimo e confiança você me dizia que tudo daria certo, e deu!

Agradeço ao meu orientador, Luciano, por todo o conhecimento compartilhado durante esses últimos anos. Com toda certeza foi uma jornada de muito aprendizado. Minha gratidão por estar sempre disposto a ouvir e contribuir com seu extenso repertório de ideias.

Um agradecimento especial aos meus amigos e companheiros de curso Johny, Levy e Michael. Foi muito bom encontrar vocês nesse tempo. Infelizmente a pandemia não permitiu nossa convivência no mesmo espaço físico (laboratório) por mais tempo, mas isso não foi um limitador. Mesmo distantes fisicamente pudemos estabelecer esse espaço de apoio, troca e contribuições. Eu agradeço por sempre estarem dispostos a somar. Espero que essa rede de amizade e colaboração perdure por muito tempo.

Agradeço aos meus amigos e amigas, especialmente Kenia, Anna, Darlany, Joise e Rafael por todo apoio moral e emocional neste último ano. É sempre bom cruzar com pessoas especiais em nosso caminho. A presença de vocês muitas vezes foi um alívio em meio a tanta pressão.

Finalizo meus agradecimentos me dirigindo à banca examinadora deste trabalho, professora Damires Souza e professores Fernando Fonseca, Altigran Silva, André Câmara e Carlos Eduardo. Gratidão por todos os comentários e contribuições durante a defesa. Com certeza foram valiosos e enriqueceram a versão final deste documento.

*"Hoje me sinto mais forte  
Mais feliz quem sabe  
Só levo a certeza de que muito pouco sei  
Ou nada sei..." (SATER, 1992)*

## ABSTRACT

General-purpose knowledge bases (KBs), e.g., DBpedia, YAGO, and Wikidata, store factual data about a set of entities. These KBs have been constructed to store cross-domain knowledge, e.g., health, entertainment, industry, sports, and arts. Most applications that use data from general-purpose KBs are domain-specific. Some tasks, such as query formulation and information extraction, require a data schema to explore the contents of a KB. However, schema-related declarations are not mandatory and, sometimes, are not provided. Therefore, these domain-specific applications face two issues: (1) they require only a subset of data that meets the domain of interest, but general-purpose KBs have a large volume of factual data within many distinct domains; and (2) the lack of schema-related information. In this thesis, we address the problem of domain-specific schema discovery from general-purpose KBs. Specifically, we build ANCHOR, an end-to-end pipeline to identify a domain-specific dataset as well as its schematic description in an automatic way. ANCHOR works in three steps: domain discovery, class identification and class schema discovery. First, it extracts a specific domain exploring category-category mappings from KB. From this, it identifies domain entities through entity-category mappings. Next, the class identification step discovers implicit classes within the dataset. For that, ANCHOR learns entity representation from entity-category mappings and uses it to identify implicit entities' classes by grouping similar entities. Finally, the class schema discovery task builds the class schema, i.e., it identifies a set of relevant attributes that best describe the entities within the same class. For that, ANCHOR runs CoFFee, an approach based on attributes co-occurrence and frequency to identify a set of core attributes for each class discovered in the previous step. We have performed an extensive experimental evaluation on four distinct DBpedia domains. For the class identification task, we compare ANCHOR against some traditional and embedding-based baselines. The results show that applied to standard clustering algorithms, our entity representation outperforms the baselines and is effective for the class identification task. For the class schema discovery task, we compare CoFFee against two state-of-the-art approaches. The results show that CoFFee proved to be effective in filtering out less relevant attributes. It selects a set of core attributes keeping its retrieval rate high and producing a higher-quality schema class for the identified classes.

**Keywords:** schema discovery; domain discovery; entity representation; class identification; class schema discovery.

## RESUMO

Bases de conhecimento de propósito geral, e.g., DBpedia, YAGO e Wikidata, armazenam dados factuais sobre um conjunto de entidades. Elas são construídas para armazenar conhecimento de múltiplos domínios, e.g., saúde, entretenimento, indústria, esportes e artes. A maioria das aplicações que utilizam dados de bases de conhecimento de propósito geral é específica para um domínio. Algumas tarefas, tais como, formulação de consulta e extração da informação, requerem um esquema de dados para explorar o conteúdo de uma base de conhecimento. Entretanto, declarações específicas de esquema não são obrigatórias e, algumas vezes, não são fornecidas. Portanto, aplicações específicas para um domínio enfrentam dois problemas: (1) elas requerem apenas um subconjunto de dados de interesse ao domínio da aplicação, mas as bases de conhecimento de propósito geral possuem um grande volume de dados factuais em diferentes domínios; e (2) a falta de informações relacionadas ao esquema. Nesta tese, endereçamos o problema da descoberta de esquema para um domínio específico a partir de bases de conhecimento de propósito geral. Especificamente, desenvolvemos ANCHOR, um *pipeline* ponta-a-ponta que tem como objetivo identificar, de maneira automática, um conjunto de dados para um domínio específico bem como a sua descrição de esquema. ANCHOR é dividido em três etapas: descoberta de domínio, identificação de classe e descoberta do esquema da classe. Inicialmente, ANCHOR extrai um domínio específico explorando os mapeamentos categoria-categoria fornecidos pela base de conhecimento. Em seguida, a etapa de identificação de classe descobre classes implícitas no conjunto de dados. ANCHOR aprende uma representação para cada entidade utilizando os mapeamentos entidade-categoria. Essa representação é usada para agrupar entidades similares com o objetivo de identificar classes de entidades implícitas no conjunto de dados. Por fim, a etapa de descoberta do esquema da classe identifica um conjunto de atributos relevantes que melhor descreve as entidades de uma mesma classe. ANCHOR executa CoFFee, uma abordagem baseada na coocorrência e frequência dos atributos para identificar um conjunto de atributos centrais em cada classe descoberta na etapa anterior. Realizamos experimentos em quatro domínios da DBpedia. Na tarefa de identificação de classe, comparamos ANCHOR com baselines tradicionais e baseadas em *embeddings*. Os resultados mostraram que, utilizando os algoritmos de agrupamento clássicos, a representação de entidade proposta nesta tese superou os baselines, mostrando ser eficiente para a tarefa de identificação de classe. Na tarefa de descoberta do esquema da classe, comparamos CoFFee com duas abordagens do estado da arte. Os resultados indicam que CoFFee é eficaz para filtrar atributos menos relevantes. Ele seleciona um conjunto de atributos centrais mantendo a taxa de recuperação alta e produzindo um esquema de alta qualidade para as classes identificadas.

**Palavras-chave:** descoberta de esquema; descoberta do domínio; representação de entidade; identificação de classe; descoberta do esquema da classe.

## LIST OF FIGURES

Figure 1 – Illustration of different views of a semantic graph extracted from DBpedia.	16
Figure 2 – Graphical abstract - ANCHOR pipeline steps.	21
Figure 3 – Research methodology overview.	22
Figure 4 – Snippet from Bill Gates’s page. The main elements are highlighted by color: abstract/text (orange); infobox (green); sections (purple); categories (red).	25
Figure 5 – Guidelines for Infobox_person template.	26
Figure 6 – Example of triples from infobox data within DBpedia datasets.	27
Figure 7 – Excerpt of the Wikipedia category graph showing the category Albums together with some of its subcategories.	28
Figure 8 – Example of triples from categories data within DBpedia datasets.	30
Figure 9 – Excerpt of a knowledge base created from a set of triples.	31
Figure 10 – Training sample example - Word2vec.	34
Figure 11 – Word2vec architecture: CBOW model (left) and Skip-gram model (right).	35
Figure 12 – Deep Walk pipeline.	36
Figure 13 – Deep Walk Intuition. Input: Karate Club Graph (left-side). Output: Node Embeddings (right-side).	37
Figure 14 – Visualizations of Les Misérables coappearance network generated by node2vec. Label colors reflect patterns in the network: homophily (top) and structural equivalence (bottom).	37
Figure 15 – Different ways for clustering the same dataset.	38
Figure 16 – Execution of the K-means algorithm in three iterations.	39
Figure 17 – A hierarchical clustering from four points is illustrated as a dendrogram (a) and nested clusters (b).	40
Figure 18 – Approaches to calculating cluster proximity.	40
Figure 19 – Example of a 12-point cluster using the DBSCAN approach.	41
Figure 20 – Type profile (example)	52
Figure 21 – Pattern extraction (left-side) and clustering (right-side) from a RDF dataset.	53
Figure 22 – Pipeline for incremental schema discovery.	54
Figure 23 – Snippet of the a KB content in the oil and gas domain.	63
Figure 24 – A sample of the graphs created from the entity-category (a) and category-category mappings (b).	65
Figure 25 – ANCHOR’s pipeline.	66
Figure 26 – Domain graph sample (Oil and gas).	68

Figure 27 – Bipartite graph (sample) built from the entity-category mappings of a KB (oil and gas domain). The left-side shows the bipartite graph from the original mapping, whereas the right-side shows the bipartite graph created from the mapping extension. Dashed edges illustrate new relations added from the extension. . . . .	70
Figure 28 – Entity graph created from the bipartite graph in Fig 27. . . . .	70
Figure 29 – Illustration of the vector representation generated by node embedding algorithms. Each node (entity) is represented by a 5-dimensional vector. At the bottom, we illustrate a 2-dimensional view. Nearby entities in d-dimensional space, e.g., Ghawar Field and Khurais Field belong to the same class. . . . .	71
Figure 30 – Attribute frequency (Oil field class). . . . .	73
Figure 31 – CoFFee’s pipeline. . . . .	74
Figure 32 – Example of the graphs used by CoFFee. In (a) the bipartite graph created from the set of entity schemata, and (b) the attribute graph created from the relationships between the attributes of the set of entity schemata. . . . .	75
Figure 33 – Summary of experimental evaluation. . . . .	80
Figure 34 – Overlap between data schemas for each class. . . . .	86
Figure 35 – Visualizations of (a) ANCHOR’s vectors, (b) RDF2vec vectors and (c) Wikipedia2vec vectors for the Formula One domain. . . . .	88
Figure 36 – Performance (effectiveness) of the approaches to summarize the class schema. . . . .	92
Figure 37 – Effectiveness of approaches to summarize the company class schema. .	94
Figure 38 – Attribute frequency (Race class). . . . .	94
Figure 39 – Quality of the class schema ordered by the relevance of the attributes. .	95
Figure 40 – Class schema quality compared to the reference schema. . . . .	97

## LIST OF TABLES

Table 1 – Set of triples representing the knowledge of a KB. . . . .	30
Table 2 – Comparative analysis between works that perform the domain discovery task. . . . .	49
Table 3 – Comparative analysis between works that perform the schema discovery task. . . . .	60
Table 4 – Summary of the datasets used in each domain. . . . .	79
Table 5 – Overview of the class identification’s evaluation set. . . . .	81
Table 6 – Class identification results. . . . .	84
Table 7 – Results for the two sample proportion Z-test. . . . .	85
Table 8 – Difference between CoFFee and Universal approaches . . . . .	91
Table 9 – Class schema quality compared to the reference schema - Oil and gas domain. . . . .	98
Table 10 – Dataset statistics - Oil and gas domain. . . . .	115
Table 11 – Dataset statistics - Formula one domain. . . . .	115
Table 12 – Dataset statistics - Martial arts domain. . . . .	116
Table 13 – Dataset statistics - Tourism domain. . . . .	116
Table 14 – Schema reference information - Oil and gas domain (*short version). . .	117
Table 15 – Schema reference information - Formula one domain. . . . .	117
Table 16 – Schema reference information - Martial arts domain. . . . .	118
Table 17 – Schema reference information - Tourism domain. . . . .	118

## LIST OF ABBREVIATIONS AND ACRONYMS

<b>BFS</b>	Breadth-First-Sampling
<b>CBOW</b>	Continuous Bag-of-Words
<b>CS</b>	Characteristic Set
<b>DFS</b>	Depth-First-Sampling
<b>FCA</b>	Formal Concept Analysis
<b>HAC</b>	Hierarchical Agglomerative Clustering
<b>JSON</b>	JavaScript Object Notation
<b>KB</b>	Knowledge Base
<b>KG</b>	Knowledge Graph
<b>LSA</b>	Latent Semantic Analysis
<b>OWL</b>	Web Ontology Language
<b>RDF</b>	Resource Description Framework
<b>RDFS</b>	RDF Schema
<b>URI</b>	Uniform Resource Identifier
<b>W3C</b>	World Wide Web Consortium
<b>WCH</b>	Wikipedia Category Hierarchy
<b>XML</b>	Extensible Markup Language
<b>YAGO</b>	Yet Another Great Ontology

## CONTENTS

<b>1</b>	<b>INTRODUCTION . . . . .</b>	<b>15</b>
1.1	CONTEXT AND MOTIVATION . . . . .	15
1.2	PROBLEM AND RESEARCH QUESTIONS . . . . .	19
1.3	OBJECTIVES . . . . .	20
1.4	SOLUTION OVERVIEW . . . . .	21
1.5	RESEARCH METHODOLOGY . . . . .	22
1.6	CONTRIBUTIONS . . . . .	22
1.7	THESIS ORGANIZATION . . . . .	23
<b>2</b>	<b>BACKGROUND . . . . .</b>	<b>24</b>
2.1	WIKIPEDIA . . . . .	24
2.1.1	<b>Infoboxes . . . . .</b>	<b>25</b>
2.1.2	<b>Categories . . . . .</b>	<b>28</b>
2.2	KNOWLEDGE BASE . . . . .	30
2.3	REPRESENTATION LEARNING . . . . .	33
2.3.1	<b>Word Representation . . . . .</b>	<b>33</b>
2.3.2	<b>Node Representation . . . . .</b>	<b>35</b>
2.4	CLUSTERING . . . . .	37
2.4.1	<b>Clustering Algorithms . . . . .</b>	<b>39</b>
2.4.2	<b>Cluster Evaluation . . . . .</b>	<b>42</b>
2.5	CONSIDERATIONS . . . . .	43
<b>3</b>	<b>RELATED WORK . . . . .</b>	<b>45</b>
3.1	DOMAIN DISCOVERY . . . . .	45
3.1.1	<b>Domain taxonomies and ontologies . . . . .</b>	<b>45</b>
3.1.2	<b>Domain-specific data from knowledge bases . . . . .</b>	<b>46</b>
3.1.3	<b>Semantic annotations . . . . .</b>	<b>47</b>
3.1.4	<b>Comparative analysis . . . . .</b>	<b>48</b>
3.2	SCHEMA DISCOVERY . . . . .	50
3.2.1	<b>Structured data . . . . .</b>	<b>50</b>
3.2.2	<b>Semi-structured data . . . . .</b>	<b>51</b>
3.2.3	<b>Comparative analysis . . . . .</b>	<b>56</b>
3.3	CONSIDERATIONS . . . . .	61
<b>4</b>	<b>A PIPELINE FOR SCHEMA DISCOVERY FROM GENERAL-PURPOSE KNOWLEDGE BASES . . . . .</b>	<b>62</b>

4.1	MOTIVATING SCENARIO . . . . .	62
4.2	PRELIMINARES . . . . .	63
4.3	SCHEMA DISCOVERY PIPELINE . . . . .	66
<b>4.3.1</b>	<b>Domain Discovery . . . . .</b>	<b>67</b>
<b>4.3.2</b>	<b>Class Identification . . . . .</b>	<b>68</b>
<b>4.3.3</b>	<b>Class Schema Discovery . . . . .</b>	<b>72</b>
4.4	CONSIDERATIONS . . . . .	78
<b>5</b>	<b>EXPERIMENTS . . . . .</b>	<b>79</b>
5.1	DOMAIN DATA . . . . .	79
5.2	CLASS IDENTIFICATION ASSESSMENT . . . . .	80
<b>5.2.1</b>	<b>Experimental setup . . . . .</b>	<b>80</b>
<b>5.2.2</b>	<b>Results . . . . .</b>	<b>83</b>
5.3	CLASS SCHEMA DISCOVERY ASSESSMENT . . . . .	89
<b>5.3.1</b>	<b>Experimental setup . . . . .</b>	<b>89</b>
<b>5.3.2</b>	<b>Results . . . . .</b>	<b>91</b>
5.4	CONSIDERATIONS . . . . .	99
<b>6</b>	<b>CONCLUSION . . . . .</b>	<b>101</b>
6.1	OVERVIEW . . . . .	101
6.2	CONTRIBUTIONS . . . . .	103
6.3	LIMITATIONS AND FUTURE WORK . . . . .	103
6.4	PUBLICATIONS . . . . .	105
	<b>REFERENCES . . . . .</b>	<b>106</b>
	<b>APPENDIX A – DATASET STATISTICS . . . . .</b>	<b>115</b>
	<b>APPENDIX B – SCHEMA REFERENCE . . . . .</b>	<b>117</b>

# 1 INTRODUCTION

In this chapter, we first provide an overview and introduce the context of this thesis. Then, we present the motivation, research problem, and guiding research questions for the development of this work. Finally, we address the objectives, contributions, and a brief description of the organization of the content of this document.

## 1.1 CONTEXT AND MOTIVATION

A Knowledge Base (KB) is a structured database that stores a collection of factual data about a set of entities in triple format  $\langle \text{subject}, \text{predicate}, \text{object} \rangle$  (LAN et al., 2022). In this sense, an entity refers to a real-world object (e.g., person, animal, place).

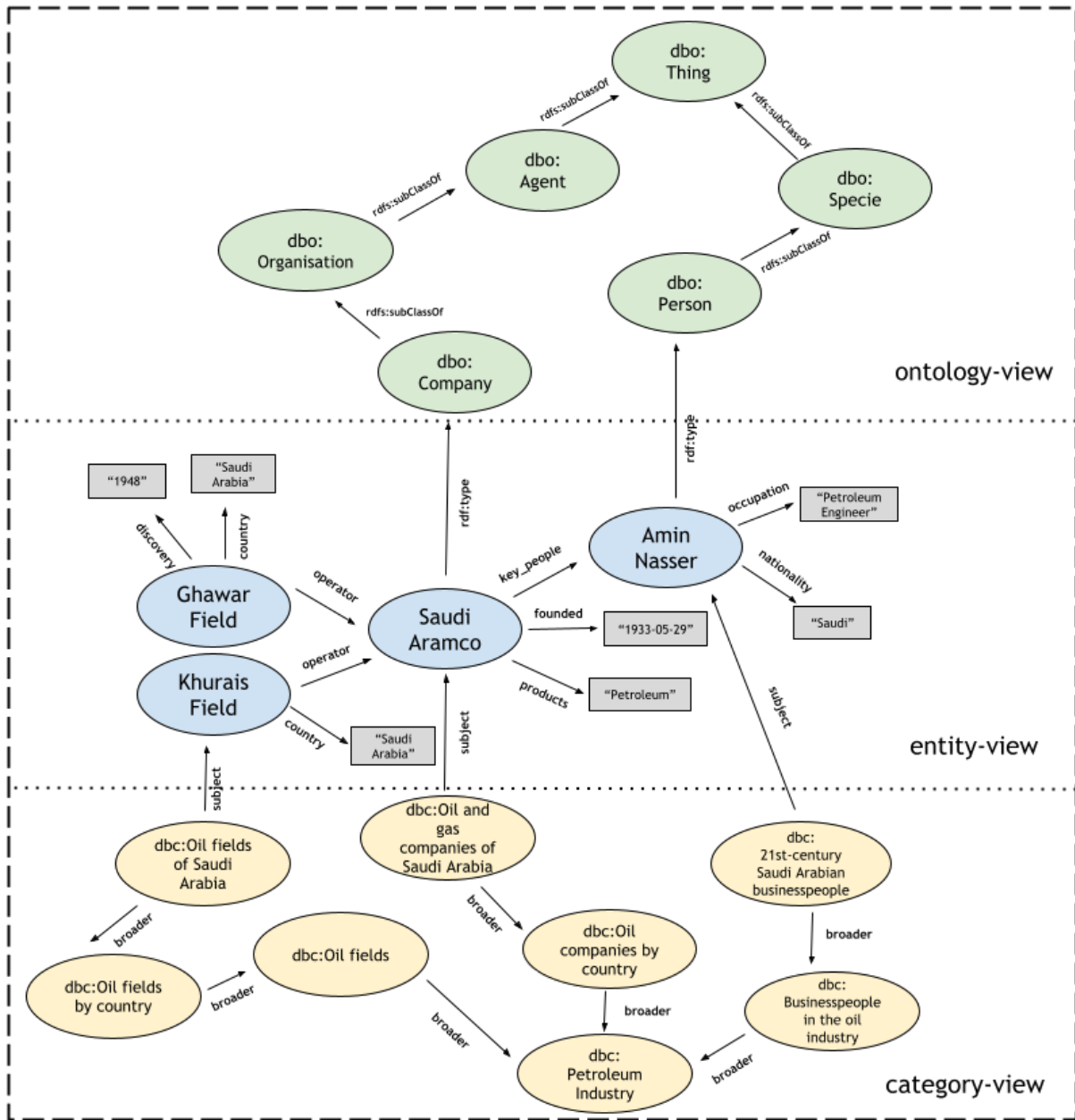
KBs are built to organize and share knowledge about a domain of interest (e.g., health, entertainment, bibliography, arts). Many KBs are publicly available such as DBpedia (LEHMANN et al., 2015), Wikidata (VRANDEČIĆ; KRÖTZSCH, 2014), and YAGO (TANON; WEIKUM; SUCHANEK, 2020). A common characteristic among these KBs is that they have been constructed to store cross-domain knowledge. For example, DBpedia is a large-scale popular general-purpose KB. It is estimated to have approximately 850 million structured and semantic facts about more than 5 million entities<sup>1</sup>. As a result, many applications take advantage of this semantic structure to carry out specific tasks.

Silva, Ziviani and Porto (2019) highlight the use of KBs both in industry and academia. For instance, in industry, search engines, e.g., Google and Bing, have been using KBs to augment their search results (YIH et al., 2015) by using factual knowledge about entities to refine a query and present structured content about a search entity to the user. In addition, this knowledge can be used in conversation and user interaction applications, such as chatbots and virtual assistants (e.g., Alexa and Cortana). In academia, KBs are used in several contexts. For example, natural language processing models have leveraged entities and their attributes from KBs for information extraction tasks (MOREIRA; BARBOSA, 2021) and machine learning models have used knowledge embedded in KBs to detect fake news (PAN et al., 2018; XU et al., 2022).

In this thesis, we are interested in general-purpose KBs. Commonly, general-purpose KBs, like DBpedia, extract their knowledge from Wikipedia. Wikipedia is a multilingual encyclopedia. It has a set of pages, in which each one describes the content of an entity. In DBpedia, for example, each Wikipedia page is equivalent to an entity. The main elements extracted from each Wikipedia page are data from infoboxes and categories. They form a large semantic graph, as illustrated in Figure 1.

<sup>1</sup> <https://www.dbpedia.org/blog/dbpedia-snapshot-2022-09-release/>

Figure 1 – Illustration of different views of a semantic graph extracted from DBpedia.



Source: Created by the author

Figure 1 illustrates the organization of the DBpedia from different views. The *ontology-view* presents a snippet of the ontology used by DBpedia, in which each class defined in the semantic graph is represented by green ellipses. Generally, KBs use an ontology to provide a formal representation of existing concepts, i.e., classes (e.g., `Person`). In this sense, each class has attributes (e.g., `nationality`, `occupation`). Classes are organized into a taxonomic hierarchy (e.g., `Company` is a subclass of `Organization`) (SILVA; ZIVIANI; PORTO, 2019). The *entity-view* presents entity factual data mainly extracted from Wikipedia infoboxes. In this view, entities are nodes (represented by blue ellipses) connected to other nodes (entity or literals - represented by gray rectangles), through edges that have labels,

which indicate the name of an attribute. The *category-view* shows mappings extracted from Wikipedia categories. In this view, categories are nodes (represented by yellow ellipses) and are connected to other categories (through the `broad` property) or to entities (through the `subject` property). In this context, categories are a way of organizing entities thematically. Therefore, they can be seen from two perspectives: the `broad` property relates two categories, and it captures the subcategory relationship, e.g., `dbc:Oil fields of Saudi Arabia` is a subcategory of `dbc:Oil fields by country`, whereas the `subject` property relates an entity to a category, mapping themes related to the content of an entity, e.g., the `Khurais Fields` entity is associated with the `dbc:Oil fields of Saudi Arabia` category.

One way to explore the contents of a KB is through the **data schema**. Specifically, KBs represent their knowledge through the Resource Description Framework (RDF) model<sup>2</sup>. In this context, the RDF Schema (RDFS)<sup>3</sup> model can be used to create an ontology, which defines a set of classes and attributes used to organize entities in a KB. For example, `rdfs:type` declarations specify the kind of an entity, i.e., which class an entity belongs to. In Figure 1, it is possible to identify some entities that have the `rdf:type` declaration. For instance, the entity `Saudi Aramco` has a relationship with the class `dbo:Company` (in the DBpedia ontology), which indicates that it is a company.

According to Kellou-Menouer and Kedad (2015), *"in the Web of data, the notion of schema is understood as a guide, not as a strict representation to which the data must conform"*. It is important to highlight that schema-related statements, such as `rdf:type`, are not mandatory and may not always be provided (BOUHAMOUM; KEDAD; LOPES, 2020). In the DBpedia case, there is a significant amount of entities without this information (HEIST; PAULHEIM, 2019). In the example shown in Figure 1, the entities `Ghawar Field` and `Khurais Field` do not have this declaration. Also, another problem is that many entities are mapped to generic classes, e.g., `Tom Hanks` is mapped to the `dbo:Person` class, when in fact, the `dbo:Actor` class best represents him (SOFRONOVA et al., 2020). Furthermore, KBs have a flexible structure. In this sense, entities of the same kind have a set of distinct attributes. In other words, they do not use schema-based constraints to define an entity. For instance, the entities `Ghawar Field` and `Khurais Field` are both oil fields; however, the first has the `discovery` attribute, while the second does not (Figure 1).

Specifically, a lack of schema-related information makes it difficult to understand and consume this data. To overcome this problem, schema discovery approaches have been proposed in the literature to identify a data schema from a dataset (CHRISTODOULOU; PATON; FERNANDES, 2015; KELLOU-MENOUER; KEDAD, 2015; BOUHAMOUM; KEDAD; LOPES, 2020). Recently, Kellou-Menouer et al. (2022) published a survey identifying and classifying the main approaches to schema discovery according to the target problem. Among them, approaches to implicit schema discovery stand out. According to them,

<sup>2</sup> <<https://www.w3.org/RDF/>>

<sup>3</sup> <<https://www.w3.org/TR/rdf-schema/>>

these approaches *"try to discover the schema of a data source, usually without needing any additional information about the schema declared in the dataset"*.

In this thesis, we address the problem of implicit schema discovery from general-purpose KBs. As mentioned earlier, KBs are a valuable dataset for many applications. Its flexible nature and lack of schema-related information are issues that make the consumption of this data a challenge for interested users and applications.

According to Lalithsena et al. (2017), most applications that use data from general-purpose KBs are domain-specific. A domain-specific application is designed to address specific needs and requirements within a particular domain. For example, an application in the *oil and gas* domain is interested in data from oil companies, oil fields, and refineries, among other data related to the domain. As general-purpose KB stores data that cover many domains, these applications only require a subset of KB data that meets the domain of interest. In this context, domain-specific applications face two issues: (1) general-purpose KBs are broad and have a large volume of knowledge in the most varied domains (e.g., arts, politics, industry, entertainment); (2) the lack of schema-related information. Therefore, we conclude that investigating methods to identify a dataset related to the domain of interest as well as its schematic description from general-purpose KB is a relevant task that can benefit different applications.

Some data processing and management applications rely on a schema to perform their tasks. In this context, we present below examples of applications that can benefit from schema discovery approaches:

- **Query Formulation** - The lack of schema-related information in a dataset makes its use difficult. To formulate a query, it is necessary to know the dataset's structure, e.g., classes, attributes, and resources. Prior knowledge of this structure helps in the process of formulating a structured query (ISSA et al., 2019). For example, in Campinas et al. (2012), the authors propose a prototype that relies on a graph summary approach to help users formulate SPARQL queries without prior knowledge about the RDF dataset structure.
- **Information Extraction** - Applications for information extraction based on the slot-filling task depend on a schema to guide the extraction process. For example, Moreira and Barbosa (2021) presents DeepEx, a system that autonomously extracts missing attributes of entities in knowledge bases from unstructured text. In this context, DeepEx relies on a schema to extract values for attributes missing in an entity. Thus, providing a description containing the main attributes of a class can be helpful for this task.
- **Q&A Systems** - One of the challenges of Q&A systems is to map a question written in natural language to a structured query, e.g., SPARQL query, as in Adolphs et al. (2011), Han, Finin and Joshi (2011), Zhang et al. (2020). In this sense, making the

schema of the dataset used by such a system available in machine-readable format can be an alternative to facilitate this task.

- **Data Integration** - Data integration allows multiple data sources to combine their data and provide an integrated view (DONG; SRIVASTAVA, 2015). One of the steps in this process is schema mapping. Specifically, this step relies on the schema of the data sources to create an integrated schema and consequently provide a single view of data from multiple datasets (MA et al., 2022).

## 1.2 PROBLEM AND RESEARCH QUESTIONS

Given the above context, we focus on the problem of domain-specific schema discovery from general-purpose KBs.

In this thesis, we formalize the schema discovery problem similar to Bouhamoum, Kedad and Lopes (2020): Given a set of entities  $E$  in a dataset  $D$ , the goal is to discover the schema of  $D$ , which is composed of a set of classes  $C = \{c_1, \dots, c_n\}$ , where each  $c_i$  corresponds to a subset of  $E$ , representing a set of similar entities. Each class  $c_i$  is represented by a set of attributes  $\{a_1^i, \dots, a_m^i\}$ .

Thus, given a dataset  $D$  in a specific domain, we split the schema discovery problem into two complementary tasks: *class identification* and *class schema discovery*. In this thesis, a class is represented by a set of entities that share some attributes.

First, the class identification task seeks to discover implicit classes in the dataset by grouping similar entities. Second, the class schema discovery task builds the class schema, i.e., it identifies a set of relevant attributes that best describe the entities of a class. KBs have a flexible structure. In other words, entities within the same class can have different attributes. If, on the one hand, this flexibility facilitates the publication of data, on the other hand, the heterogeneity between the attributes of entities within the same class can make it difficult to understand and consume this data. Considering attributes are not equally relevant to a class, the class schema discovery task explores a set of entities within the same class to select a set of core attributes to provide a unified description for a class.

In this sense, we address the following research questions that we intend to answer throughout this thesis work:

### ***RQ1: How to extract a specific domain from KB?***

General-purpose KBs are broad and domain-specific applications require only a portion of data. Therefore, identifying a subset of entities of interest to the application is a relevant task. Previous schema discovery approaches deal with datasets in a well-defined domain, unlike the context under study in this thesis.

---

***RQ2: How to identify similar entities in a KB, and which entity characteristics are the most effective for performing the class identification task?***

A primary task in schema discovery is identifying the groups of similar entities in a dataset. Previous approaches (NGUYEN et al., 2012; KELLOU-MENOUER; KEDAD, 2015; CHRISTODOULOU; PATON; FERNANDES, 2015; BOUHAMOUM; KEDAD; LOPES, 2020) use an attribute-based strategy, i.e., group entities according to the attributes describing them. However, KBs have a flexible structure and do not establish schema-based restrictions on defining an entity. In this context, entities of the same kind might have different attributes. Furthermore, groups of similar entities can have overlapping sets of attributes. Specifically, in this context, attribute-based approaches may not perform well in this task.

***RQ3: How to select which attributes are relevant to represent a set of entities within an entity class?***

Since entities within the same class have heterogeneous sets of attributes, this task seeks to find a set of core attributes for a class. Previous approaches (CHRISTODOULOU; PATON; FERNANDES, 2015; KELLOU-MENOUER; KEDAD, 2015) consider the union of the set of attributes of all entities. In our context, this strategy may be unsuitable because using all attributes can be cumbersome (WANG et al., 2015) and attributes are not equally relevant. Some works have proposed approaches based on the frequency of attributes (WEISE; LOHMANN; HAAG, 2016; ISSA et al., 2019). Overall, they depend on parameters to specify the most appropriate threshold for attribute selection.

### 1.3 OBJECTIVES

The main objective of this thesis work is to propose an end-to-end pipeline for the specific-domain schema discovery task from general-purpose KB. To achieve this goal, we have set some specific objectives:

- Explore the organization of general-purpose KBs to find specific domains and define specific schemata.
- Extract a specific domain from a general-purpose KB.
- Identify a set of entities within a specific domain.
- Create a novel approach to identify implicit classes, i.e., groups of similar entities from a dataset.
- Devise a novel approach to build the class schema, i.e., select a set of core attributes from a set of entities within the same class.

- Perform experiments in different domains to evaluate the proposed solutions in each task of the schema discovery process.

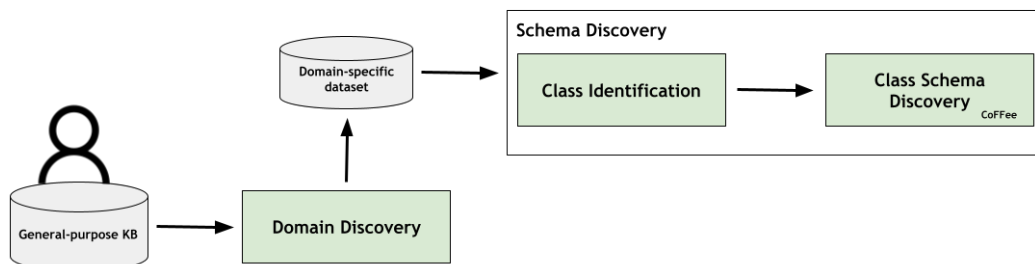
#### 1.4 SOLUTION OVERVIEW

In this thesis, we present ANCHOR (dom**A**iN s**CH**ema disc**O**ve**R**y), an end-to-end pipeline to perform domain-specific schema discovery from a general-purpose KB in an automated way. To do this, we exploit some resources provided by the general-purpose KB.

One of them is categories (SENOUSSI, 2018). Categories organize the KB entities thematically. In this sense, we see categories from two perspectives. There are *entity-category* mappings, which map an entity to a category. There are also *category-category* mappings, which map the relationships between categories forming an extensive taxonomy (HEIST; PAULHEIM, 2019) - see Figure 1 (category-view). In this context, we believe that categories can be valuable resources to reach our goal. For this reason, in this work, we seek to answer the research questions addressed by leveraging both mappings.

ANCHOR works in three steps (Figure 2). First, it extracts a specific domain  $D$  of a  $KB$  based on a user-selected seed category  $c$  used for category-category mappings exploration. In this step, we aim to find a set of subcategories of  $c$  that represents the domain of interest (Domain Discovery). Next, ANCHOR identifies domain entities through *entity-category* mappings and groups them into classes using an unsupervised learning approach (Class Identification). Previous work has proposed attribute-based approaches to perform this task (NGUYEN et al., 2012; KELLOU-MENOUER; KEDAD, 2015; CHRISTODOULOU; PATON; FERNANDES, 2015; MEHRI; VALTCHEV, 2017). However, in some cases looking only at the attributes of entities is not enough. To overcome this limitation, we propose a new approach to entity representation. In this direction, we use a node embedding algorithm to learn a representation of entities from the *entity-category* mappings. We assume that entities that share similar categories may be closer in representation space. Finally, in the last step, ANCHOR runs CoFFee, an approach based on attributes co-occurrence and frequency to identify a set of core attributes for each class discovered in the previous step (Class Schema Discovery).

Figure 2 – Graphical abstract - ANCHOR pipeline steps.

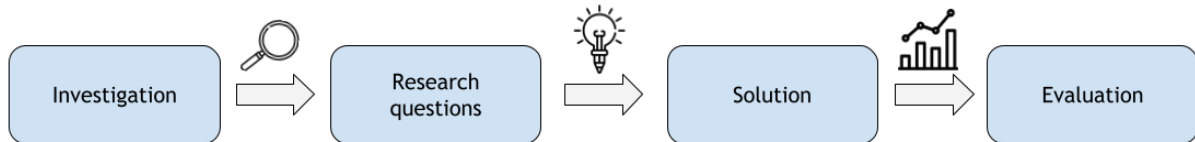


Source: Created by the author

## 1.5 RESEARCH METHODOLOGY

This section briefly describes the methodology used in this thesis work. Figure 3 illustrates each methodology phase.

Figure 3 – Research methodology overview.



**Source:** Created by the author

Initially, we investigated the research topic of this thesis (schema discovery). We conducted an exploratory study to obtain an overview of the schema discovery approaches. Furthermore, this study allowed identifying limitations and possible improvements to contribute to the state-of-the-art. Next, we investigate the structure of the general-purpose KBs (research context) to identify their resources and understand how these repositories organize their content. This initial phase allowed obtaining a background for the research topic and delimiting the research context: domain-specific schema discovery from general-purpose KBs.

Based on this context, we address three research questions (described in Section 1.2) that guided the development of this thesis. This work is classified as applied research since it seeks to advance the state-of-the-art from practical applications to solve a specific problem.

In the solution phase, we devise an end-to-end pipeline split into three steps (described in Section 4), in which each step seeks to answer a research question. In this phase, we carry out the implementation of the solution. Finally, in the last research phase, we evaluated the proposed solution. We define some question and answer them from an extensive experimental evaluation (described in Section 5). We analyzed the results from a quantitative approach. In this context, we compared the results with state-of-the-art baselines to measure the contributions of this thesis work. Also, we write some papers. These manuscripts have been submitted to journals and conferences (see Section 6.4).

## 1.6 CONTRIBUTIONS

These are the main contributions of this thesis:

- An end-to-end solution that, given a category of interest that defines a domain, outputs the domain schema with a set of entity classes and their respective attributes.
- A novel strategy to create entity representations based on their category mappings.

- 
- An approach based on attribute co-occurrence and frequency that, given a set of entities within the same class, identifies a set of relevant attributes for the class.
  - An extensive experimental evaluation in four specific domains, extracted from DBpedia, to evaluate the proposed solutions.

## 1.7 THESIS ORGANIZATION

In addition to this chapter, this thesis is organized as follows:

- Chapter 2 presents the concepts and techniques related to the solutions proposed in this thesis.
- Chapter 3 discusses some related work and compares it with this thesis work. We've organized these works into two sections, grouping them by task (Domain Discovery and Schema Discovery).
- Chapter 4 describes the ANCHOR pipeline. Initially, we formalize some definitions and detail each step of the pipeline. We describe the process used to learn an entity representation from entity-category mappings, i.e., entity embeddings, and specify how CoFFee, our solution for the class schema discovery task, works. Both are the main contributions of this work.
- Chapter 5 details the experimental evaluation carried out in this work. Specifically, we performed experiments to evaluate Class Identification and Class Schema Discovery tasks using four domains from DBpedia.
- Chapter 6 concludes this thesis work. We present the contributions achieved and address some limitations and future work. In addition, we list the papers produced during the doctoral course.

## 2 BACKGROUND

This chapter introduces some concepts and techniques applied in this thesis. Section 2.1 shows an overview of Wikipedia content and discusses two of its main elements: infoboxes and categories. Section 2.2 briefly discusses the KB concept and presents an overview of some general-purpose KBs. Section 2.3 introduces Representation Learning and focuses mainly on two techniques: Word Embedding and Node Representation. Finally, Section 2.4 addresses the concept of unsupervised learning, specifically discussing clustering approaches and assessment measures for clustering data.

### 2.1 WIKIPEDIA

Wikipedia is a collaborative, universal, and multilingual encyclopedia established on the internet under the *wiki* principle (MOREIRA; COSTA-NETO; BARBOSA, 2019). Wikipedia consists of a set of pages describing the content of an entity (e.g., politicians, artists, athletes). The structure of a page has different elements, such as text (organized in sections), an infobox, and categories.

Figure 4 shows Bill Gates’s page<sup>1</sup> extracted from Wikipedia. In this example, we identify these elements, e.g., text (orange), infobox (green) and categories (red). The page has text describing the entity (**Bill Gates**). It is organized into sections and has links to other entities mentioned in the text. These links allow for integration and navigation between Wikipedia contents. Furthermore, the page has an infobox (summarizing the content in a structured way) and categories (mapping the page to themes related to its content).

Wikipedia has a large number of pages and covers a variety of domains. In addition, its content is rich in semantics. According to its official page<sup>2</sup>, the English version surpassed six million articles in January 2020. Due to these characteristics, Wikipedia has been widely used as a data source in different works.

The initiative to extract and structure data from Wikipedia has helped create some knowledge bases, such as DBpedia (LEHMANN et al., 2015), YAGO (REBELE et al., 2016) and Wikidata (VRANDEČIĆ; KRÖTZSCH, 2014). Specifically, the DBpedia project is one of the main contributions of Wikipedia. DBpedia has several datasets<sup>3</sup> that store data extracted from more than 6.6 million entities (pages). Among the data extracted from Wikipedia, *infoboxes* and *categories* data stand out.

<sup>1</sup> <[https://en.wikipedia.org/wiki/Bill\\_Gates](https://en.wikipedia.org/wiki/Bill_Gates)>

<sup>2</sup> <[https://en.wikipedia.org/wiki/English\\_Wikipedia](https://en.wikipedia.org/wiki/English_Wikipedia)>

<sup>3</sup> <<https://databus.dbpedia.org/dbpedia/collections/latest-core>>

Figure 4 – Snippet from Bill Gates’s page. The main elements are highlighted by color: abstract/text (orange); infobox (green); sections (purple); categories (red).


**William Henry Gates III** (born October 28, 1955) is an American business magnate, software developer, and philanthropist. He is best known as the co-founder of **Microsoft Corporation**.<sup>[2][3]</sup> During his career at Microsoft, Gates held the positions of **chairman**, **chief executive officer** (CEO), **president** and **chief software architect**, while also being the largest individual **shareholder** until May 2014. He is one of the best-known entrepreneurs and pioneers of the **microcomputer revolution** of the 1970s and 1980s.

Born and raised in **Seattle, Washington**, Gates co-founded Microsoft with childhood friend **Paul Allen** in 1975, in **Albuquerque, New Mexico**; it went on to become the world's largest **personal computer** software company.<sup>[4][a]</sup> Gates led the company as chairman and CEO until stepping down as CEO in January 2000, but he remained chairman and became chief software architect.<sup>[7]</sup> During the late 1990s, Gates had been criticized for his business tactics, which have been considered **anti-competitive**. This opinion has been upheld by numerous court rulings.<sup>[8]</sup> In June 2006, Gates announced that he would be transitioning to a part-time role at Microsoft and full-time work at the **Bill & Melinda Gates Foundation**, the private charitable foundation that he and his wife, **Melinda Gates**, established in 2000.<sup>[9]</sup> He gradually transferred his duties to **Ray Ozzie** and **Craig Mundie**.<sup>[10]</sup> He stepped down as chairman of Microsoft in February 2014 and assumed a new post as technology adviser to support the newly appointed CEO **Satya Nadella**.<sup>[11]</sup> In March 2020, Gates left his board positions at Microsoft and **Berkshire Hathaway** to focus on his philanthropic endeavors including **climate change**, **global health** and **development**, and **education**.<sup>[12]</sup>

Since 1987, he has been included in the **Forbes** list of the world's wealthiest people.<sup>[13][14]</sup> From 1995 to 2017, he held the **Forbes** title of the richest person in the world all but four of those years.<sup>[1]</sup> In October 2017, he was surpassed by **Amazon** founder and CEO **Jeff Bezos**, who had an estimated net worth of US\$90.6 billion compared to Gates's net worth of US\$89.9 billion at the time.<sup>[15]</sup> As of August 2020, Gates had an estimated net worth of US\$113.7 billion, making him the second-wealthiest person in the world, behind Bezos.<sup>[16][b]</sup>

Later in his career and since leaving day-to-day operations at Microsoft in 2008, Gates has pursued a number of philanthropic endeavors. He has given sizable amounts of money to various charitable organizations and scientific research programs through the **Bill & Melinda Gates Foundation**, reported to be the world's largest **private charity**.<sup>[18]</sup> In 2009, Gates and **Warren Buffett** founded **The Giving Pledge**, whereby they and other billionaires pledge to give at least half of their wealth to philanthropy.<sup>[19]</sup>

**Bill Gates**



Gates in 2018

<b>Born</b>	William Henry Gates III October 28, 1955 (age 64) Seattle, Washington, U.S.
<b>Education</b>	Lakeside School, Harvard University (dropped out)
<b>Occupation</b>	Software developer · investor · entrepreneur
<b>Years active</b>	1975–present
<b>Known for</b>	Co-founder of Microsoft
<b>Net worth</b>	US\$105.6 billion (May 2020) <sup>[1]</sup>
<b>Title</b>	Co-chairman and co-founder of the Bill & Melinda Gates Foundation Chairman and founder of Branded Entertainment Network Chairman and co-founder of TerraPower

**Contents** [hide]

- Early life
- Microsoft
  - BASIC
  - IBM partnership
  - Windows
  - Management style
  - Antitrust litigation

**Categories:** Bill Gates | 1955 births | 20th-century American businesspeople | 20th-century American engineers | 21st-century American businesspeople | 21st-century American engineers | 21st-century philanthropists | American billionaires | American chairmen of corporations | American computer businesspeople | American computer programmers | American corporate directors | American financiers | American humanitarians | American inventors | American investors | American nonprofit chief executives | American people of English descent | American people of German descent | American people of Scotch-Irish descent | American people of Scottish descent | American philanthropists | American software engineers | American technology chief executives | American technology company founders | American technology writers | American venture capitalists | Big History | Bill & Melinda Gates Foundation people | Businesspeople from Seattle | Businesspeople in software | Cornell family | Directors of Berkshire Hathaway | Directors of Microsoft | Dudley–Winthrop family | Fellows of the British Computer Society | Foreign members of the Chinese Academy of Engineering | Freemen of the City of London | Gates family | Giving Pledgers | Grand Cordons of the Order of the Rising Sun | Harvard College alumni | History of Microsoft | History of computing | Honorary Knights Commander of the Order of the British Empire | HuffPost bloggers | Lakeside School alumni | Living people | Members of the United States National Academy of Engineering | Microsoft employees | National Medal of Technology recipients | Nerd culture | People from Medina, Washington | People from Seattle | Personal computing | Phillips family (New England) | Placards of the Order of the Aztec Eagle | Presidential Medal of Freedom recipients | Recipients of the Cross of Recognition | Recipients of the Padma Bhushan in social work | Spokespersons | Windows people | Wired (magazine) people | Writers from Seattle

Source: Wikipedia (2023)

### 2.1.1 Infoboxes

An infobox has a set of attribute-value pairs that summarize in a structured way information about an entity. Due to their structure, some communities, e.g., Databases, Semantic Web, and Natural Language Processing, use infobox data to leverage their tasks (LANGE; BOHM; NAUMANN, 2010; NGUYEN et al., 2011; SERRA et al., 2011; NGUYEN et al., 2012; KUZHEY; WEIKUM, 2012; WECHEL; LEWONIEWSKI, 2015; MORALES et al., 2016; SáEZ; HOGAN, 2018).

An infobox is created from an *infobox template*. Overall, an infobox template suggests a set of attributes to related pages. For example, Bill Gates’ infobox (Figure 4) was created from the INFOBOX\_PERSON template. This template suggests generic attributes that describe people, such as **name**, **birthplace**, **nationality**, and **occupation**. Figure 5 illustrates the template INFOBOX\_PERSON. Note that on the left side are suggested attributes for an infobox describing a person. The right side provides a preview of the infobox arrangement on the page. It is important to highlight that there are no restrictions on filling in the

suggested attributes. In other words, an editor, i.e., the person who will create an infobox, can fill in all or only part of the values for the suggested attributes. Furthermore, it is possible to include other attributes outside the infobox template because there are no constraints to prevent this. In short, an infobox can be freeform defined.

Figure 5 – Guidelines for Infobox\_person template.

### Usage

The infobox may be added by pasting the template as shown below into an article and then filling in the desired fields. Any parameters left blank or omitted will not be displayed.

#### Blank template with basic parameters

```

{{Infobox person
| name           = <!-- defaults to article title when left blank -->
| image          = <!-- filename only, no "File:" or "Image:" prefix, and no enclosing [[brackets]] -->
| alt            = <!-- descriptive text for use by speech synthesis (text-to-speech) software -->
| caption        =
| birth_name     = <!-- only use if different from name -->
| birth_date     = <!-- {{Birth date and age|YYYY|MM|DD}} for living people supply only the year with {{Birth year and age|YYYY}} unless the exact date is already widely published, as per [[WP:DOB]]. For people who have died, use {{Birth date|YYYY|MM|DD}}. -->
| birth_place    =
| death_date     = <!-- {{Death date and age|YYYY|MM|DD|YYYY|MM|DD}} (DEATH date then BIRTH date) -->
| death_place    =
| nationality     = <!-- use only when necessary per [[WP:INFONAT]] -->
| other_names    =
| occupation     =
| years_active   =
| known_for      =
| notable_works  =
}}
```

<b>name</b>	
This is an example image	
<b>caption</b>	
<b>Born</b>	birth_name birth_date birth_place
<b>Died</b>	death_date death_place
<b>Nationality</b>	nationality
<b>Other names</b>	other_names
<b>Occupation</b>	occupation
<b>Years active</b>	years_active
<b>Known for</b>	known_for
<b>Notable work</b>	notable_works

Source: Wikipedia (2023)

There is an effort to standardize the use of specific templates for entities of the same kind. For example, the INFOBOX\_COMPANY template for entities that describe *companies* and the INFOBOX\_FILM template for entities describing *films*. However, entities of the same kind can be assigned to a different template (NGUYEN et al., 2012). In this sense, there is an effort to map infobox templates to DBpedia ontology classes. This mapping also seeks to resolve conflicts at the attribute level. For instance, the same attribute is written in different ways, e.g., *birthplace* and *place-of-birth* (HAHN et al., 2010).

Infobox data extracted from Wikipedia is available from DBpedia in the following datasets:

- **infobox-properties** - Infobox data is extracted from a generic parser and organized in a triple format. Figure 6(a) shows an example of a triple contained within this dataset. In this example, the entity *The Beatles* has the attribute *genre* whose value is *Rock music*. In this dataset, an entity has several triples since each triple refers to an attribute of its infobox.
- **mappingbased objects** - It uses a specific parser that maps an infobox template to an ontology class and, consequently, an infobox attribute to a class property. This dataset has a higher quality than the previous dataset. The data is cleaner, and

attribute heterogeneity conflicts are resolved. However, there is a significant number of templates that are not mapped to ontology. Consequently, there is less coverage concerning the number of entities and attributes in this dataset (MOREIRA; COSTA-NETO; BARBOSA, 2019). Specifically, this dataset stores triples whose attribute value is an object, i.e., a mention of another entity. For instance, Figure 6(b) shows an example of a triple contained within this dataset. In this example, the entity **Bill Gates** has the attribute **education**. Note that this attribute is mapped to a property of the **Person** class in the DBpedia ontology. Also, the object value makes reference to the other entity (**Havard University**).

- **mappingbased literals** - This dataset is similar to the one described previously (mappingbased objects) concerning the parser used. The difference between them is that this dataset stores triples whose attribute value is literal, i.e., a text, numerical, or date value. For instance, Figure 6(c) shows an example of a triple contained within this dataset. In this example, the entity **Bill Gates** has the attribute **birthDate** whose value is a data (*1955-10-28*).

Figure 6 – Example of triples from infobox data within DBpedia datasets.

```

1 (
2   <http://dbpedia.org/resource/The_Beatles>
3   <https://dbpedia.org/property/genre>
4   "Rock_music"@en.
5 )
7 Subject = The_Beatles
8 Predicate = genre
9 Object = Rock_music

```

(a) – infobox-properties dataset

```

1 (
2   <http://dbpedia.org/resource/Bill_Gates>
3   <https://dbpedia.org/ontology/education>
4   <http://dbpedia.org/resource/Havard_University>
5 )
7 Subject = Bill_Gates
8 Predicate = education
9 Object = Havard_University

```

(b) – mappingbased objects dataset

```

1 (
2   <http://dbpedia.org/resource/Bill_Gates>
3   <https://dbpedia.org/ontology/birthDate>
4   "1955-10-28"
5 )
7 Subject = Bill_Gates
8 Predicate = birthYear
9 Object = "1955-10-28"

```

(c) – mappingbased literals dataset

**Source:** Created by the author

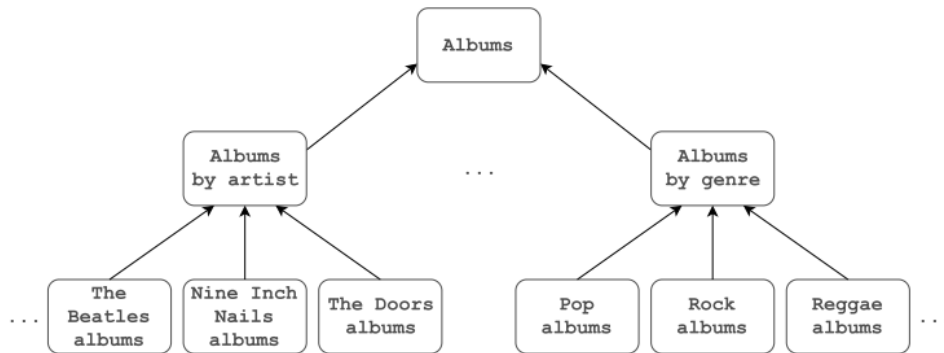
### 2.1.2 Categories

Wikipedia categories are valuable sources of information. According to Boldi and Monti (2016), categories group pages related to similar subjects. It allows users to browse a set of related articles. Each page on Wikipedia is assigned one or more categories. For example, in Figure 4, it is possible to observe that Bill Gates' page is assigned to some categories, e.g., **1955 births** (people born in the year 1955) and **Directors of Microsoft** (people who took a management position at Microsoft). In this context, it is possible to see that the categories assigned to this page have a relationship with the entity described on that page.

Organizing content by category allows the user to explore Wikipedia pages by theme. For example, when browsing the category **1955 births**, it is possible to find a set of pages associated with other people born in the same year.

In addition to the mapping between a page and a category, Wikipedia has a mapping between categories. In other words, this mapping captures the subcategory relationship and creates a structure known as a *wikipedia category graph*<sup>4</sup>. Figure 7 shows an excerpt from a category graph extracted from Wikipedia. In this example, the category **The Beatles albums** is a subcategory of **Albums by artist**, which is a subcategory of **Albums**.

Figure 7 – Excerpt of the Wikipedia category graph showing the category Albums together with some of its subcategories.



**Source:** Heist and Paulheim (2019)

Categories are created and organized by Wikipedia editors and contributors. Editors establish guidelines for creating and mapping a category. For example, they recommend avoiding cycles between category mappings. In the category graph, a cycle hinders navigation between categories and prevents automated processes in this structure (AOUICHA; TAIEB; EZZEDDINE, 2016). Furthermore, it is recommended to observe the semantics between the mappings of a category and its subcategories. Disconnected categories make it difficult for the user to navigate the graph.

There is an effort for categories to follow standards that facilitate content navigation. One of these patterns is the specialization by theme. In Figure 7, it is possible to observe

<sup>4</sup> <[https://en.wikipedia.org/wiki/Wikipedia:Category\\_tree](https://en.wikipedia.org/wiki/Wikipedia:Category_tree)>

the specialization of the **Albums** category by two attributes: artist and genre. For example, the **Albums by genre** category maintains a subcategory relationship between different music genres, such as pop, rock, and reggae. In this sense, the semantics of these mappings contribute so that the content can be explored in a fine-grained way.

Some works explored the potential of the categories in several contexts, such as: creating taxonomies for a specific domain (KOTLERMAN et al., 2011; VIVALDI; RODRÍGUEZ, 2010), defining semantic annotations for entities (TITZE et al., 2014), and refining the navigation of the content from the perspective of the entity (RAYNAUD; SUBERCAZE; LAFOREST, 2018). These works are briefly discussed in Section 3. Other papers focused on performing a semantic analysis to extract knowledge from categories (NASTASE; STRUBE, 2008) and infer new facts for knowledge base augmentation (HEIST; PAULHEIM, 2019).

Specifically, Wikipedia category data is available by DBpedia in two datasets: *articles-categories* e *skos-categories*.

- **articles-categories** - This dataset stores, in triples format, the mappings between a page (entity) and a category. These elements are connected through the **subject** predicate. For instance, Figure 8(a) shows an example of a triple within this dataset. In this example, the entity **Bill Gates** has a relationship with the category **1955 births**.
- **skos-categories** - This dataset stores, in triples format, the relationship between a category pair. These categories are mapped through the **broader** predicate. In this sense, given two categories  $c_i$  *broader*  $c_j$ , it is understood that  $c_i$  is a subcategory of  $c_j$ . For instance, Figure 8(b) shows an example of a triple within this dataset. In this example, the category **Albums by genre** has a subcategory relationship with the category **Albums**.

Figure 8 – Example of triples from categories data within DBpedia datasets.

```

1 (
2   <http://dbpedia.org/resource/Bill_Gates>
3   <http://purl.org/dc/terms/subject>
4   <http://dbpedia.org/resource/Category:1955_births>
5 )
6
7 Subject = Bill_Gates
8 Predicate = subject
9 Object = Category:1955_births

```

(a) – articles-categories dataset

```

1 (
2   <http://dbpedia.org/resource/Category:Albums_by_genre>
3   <http://www.w3.org/2004/02/skos/core#broader>
4   <http://dbpedia.org/resource/Category:Albums>
5 )
6
7 Subject = Category:Albums_by_genre
8 Predicate = broader
9 Object = Category:Albums

```

(b) – skos-categories dataset

**Source:** Created by the author

## 2.2 KNOWLEDGE BASE

KB or Knowledge Graph (KG) are repositories that store knowledge about a domain. In this context, knowledge represents a collection of facts (triples) in the form <subject, relation, object> (LAN et al., 2022). KB content is modeled as a graph. In this sense, nodes symbolize entities, e.g., real-world objects, and classes assigned to them, whereas edges represent assertions about entities and/or classes (SILVA; ZIVIANI; PORTO, 2019).

Figure 9 illustrates a KB created from the set of triples presented in Table 1. Note that each node in the graph represents a real-world object (entity), e.g., **Douglas Adams** and **Cambridge**, or a class assigned to an entity, e.g., **dbo:Person** and **dbo:Writer**. Edges indicate relationships between a pair of nodes, e.g., **Douglas Adams** and **Cambridge** are connected by the **birthPlace** relationship.

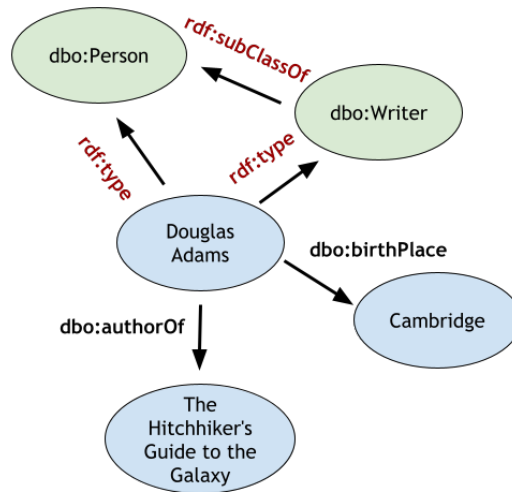
Table 1 – Set of triples representing the knowledge of a KB.

subject	relation	object
Douglas Adams	dbo:authorOf	The Hitchhiker’s Guide to the Galaxy
Douglas Adams	dbo:birthPlace	Cambridge
Douglas Adams	rdf:type	dbo:Writer
Douglas Adams	rdf:type	dbo:Person
dbo:Writer	rdfs:subClassOf	dbo:Person

**Source:** Created by the author

KB provides an adequate semantic structure for computer systems to be able to process this knowledge in an automated way. For this reason, applications in different fields, e.g., search engines (YIH et al., 2015), Q&A systems (LI et al., 2021; LAN et al., 2022), information

Figure 9 – Excerpt of a knowledge base created from a set of triples.



**Source:** Created by the author

extraction (MOREIRA; BARBOSA, 2021), recommendation systems (XU et al., 2021), and fact check (PAN et al., 2018), use KBs to perform their tasks.

KBs use the RDF model to represent and publish its content (ALI et al., 2022). The RDF model is a World Wide Web Consortium (W3C)<sup>5</sup> standard for describing web data. It allows data about a given resource to be represented in a structured and connected way. A resource has a Uniform Resource Identifier (URI) to identify it. Statements about a resource are defined in triples format. In this context, a triple has three main components: resource, property, and value. Considering the example in Figure 9, the resource can be an entity (e.g., *Douglas Adams*). Property is an attribute or relation to describe a resource (e.g., *rdf:type*). The value can be another resource (e.g., *dbo:Person*) or a literal value, i.e., a string or other datatype. RDF is compatible with other web standards such as RDFS and Web Ontology Language (OWL)<sup>6</sup>.

Specifically, the RDFS standard models classes and relationships used to describe resources on the Web in a more precise and structured way. RDFS provides a semantic vocabulary with some classes and properties to model the RDF data structure. The most common element is *rdfs:Class*. In this context, a class groups a set of resources. Members of a class are called instances of the class. The *rdf:type* property states that a resource is an instance of a class. For example, *Douglas Adams* is an instance of the *dbo:Person* and *dbo:Writer* classes (Figure 9).

The relationship between two classes is represented by the *rdfs:subClassOf* property. For instance, in Figure 9, the *dbo:Writer* class is a subclass of *dbo:Person*. A resource has a set of attributes (properties) identified by a name and *rdfs:domain* and *rdfs:range* statements. In other words, the *rdf:domain* statement defines the domain of the attribute,

<sup>5</sup> <<https://www.w3.org/>>

<sup>6</sup> <<https://www.w3.org/OWL/>>

i.e., which class it belongs to, whereas the `rdf:range` statement defines the data type or class of the attribute. For example, for the `dbo:birthPlace` attribute (Figure 9), the values for the properties `rdfs:domain` and `rdfs:range` is `dbo:Person` (indicating that this attribute belongs to the Person class) and `dbo:Place` (indicating that the value of this attribute is an instance of the Place class), respectively. In short, the RDFS model can be used to provide a structural description for a set of resources.

KBs are created by gathering information and data from a variety of sources. For instance, structured sources, e.g., Wikipedia infoboxes and/or web tables, and unstructured sources, e.g., text, articles, web pages (FENSEL et al., 2020). KBs can be built for a specific or for a cross-domain. In this thesis, we are interested in general purpose knowledge bases. In short, a general-purpose knowledge contains a broad and diverse range of information and knowledge, covering multiples domains. Examples of general-purpose knowledge bases include DBpedia, Yet Another Great Ontology (YAGO), and Wikidata. Below, we provide a brief description of each of these.

- **DBpedia** (BIZER et al., 2009; LEHMANN et al., 2015) - It is a general-purpose knowledge base created from data extracted from Wikipedia. DBpedia was launched in 2007 (AUER et al., 2007), and nowadays, it has consolidated itself as one of the most relevant KBs. The result of this project is datasets storing knowledge about multiple domains in different languages. DBpedia is estimated to have approximately 900 million triples (January 2021)<sup>7</sup>. The trend is for this number to continue to grow. DBpedia makes its content available on different datasets. The best-known dataset is *mappings-based*. In this dataset, the data is extracted from infoboxes and mapped to a cross-domain ontology. Currently, the DBpedia ontology has 768 classes and is described by 3.000 different properties<sup>8</sup>. Data can be consumed in two ways: by downloading datasets or using an endpoint to submit queries to your content<sup>9</sup>.
- **YAGO** - It is a general-purpose, multilingual knowledge base. Its first version was released in 2007 (SUCHANEK; KASNECI; WEIKUM, 2007). Over the years, YAGO has been changing. Its first version extracted facts mainly from Wikipedia category names and infoboxes. The next version includes temporal and geographic facts extracted from GeoNames (HOFFART et al., 2013). Version 3 scaled to multilingual fact extraction covering 10 different languages (MAHDISOLTANI; BIEGA; SUCHANEK, 2014). The current version, YAGO4, has facts extracted from Wikidata, and its taxonomy is based on schema.org (TANON; WEIKUM; SUCHANEK, 2020). YAGO content is organized into some datasets, and the most popular is the *Facts* dataset, which

<sup>7</sup> <<https://www.dbpedia.org/resources/latest-core/>>

<sup>8</sup> <<https://www.dbpedia.org/resources/ontology/>>

<sup>9</sup> <<https://dbpedia.org/sparql/>>

stores assertions about the extracted entities. Access to this content is through the project website<sup>10</sup> or SPARQL endpoint<sup>11</sup>.

- **Wikidata** - Wikidata is an open, collaborative, and multilingual Knowledge Base. Its content is extracted from Wikipedia pages (VRANDEČIĆ; KRÖTZSCH, 2014). Wikidata uses the notion of items to organize its content. In this context, items represent any knowledge stored in Wikidata, e.g., resources, topics, and concepts. Each item has a unique identifier. For example, *Douglas Adams* (entity) is equivalent to item Q42. In addition, a property (attribute) also has a unique identifier, e.g., *place\_of\_birth* (P19). For instance, the triple (Q42, P19, Q350) in Wikidata corresponds to the second triple in Table 1. Wikidata provides an endpoint for submitting SPARQL queries online<sup>12</sup>.

## 2.3 REPRESENTATION LEARNING

Representation Learning is a subfield of Machine Learning that seeks to learn helpful and meaningful representations for data, such as text, graphs, and images. In this context, this data representation can be used as input for machine learning algorithms in several tasks, e.g., classification and clustering (BENGIO; COURVILLE; VINCENT, 2013). The idea is to automatically learn a lower-dimensional representation that captures its discriminative features rather than using manually engineered features. In this thesis, we apply an approach for representing nodes in a graph. So, in this section, we focus on discussing two types of representation: words and graphs.

### 2.3.1 Word Representation

The use of unstructured data (text) is increasingly common in applications. However, dealing with natural language is not a trivial task. A text has a set of words that have meanings and need to be interpreted by a human or a machine. Specifically, if the latter is the case, it is necessary to use a strategy to build a word representation machine-readable.

In short, a word representation is a feature vector that expresses its meaning. There are some strategies for word representation, e.g., one-hot-encoding and distributed word representation. Next, we present a brief intuition of these strategies.

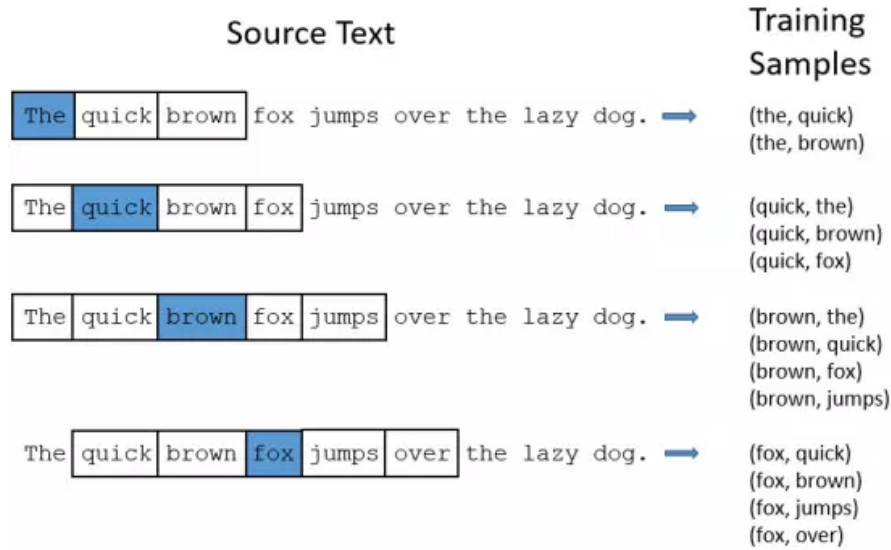
The idea behind the one-hot-encoding approach is to map each word in the vocabulary to an index. In other words, given a vocabulary  $V = \{w_1, w_2, \dots, w_n\}$ , each word  $w_i$  is represented as  $|V|$ -dimensional vector, where each dimension of  $w_i$  is 1 (if  $w = w_i$ ) or 0 (otherwise). This approach is simple, but it has some limitations. Liu, Lin and Sun (2020) points out some of them: i) it does not capture the semantic relatedness among words; ii)

<sup>10</sup> <<https://yago-knowledge.org/>>

<sup>11</sup> <<https://yago-knowledge.org/sparql>>

<sup>12</sup> <<https://query.wikidata.org/>>

Figure 10 – Training sample example - Word2vec.



Source: <<https://cutt.ly/mgZBVkQ>>

it has a high-dimensional representation; iii) it is sensitive to changes since the insertion of new words requires the assignment of new indices and would change the dimensions of the representation.

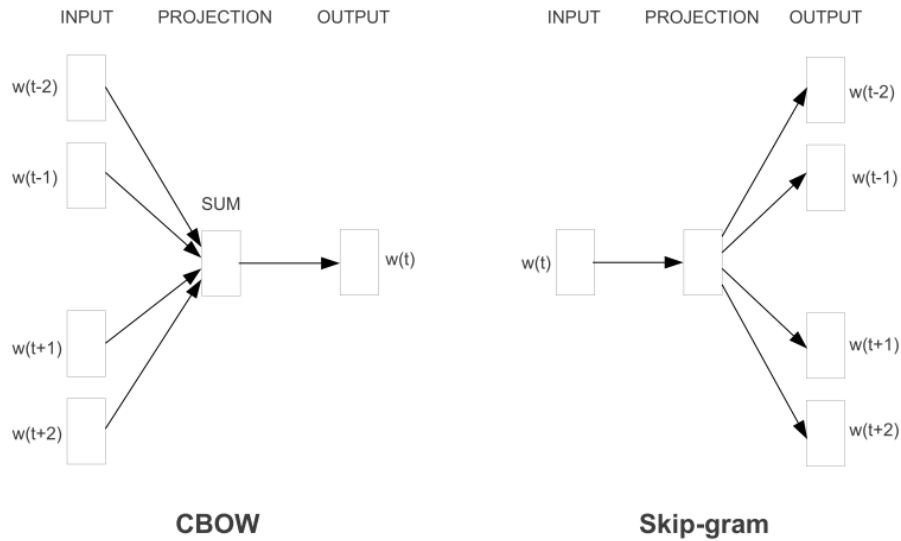
To overcome the limitations presented above, approaches that deal better with the semantics of words were proposed, e.g., Latent Semantic Analysis (LSA) (WIEMER-HASTINGS, 2006) and Word2vec (MIKOLOV et al., 2013a). In the remainder of this section, we focus on the Word2vec approach since the node embeddings algorithm used in this thesis applies Word2vec as a background.

Word2vec (MIKOLOV et al., 2013a; MIKOLOV et al., 2013b) is a technique to learn dense vector representations of words, i.e., word embeddings. Overall, given a text corpus, Word2vec builds a model from a neural network to capture the relationships and meanings between words. The Word2vec architecture has two models: Skip-gram and Continuous Bag-of-Words (CBOW).

The idea of Word2vec is to use contextual information to learn meaningful representations. In this sense, the context is a set of words around a target word. So, the models depend on the setting of the hyperparameter *windows size*. For example, consider the sentence "The quick brown **fox** jumps over the lazy dog", and a windows size of 2 (Figure 10). If the target word is **fox**, its neighboring words will be (quick, brown, jumps, over). Based on this, the training samples will be: (fox, quick), (fox, brown), (fox, jumps), (fox, over).

Once the training sample is extracted, the model can be applied. The main difference between the models is that Skip-gram learns the embeddings that can predict the context words given a target word, while CBOW optimizes the embeddings so that they can predict a target word given its context words. Figure 11 illustrates the architecture of

Figure 11 – Word2vec architecture: CBOW model (left) and Skip-gram model (right).



Source: Mikolov et al. (2013a)

both models.

In the Skip-gram model, given a target word and a window size, the model predicts neighboring words. The word is represented by a vector *one hot encoding*. The model uses a hidden layer with  $E$  neurons, in which  $E$  is the size of the embedding, i.e., features. The input is multiplied by the hidden layer weights. The output layer uses a softmax that produces a vector, in which each element is equivalent to a probability value (between 0 and 1) referring to each vocabulary word.

The CBOW model works in reverse to the skip-gram model. CBOW tries to predict a target word (output) from a context of words (input). For example, if we have four context words used to predict a target word, the input layer of the network will be composed of the vectors that represent those surrounding words. These vectors move to the hidden layer, which returns a  $1 \times E$  dimension vector. The output of the hidden layer is the average of the vectors of the context words.

In short, the representations learned by Word2vec have shown significant results since they capture the semantics between a set of words. In this sense, semantically similar words are close together in a vector space. From these representations, it is possible to perform several tasks, e.g., word similarity, clustering, and classification.

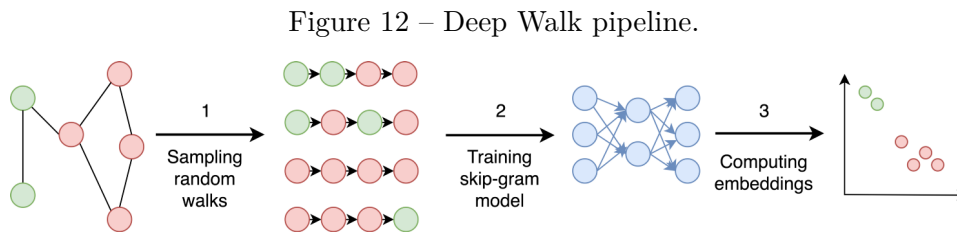
### 2.3.2 Node Representation

A graph is a data structure that consists of a set of vertices (also called nodes) and edges, which edges represent relationships or connections between the vertices. Many real-world applications use graph as a data structure, e.g., social networks (to model social interactions) and knowledge bases (to represent relationships between entities).

According to Cai, Zheng and Chang (2018), graph analytics allows a deeper understanding of the meaning of the data and is beneficial for many applications, e.g., node recommendation and link prediction. For instance, in a social network, the link prediction task can identify real-world friends based on relationships between your neighborhood (friends). Towards this goal, an efficient way to capture the topology of the graph and/or the relationship between the nodes is using techniques to learn node representation, i.e., node embeddings.

Node embedding is a technique for representing nodes from a graph as low-dimensional vectors in representation space. In other words, a node embedding algorithm learns a mapping from the neighborhood, i.e., relationships with other nodes, to a vector representation. In this context, node embeddings are appropriate to encode their structure. Deep Walk and node2vec are two algorithms that stand out in this segment.

DeepWalk (PEROZZI; AL-RFOU; SKIENA, 2014) is an approach for learning node representation using random walk and word2vec. Word2vec is an algorithm for word embeddings. Hence, instead of a sequence of words, it uses a sequence of nodes generated from a random walk. The walk length is a parameter. In short, DeepWalk selects a node and walks randomly among its neighbors. Figure 12 illustrates the DeepWalk pipeline. First, for each node, it extracts a sequence of nodes from the random walks. These sequences are given as input for the skip-gram model. This model learns node embeddings to capture topological relationships from the neighborhood.

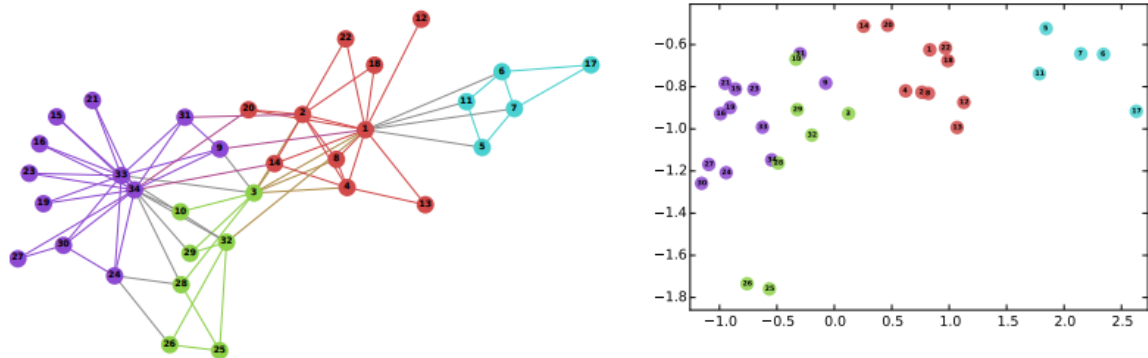


**Source:** <[https://github.com/ashishpatel26/graph\\_nets-1](https://github.com/ashishpatel26/graph_nets-1)>

The authors use Figure 13 to illustrate the intuition behind DeepWalk. The input to the algorithm is a social network connecting 34 members of a karate club (left side). Note that the output is node embeddings. The projection in two dimensions (right-side) shows that nodes connected have their embeddings close in the representation space.

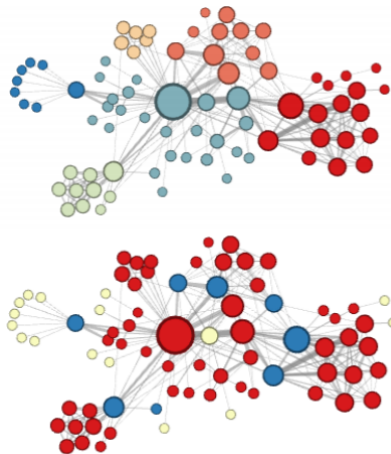
The main limitation of DeepWalk is that the walk is completely random, i.e., without considering any weight. In this sense, node2vec (GROVER; LESKOVEC, 2016) was proposed to address this limitation. The authors propose a biased random walk strategy. The idea is to combine both strategies, i.e., Breadth-First-Sampling (BFS) and Depth-First-Sampling (DFS), to explore the graph structure. The purpose is to control the trade-off between exploring deeper into the graph versus exploring wider across the graph. For this, node2vec depends on two main parameters:  $p$  and  $q$ . These parameters control the probability that a given node will be visited during the walk. Grover and Leskovec (2016) highlight that

Figure 13 – Deep Walk Intuition. Input: Karate Club Graph (left-side). Output: Node Embeddings (right-side).



**Source:** Perozzi, Al-Rfou and Skiena (2014)

Figure 14 – Visualizations of Les Misérables coappearance network generated by node2vec. Label colors reflect patterns in the network: homophily (top) and structural equivalence (bottom).



**Source:** Grover and Leskovec (2016)

the parameter  $p$  controls the likelihood of immediately revisiting a node in the walk, while the parameter  $q$  allows the search to differentiate between “inward” and “outward” nodes.

In short, these parameters help to capture patterns in the graph, e.g., homophily and structural equivalence. Figure 14 shows the Misérables network. Each node represents a character and edges represent cooperation between the characters. The authors use this example to explain the intuition behind these parameters. When  $p = 1$  and  $q = 0.5$  (top), it is possible to identify node clusters based on characters interaction (homophily). When  $p = 1$  and  $q = 2$  (bottom), the grouping takes structural similarity into account. In summary, the parameters are defined according to the goal.

## 2.4 CLUSTERING

The ability to produce and collect data is growing every day. Understanding how data and its relationships help in the knowledge discovery process is a challenge in the data

mining task. In this context, cluster analysis is a way to assist these demands.

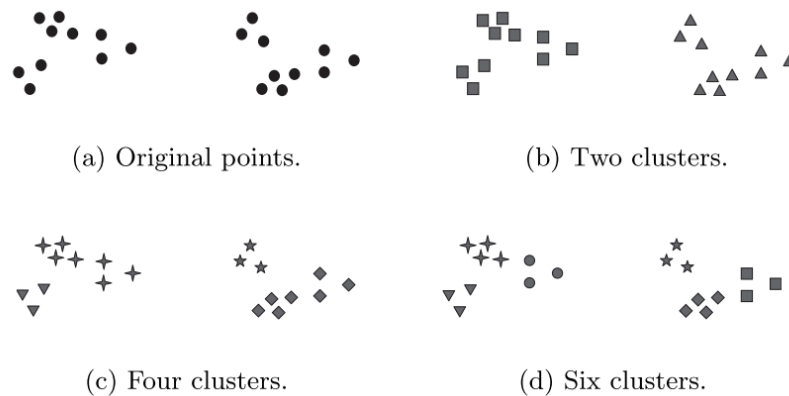
Cluster analysis or clustering aims to group data objects (data points), i.e., records, tuples, or entities, existing in a dataset. According to Steinbach, Kumar and Tan (2005), objects within the same group must be similar to one another (e.g., homogeneous) and different from the objects within other groups (e.g., isolated).

Clustering is an unsupervised task, i.e., it seeks to discover patterns in the data to find similar objects. Unlike the classification task, it doesn't require labels. Some applications use clustering techniques to label data and use it for classification tasks.

Cluster analysis is applied in many fields. For example, in the marketing context, it is used to identify customers with similar purchasing profiles. From this analysis, the marketing team directs advertising actions more suitable for each profile (KASSAMBARA, 2017).

The main challenge in cluster analysis is to define the number of clusters. Steinbach, Kumar and Tan (2005) argue that sometimes this is not well defined. Figure 15 presents an example that justifies this assertion. Figure 15(a) shows the original data, whereas Figure 15(b-d) shows different ways for clustering this data. In short, the number of clusters depends on some aspects, e.g., who is analyzing and the results that they want to obtain. Some methods can help with this task, e.g., elbow and silhouette index.

Figure 15 – Different ways for clustering the same dataset.



**Source:** Steinbach, Kumar and Tan (2005)

The literature appoints some clustering strategies, e.g., *partitional*, *hierarchical*, and *density-based*. In short, the partitional approach divides the dataset into a predefined number of groups. The hierarchical approach creates a tree of hierarchical groups. It uses two techniques: Agglomerative and Divisive. Finally, the density-based strategy uses a proximity measure to identify clusters. According to Sander (2010), this approach is "based on the idea that a cluster in a data space is a contiguous region of high point density, separated from other such clusters by contiguous regions of low point density".

### 2.4.1 Clustering Algorithms

Many clustering algorithms have been proposed over the years (FAHAD et al., 2014). Next, we highlight three best-known algorithms that implement each strategy mentioned above.

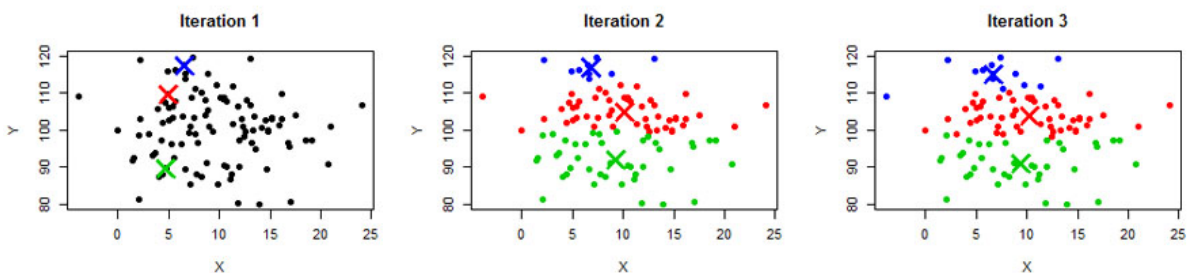
#### K-Means

K-means is an algorithm prototype-based (MACQUEEN, 1967). It divides the dataset into  $k$  partitions. Each partition is a cluster containing similar data objects. First, it randomly defines  $k$  centroids. The centroids are adjusted according to the data distribution. Then, the algorithm iteratively works in two steps until it reaches the convergence point.

The first step assigns a data object to the partition nearest. The most used distance measure is the Euclidean distance. The second step recalculates the centroid of each partition. The centroid is the average value of the partition's data objects.

This process is executed until converges, i.e., when no data object changes partitions. Figure 16 illustrates the behavior of the algorithm in three instants of time. In this example,  $k$  is equal to 3. Note that in the first interaction, the centroids are assigned randomly. In the following iterations, the centroids are recalculated from the changes that occurred in the partition. Over time, improved versions of K-means, which optimizes the initialization of the centroid, have emerged, e.g., K-means++.

Figure 16 – Execution of the K-means algorithm in three iterations.



Source: <<http://www.bigdataexaminer.com/2017/08/07/possibly-the-simplest-way-to-explain-k-means-algorithm/>>

#### Hierarchical Agglomerative Clustering

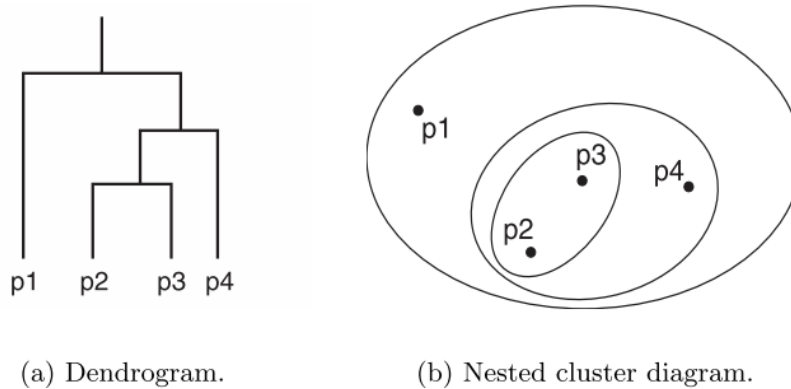
Hierarchical grouping is one of the popular clustering techniques (MONATH et al., 2021). There are two approaches to generating hierarchical clusters: Agglomerative and Divisive. In the agglomerative approach, each data object is a single cluster, which at each interaction merges with the closest cluster until there is a single cluster with all data objects. Oppositely, the divisive approach starts with a single cluster with all data objects and divides data objects until only singleton clusters with individual points remain.

The Hierarchical Agglomerative Clustering (HAC) algorithm works by following these steps:

1. Compute the similarity matrix;
2. Each data object is a cluster;
3. Repeat until only a single cluster remains: Merge the two closest clusters and update the similarity matrix;

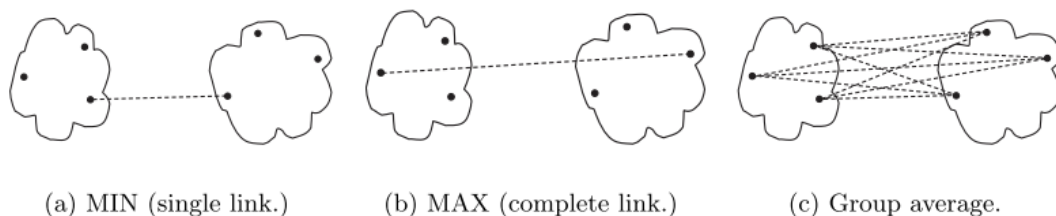
Figure 17 illustrates how the HAC algorithm works. Figure 17(b) shows the grouping process from 4 data points. From the second iteration, points p2 and p3 are merged, then point p4 is added to the cluster of p2 and p3, and finally, point p1 is included. The clustering process forms a Dendrogram, e.g., a tree (Figure 17(a)). A dendrogram allows viewing of the clustering process.

Figure 17 – A hierarchical clustering from four points is illustrated as a dendrogram (a) and nested clusters (b).



**Source:** Steinbach, Kumar and Tan (2005)

Figure 18 – Approaches to calculating cluster proximity.



**Source:** Steinbach, Kumar and Tan (2005)

HAC use several techniques to calculate the proximity between two clusters, e.g., Single Link (MIN), Complete Link (MAX), and Group Average. Zepeda-Mendoza and Resendis-Antonio (2013) summarize these techniques as follows: "*single link* takes into

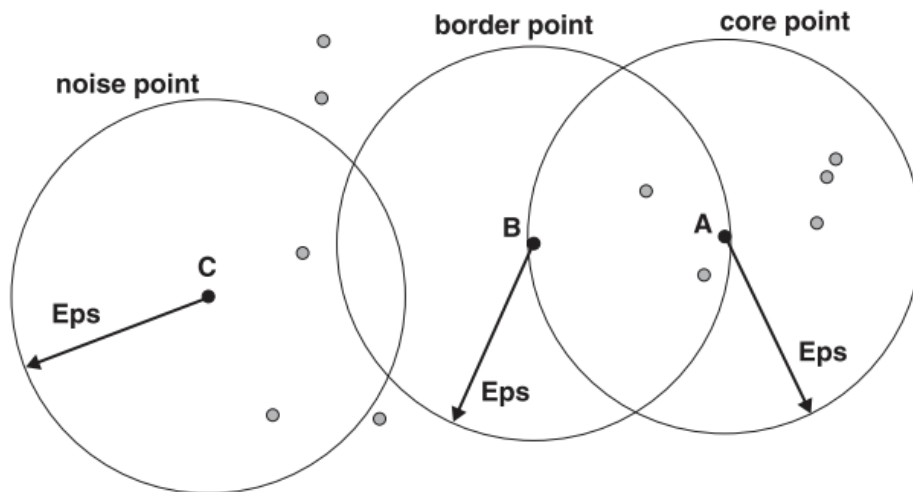
account the shortest distance of the distances between the elements of each cluster; **complete link** takes the longest distance between the elements of each cluster; and **group average** takes the mean of the distances between the elements of each cluster, thus, the merged clusters are the ones with the minimum mean distance". These approaches are illustrated in Figure 18.

## DBSCAN

DBSCAN is an algorithm that seeks to identify regions of high density in data space to form clusters (ESTER et al., 1996). It uses two parameters: epsilon ( $\epsilon$ ) and minimum points ( $MinPts$ ). The parameter  $\epsilon$  defines the radius of the neighborhood around a point  $x$ , whereas the parameter  $MinPts$  is the minimum number of neighbors within  $\epsilon$  radius.

Before presenting how DBSCAN works, we highlight three concepts: core points, border points, and noise points. A point  $x$  is a *core point* if the number of neighboring points is greater than or equal to  $MinPts$  within a radius size of  $\epsilon$ . A point  $x$  is a *border point* if the number of neighbors is less than  $MinPts$ , but it belongs to the neighborhood of a  $z$  point. Finally, a point  $x$  is a *noise point* if it does not fit into any of the previous situations.

Figure 19 – Example of a 12-point cluster using the DBSCAN approach.



Source: Steinbach, Kumar and Tan (2005)

Figure 19 illustrates the three situations. Suppose  $\epsilon$  is any fixed value and  $MinPts$  is equal to 7. The point **A** is considered a *core point* because around it there are 7 points (counting with itself). The point **B** is considered a *border point* because it is positioned at the radius limit and around it there are no 7 or more points to form a new cluster. Finally, the point **C** is considered a *noise point* because it does not meet any of the other requirements.

The DBSCAN algorithm is described as follows:

1. Given a data object (point)  $x_i$ , it computes the distance between  $x_i$  and all points within a neighborhood radius (eps). If the number of neighbors is greater than or equal to  $MinPts$ , then this point is considered a core point, and a new cluster is defined.
2. This process is performed on all data objects in the dataset. If a data object meets the previous requirements it is added to the current cluster, otherwise, it forms a new cluster.
3. The algorithm ends when all data objects are visited. A data object is marked as a noise point if it has not been assigned to a cluster.

### 2.4.2 Cluster Evaluation

Cluster evaluation is an important step in the cluster analysis process. Overall, evaluation measures consider two aspects: whether the clusters are *homogeneous*, e.g., if all data objects within the same group are close, and whether the clusters are *isolated*, e.g., if there is a separation between two groups. This assessment is not always trivial since the cluster analysis imposes some challenges, as discussed previously.

Cluster assessment metrics are classified as **unsupervised** and **supervised**. The unsupervised metrics measure cohesion and separation. In other words, cohesion verifies how related the objects in a cluster are, while separation determines how well separated a cluster is from others. These metrics are called internal indices because they use only the information in the dataset to perform the assessment (STEINBACH; KUMAR; TAN, 2005; LIU et al., 2010). Differently, supervised metrics use external information (i.e., ground truth) to compare the results obtained by a clustering algorithm. For this reason, these measures are called external indices. We briefly show three widely used metrics: silhouette index (unsupervised), V-score, and Adjusted Rand Index (supervised).

- **Silhouette index** (ROUSSEEUW, 1987) measures how well a data object is grouped. Also, it estimates the average distance between clusters. The silhouette index is used to analyze the separation distance between the clusters. For each data object  $i$ , the silhouette index  $s_i$  is calculated as follows<sup>13</sup>:

1. For each data object  $i$ , calculate the average dissimilarity  $a_i$  between  $i$  and all other points of the cluster to which  $i$  belongs.
2. For all other clusters  $C$ , to which  $i$  does not belong, calculate the average dissimilarity  $d(i, C)$  of  $i$  to all observations of  $C$ . The smallest of these  $d(i, C)$  is

<sup>13</sup> <<https://www.datanovia.com/en/lessons/cluster-validation-statistics-must-know-methods/>>

defined as  $b_i = \min_C d(i, C)$ . The value of  $b_i$  can be seen as the dissimilarity between  $i$  and its “neighbor” cluster, i.e., the nearest one to which it does not belong.

3. Finally the silhouette width of the observation  $i$  is defined by the formula:  $S_i = \frac{(b_i - a_i)}{\max(a_i, b_i)}$ .

The index has values between -1 and 1, where values close to 1 indicate that a data object is well grouped, values close to 0 indicate that a data object is between two clusters, and values close to -1 indicate that a data object is on the wrong cluster.

- **V-score** (ROSENBERG; HIRSCHBERG, 2007). Given a ground truth, this metric calculates the harmonic mean considering the *homogeneity* and *completeness* of the clusters. V-score measure is calculated as follows:  $V = \frac{(1+\beta)*h*c}{(\beta*h)+c}$ . Where, (h) is the *homogeneity*, i.e., percentage of data objects within a cluster that belong to a same class in a ground truth, and (c) is the *completeness*, i.e., percentage of data objects that belong to a class within the same cluster. The  $\beta$  parameter is a ratio of weight attributed to homogeneity *versus* completeness. If  $\beta$  is greater than 1, completeness is weighted more strongly in the calculation. If  $\beta$  is less than 1, homogeneity is weighted more strongly (ROSENBERG; HIRSCHBERG, 2007).
- **Adjusted Rand Index** (HUBERT; ARABIE, 1985) According to Santos and Embrechts (2009), “the Adjusted Rand Index (ARI) is frequently used in cluster validation since it is a measure of agreement between two partitions: one given by the clustering process and the other defined by external criteria (ground truth)”. In this sense, given two partitions  $(C, G)$ , where the  $C$  partition is the clusters created by the clustering algorithm, and the  $G$  partition is the ground truth, ARI is calculated as follows:  $ARI = \frac{\binom{n}{2}(a+d) - [(a+b)(a+c) + (c+d)(b+d)]}{\binom{n}{2}^2 - [(a+b)(a+c) + (c+d)(b+d)]}$ .

Where,  $a$  is the number of data objects within the same cluster in  $C$  and in the same cluster in  $G$ .  $b$  is the number of data objects within the same cluster in  $C$  and in different clusters in  $G$ .  $c$  is the number of data objects within the different clusters in  $C$  and in different clusters in  $G$ .  $d$  is the number of data objects within the different clusters in  $C$  and in the same cluster in  $G$ .

## 2.5 CONSIDERATIONS

This chapter introduced some concepts for understanding the contribution proposed in this thesis. Initially, we present an overview of Wikipedia’s Infoboxes and Categories data. These contents are the principal data source for building general-purpose knowledge bases, such as DBpedia and YAGO. Next, we discuss the concept of Knowledge Bases and introduce the RDF and RDFS models. In this context, these semantic models are used to model a KB.

We contextualized Representation Learning, specifically discussing Word Representation (e.g., Word2vec models) and Node Representation (e.g., node embedding algorithms). As discussed in this chapter, node embedding algorithms use approaches based on word embedding to create node representations. In this context, node embeddings are appropriate to encode the graph structure. Finally, we discuss strategies and metrics for the clustering process. Specifically, many schema discovery approaches apply clustering techniques to their solutions.

In the next chapter, we present a compilation of related work to the goals of this thesis. We categorize these papers and perform a comparative analysis between them and this thesis work.

### 3 RELATED WORK

In this chapter, we discuss some papers related to the objectives proposed in this thesis. These papers are organized into two sections: Section 3.1 discusses articles that focus on domain-specific discovery from general-purpose KBs, whereas Section 3.2 discusses some approaches to schema discovery.

#### 3.1 DOMAIN DISCOVERY

General-purpose KBs cover data from several domains. However, many applications are domain-specific. In this context, some papers in the literature propose approaches and methods to extract a domain of interest from KBs. In this context, a domain refers to a set of terms related to a subject of interest. For example, the **movie** domain is formed by *film*, *actor*, *director*, among other terms. Some studies (FONT; ZOUAQ; GAGNON, 2015; TITZE et al., 2014) discuss the effort to include domain information in general-purpose KBs.

Some papers use Wikipedia category data to extract a domain of interest from KBs. In Section 2.1.2, we discuss the concept of categorical data and how they are organized in a KB. To this end, Wikipedia Category Hierarchy (WCH) has been a rich source of information since its function is to organize categories by related subjects (BOLDI; MONTI, 2016). Also, it is a straight method for dealing with DBpedia data. Several studies explore this hierarchy for this purpose. These studies are organized according to their objectives: i) create taxonomies and ontologies for a specific domain (VIVALDI; RODRÍGUEZ, 2010; KOTLERMAN et al., 2011; MIRYLENKA; PASSERINI; SERAFINI, 2015); ii) identify and consume data from a domain of interest (LALITHSENA et al., 2017; RAYNAUD; SUBERCAZE; LAFOREST, 2018); and iii) create semantic annotations with domain information (TITZE et al., 2014). Thus, this section aims to present and discuss these papers.

##### 3.1.1 Domain taxonomies and ontologies

Taxonomies and ontologies are two ways to represent knowledge. Although similar, there is a difference between them. Taxonomies organize a set of terms into a hierarchy, while ontologies define semantic relationships between concepts in a domain of interest. In this context, Vivaldi and Rodríguez (2010), Kotlerman et al. (2011), Mirylenka, Passerini and Serafini (2015) create specific-domain taxonomies and ontologies from Wikipedia.

Vivaldi and Rodríguez (2010) and Kotlerman et al. (2011) argue that organizing data into taxonomies is advantageous for content discovery, search, exploration, and analysis. These works use the WCH as background to automatically create domain taxonomies. In

other words, the taxonomy originates from a WCH sub-tree containing the most relevant terms for a domain of interest.

The proposed method by Kotlerman et al. (2011) follows three steps: i) it selects a subset of categories from the WCH using a set of keywords; ii) it derives a sub-tree, creating a unique path between a category and the root of the sub-tree; and iii) it prunes the hierarchy tree to retain only the most relevant categories and obtain a taxonomy of the desired size. For each category from the derived sub-tree, the method calculates its sub-tree weight and prunes categories whose sub-tree weight is lower than a threshold.

Vivaldi and Rodríguez (2010) propose an approach with some particularities that differentiates it from the previous method. For example, instead of keywords, authors use a high-level category (which represents the domain of interest) to find a subset of related categories. For this, the authors use a depth-first search approach to visit child categories. They also implement a constraint that considers the content of pages related to each category.

Mirylenka, Passerini and Serafini (2015) proposed an automatic method to extract a domain ontology from the WCH. To do this, the authors define a category (which represent the domain of interest) and apply a breadth-first approach to finding related categories. The approach works into three steps: (i) selecting the relevant categories, (ii) splitting them into classes and individuals, and (iii) classifying the relations. These steps are modeled as a classification problem using machine learning techniques. The authors trained a binary classifier (SVM) to predict if a category is relevant based on its features, e.g., depth (the distance from the root), title, and parent categories. From the set of relevant categories, the authors split them into classes and individuals. They consider the grammatical number (singular or plural) from the category title, e.g., singular titles indicate broader categories. Finally, the last step establishes the semantic relationship (related\_to, subclass\_of, instance\_of) between classes and entities in the ontology. For each type of relation, a classifier was trained from a set of features based on the category title. A weakness of this work is the dependence on labeled data to train the classifier at each step. Generating training examples for different domains is not an easy task. Furthermore, the evaluation considered only two domains (Computing and Music).

### 3.1.2 Domain-specific data from knowledge bases

General-purpose KBs, such as DBpedia, contain a significant number of facts expressed as hierarchical relationships and are used by many applications. However, most applications are domain specific and require only a subset of this data. In this context, some approaches (LALITHSENA et al., 2017; RAYNAUD; SUBERCAZE; LAFOREST, 2018; ROCHA; ZUCKER; GIBOIN, 2018) identify a domain, within a KB, to facilitate the access and consumption only of the data required by the applications.

Lalithsena et al. (2017) propose an approach to extract a domain-specific hierarchical

subgraph. To this end, the authors consider a set of domain entities as the input. These domain entities represent the domain of interest. From these entities, other entities are discovered. To expand the entity set, the authors use the WCH and explore the relationships between the categories of the entities used as input. To filter categories related to the domain of interest, the authors detail a method that explores three semantic relationship types between categories: type semantic, lexical-semantic, and structural-semantic. The output produced by the approach is a sub-graph containing only hierarchical relationships between entities in a specific domain.

In a similar way, Raynaud, Subercaze and Laforest (2018) proposed Fouilla, a tool that allows a user to browse, from a general-purpose KB, on triples corresponding to a target domain. Fouilla uses structural resources provided by Wikipedia and DBpedia, such as Categories, Portals, Outlines, and Lists. Its architecture is organized into three modules. The first module (Topic Identification) identifies domains (topics) existing in a dataset. The second module (Topic Delimitation) maps a relationship (triple x domain) using techniques such as Latent Semantic Analysis (LSA), and the last module (Triples Ranking and Filtering) refines the previously mapped triples, calculating an adequacy score for a given domain. Triples and mappings are stored in a repository. The idea is that the user selects a domain from a pre-defined list, and Fouilla presents only the triples to that domain.

Rocha, Zucker and Giboin (2018) propose a heuristic to find entities that belong to a given specific domain. The idea is to represent them in a specialized subgraph. To do this, the authors use a set of representative keywords. These keywords are used to find relevant categories and, from this, discover other related entities. In addition, the selected categories help to enrich the subgraph with links between the identified entities.

### 3.1.3 Semantic annotations

Despite being a rich semantic repository, KBs do not provide domain information, i.e., a kind of semantic annotation describing which concepts are associated with an entity. In this sense, Titze et al. (2014) propose an automatic method to provide thematic labels for DBpedia entities using Wikipedia categories. Overall, the categories are fine-grained, so the idea is to group them into more specific subjects. For example, Barack Obama is associated in some categories, including *Politicians from Chicago, Illinois*; *African American United States Senators*; and *Democratic Party Presidents of the United States*. These categories can be grouped to capture the notion of (U.S.) Politics. The proposed approach groups categories using a kernel-based k-means clustering algorithm. Specifically, the authors consider the four kernel functions: co-occurrence kernel, distributional kernels, string kernel, and category tree kernel. As output, the approach provides a set of categories grouped according to the related semantic term. This label can be associated with entities mapped to categories in order to provide a more detailed description of the

entity’s semantics.

### 3.1.4 Comparative analysis

KBs are useful for many applications. As mentioned earlier, these repositories are massive (in terms of the number of triples) and cross-domain. Also, there are applications interested in consuming data for a specific domain. In this context, approaches focused on this purpose are helpful for applications and users. The main objective of this thesis is to propose a pipeline for specific domain schema discovery from a general-purpose KB. To this end, first, we need to extract a domain of interest. Table 2 highlights some characteristics existing in the papers presented above. We discussed these features and performed a brief comparative analysis.

All works apply resources taken from Wikipedia. Specifically, most works utilize the WCH to leverage their final task. In addition, despite being applied in different contexts, the main purpose of these works is to facilitate access to data in a domain of interest. For example, papers that propose to create a taxonomy and ontology (VIVALDI; RODRÍGUEZ, 2010; KOTLERMAN et al., 2011) are interested in mapping terms and concepts to improve the discovery and exploration of content in a target domain. In our pipeline, the domain discovery step aims to find categories belonging to a given domain of interest, identify related entities and then discover their schema.

A domain can be modeled in different ways. For example, some works consider keywords containing terms relevant to a domain of interest (KOTLERMAN et al., 2011; ROCHA; ZUCKER; GIBOIN, 2018), while others use a set of entities as a seed to expand their search space in a KB (LALITHSENA et al., 2017). On the other hand, the approach applied in this thesis extracts a domain from a high-level category (root). The idea is that this category is used as a seed to find related categories using WCH. The approach used in this thesis is similar to Vivaldi and Rodríguez (2010). However, the pruning strategy used during the expansion of the categories is different. Vivaldi and Rodríguez (2010) sets a score based on some features of the categories to filter out non-relevant categories. Oppositely, we implement a pruning strategy based on attribute similarity from entities mapped to a given category. In the next chapter, specifically in Section 4.3.1, we discuss the pruning strategy used for the Domain Discovery task proposed in our pipeline.

Table 2 – Comparative analysis between works that perform the domain discovery task.

<b>Work</b>	<b>Goal</b>	<b>Domain</b>	<b>Resources</b>	<b>Output</b>
Vivaldi and Rodrigues (2010)	Create a domain taxonomy from Wikipedia	Category (root)	WCH (Categories) and Pages	Set of domain terms extracted from the categories
Kotlerman et al. (2011)	Create a domain taxonomy from Wikipedia	Keywords	WCH (Categories)	Set of domain terms extracted from the categories
Myrilenka; Passerine; Serafini (2015)	Extract a domain ontology from DBpedia	Category (root)	WCH (Categories)	Domain ontology
Lalithsena et al. (2017)	Extract a domain-specific hierarchical subgraph	Set of entities	WCH (Categories)	Set of hierarchical relationships between entities in a specific domain
Raynaud; Subercaze; Laforeste (2018)	Browse triples related to a specific domain	Portals	Categories, Portals, Outline e Lists	Set of triples related to a domain of interest
Rocha; Zucker; Gi-boin (2018)	Extract an entity set from a target domain	Keywords	WCH (Categories)	Set of entities specific to a domain
Titze et al. (2014)	Create thematic labels	-	WCH (Categories)	Categories cluster
<b>This thesis</b>	<b>Find categories related to a domain of interest</b>	<b>Category (root)</b>	<b>WCH (Categories)</b>	<b>Set of domain categories</b>

Source: Created by the author

## 3.2 SCHEMA DISCOVERY

Schema discovery is a research topic studied over the years. This subject began to be explored in the last two decades mainly due to the growth in the adoption of semi-structured data formats such as Extensible Markup Language (XML), JavaScript Object Notation (JSON), and RDF, as described by Gómez et al. (2018).

Recently, Kellou-Menouer et al. (2022) published a survey discussing and classifying schema discovery approaches into three types: i) implicit - these approaches seek to exploit the implicit structure of the dataset without considering explicit statements, e.g., datasets do not have class/type labels; ii) explicit - these approaches use explicit statements in the dataset to complement and enrich the schema; and iii) structural pattern - these approaches provide an overview of the attribute patterns that exist in dataset entities.

The schema discovery problem presented in this thesis is in line with implicit schema discovery approaches. In other words, given a dataset, the goal is to identify groups of similar entities and, for each group, to infer a common schema. Thus, we will discuss some related papers with the objective of this thesis. These papers are organized by data types, namely: structured and semi-structured.

### 3.2.1 Structured data

Most of the data available in general-purpose KBs such as DBpedia and YAGO comes from infobox data (MOREIRA; COSTA-NETO; BARBOSA, 2019). Infobox data is structured but heterogeneous. As mentioned in previous chapter, it is required a template to define an entity infobox. However, currently, these templates lack mechanisms to validate data constraints. For this reason, infoboxes created from the same template can have a set of different attributes.

In this context, Nguyen et al. (2012) proposes WIClust (Wikipedia Infobox Cluster), a clustering algorithm aiming to group a set of Wikipedia infoboxes. The idea is to group infoboxes with similar structures by type (class) and, for each type, define a global schema containing the discriminating attributes of that set of entities (infoboxes). The authors aim to provide this schema for applications such as structured-query systems. Thus, given a set of infoboxes as input, WIClust works in two steps. First, there is an attribute clustering step. Ambiguous attributes are excluded because they can appear in different types and are not representative for this step. The authors use an approach based on attribute correlation. The idea is to cluster pairs of attributes with high correlation, separate uncorrelated attributes pairs, and merge groups of attributes that intersect. In the next step, WIClust calculates the similarity between an infobox instance (entity) and each attribute cluster identified in the previous step. An infobox is assigned to the attribute cluster with the highest similarity.

Similarly, Wu and Weld (2008) recognize the potential of infobox data for multiple

applications and proposes an approach called Kylin Ontology Generator (KOG). Given a set of infoboxes, KOG generates an ontology describing the main classes and attributes contained in the dataset. The idea is that this ontology supports applications that need to access and query this data, e.g., an advanced query system based on infobox data. KOG’s architecture contains three modules: schema cleaner, subsumption detection, and schema mapping. First, each infobox template is seen as an ontology class. However, different templates can possibly represent the same concept. Thus, the schema cleaner module works by identifying and grouping templates with similar structures. Next, the subsumption detection module applies Markov Logic Network models to infer the relationship between the classes using WordNet and other features as a background for this task. Finally, the schema mapping module resolves conflicts at attribute levels, performing the mapping of attributes in the hierarchy of the generated classes.

Yao et al. (2008) and Xu et al. (2010) proposed to learn a global schema for a set of entities of the same class (type). These works seek to resolve heterogeneity conflicts between schemas, i.e., attributes representing the same concept but are described differently, e.g., `birthday` and `born` for the `Person` class. In this direction, Yao et al. (2008) proposed a framework that works in two steps: first, it extracts web entities from a set of pages and, later, learns the global schema of a given type using the collected entities. Specifically, we are interested in this second part of the framework. To do this, the authors use a method based on entropy. The idea is to learn the global schema from a combination of features, such as frequent labels, entities, and related pages. The approach described in Xu et al. (2010) is an evolution of the previous work. The authors consider the dynamic factor that data on the web has. Furthermore, the global schema learning process uses a SVM supervised model. The model training stage uses a set of features combining attribute name (label), data type, and contextual aspects such as size and position in which the attribute appears in the schema.

### 3.2.2 Semi-structured data

Semi-structured datasets has become popular over the years due to the consolidation of the Web as a platform for publishing and consuming data. Specifically, RDF datasets are widely available. Bouhamoum, Kedad and Lopes (2020) argues that despite this, many datasets do not provide schema-related information since these declarations are not mandatory. This makes it difficult for applications to consume this data. In this direction, some papers address the problem of schema discovery for RDF datasets. Most approaches infer the dataset schema by grouping similar entities.

Kellou-Menouer and Kedad (2015) propose an approach to schema discovery in RDF datasets. The authors consider the dataset schema as a set of types (e.g., classes) and link definitions. They argue that information about the data schema facilitates the use of the RDF dataset. The proposed approach relies on a density-based clustering algorithm.

The idea is that similar entities are assigned to the same cluster. To do this, the authors use Jaccard similarity. The similarity between a pair of entities is calculated considering their respective sets of attributes (properties). Each cluster has a type profile. A type profile is a description of a class (Figure 20). It is composed of the union of all attributes of the entities that are part of that class, as well as a probability value that indicates the frequency with which an attribute appears in the entities. The type profile is used to find overlapping between classes (generalization/specialization), e.g., person/actor or person/singer.

Figure 20 – Type profile (example)

- Politician:  $\langle (\vec{n\grave{a}me}, 1), (\vec{party}, 0.73), (\vec{children}, 0.21), (\vec{birthDate}, 0.94), (\vec{nationality}, 0.15), (\vec{succ\grave{e}ssor}, 0.78), (\vec{deathDate}, 0.68), \dots \rangle$ .
- SoccerPlayer:  $\langle (\vec{n\grave{a}me}, 1), (\vec{height}, 0.46), (\vec{surname}, 0.93), (\vec{birthDate}, 1), (\vec{nationalteam}, 0.86), (\vec{currentMember}, 0.8), (\vec{deathDate}, 0.06), \dots \rangle$ .

**Source:** Kellou-Menouer and Kedad (2015)

In the experimental evaluation, the authors considered some synthetic and real RDF datasets. For instance, a DBpedia dataset with entities from the classes Politician, SoccerPlayer, Museum, Movie, Book, and Country. They measured the approach performance considering a gold standard (using precision and recall metrics). DBpedia dataset considered classes very distinct. In this context, clustering entities looking only at attributes is trivial. In scenarios in which entities within different classes have similar attributes, e.g., Actor and Singer, this approach tends to underperform.

Similar to the previous paper, Christodoulou, Paton and Fernandes (2015) proposes to use a hierarchical clustering algorithm to infer the schema by looking at entity-level data from an RDF dataset. An entity is represented by a vector containing the attributes used to describe it. The hierarchical tree created from the clustering allows for analyzing the similarity between entities and defining a similarity threshold to find a suitable number of clusters. Furthermore, this approach includes a cluster annotation step to assign a label to the identified classes. The authors assume that some entities have labels identifying their type, e.g., `rdf:type` predicate for DBpedia entities. From this, a naive strategy is used that consists of identifying the most frequent type among the labeled entities and generalizing to the cluster.

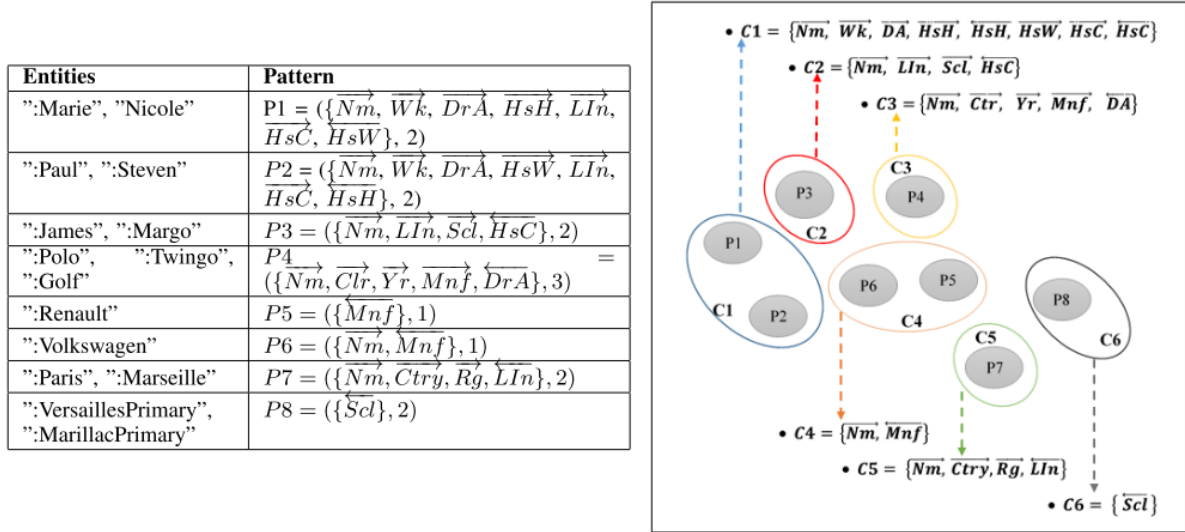
Mehri and Valtchev (2017) proposed a method that uses Formal Concept Analysis (FCA) to explore an RDF dataset and infer its schema. This approach identifies existing classes and relationships in the dataset. In short, this method creates a hierarchical structure based on concepts (classes) discovered from a relationship matrix composed of the attributes of the entities.

Bouhamoum et al. (2018) propose an approach for schema discovery considering the scalability issue. This paper is an evolution of the previous work (KELLOU-MENOUER;

KEDAD, 2015). In the previous paper, the authors use a density-based clustering algorithm (DBSCAN). However, they justify that this algorithm is expensive to process large datasets. In this context, Bouhamoum et al. (2018) proposes a new strategy for representing entities to solve the problem of processing costs. Instead of using a vector of attributes for each entity, they use a condensed representation of the dataset obtained from the attribute patterns found in the entities. To do this, the authors split their approach into two steps: (i) pattern extraction and (ii) pattern clustering.

A pattern is a set of distinct attributes that occur in at least one entity. Patterns are extracted from triples present in the dataset and organized in lists. The pattern extraction process is parallel using a Spark-based implementation. After creating this condensed representation, the next step is to cluster the extracted patterns to discover the existing classes in the dataset. Figure 21 exemplifies an overview of the approach. On the left are attribute patterns extracted from an RDF dataset. On the right, the patterns are grouped into classes. Note that an attribute pattern can represent more than one entity. This reduces the items that will be grouped by the algorithm.

Figure 21 – Pattern extraction (left-side) and clustering (right-side) from a RDF dataset.



Source: Bouhamoum et al. (2018)

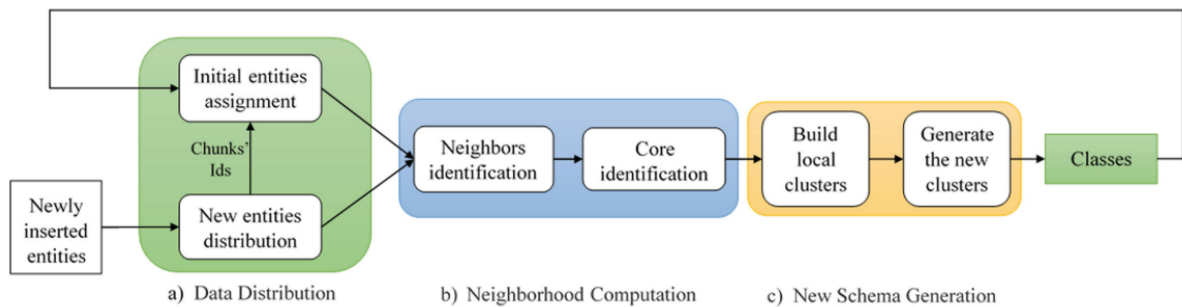
The authors performed experiments on four large RDF datasets, e.g., DBpedia and DBLP. Pattern extraction made it possible to reduce the time required by DBSCAN to group the dataset entities.

Despite encouraging results, the authors improved the performance of the approach. In a heterogeneous dataset, the number of attribute patterns was large. The idea was to propose a parallel clustering algorithm to solve this issue. Toward this goal, Bouhamoum, Kedad and Lopes (2020) proposed SC-DBSCAN, an algorithm that groups structurally similar entities into classes. This algorithm is an adaptation of DBSCAN. However, the main strategy used by the authors is to use parallelization technologies, e.g., Spark, to divide the dataset into subsets and perform the clustering process on the created par-

titions. Overall, SC-DBSCAN identifies similar local clusters and outputs final clusters resulting from a merge process. The experiments showed that SC-DBSCAN obtained an equal quality and superior performance to a sequential clustering algorithm, i.e., DBSCAN. In summary, SC-DBSCAN can group a large number of attribute patterns faster without losing quality in the results.

Bouhamoum, Kedad and Lopes (2021) propose an improvement in the SC-DBSCAN algorithm. This change considers that massive datasets are dynamic. Over time, it is common for datasets to change with the insertion of new entities. For example, DBpedia is a periodically updated dataset. A new version of this dataset adds hundreds or thousands of new entities. The authors argue that new entities influence the schema generated, so it is necessary to keep the schema updated. In this context, the main contribution is an adaptation of the previously proposed pipeline (BOUHAMOUM; KEDAD; LOPES, 2020). When a set of new entities is given to the algorithm, these entities are assigned in the most similar partitions. These partitions were created in previous iterations. After this, the neighborhood is checked and the process is adjusted so that the final result is consistent, eliminating the need to process the complete dataset (Figure 22).

Figure 22 – Pipeline for incremental schema discovery.



**Source:** Bouhamoum, Kedad and Lopes (2021)

Bouhamoum, Kedad and Lopes (2022) propose an extension of Bouhamoum, Kedad and Lopes (2021). The main difference between them is the scenario of entity set exclusion. In the pipeline, the process of deleting an entity set is similar to adding a set. The authors justify that dealing with exclusion is as important as inclusion, as both can cause inconsistency in the schema that describe the dataset.

Meimaris (2019) and Zouaq and Martel (2020) propose using entity embedding for the schema discovery task. The idea is that the learned entity vector representations are good features and contribute to the schema discovery process, specifically in the entity clustering step.

Meimaris (2019), in their vision paper, propose two directions to visualize an implicit schema from entities of a RDF dataset. The first proposal applies a Characteristic Set (CS), i.e., a set of distinct attributes that describe an entity. The idea is to use a pre-trained word embedding model (e.g., Google News Corpus) to map each attribute contained in

a CS to a word vector and subsequently aggregate the vectors of the assigned words to derive a single vector (e.g., average or sum). The goal is to use the aggregated vector to represent an entity. The authors justify that word embeddings are useful to capture semantics and the relationship between words. Consequently, mapping an attribute to a word vector could contribute towards discovering relationships between attributes and identifying entities that have a similar structure, i.e., described by similar CSs. The second direction proposed by the authors is to use graph embedding techniques. RDF datasets can be represented in a graph. The idea is to use these techniques to create a representation for each node of an RDF graph, similar to the approach we are proposing in this thesis (see Section 4.3.2).

Zouaq and Martel (2020) propose an approach to extract a taxonomy for a knowledge graph. This taxonomy is represented hierarchically by a set of classes, summarizing the content of a knowledge graph. The authors investigated the possibility of using knowledge graph embeddings to derive this taxonomy. In this direction, several models were evaluated, such as TransE (BORDES et al., 2013) and RDF2vec (RISTOSKI; PAULHEIM, 2016). In a knowledge graph, each node is an entity represented by a vector of features learned by each model. The idea is to use these representations to cluster entities into classes. To do this, the authors propose to use a hierarchical clustering algorithm and also some data-based statistics to generate the taxonomy.

Issa et al. (2019) proposed LOD-COM, a tool based on completeness to reveal the conceptual schema of RDF datasets. The authors argue that the lack of metadata describing dataset conceptual information makes exploration and usability of RDF data difficult. Based on this premise, the authors use an item mining-based approach to find frequent attribute patterns from a set of entities within the same class. The implementation of this approach considers the FP-growth algorithm. Thus, a parameter (support vector) is required to find frequent attribute patterns. As output, the tool returns a class diagram containing the classes and their relationships, the attributes, and an associated completeness value (i.e., percentage of entities that have that attribute). The authors performed a case study in 4 DBpedia classes (Film, Settlement, Organization, and Scientist) to illustrate how the tool works. They varied the parameter values between 0.9 and 0.1, showing that lower thresholds produce more complex schemas with the highest number of selected attributes.

Weise, Lohmann and Haag (2016) proposed LD-VOWL, a tool for extracting and visualizing schema information for Linked Data. The authors use the class-centring perspective to extract schema information for a data source. In other words, SPARQL queries are submitted over the entities of a class to reveal their schema. Specifically, a query identifies the  $k$  most frequent attributes, and the class schema is defined from this result.

Wang et al. (2015) propose a framework for managing JSON records. The framework supports three tasks: Schema Extraction & Discovery, Schema Repository, and Schema

Consuming. Specifically, we are interested in the Schema Consuming task. This task involves two sub-tasks: Query and Schema Presentation. In this sense, the authors proposed the Skeleton to summarize record schemas. Skeleton is an approach that seeks, from a set of schemas of the same type (class), to build a common schema with the most relevant properties (attributes). The authors present conceptualization, formalization of the Skeleton, and detail the algorithm for building this global schema. The authors use a gain and cost function that measures the quality of a global schema generated by the approach. The core of Skeleton is maximizing the attributes by appearing in identical schemas. In the experiment that evaluates the effectiveness of Skeleton, the authors used datasets extracted from some sources, such as DBpedia and Freebase, with entities of three different classes (types): Drug, Movie, and Company.

### 3.2.3 Comparative analysis

Over the years, the Web has become a consistent platform for publishing and consuming data. Organizations such as the W3C have emphasized the importance of consolidating good practices related to publishing and using data on the Web. In addition to providing the dataset, it is important that these data can be easily interpreted by those who will consume them. One of the best practices defined by the W3C is to provide schema-related information<sup>1</sup>. In general, a dataset schema provides a content overview and leverages its use by both applications and users. However, some authors (ISSA et al., 2019; BOUHAMOUM; KEDAD; LOPES, 2020; KELLOU-MENOUER et al., 2022) highlight that it is common the lack of this kind of information or even the incomplete availability.

The problem of schema discovery can be divided into two tasks that we call *class identification* and *class schema discovery*. The first task seeks to cluster similar entities into classes, while the second task seeks to find a common (or global) schema for each identified class. Most of the papers we analyzed cover these two tasks (NGUYEN et al., 2012; KELLOU-MENOUER; KEDAD, 2015; BOUHAMOUM; KEDAD; LOPES, 2020; BOUHAMOUM; KEDAD; LOPES, 2021), while some only cover the second task (WU; WELD, 2008; ISSA et al., 2019; WEISE; LOHMANN; HAAG, 2016; WANG et al., 2015).

We present a set of papers that have explored the problem of schema discovery over the past 15 years. In this sense, this subsection aims to compare these papers against this thesis work. Table 3 presents some characteristics that we will highlight below.

We have observed that in the last decade, most papers have focused on semi-structured datasets such as web tables, RDF, and JSON. Semi-structured data has a flexible structure, i.e., they do not present restrictions to validate its organization. This characteristic justifies the adoption of these formats since this flexibility favors data publication. On the other hand, the lack of a well-defined structure makes it difficult to understand and, consequently, the consumption of this data.

<sup>1</sup> <<https://www.w3.org/TR/dwbp/>>

Regarding the goals, all works highlight the need to define a schema to facilitate the understanding and consumption of data by applications and users. Among the main motivations are applications for data integration, query formulation, and data manipulation. In this thesis work, we seek to provide a schema for domain-specific applications that need to use data from a general-purpose KB. We believe that applications can benefit from a specific schema to consume and explore the data needed for their tasks in a more straightforward manner.

We focus on presenting and discussing papers that perform implicit schema discovery. In other words, this task is done by looking at entities of a dataset. To do this, some works use unsupervised learning techniques, such as clustering, to group similar entities. The idea of clustering is to find entity classes present in a dataset. Works such as Nguyen et al. (2012) and Bouhamoum, Kedad and Lopes (2020) contribute on proposing specific algorithms to perform this clustering (WIClust and SD-DBSCAN, respectively). Similarly, in this thesis, we are proposing a pipeline that includes a clustering step (see Section 4.3). However, in this step, our contribution is to apply node embedding techniques to learn a representation for each entity within a dataset. The idea is that this representation is useful to capture the similarity between entities. Our goal is to use these learned representations as features (input) for classic clustering algorithms, e.g., DBSCAN, HAC, K-means++, among others.

Some papers use classical clustering algorithms (DBSCAN and HAC) to cluster similar entities (KELLOU-MENOUER; KEDAD, 2015; CHRISTODOULOU; PATON; FERNANDES, 2015; BOUHAMOUM et al., 2018). These papers use the attributes that describe the entities as features that are given as input to these algorithms. The authors present interesting results in the performed experiments. However, in some scenarios, these approaches tend to have lower performance, as reported in Bouhamoum, Kedad and Lopes (2020). For example, in some cases, the entities within different classes are described by similar attribute sets. Thus, the algorithm has difficulty correctly grouping entities since attributes are the main features considered.

Recently, Zouaq and Martel (2020) showed that it is possible to use knowledge graph embedding models as a feature extractor. The idea is that the entities and relations present in a knowledge graph are mapped to low-dimensional vector representations. The assumption is that close embeddings in a vector space imply a semantic similarity. Thus, entity embeddings are used as features in a hierarchical clustering algorithm to group similar entities in class. In this paper, the models are trained from relations between entities present in the knowledge graph. Differently, in this thesis, we are proposing to create entity embeddings from category mappings provided by the KB. Meimaris (2019) discusses other ways to create entity embeddings, such as using pre-trained word models. According to the author’s proposal, the main difficulty in using pre-trained word models is the mapping between a word and an attribute. This is due to the usual differences in

spelling between an attribute and a word, which has a semantic relationship.

In addition to clustering the entities, most works address the need to generate a description for each identified class (cluster). In short, we call this description of class schema, i.e., the set of attributes that describe the previously grouped entities. Kellou-Menouer and Kedad (2015) and Bouhamoum et al. (2018) define this concept as type profile. In these papers, a type profile is the union of all attributes that describe the entities of that class. This strategy to define the class schema is naive since, in most cases, the entities within the same class are described by a set of distinct attributes (BOUHAMOUM; KEDAD; LOPES, 2022). Furthermore, the union of all attributes can generate a description with a high number of attributes that are not equally relevant. In Section 4.3.3 we discuss this problem in more detail.

A set of works deal with this problem in isolation (WEISE; LOHMANN; HAAG, 2016; ISSA et al., 2019; WANG et al., 2015), seeking to define approaches to find a summarized schema for a set of entities of the same class. Issa et al. (2019) define an approach using a mining-based algorithm to find frequent attribute patterns from a set of entities of a class. Weise, Lohmann and Haag (2016) uses a simple strategy based purely on attribute frequency. Both approaches have a weakness, which is the definition of a parameter. Defining this parameter is not a trivial task, especially when there is no prior knowledge of the dataset. Also, the frequency distribution changes for each dataset.

Wang et al. (2015) presents a parameter-free approach based on weighted frequency. The proposed approach gives priority to attribute appearing in equivalent schemas. The authors assume that the frequency of a schema represents its importance and significance in the dataset. This is done using a gain and cost function that measures the quality of a set of attributes against a set of schemas. Similarly, in this thesis, we propose a pipeline that has a step to build the class schema. Our goal is to generate a summarized schema containing the most significant attributes to describe the previously clustered entities. Our class schema discovery approach is parameter-free, unlike (ISSA et al., 2019) and (WEISE; LOHMANN; HAAG, 2016), and is based on the frequency and co-occurrence of attributes, unlike (WANG et al., 2015). We want to capture which are the most relevant and discriminatory attributes to represent a set of similar entities without user intervention.

In another direction, related to the data integration task, Yao et al. (2008) and Xu et al. (2010) seek to define an integrated schema for a set of entities that belong to the same concept (class). To do this, Yao et al. (2008) apply an entropy-based approach, while Xu et al. (2010) use a machine learning-based approach.

The last dimension we want to highlight in this comparative analysis is the output provided by the approaches. There are different ways to represent a schema. Similar to ours, some papers consider that the schema is represented by a set of classes and their respective attributes (NGUYEN et al., 2012; BOUHAMOUM; KEDAD; LOPES, 2020; BOUHAMOUM; KEDAD; LOPES, 2022). In addition, other papers included in the schema links between

classes (KELLOU-MENOUER; KEDAD, 2015; CHRISTODOULOU; PATON; FERNANDES, 2015; MEHRI; VALTCHEV, 2017; BOUHAMOUM et al., 2018). Finally, other papers consider the schema as only a set of attributes to describe entities within the same class (ISSA et al., 2019; WANG et al., 2015; YAO et al., 2008; Xu et al., 2010).

Table 3 – Comparative analysis between works that perform the schema discovery task.

Work	Data	Schema Discovery			
		Class Identification		Class Schema Discovery	
		Techniques	Features	Approach	Parameters
Nguyen et al. (2012)	Infobox	Clustering (WIClust)	Attributes	Union	no
Wu and Weld (2008)	Infobox	Classification (SVM), Markov Logic Networks	Attribute-based similarity metrics	x	x
Yao et al. (2008)	Web table	x	x	Entropy	yes
Xu et al. (2010)	Web table	x	x	Classification (SVM)	yes
Kellou-Menouer and Kedad (2015)	RDF	Clustering (DBSCAN)	Attributes	Union	no
Christodoulou, Paton and Fernandes (2015)	RDF	Clustering (Hierarchical)	Attributes	Union	no
Mehri and Valtchev (2017)	RDF	Formal Concept Analysis	Attributes	Union	no
Bouhamoum et al. (2018)	RDF	Clustering (DBSCAN), Spark	Attribute pattern	Union	no
Bouhamoum, Kedad and Lopes (2020)	RDF	Clustering (SC-DBSCAN), Spark	Attribute pattern	Union	no
Bouhamoum, Kedad and Lopes (2021)	RDF	Clustering (SC-DBSCAN), Spark	Attribute pattern	Union	no
Bouhamoum, Kedad and Lopes (2022)	RDF	Clustering (SC-DBSCAN), Spark	Attribute pattern	Union	no
Meimaris (2019)	RDF	Clustering (Hierarchical), Entity embedding (word and graph embedding)	Entity attributes and relations	Union	no
Zouaq and Martel (2020)	RDF	Clustering (Hierarchical), Statistics, Entity embedding (node embedding)	Entity relations	Union	no
Issa et al. (2019)	RDF	x	x	Pattern-frequency	yes
Weise, Lohmann and Haag (2016)	RDF	x	x	Frequency	yes
Wang et al. (2015)	JSON	x	x	Frequency weight	no
<b>This thesis</b>	<b>RDF</b>	<b>Clustering (classic algorithms), Entity embedding (node embedding)</b>	<b>Category mappings</b>	<b>Co-occurrence and Frequency</b>	<b>no</b>
					<b>Classes and attributes</b>

Source: Created by the author

### 3.3 CONSIDERATIONS

In this chapter, we present and discuss some papers in line with the proposal covered by this thesis.

Initially, we discussed some papers related to the domain discovery task. In the context of general-purpose KBs, this task is important since many applications are interested in consuming data for a domain of interest. From the analysis accomplished, we observed that the use of the hierarchy of categories is an alternative widely used in the literature since it allows the exploration of the content of the KB by themes. KBs like DBpedia and YAGO map entities to categories, and by navigating between categories, it is possible to identify which entities are related to a given theme.

Next, we discuss papers that focus on schema discovery. We emphasize approaches that do this implicitly, i.e., considering only entities within a dataset. Most papers use unsupervised learning methods to group entities and identify existing classes in a dataset. Furthermore, we observed that most papers use a naive strategy (attribute union or frequency) to define a global schema for the identified entity classes.

In the next chapter, we will discuss the proposed solution to the problem we are addressing in this thesis work. Our main contribution is an end-to-end pipeline for schema discovery from general-purpose KBs.

## 4 A PIPELINE FOR SCHEMA DISCOVERY FROM GENERAL-PURPOSE KNOWLEDGE BASES

This chapter presents ANCHOR, our end-to-end pipeline, which performs schema discovery for a specific domain from a general-purpose KB. ANCHOR works in three steps. Each step seeks to answer a research question addressed in this thesis work. Initially, we describe a scenario in which a domain-specific application can benefit from our solution (Section 4.1). Next, we introduce some preliminary concepts to understand our contributions (Section 4.2). In the next section (Section 4.3), we detail each step in the pipeline. Specifically, in Section 4.3.1, we describe the strategy used to extract a specific domain from a general-purpose KB. Section 4.3.2 details how ANCHOR works to build entity representations from entity-category mappings. It uses these representations for the Class Identification task. In Section 4.3.3, we describe CoFFee, a solution that has been integrated into our pipeline to solve the Class Schema Discovery task. Finally, in Section 4.4, we address the final considerations of this chapter.

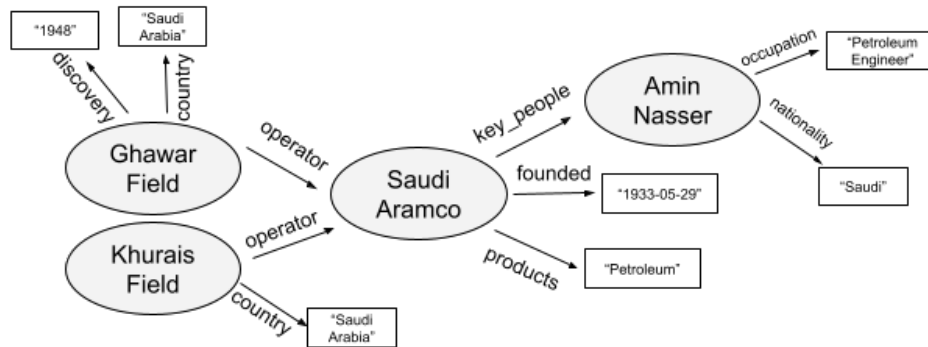
### 4.1 MOTIVATING SCENARIO

Some applications rely on a data schema to perform their tasks. For example, in Adolphs et al. (2011), the authors propose a system of questions and answers using knowledge bases. In this context, the main challenge is to map a natural language query to a structured query. Considering this scenario, suppose we have a domain-specific application related to the *oil and gas* domain. This application performs structured queries on a knowledge base as in Adolphs et al. (2011). For this application, schema-related information is essential in the query formulation process since this information offers a description of the dataset structure (e.g., entity classes and their attributes).

Consider that we have a dataset in this domain extracted from a general-purpose KB, as illustrated in Figure 23. Entities are represented by ellipses, edge labels represent attribute names, and rectangles represent literal values. As can be seen, this dataset does not provide schema-related information. To explore its content, it is necessary to infer this information. When analyzing the dataset, it is possible to identify entities of different kinds. For example, *Ghawar Field* and *Khurais Field* are both oil fields, whereas *Saudi Aramco* and *Amin Nasser* are a company and businessman, respectively. Assume that among the possible queries submitted to this application are:

- Q1 - *"which oil fields are located in Saudi Arabia?"*
- Q2 - *"which companies supply Petroleum?"*

Figure 23 – Snippet of the a KB content in the oil and gas domain.



**Source:** Created by the author

To process  $Q1$  and  $Q2$ , for instance, it is necessary to know that the dataset has *oil field* and *oil company* classes. In addition, entities within the oil field class have the **country** attribute, which indicates a country where an oil field is located, as well as entities within the oil company class, have an attribute called **products**, which indicates which types of products are marketed or supplied by a company.

Note that manually exploring this information can be a tedious and non-trivial task. However, the task of mapping a question to a structured query can be facilitated if there is a data schema intermediating this task. In this context, ANCHOR can be useful for this application since it can describe the structure of the dataset.

## 4.2 PRELIMINARES

In this section, we present some concepts and definitions to help the understanding of the problem we tackle in this thesis.

**(Entity).** An entity  $e$  is a real-world object described by a set of attributes.

**(Knowledge Base).** A knowledge base  $KB$  is a graph that stores factual data about entities in the form of triples  $(s, p, o)$ :  $s$  is the relation subject, which represents an entity  $e$ ;  $p$  is a predicate, corresponding to an attribute; and  $o$  is a value for that attribute which can be an entity or a literal.

**Example.** Figure 23 presents a snippet of a KB, where it is possible to identify four entities (Ghawar Field, Khurais Field, Saudi Aramco, and Amin Nasser) and their respective attributes. In this example, the entity Ghawar Field has the following attributes: discovery, country, and operator.

General-purpose KBs, e.g., DBpedia and YAGO, use Wikipedia categories to build rich taxonomies (HEIST; PAULHEIM, 2019) and organize their entities thematically (LALITH-

SENA et al., 2017).

**(Category).** According to Senoussi (2018), "*content categories are used to group articles dealing with the same subject*". A *subject* can be any semantic mapping between an entity  $e$  (article) and a category  $cat$ .

**Example.** In DBpedia, the entities `Ghawar Field` and `Khurais Field` were stated to belong to the `Oil fields of Saudi Arabia` category using the Dublin Core (DC) subject term.

In general-purpose KBs as DBpedia, there are two types of mappings related to the categories: The *entity-category* and the *category-category* mappings. Entity-category maps an entity  $e$  to a category  $cat$ , while category-category maps the relationship between two categories  $cat_i$  and  $cat_j$  to define a subcategory hierarchy. In this work, we explore these DBpedia mappings. These mappings are modeled as a graph. Here, we call them Entity-Category Graph and Category Graph, respectively. We highlight that other general-purpose KBs, e.g., YAGO, organize their content based on Wikipedia categories and have mappings similar to those described above.

**Definition 1 (Entity-Category Graph).** An *Entity-Category Graph ECG* models the entity's mapping with one or more categories as a bipartite graph. We formally define  $ECG = \{E, CAT, ES\}$ , where  $E$  is a set of entities,  $CAT$  is a set of categories, and  $ES$  is a set of edges defined by the predicate *subject*.

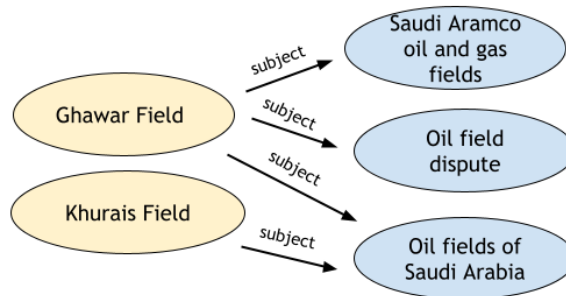
**Example.** Figure 24(a) illustrates an *ECG* created from *entity-category* mappings. As can be seen, the entity `Ghawar Field` belongs to three categories: `Oil field dispute`, `Oil fields of Saudi Arabia`, and `Saudi Aramco oil and gas field`. Hence, these categories describe the themes in which the entity `Ghawar Field` is related.

**Definition 2 (Category Graph).** A *Category Graph CG*  $CG = \{CAT, EB\}$  organizes a set of categories  $CAT$  as a taxonomy using the *IS-A* relationship. This relationship is represented by a set of edges  $EB$  defined by the predicate *broader*. For two categories  $cat_1$  and  $cat_2$ , we have  $cat_1$  *broader*  $cat_2$  if  $cat_1$  *IS-A* subcategory of  $cat_2$ .

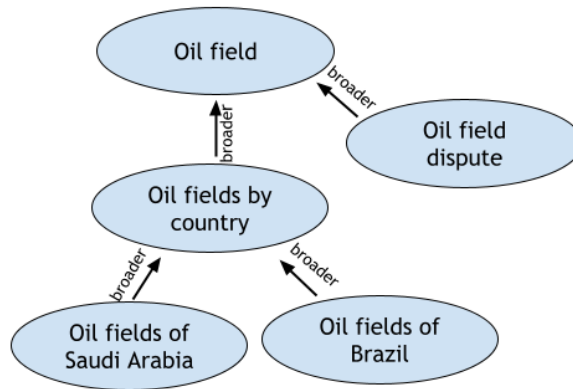
**Example.** Figure 24(b) illustrates an *CG* created from *category-category* mappings. It is possible to see the graph created from a *broader* relationship between categories forming a taxonomy. In this graph, `Oil field` is a root category (broad) that contains `Oil fields by country` and `Oil field dispute` as subcategories, i.e., specialized subjects inside the broader one. Additionally, `Oil fields by country` maintains a subcategory relationship to `Oil fields of Saudi Arabia`.

As introduced in Senoussi (2018), some categories only present mapping to other categories. In this work, we call a *bridge category* a category that has no entities assigned to

Figure 24 – A sample of the graphs created from the entity-category (a) and category-category mappings (b).



(a) Entity-Category Graph



(b) Category Graph

**Source:** Created by the author

it, only other categories. In the KB organization, these categories work as hub interconnecting fine-grained categories. Formally, we define a bridge category as follows:

**Definition 3 (Bridge Category).** A category  $cat$  is classified as a bridge category if  $cat \in CG$  and  $cat \notin ECG$ , i.e., if it has no mappings in  $ECG$ .

**Example.** The Oil field by country category is a bridge category because it has no entities assigned to it. Specifically, this category is a bridge between the Oil fields category and specialized categories by countries, such as Oil fields of Saudi Arabia and Oil fields of Brazil, as illustrated in Figure 24(b).

**(Domain).** Like in Font, Zouaq and Gagnon (2015), domain  $D$  is a set of categories organized under the seed category  $cat$ . In other words, starting from  $cat$ , we explore the category graph looking for a set of subcategories of  $cat$  that represent the domain we want to extract.

**Example.** Figure 26 illustrates a sample of the *Oil and Gas* domain extracted from the Petroleum industry category. Note that it is possible to reach a subset of the Petroleum

industry subcategories related to the core theme of the domain, such as *Oil fields*, *Oil companies*, among others.

**Definition 4 (Class).** A class  $c$  represents a set of semantically similar entities, i.e., entities which are of the same kind in a specific domain.

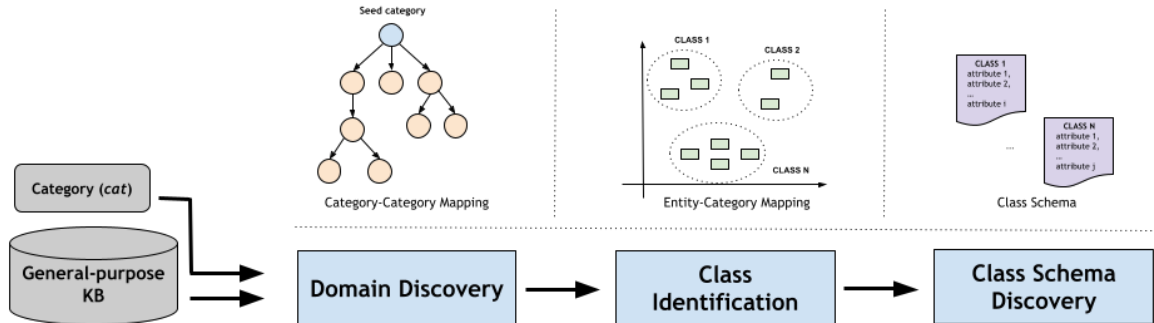
**Example.** It is possible to find entities representing different classes in a domain. For example, considering the *oil and gas* domain, entities *Ghawar Field* and *Khurais Field* represent *oil fields*. In contrast, *Saudi Aramco* represents *oil company* (Figure 23). In this context, *Oil field* and *Oil company* are classes belonging to the domain *oil and gas*.

**Definition 5 (Class schema).** A class schema  $S_c = \{a_1, \dots, a_m\}$  is the set of core attributes that better describe entities belonging to  $c$ .

Based on those concepts, we define our research problem as follows: *Given a category seed (root) cat and a general-purpose knowledge base KB, our goal is to extract a domain D and find a set of classes  $C = \{c_1, \dots, c_i\} \in D$  and their respective class schema.*

### 4.3 SCHEMA DISCOVERY PIPELINE

Figure 25 – ANCHOR’s pipeline.



Source: Created by the author

Figure 25 shows our solution (ANCHOR) for schema discovery. Given a general-purpose KB and a (seed) category  $cat$ , ANCHOR executes three steps to identify the schema of  $D$ : Domain Discovery, Class Identification and Class Schema Discovery. First, the Domain Discovery step extracts the domain  $D$  of  $cat$  by exploiting the hierarchical mappings of the categories of the  $KB$ . Next, the Class Identification discovers the implicit classes of  $D$  by applying a clustering algorithm on the learned representations of the entities of  $D$ . Finally, the Class Schema Discovery step builds the class schema for each class by applying a strategy that relies on attribute co-occurrence and frequency. In the remainder of this section, we describe each component of our proposed solution in further detail.

### 4.3.1 Domain Discovery

Unlike some strategies that propose a schema discovery approach to well-defined datasets, in this thesis, we are interested in exploring general-purpose KBs. Therefore, the first step is to identify the application’s domain of interest to find a dataset with entities related to that domain and proceed with the other tasks of the schema discovery process.

As discussed in the previous chapter, a common strategy to identify domains within KBs is to rely on their category information. Previous approaches have done this to build domain-specific taxonomies (KOTLERMAN et al., 2011; VIVALDI; RODRÍGUEZ, 2010) and extract a domain-specific hierarchical subgraph (LALITHSENA et al., 2017). Similarly, we use the category graph to identify domains within KBs.

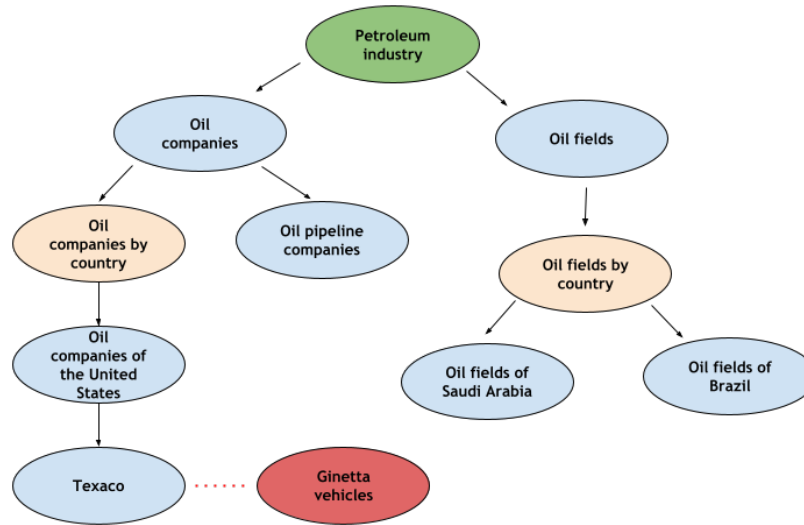
Our solution applies a depth-first search (DFS) strategy given a starting seed category *cat*. With DFS, we navigate on the KB category graph aiming to create a domain graph *D*. Like Moreira, Costa-Neto and Barbosa (2021), the DFS strategy considers the non-bridge categories *cn*, and it is based on the Jaccard Similarity (JACCARD, 1901). We compute the similarity measure by considering the global schema, i.e., the union of all entities’ attributes, of the non-bridge category *cn<sub>i</sub>* against the global schema of each child node *cn<sub>j</sub>* in its subcategories tree. We assign each subcategory *cn<sub>j</sub>* to *D* if the similarity between it and *cn<sub>i</sub>* is higher than a certain threshold. The Jaccard similarity computation is present in Equation 4.1. In short, it considers the attribute sets intersection size divided by the size of their union.

$$sim(cn_i, cn_j) = \frac{|cn_i \cap cn_j|}{|cn_i \cup cn_j|} \quad (4.1)$$

Following the study conducted by Moreira, Costa-Neto and Barbosa (2021), we recommend avoiding setting a high threshold, e.g., above 0.5, as this may restrict navigation between subcategories and fail to consider related categories. Mainly because there could be entities in the category *cn* sharing only a few attributes with a subcategory. Hence, a high threshold can miss these relations. We apply this pruning strategy to avoid adding category nodes in *D* whose entities’ attributes differ too much from the domain defined by *cat*. We add bridge categories to *D* without pruning.

Figure 26 illustrates a sample of the domain graph with the **Petroleum industry** category as the seed category (green node). Applying the DFS strategy from this category, it is possible to reach the **Texaco** category, which has **Ginetta vehicles** as its subcategory. **Texaco** category contains entities such as **Getty Oil** and **Chevron Corporation**. The attributes of these category’s entities mainly describe companies’ properties and location information (e.g., *num\_employees*, *founded*, and *location\_city*). The **Ginetta vehicles** category is associated with entities describing cars’ properties, such as *capacity*, *engine\_name*, and *car\_name*, e.g., **Ginetta G60** and **Ginetta F400**. Because of that, the Jaccard similarity between the attributes of the entities in **Ginetta vehicles** and **Texaco** categories is

Figure 26 – Domain graph sample (Oil and gas).



Source: Created by the author

very low. As a result, the **Ginetta vehicles** category is not added to the domain graph (red node). Adding the **Ginetta vehicles** category to  $D$  would lead to categories unrelated to the **Petroleum industry**.

Categories classified as bridges (orange nodes), e.g., **Oil companies by country** and **Oil fields by country**, are added to the  $D$  without pruning, as they work as a hub between more specific categories and do not contain entities.

In this step, the choice of the seed category is an important issue. This seed must be representative of the domain of interest. Nevertheless, broader categories may be better suited for this purpose, as the category hierarchy follows a fine-grained organization. In this context, other categories can be reached from it, and the domain can be further refined.

#### 4.3.2 Class Identification

From the domain graph  $D$  (built in the previous step), we find a set of domain entities  $E$  through entity-category mappings provided by KB. Therefore, the next step of our pipeline discovers implicit classes in  $D$  (see Definition 4) by clustering similar entities.

Previous approaches have clustered entities based on the similarity of their attributes to perform this task (NGUYEN et al., 2012; CHRISTODOULOU; PATON; FERNANDES, 2015; MEHRI; VALTCHEV, 2017; BOUHAMOUM; KEDAD; LOPES, 2020). However, this strategy underperforms in some scenarios. For example, when entities within different classes have similar attribute sets (BOUHAMOUM; KEDAD; LOPES, 2020) or entities in the same class have a very heterogeneous set of attributes. In this context, attribute-based similarity can wrongly assign an entity to a class, as we show in Section 5.2.2.

Thus, instead of relying on the entities' attributes, we propose learning representations of the entities in  $D$  from their shared categories and then clustering them in classes based

on those representations, applying an off-the-shelf clustering algorithm. We assume that entity-category mappings reflect the organization of entities in a KB and therefore help to capture the relationship between similar entities. As a result, entities that share categories would be closer in a new representation space created based on entity-category mappings.

We split the class identification step into three tasks: (1) building a bipartite graph  $B$  from the *entity-category* mappings in  $D$ ; (2) applying a node embedding algorithm to learn entity representations; and (3) applying a clustering algorithm on the learned representations for class identification.

Regarding the first task, Algorithm 1 presents how we build the bipartite graph  $B$ . At first, ANCHOR obtains the set of categories  $(CATe_i) \in D$  of each entity  $e_i$  in  $E$  (line 4). Next, for each category  $cat_k$  in  $CATe_i$ , the algorithm finds the shortest path between  $cat_k$  and the seed category  $cat$  (line 5). If the length of this path is higher than 2, it adds to  $B$  edges between  $e_i$  and the categories in this path below the first hierarchy level from  $cat$  (line 8). If the shortest path is lower than 2, it only adds the edge between  $e_i$  and  $cat_k$  to  $B$  (line 10). The idea behind expanding  $CATe_i$  is that bridge categories may connect the fine-grained categories. Hence, this extension allows the bipartite graph to become a denser structure enabling the capture of bridge relationships between entities.

---

**Algoritmo 1:** Bipartite Graph Creation

---

**Input:** Domain graph ( $D$ ), Seed category ( $cat$ )

**Output:** Bipartite graph ( $B$ )

```

1  $B \leftarrow \text{new Graph}()$ 
2  $E \leftarrow \text{new Graph}()$  // Captures entities mapped to categories that belong
   to  $D$ .
3 for  $e_i \in E$  do
4    $CATe_i \leftarrow E.\text{categories}(e_i)$  // Captures the categories  $\in D$  mapped to  $e_i$ .
5   for  $cat_k \in CATe_i$  do
6      $path \leftarrow \text{shortest\_path}(D, cat, cat_k)$ 
7     if  $\text{length}(path) > 2$  then
8        $B.\text{add\_nodes\_from}(e_i, path[2:])$ 
9     else
10       $B.\text{add\_nodes\_from}(e_i, cat_k)$ 

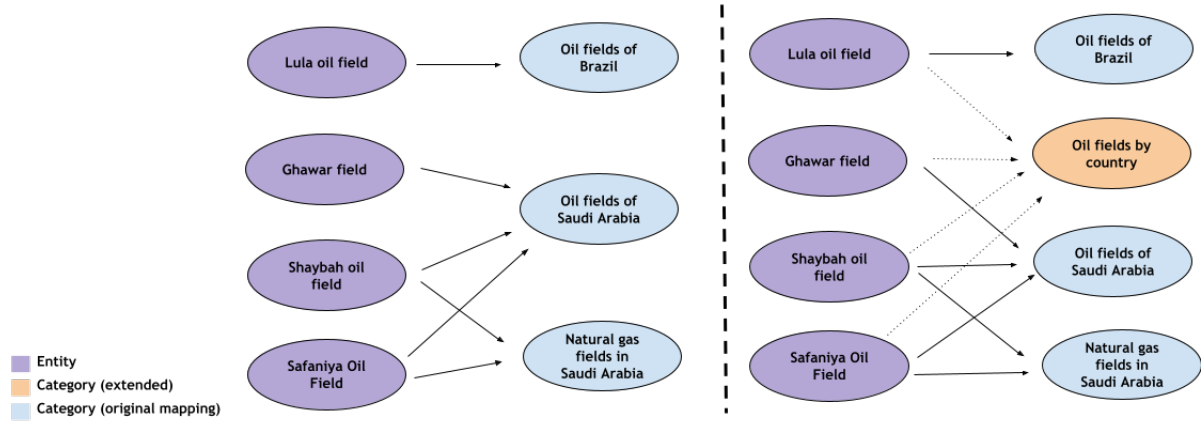
```

---

This extension allows to add more context to the entity mappings. For example, it is common for entities to be mapped to fine-grained categories. In this sense, there may be categories mapped to few entities. By extending the entity mappings, these entities are approximated. Commonly, the categories at the highest level in  $D$  are generalist. As our idea is to seek a trade-off between generalization and specialization, in our implementation, we chose to disregard the categories localized in the first two levels of  $D$ . This choice was made empirically based on the evaluated scenarios.

Figure 27 shows a sample of the bipartite graph for the *oil and gas* domain. The

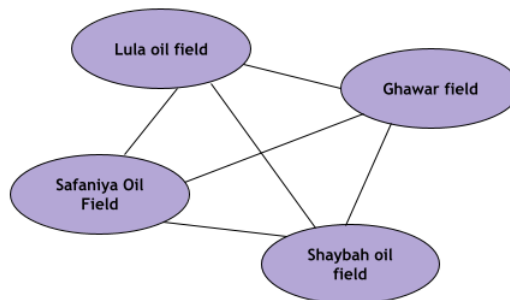
Figure 27 – Bipartite graph (sample) built from the entity-category mappings of a KB (oil and gas domain). The left-side shows the bipartite graph from the original mapping, whereas the right-side shows the bipartite graph created from the mapping extension. Dashed edges illustrate new relations added from the extension.



Source: Created by the author

entities `Lula oil field` and `Ghawar field` share the same category (`Oil fields by country`) (right-side). Initially, these two entities do not directly share any category (left side). However, the categories `Oil fields of Brazil` and `Oil fields of Saudi Arabia` are connected by the bridge category `Oil fields by country`. We leverage, therefore, bridge categories to extend the bipartite graph, making it less sparse (right-side). Building a dense graph in which related entities are closer allows the learning representation algorithm to produce better representations of entities.

Figure 28 – Entity graph created from the bipartite graph in Fig 27.



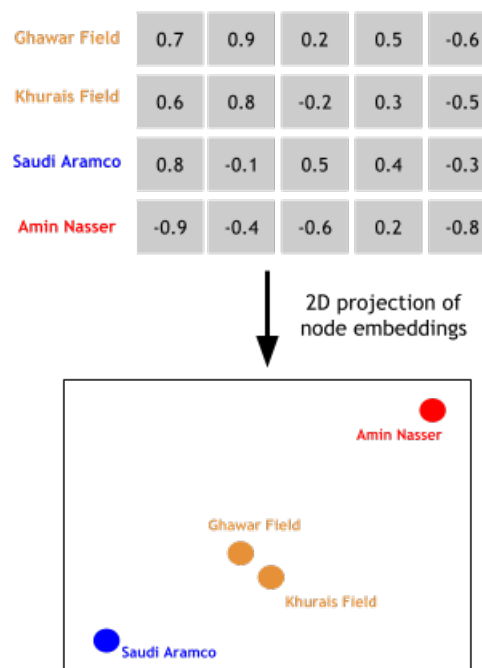
Source: Created by the author

Next, from the bipartite graph  $B$ , ANCHOR builds an entity graph  $EG$  since our goal is to capture the relationships between entities from category mappings. Therefore, entities that share at least one category are connected nodes in  $EG$ . Figure 28 shows the  $EG$  built from the bipartite graph in Figure 27 (right-side).

The core of our class identification solution is to apply embedding techniques to learn a vector representation for the entities. Then, ANCHOR runs a node embedding algorithm on  $EG$  that maps each node (entity) in the graph to a vector representation (embedding) in a low dimensional space based on its category mappings, as illustrated in Figure 29. Node

embeddings encode the structure of nodes from their neighborhood (ZHOU et al., 2022) in such way that nodes with similar neighborhood have similar representations. In our implementation, we use the node-embedding algorithm *node2vec* (GROVER; LESKOVEC, 2016) since it demonstrated to produce high-quality embeddings (DALMIA; J; GUPTA, 2018). Furthermore, it uses a walk-flexible strategy to capture the network neighborhood of a given node. In other words, neighborhood help to capture the semantic context existing between a node and its neighbors. We specified how the node2vec algorithm works in Section 2.3.2.

Figure 29 – Illustration of the vector representation generated by node embedding algorithms. Each node (entity) is represented by a 5-dimensional vector. At the bottom, we illustrate a 2-dimensional view. Nearby entities in d-dimensional space, e.g., Ghawar Field and Khurais Field belong to the same class.



**Source:** Adapted from: <<https://bityli.com/XSR6G>>

The final step of Class Identification applies a clustering algorithm on the learned representation of the entities in  $D$  to group them into distinct classes. There are several available clustering algorithms that we could use for this task, such as K-means (MACQUEEN, 1967), DBSCAN (ESTER et al., 1996), and Agglomerative clustering (KAUFMAN; ROUSSEEUW, 1989). We do not adopt a particular clustering algorithm for this purpose since our goal is to produce good clusters with the entities' learned vectors regardless of the chosen algorithm. In the experiments carried out, we evaluate distinct clustering strategies using the learned vectors.

The two most complex tasks of the class identification step are learning representations of entities and grouping them into classes. However, node2vec is a parallelized and scalable model, as mentioned by Grover and Leskovec (2016). Regarding the clustering approaches,

ANCHOR is flexible and can use classic clustering algorithms, such as K-means++ and Hierarchical Agglomerative Clustering (HAC). Thus, the computational complexity for clustering depends on the chosen algorithm, e.g., K-means++ presents a linear complexity, whereas the standard HAC is  $O(n^3)$ . In short, this step uses methods that can be optimized and scaled for large data.

### 4.3.3 Class Schema Discovery

The last step of the pipeline aims to define the class schema for each of the classes discovered in the previous step (see Definition 5). Similar entities, i.e., belonging to the same class, are often described by a set of distinct attributes. For instance, the *Khurais Field* entity does not contain the attribute *discovery*, which is present in *Ghawar Field* entity (Figure 23). This illustrates the lack of schema standardization our solution needs to deal with to define the class schema.

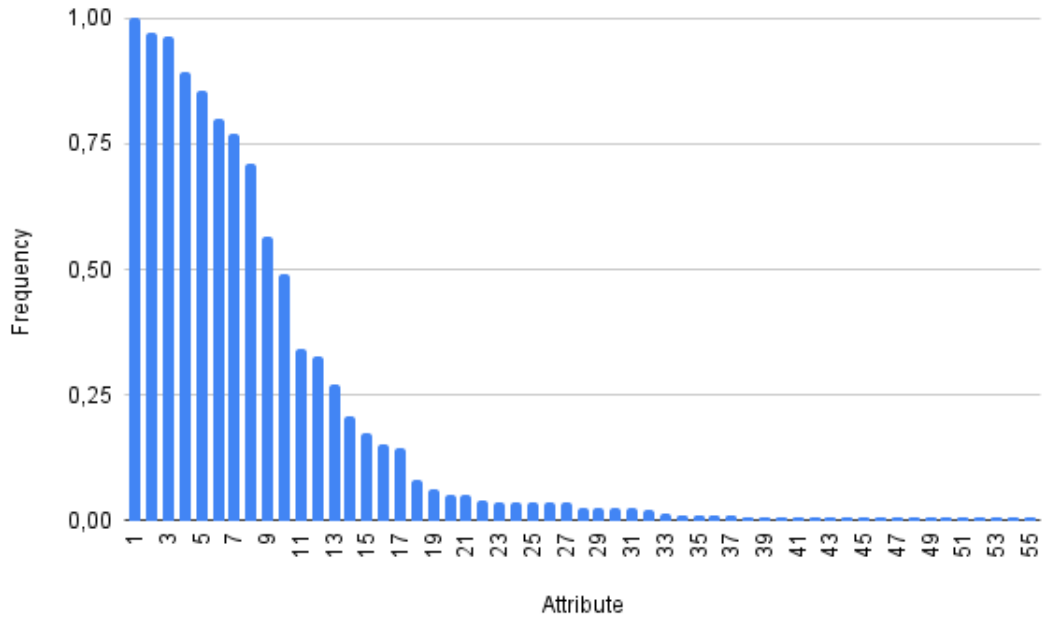
After identifying similar entities, previous approaches (KELLOU-MENOUER; KEDAD, 2015; CHRISTODOULOU; PATON; FERNANDES, 2015) define the class schema with the union of all attributes that describe the entities. However, in the scenario we are dealing with, this naive strategy may present some inconsistencies, e.g., the set of all attributes can be large, and the attributes are not equally relevant.

To illustrate this situation, consider the *oil field* class in the oil and gas domain. In the dataset used in this thesis, the union of the attributes of all its entities is equivalent to 55 attributes, which are not equally relevant. Figure 30 shows the frequency distribution of the attributes of the entities in the *oil and gas* class. Note that only 9 (16%) attributes occur in more than 50% of entities, while most attributes have a low frequency. In other words, the union strategy may include attributes not relevant to describe the set of entities of a class. Other papers have proposed some related approaches (MOREIRA; BARBOSA, 2021; ISSA et al., 2019; WEISE; LOHMANN; HAAG, 2016); however, the core of these solutions is based only on the frequency of the attributes. Furthermore, these approaches rely on a parameter to define a suitable frequency threshold. Specifically, this is a challenge when there is no prior knowledge of the dataset.

Thus, to fill this gap it is necessary to find a way to define a concise representation, i.e., a summarized schema, for an entity class. A summarized schema is useful for applications that need a well-defined schema to perform their tasks. In this sense, our goal is to explore a set of heterogeneous entity schemas  $S$  to find a class schema  $S_c$ , which contains the most relevant attributes for a class  $c$ . For conceptual purposes, we define an entity schema as follows.

**Definition 6 (Entity schema).** *An entity schema  $s(e) = \{a_1, \dots, a_n\}$  consists of a set of attributes that describe an entity  $e$ , e.g., for *Ghawar Field* its entity schema is  $s(\text{GhawarField}) = \{\text{discovery}, \text{country}, \dots, \text{operator}\}$  (see Figure 23).*

Figure 30 – Attribute frequency (Oil field class).



Source: Created by the author

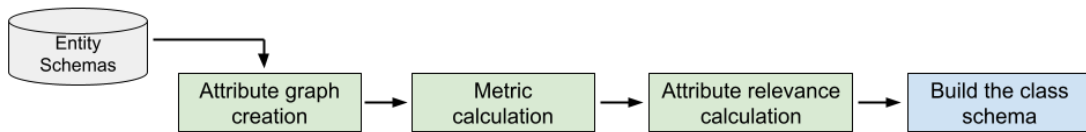
To achieve this goal, we added to the pipeline an approach called **CoFFee** (COSTA-NETO et al., 2022). CoFFee is a free-parameter approach that balances *co-occurrence* and *frequency* of attributes. Our intuition is that less frequent attributes which co-occur with the frequent ones are also important to compose a class schema. Aligned to the most frequent attributes the less frequent ones can also introduce some relevance to the context and provide a more complete schema.

Returning to the example, suppose we are interested in finding the schema of the class *oil field*. As seen earlier, the attributes of this class have a long-tail distribution, e.g., only 16% of attributes (9 of 55) have a frequency greater than 50%. Analyzing a less frequent attribute, e.g., *location* (frequency = 34%), we verify that it has a high co-occurrence value with the most frequent attributes in the schema set, such as *name*, *country*, and *operator*. In this direction, the core of our approach is to combine these two aspects to find a high-quality summarized schema for a class.

Figure 31 illustrates the pipeline executed to achieve this goal. Each step is detailed below. CoFFee models the entity schemas as a graph and uses centrality metrics (degree centrality and closeness) to capture the notion of co-occurrence between attributes. In addition, we propose a novel score that calculates the relevance of an attribute for a set of entity schemas, combining the centrality and frequency values. We use this score to rank and select a set of core attributes for the class.

### Attribute graph creation

Figure 31 – CoFFee’s pipeline.



Source: Created by the author

We model a set of entity schemas as a bipartite graph  $BG = \{E, A, EA\}$ , where  $E$  is a set of entities,  $A$  is a set of attributes, and  $EA$  is a set of edges between an entity and an attribute. Our goal is to capture the co-occurrence relationship between the attributes by generating an *attribute graph*. We derive the attribute graph from  $BG$ . In this graph, an attribute has an edge with another attribute if they appear in the same entity schema. Formally, we define the attribute graph as follows.

**Definition 7 (Attribute graph).** *An attribute graph  $AG = \{A, ES\}$  is a graph where  $A$  is a set of attributes, and  $ES$  is a set of edges, in which there is an edge between two attributes  $a_k$  and  $a_j$  if they occur in the same entity schema.*

We assume that attributes belonging to a set of entity schemas have been submitted to a schema alignment step, i.e., attributes that are homonyms and synonyms have been identified and aligned (DONG; SRIVASTAVA, 2015), since most of the KBs are built upon an ontology that already treats the cases of attributes with different spellings and similar semantic meaning<sup>1</sup>.

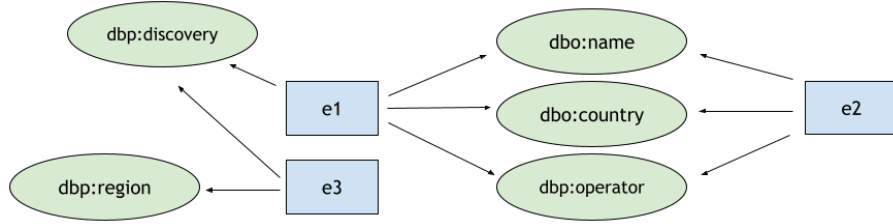
Figure 32a illustrates an example of a bipartite graph created from a set of entity schemas. Blue rectangles represent an entity, while green ellipses represent an attribute. The edges between an entity and attribute indicate that an entity  $e_i \in E$  is described by an attribute  $a_j \in A$ . Figure 32b illustrates an attribute graph resulting from the bipartite graph shown in Figure 32a.

## Metric calculation

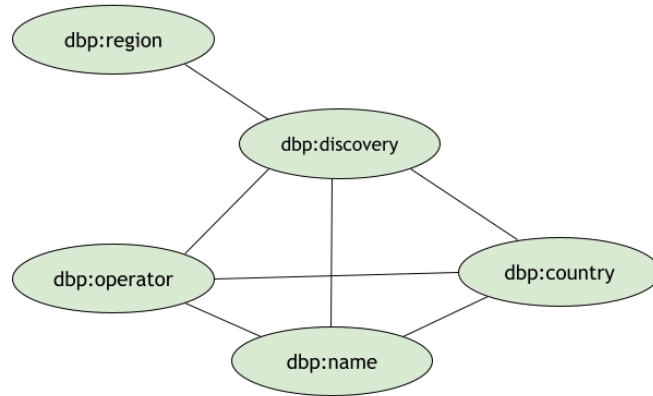
From  $AG$ , we use two centrality metrics to capture the relationship between attributes: *degree* and *closeness centrality* (ZHANG; LUO, 2017). These metrics aim to identify the central nodes of the graph. Each metric expresses a dimension of centrality observed from the graph. The values for each metric are normalized and are in the range of 0 to 1. These centrality measures are defined below.

<sup>1</sup> In the dataset used in this thesis, we identify some conflicts between the names of some attributes. To resolve this, we perform a string match in pairs of attributes. As the attribute alignment task is not the focus of this thesis, we will not discuss it in detail.

Figure 32 – Example of the graphs used by CoFFee. In (a) the bipartite graph created from the set of entity schemata, and (b) the attribute graph created from the relationships between the attributes of the set of entity schemata.



(a) Bipartite graph



(b) Atributte graph

**Source:** Created by the author

**(Degree centrality).** It expresses the number of edges assigned to a node. The centrality degree of an attribute (node)  $a_k$  is calculated as follows:

$$DC(a_k) = \frac{m_i}{(N - 1)} \quad (4.2)$$

Where  $m_i$  number of edges assigned to  $a_k$ , and  $N$  is the number of attributes in  $AG$ .

**(Closeness centrality).** It denotes how close a node is to all nodes of the graph. This measure is the reciprocal of the sum of the distances from a node to the other nodes. The closeness centrality of an attribute  $a_k$  is calculated as follows:

$$Clo(a_k) = \left( \frac{\sum_{j=1}^N d(a_k, a_j)}{(N - 1)} \right)^{-1} \quad (4.3)$$

Where  $d(a_k, a_j)$  is the shortest distance between  $a_k$  and  $a_j$  in  $AG$ .

We chose these metrics to capture the notion of co-occurrence, focusing on two main aspects: *linkage* and *influence*. For example, an attribute  $a_k$  with a high centrality degree

indicates that there is a high number of attributes co-occurring with it. On the other hand, an attribute  $a_k$  with a high value of closeness indicates its high influence to other attributes, i.e., the attribute is close to attributes in the center of the graph. The idea is to capture with which attributes  $a_k$  co-occur. If they occur with core attributes, their degree of closeness is high.

We also calculate the frequency of an attribute  $a_k$  on a set of entity schemas  $S$ . The frequency is calculated as follows:

$$F(a_k) = \frac{n_k}{|S|} \quad (4.4)$$

Where  $n_k$  is the number of times  $a_k$  occurs in  $S$ .

### Attribute relevance calculation

We propose a novel score to calculate the relevance of an attribute  $a_k$  concerning  $S$ . We use this score to define the class schema. We combine the degree and closeness centrality metrics with the frequency. This score helps to capture less frequent attributes that keep relevant interconnections to core attributes. The attribute relevance is calculated as follows:

$$R(a_i) = DC(a_k) * w_{dc} + Clo(a_k) * w_{clo} + F(a_k) * w_f \quad (4.5)$$

The weights for each metric are defined proportionally. In our implementation we set:  $w_{dc} = 0.25$  e  $w_{clo} = 0.25$ ,  $w_f = 0.5$ .

The choice of weights is based on some experiments. They showed that a proportional combination of the co-occurrence and frequency metrics allows CoFFee to provide a summarized schema, minimizing non-relevant attributes without compromising the data retrieval rate.

### Build the class schema

In this step, our goal is to find  $S_c$  (see Definition 8).  $S_c$  is composed of the highest qualified attribute set to describe a set of entity schemas  $S$ . The quality of  $S_c$  is measured according to Equation 4.6. This measure considers the gain and cost of  $S_c$  (defined below) concerning  $S$ .

$$q(S_c) = \sum_{i=1}^N \alpha_i G(S_i, S_c) - \sum_{i=1}^N \beta_i C(S_i, S_c) \quad (4.6)$$

$$G(S_i, S_c) = \frac{|S_i \cap S_c|}{|S_i|} \quad (4.7)$$

$$C(S_i, S_c) = 1 - \frac{|S_i \cap S_c|}{|S_c|} \quad (4.8)$$

Where,  $G(S_i, S_c)$  (Equation 4.7) is the gain of  $S_c$  in  $S_i$ , i.e., the percentage of attributes in  $S_i$  present in  $S_c$ , and  $C(S_i, S_c)$  (Equation 4.8) is the cost of  $S_c$  in  $S_i$ , i.e., the percentage of attributes of  $S_c$  that are not present in  $S_i$ . The weights  $\alpha_i$  e  $\beta_i$  indicate the importance of each  $S_i \in S$  in the gain and cost, respectively, such that  $\sum_{i=1}^N \alpha_i = \sum_{i=1}^N \beta_i = 1$ .

This quality metric was proposed by Wang et al. (2015) however, we adapted the calculation of the weights. Thus,  $\alpha_i$  and  $\beta_i$  are calculated as follows:  $\alpha_i = \frac{r(S_i)}{\sum_{i=1}^N r(S_i)}$  and  $\beta_i = \frac{\frac{1}{r(S_i)}}{\sum_{i=1}^N \frac{1}{r(S_i)}}$ , where  $r(S_i) = \sum_{a_k \in S_i} R(a_k)$  is the sum of the attribute relevance values present in  $S_i$ . In short, the weights allow the selection of the most relevant attributes to compose  $S_c$ . The assumption here is that the most relevant attributes are better at representing  $S$ .

Here, the main challenge is to find  $S_c$  that maximizes  $q(S_c)$ . Due to the size of  $A$ , it can be impractical testing all possible attributes combination. For example, considering the *oil field* class, where  $|A| = 55$ , there are  $2^{55}$  possible combinations. Thus, we propose a heuristic to find a set  $S_c$  that maximizes  $q(S_c)$  considering the attribute relevance.

Algorithm 2 details the process to find  $S_c$ . It receives as input a set of entity schemas  $S$  and a set of attributes ordered by their relevance  $R$  (Equation 4.5). It defines  $S_c$  as top- $j$  attributes in  $R$ , where  $j$  ranges from 1 to  $|R|$  (line 3). Thus, the quality for  $S_c$  is calculated using Equation 4.6 (line 5). The algorithm repeats this process until all attributes contained in  $R$  are added to  $S_c$ . For example, in the first iteration,  $S_c$  contains the most relevant attribute, while in the second iteration, it is equivalent to the two most relevant attributes, and so on. The assumption is that the quality value decreases as less relevant attributes are added to  $S_c$ . After executing lines 3-8, the algorithm checks which set of attributes maximized the quality and defines them as  $S_c$  to represent the class schema (line 9).

---

**Algoritmo 2:** Build the class schema
 

---

**Input:**  $S$ : Set of entity schemas;  $R$ : Set of attributes ordered by relevance (Eq. 4.5)

**Output:**  $S_C$ : Set of core attributes of the class

```

1  $q_{max} \leftarrow 0$ 
2  $k \leftarrow 0$ 
3 for  $j \leftarrow 1$  to  $|R|$  do
4    $S_C \leftarrow$  pick top- $j$  in  $R$ 
5    $q \leftarrow q(S_C)$  // Eq. 4.6
6   if  $q \geq q_{max}$  then
7      $q_{max} \leftarrow q$ 
8      $k \leftarrow j$ 
9  $S_C \leftarrow$  pick top- $k$  in  $R$ 

```

---

#### 4.4 CONSIDERATIONS

In this chapter, we detail our solution for dealing with the schema discovery problem for a specific domain from general-purpose KBs. As we discussed, most applications are domain-specific. In this sense, we include a step for domain discovery in the pipeline. To do this, we extract a domain of interest from categories and identify which entities are part of that domain. Given this, the schema discovery process has been divided into two main tasks: class identification and class schema discovery.

To cluster entities into classes, we propose a new way to represent entities. We model the entity-category mappings provided by the KBs in a graph and apply a node embedding algorithm to learn a representation capable of approximating, in a vector space, similar entities. To find the class schema from a set of similar entities, we propose a new approach that takes into account the frequency and co-occurrence of an attribute concerning a set of entity schemas.

In the next chapter, we detail the experiments performed to evaluate the steps related to the schema discovery task. Furthermore, we discuss and compare the results obtained with other state-of-the-art approaches.

## 5 EXPERIMENTS

In this chapter, we present the experiments performed to evaluate ANCHOR. Specifically, we performed schema discovery on four (4) domains extracted from DBpedia. First, we extract a dataset for each domain. Section 5.1 presents the datasets used in each domain and an overview of the experimental methodology. Then, we evaluated ANCHOR on class identification (Section 5.2) and class schema discovery tasks (Section 5.3). For each experiment, we present the experimental setup, objectives and discuss the results obtained. Finally, Section 5.4 addresses the final considerations of this chapter.

### 5.1 DOMAIN DATA

To evaluate the solutions proposed in this thesis work, we used four (4) distinct Wikipedia domains extracted from the DBpedia datasets (version 2016-10)<sup>1</sup>. For each domain, we executed the *domain discovery* step introduced in Section 4.3.1. We have used the generated domain graph to find the entities of each domain.

Table 4 presents the setup used to build the datasets. The *seed category* column indicates the seed category used to extract the domain. The *categories* column indicates the number of categories found from the root category, i.e., the set of subcategories that represent the domain of interest. The *entities* column indicates the number of entities found from the set of categories.

Table 4 – Summary of the datasets used in each domain.

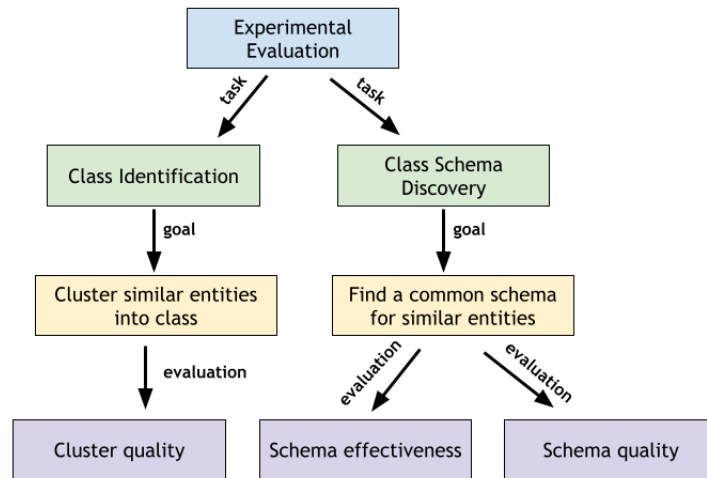
Domain	Seed category	Categories	Entities
Oil and gas	Petroleum_industry	633	2,661
Formula one	Formula_one	512	3,131
Martial arts	Martial_arts	3,591	3,361
Tourism	Tourist_attractions	29,538	3,604

**Source:** Created by the author

Figure 33 summarizes the organization of the experiments. We evaluated ANCHOR in the *class identification* and *class schema discovery* tasks. In the class identification task, our goal is to verify if the proposed strategy to represent the entities contributes to clustering similar entities and, consequently, identifies the existing classes in each one of the domains. We use some traditional clustering algorithms and evaluate the results considering cluster quality metrics. In the class schema discovery task, we verify if the proposed strategy can provide a class schema containing core attributes for the identified classes in a specific domain. We evaluated this task considering two aspects: the effectiveness and

<sup>1</sup> <<https://wiki.dbpedia.org/downloads-2016-10>>

Figure 33 – Summary of experimental evaluation.



**Source:** Created by the author

quality of the generated schema. The source code and data used in the experiments are available on github<sup>2</sup>.

## 5.2 CLASS IDENTIFICATION ASSESSMENT

In this section, we discuss the evaluation of ANCHOR in the class identification task. In this experiment, we verify if the entity’s representations are useful to build consistent entity clusters (classes) required by the task. We compare the clusters built from our entity’s representations to the ones built from state-of-the-art works. This experiment aimed to answer the following questions:

- Q1 - Is it possible to use entity-category mappings provided by general purpose KBs to learn entity representations and use them to identify similar entities?
- Q2 - Compared to other entity representation strategies, does category-based representation present better results in the class identification task?
- Q3 - Can category-based entity representation work well in less homogeneous domains?

### 5.2.1 Experimental setup

**Entity representation creation.** To generate the entities’ representation, we build an entity graph for each domain, using the strategy presented in Section 4.3.2, and then apply the *node2vec* algorithm (GROVER; LESKOVEC, 2016) on each graph with the following setup: dimension=50, walk length=20, number of walks=100, revisiting likelihood (p)=1,

<sup>2</sup> <[https://github.com/ecsn/thesis\\_experiments](https://github.com/ecsn/thesis_experiments)>

and in-out  $(q) = 0.5$ .<sup>3</sup>

**Evaluation set.** Similar to Nguyen et al. (2012), we manually labeled a random sample of entities on each domain to validate the quality of the clusters. Only one annotator carried out the labeling. Manual labeling was necessary because there were a significant amount of entities without `rdf:type` statement or with generic mappings.

We define a set of labels within each specific domain and access the Wikipedia page of each entity to assign the label according to its content. We empirically evaluated the labeling considering a sample of entities with `rdf:type` statement. For these entities, we compare the label assigned by the annotator with the `rdf:type` statement. The evaluation confirmed the adequacy of the assigned labels. We confirm the adequacy of the number of defined classes in two ways: (i) visually, by projecting the entities’ representation in 2D using the t-SNE algorithm (HINTON; ROWEIS, 2003), and (ii) by calculating the silhouette index for a grid of values.

Table 5 shows an overview of the evaluation set in each domain: the total number of entities composing the evaluation set, the proportion of entities sampled from the entire dataset, and the number of classes inside each domain. The domains represent a good variation in terms of number of classes: while *Oil and Gas* domain has 6, *Tourism* has 12 classes.

Table 5 – Overview of the class identification’s evaluation set.

Domain	#Entities	%Entities	#Classes
Oil and gas	842	31	6
Formula one	744	23	8
Martial arts	783	23	11
Tourism	772	21	12

**Source:** Created by the author

**Clustering algorithms.** We execute the scikit-learn implementation of the K-means++, DBSCAN and Hierarchical Agglomerative Clustering (HAC) algorithms to cluster similar entities. To calculate the similarity between entities, K-means++ uses Euclidean distance, whereas DBSCAN and HAC use cosine similarity. We run each algorithm using our proposed entity representation approach, presented in Section 4.3.2, and the following representations for comparison:

- **TFIDF** represents an entity as a TFIDF feature vector of its attributes. A representation based only on attributes was considered in works such as Christodoulou,

<sup>3</sup> Parameters  $p$  and  $q$  were selected according to the study reported by Grover and Leskovec (2016). The values set to both parameters were reported as effective for community detection in graphs, which is related to our task.

Paton and Fernandes (2015), Kellou-Menouer and Kedad (2015), Bouhamoum et al. (2018)

- **RDF2vec** (RISTOSKI; PAULHEIM, 2016) creates a representation vector from RDF graphs. We use the pre-trained model *DB2vec SG 200 8*<sup>4</sup>. This model was trained using data from DBpedia, considering only entity properties.
- **Wikipedia2vec** (YAMADA et al., 2016) builds representation vectors for Wikipedia words and entities in the same vector space. Specifically, representation vectors for entities are learned using the relation of entities through the link structure. We use the pre-trained model *enwiki\_20180420\_win10*<sup>5</sup>.

In this experiment, we considered clustering algorithms with different approaches, e.g., partitional and density. The idea is to verify how they behave using entity representations as input. Our intuition is that entity representations learned by ANCHOR perform well regardless of the approach used by the clustering algorithm.

We also compare our class identification approach with **WIClust** (NGUYEN et al., 2012). WIClust is an initial baseline. The idea is that it has a behavior similar to TF-IDF (K-means++) since it also uses a partitional approach and is based on attributes. To build the clusters, **WIClust** first identifies attribute clusters using a correlation strategy and some constraints. Then, it assigns an entity to its most similar attribute clusters using a TFIDF-based similarity function.

**Evaluation metrics.** We evaluate the results with the three metrics:

- Accuracy (**ACC**) measures the percentage of entities grouped correctly:

$$ACC = \frac{e}{n} \quad (5.1)$$

where  $e$  is the number of entities grouped in the correct group and  $n$  is the number of entities in the dataset.

- The **V-Score** is the harmonic mean between *homogeneity* and *completeness*, and it is calculated as follows:

$$V - Score = \frac{(1 + \beta) * h * c}{(\beta * h) + c} \quad (5.2)$$

where  $h$  is the percentage of instances of a cluster belonging to a same class (homogeneity),  $c$  is the percentage of instances of a class in the same cluster (completeness),

<sup>4</sup> <<http://data.dws.informatik.uni-mannheim.de/rdf2vec/models/DBpedia/2015-10/8depth/skipgram/>>

<sup>5</sup> <[http://wikipedia2vec.s3.amazonaws.com/models/en/2018-04-20/enwiki\\_20180420\\_win10\\_300d.pkl.bz2](http://wikipedia2vec.s3.amazonaws.com/models/en/2018-04-20/enwiki_20180420_win10_300d.pkl.bz2)>

and  $\beta$  is a weight value. For  $\beta$  values higher than 1, completeness has more weight in the V-Score calculation, while for  $\beta$  values lower than 1, the V-Score gives more attention to homogeneity. In our experiments, we do not prioritize any of them, using  $\beta = 1$ .

- Adjusted Rand Index (**ARI**) measures the agreement between two partitions (C, V), where the C partition are the classes created in the clustering process and the V partition is the ground-truth. It is calculated using the following equation:

$$ARI = \frac{\binom{n}{2}(a + d) - [(a + b)(a + c) + (c + d)(b + d)]}{\binom{n}{2}^2 - [(a + b)(a + c) + (c + d)(b + d)]} \quad (5.3)$$

where  $a$  is the number of entities of the same class and the same ground-truth group,  $b$  is the number of entities of the same class and different ground-truth groups,  $c$  is the number of entities of different classes and from the same ground-truth group, and  $d$  is the number of entities from different classes and different ground-truth groups.

All metrics return a value between  $[0 - 1]$ . A score of 1 means a perfect clustering. For the V-Score<sup>6</sup> and ARI<sup>7</sup> metrics, we use the sklearn implementation.

## 5.2.2 Results

Table 6 shows the class identification results. We performed a two sample proportion Z-test (MONTGOMERY; RUNGER, 2010) to compare the best results obtained by our method and the baselines. We considered a significance level of 0.05, with the following null hypothesis H0: The proportions of the entities assigned to the correct cluster (accuracy in our context) by the two approaches are the same. According to the results presented in Table 7, the statistical test have rejected the null hypothesis in all scenarios. Hence, these numbers confirm that our strategy in fact presents superior performance than the baselines. Overall, our method outperformed the other approaches in 3 out of the 4 domains, confirming the effectiveness of our approach.

The only exception was in the *Formula One* domain, in which K-means++ and HAC using TF-IDF had slightly better results than our approach. For instance, HAC+TFIDF obtained a V-Score of 0.93 while our approach using HAC achieved 0.90. In this case, our category-based approach poorer performance was due to some entities of different classes mapped to related categories. For example, **Martin Brundle**<sup>8</sup> and **Jack Brabham**<sup>9</sup>

<sup>6</sup> <[https://scikit-learn.org/stable/modules/generated/sklearn.metrics.v\\_measure\\_score.html](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.v_measure_score.html)>

<sup>7</sup> <[https://scikit-learn.org/stable/modules/generated/sklearn.metrics.adjusted\\_rand\\_score.html](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.adjusted_rand_score.html)>

<sup>8</sup> <[https://en.wikipedia.org/wiki/Martin\\_Brundle](https://en.wikipedia.org/wiki/Martin_Brundle)>

<sup>9</sup> <[https://en.wikipedia.org/wiki/Jack\\_Brabham](https://en.wikipedia.org/wiki/Jack_Brabham)>

Table 6 – Class identification results.

Domain: Oil and gas					Domain: Formula one				
Method	Algorithm	ACC	V-Score	ARI	Method	Algorithm	ACC	V-Score	ARI
Our method	K-means++	<b>0.97</b>	<b>0.90</b>	<b>0.91</b>	Our method	K-means++	0.85	0.86	0.79
	HAC	0.95	0.88	0.88		HAC	0.93	0.90	0.88
	DBSCAN	0.92	0.85	0.86		DBSCAN	0.91	0.88	0.88
	K-means++	0.95	0.89	0.89		K-means++	0.96	0.92	0.94
TF-IDF	HAC	0.93	0.87	0.86	TF-IDF	HAC	<b>0.96</b>	<b>0.93</b>	<b>0.94</b>
	DBSCAN	0.89	0.84	0.85		DBSCAN	0.87	0.85	0.83
	WIClust	0.91	0.81	0.80		WIClust	0.9	0.87	0.84
	K-means++	0.67	0.56	0.44		K-means++	0.61	0.58	0.45
RDF2vec	HAC	0.49	0.33	0.20	RDF2vec	HAC	0.54	0.47	0.29
	DBSCAN	0.5	0.28	0.18		DBSCAN	0.37	0.28	0.14
	K-means++	0.38	0.19	0.12		K-means++	0.35	0.22	0.12
Wikipedia2vec	HAC	0.39	0.18	0.10	Wikipedia2vec	HAC	0.29	0.18	0.10
	DBSCAN	0.44	0.16	0.11		DBSCAN	0.29	0.20	0.10
Domain: Martial arts					Domain: Tourism				
Method	Algorithm	ACC	V-Score	ARI	Method	Algorithm	ACC	V-Score	ARI
Our method	K-means++	0.89	0.88	0.83	Our method	K-means++	<b>0.98</b>	<b>0.97</b>	<b>0.96</b>
	HAC	<b>0.93</b>	<b>0.90</b>	<b>0.86</b>		HAC	0.97	0.96	0.95
	DBSCAN	0.83	0.85	0.76		DBSCAN	0.89	0.89	0.85
	K-means++	0.86	0.82	0.80		K-means++	0.74	0.75	0.62
TF-IDF	HAC	0.87	0.83	0.80	TF-IDF	HAC	0.73	0.75	0.61
	DBSCAN	0.65	0.70	0.48		DBSCAN	0.61	0.67	0.43
	WIClust	0.78	0.77	0.70		WIClust	0.67	0.59	0.51
	K-means++	0.81	0.78	0.71		K-means++	0.32	0.27	0.16
RDF2vec	HAC	0.73	0.71	0.60	RDF2vec	HAC	0.35	0.29	0.18
	DBSCAN	0.32	0.33	0.10		DBSCAN	0.33	0.20	0.13
	K-means++	0.54	0.50	0.40		K-means++	0.34	0.32	0.18
Wikipedia2vec	HAC	0.26	0.46	0.10	Wikipedia2vec	HAC	0.31	0.30	0.12
	DBSCAN	0.25	0.42	0.10		DBSCAN	0.26	0.22	0.10

Source: Created by the author.

Table 7 – Results for the two sample proportion Z-test.

Domain: Oil and gas		Domain: Formula one	
Comparisons		Comparisons	
Our method (K-means++) vs	p-value	Our method (HAC) vs	p-value
TF-IDF (K-means++)	0.036618	TF-IDF (HAC)	0.011406
RDF2vec (K-means++)	0.00001	RDF2vec (K-means++)	0.00001
Wikipedia2vec (DBSCAN)	0.00001	Wikipedia2vec (DBSCAN)	0.00001

Domain: Martial arts		Domain: Tourism	
Comparisons		Comparisons	
Our method (HAC) vs	p-value	Our method (K-means++) vs	p-value
TF-IDF (K-means++)	0.000078	TF-IDF (K-means++)	0.00001
RDF2vec (K-means++)	0.00001	RDF2vec (HAC)	0.00001
Wikipedia2vec (K-means++)	0.00001	Wikipedia2vec (K-means++)	0.00001

Source: Created by the author

were drivers who played different roles in Formula One: commentator and team owner, respectively. The *folksonomic* nature of the categories can lead to cases like that.

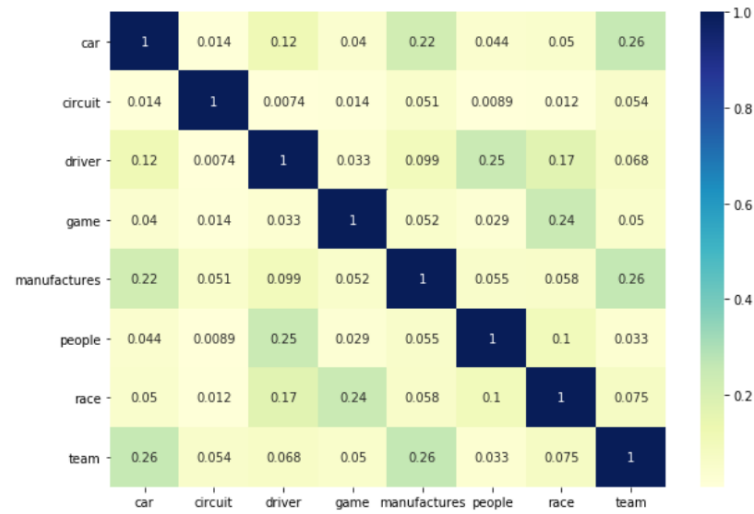
Moreover, there is little overlap between the attributes of entities in the classes of this domain, which benefits attribute-based approaches such as TFIDF since it is easier to separate the classes based on their attributes. We illustrate this assertion in Figure 34(a) by computing the Jaccard similarity between all classes' global schema (the union of all the attributes of the entities) in the Formula one domain.

On the other hand, the TFIDF approach shows inferior performance in domains where the attributes of some classes overlap. For instance, in the *Martial Arts* domain, there are some points of attribute overlap, e.g., judoka, fencer, mixedmartial, and taekwondo - see Figure 34(b). We observed that some entities in these classes are in the same cluster because these entities only have common attributes related to athletes, such as **nickname** and **nationality**. Hence, it is difficult to distinguish between them by considering only their attributes. In short, this is one of the limitations when we consider only attributes as a way of representing entities.

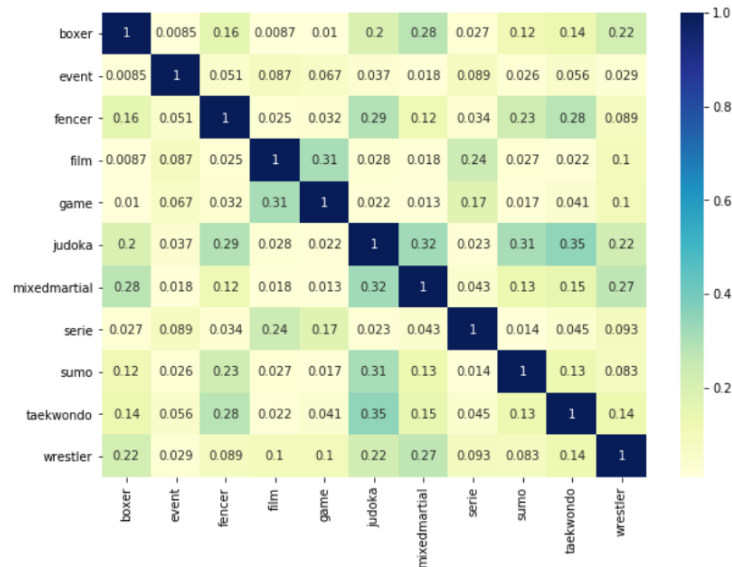
In comparison to WIClust, our approach outperformed it in all domains. The main reason for that is that WIClust, as TFIDF approaches, also relies on the attributes of the entities to build the clusters. Despite this, we observed that when we compare it with the TF-IDF approach using K-means++ it performs close (in some domains) and obtains a superior result when we compare it with the TF-IDF approach using DBSCAN. WIClust relies on a grouping of representative attributes for good performance. As expected, WIClust underperformed mainly in domains where there was overlap between the entities' schemas namely, *Martial arts* and *Tourism*.

Our approach also outperformed the results achieved by embedding-based approaches. To further illustrate the quality of our entity representation strategy based on shared categories, we project in Figure 35 the vectors of RDF2Vec, Wikipedia2vec and ours in the

Figure 34 – Overlap between data schemas for each class.



(a) Formula One



(b) Martial Arts

**Source:** Created by the author

Formula One domain using t-SNE. We note that our approach produced homogeneous and spatially separated clusters (Figure 35a) in comparison to the ones created by RDF2vec (Figure 35b) and Wikipedia2vec (Figure 35c).

In the case of Wikipedia2vec, its model builds entity embeddings by exploring their relations in the Wikipedia graph. However, connections between entities many times do not necessarily express class matching. For example, *Michael Schumacher* (driver) and *2004 Australian Grand Prix* (race) are close in the Wikipedia2vec space because Michael Schumacher won the 2004 Australian Grand Prix.

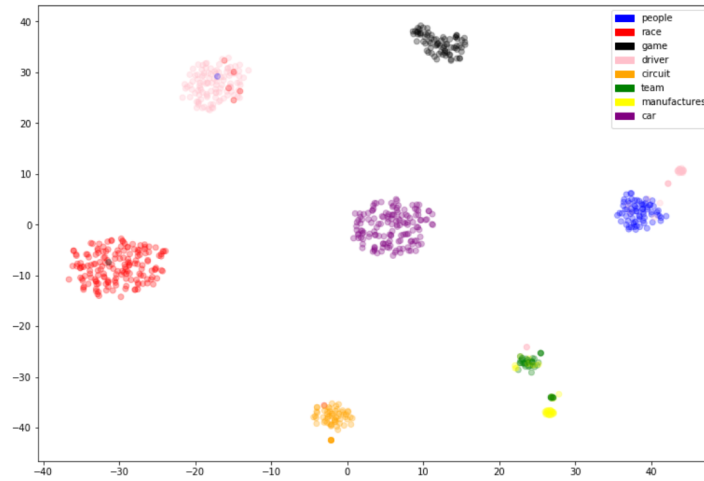
Regarding RDF2vec, this model was trained based on object properties only. In other words, this model captures the semantics between entities, but has a lower performance

---

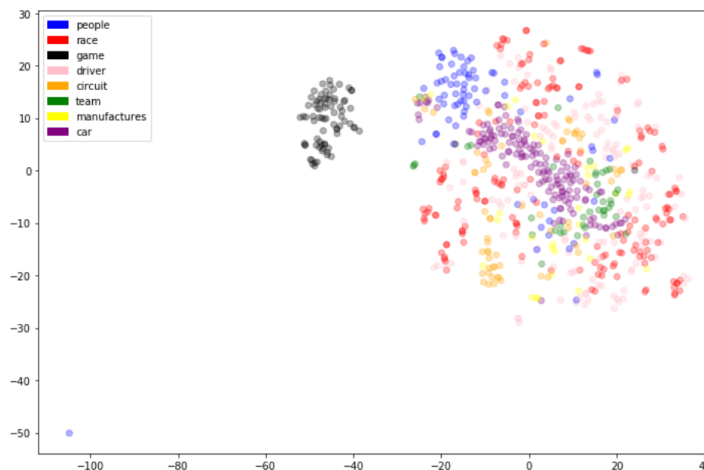
compared to approaches based on attributes (TF-IDF) and categories (our method). The vectors representing the entities in the same class are close, but the clusters (classes) are not fully isolated. This behavior represents an issue for clustering algorithms, especially density-based ones like DBSCAN. All of these observations can explain, therefore, the poor performance of Wikipedia2vec and RDF2vec, presented in Table 6.

In summary, the results presented answer the questions addressed in this experiment. We demonstrate that it is feasible to use entity-category mappings to learn a representation for entities (Q1). The results show that it is possible to apply the representations learned in traditional clustering algorithms and identify groups of similar entities. As categories are a resource for organizing the content and structure of general-purpose KBs, we believe this has contributed to creating isolated and homogeneous clusters. Consequently, our approach achieved a good performance in the evaluated metrics, confirming the quality of the provided clusters (Q2). We observed that even in heterogeneous domains (with overlap between classes), our approach overcame the limitations of the attribute-based approach maintaining a good performance (Q3).

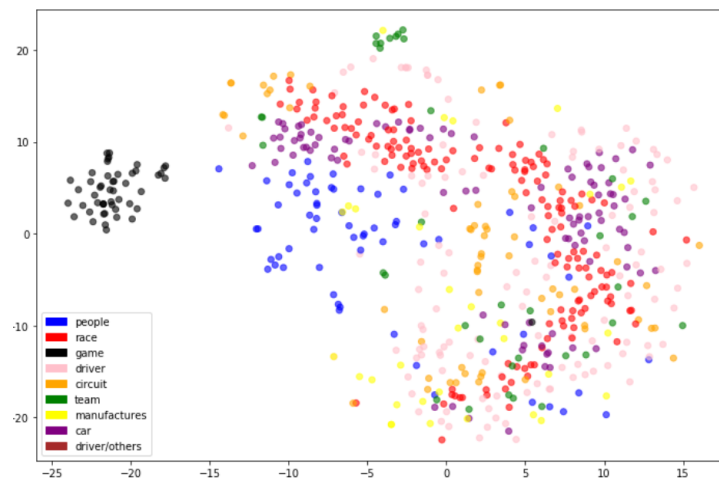
Figure 35 – Visualizations of (a) ANCHOR’s vectors, (b) RDF2vec vectors and (c) Wikipedia2vec vectors for the Formula One domain.



(a) ANCHOR’s vectors



(b) RDF2vec vectors



(c) Wikipedia2vec vectors

**Source:** Created by the author

### 5.3 CLASS SCHEMA DISCOVERY ASSESSMENT

In this section, we evaluate CoFFee, our class schema discovery approach that is part of the pipeline proposed in this thesis. In this experiment, we verify whether CoFFee is effective for identifying the most significant attributes that describe entities within the same class. Our evaluation compares CoFFee with other similar state-of-the-art approaches. This experiment aimed to answer the following questions.

- Q1 - Is CoFFee effective for selecting a set of relevant attributes concerning a set of entity schemas within the same class?
- Q2 - Does CoFFee produces a summarized schema without losing information relevant to the class?
- Q3 - When compared to a reference schema, does the result produced by CoFFee provide a good quality schema for representing an entity class?

#### 5.3.1 Experimental setup

**Dataset.** We evaluate CoFFee on the DBpedia dataset *infobox-properties* (version 12/2021). For each domain, we consider the classes identified by our class identification approach in the previous experiment. More specifically, we pick the output of the configurations with the best V-scores: K-means++ for the Oil and Gas and Tourism domains, and HAC for Formula One and Martial Arts. The distribution of classes in each domain is organized as described in Appendix A.

**Baselines.** We compare the performance of our approach against Skeleton (WANG et al., 2015) and LOD-CM (ISSA et al., 2019) since these solutions are highly aligned with the objective of this thesis. We briefly discuss the intuition behind each approach below.

- **Skeleton.** It is a parameter-free approach that aims to present a summarized representation, i.e., a set of core attributes, for a set of schemas. It considers equivalence between schemas, and the class schema is inclined towards attributes that occur in equivalent schemas.
- **LOD-CM.** It uses the FP-growth algorithm to find patterns (i.e., a set of attributes) that co-occur frequently above a user-defined threshold. The class schema is the set of attributes contained in the set of patterns identified by the algorithm. In our experiments, we varied this parameter considering the values 0.5, 0.3 e 0.1<sup>10</sup>.

<sup>10</sup> When the entity schemata are very heterogeneous, higher thresholds can restrict the number of attributes selected. Therefore, we chose to vary this threshold starting from 0.5.

**Experiments and Evaluation metrics.** Below we summarize the objectives of the experiments we perform to evaluate CoFFee.

- **Experiment 1** aims at analysing the **effectiveness** of the class schema generated by the approaches, i.e., we check if the approaches provide a summarized schema without losing information that is relevant to the class. To this end, we use two metrics proposed in (WANG et al., 2015): Retrieval Rate (RR) (Equation 5.4) and Relative Size (RS) (Equation 5.5).

$$RR = \frac{\sum_{i=1}^N \frac{|S_i \cap S_c|}{|S_i|}}{|S|} \quad (5.4)$$

$$RS = \frac{|S_c|}{|A|} \quad (5.5)$$

Where  $S$  is a set of entity schemas of the same class,  $A$  is a set of distinct attributes in  $S$  and  $S_c$  is the class schema. In other words, RR measures the gain of information obtained using the class schema, while RS measures the size of the class schema concerning the universal attribute set.

- **Experiment 2** analyzes the quality of the class schema in comparison to a reference schema. We defined the reference schema (for each class) from the set of attributes belonging to the infobox template most used by its entities. Infoboxes are one of the resources used by DBpedia to extract structured information from Wikipedia (MOREIRA; COSTA-NETO; BARBOSA, 2021), and infobox templates are created by a crowdsourcing effort and are a reasonable approximation of the class schema. Thus, we believe that the infobox template provides a closer reference schema for the entities of a class. Section 2.1.1 introduced the infobox and infobox template concepts and how they are associated with an entity. In the Appendix B, we present the infobox template considered in each class of the respective domains. It is important to note that we excluded some attributes defined as metadata, such as: **image**, **alt** and **caption**. To measure the quality of the schema generated by the approaches, we used the metrics: Precision (P), Recall (R), and F-measure (F1).

$$P = \frac{TP}{TP + FP} \quad (5.6)$$

$$R = \frac{TP}{TP + FN} \quad (5.7)$$

$$F1 = \frac{2 * P * R}{P + R} \quad (5.8)$$

Where  $TP$  is the number of selected attributes that belong to the reference schema;  $FP$  is the number of attributes that were selected but that do not belong to the reference schema; and  $FN$  is the number of attributes that belong to the reference schema but have not been selected.

### 5.3.2 Results

#### Experiment 1

Figure 36 shows the performance of the approaches concerning the retrieval rate (RR) and relative size (RS) indices. For comparison, we consider the universal attribute set (i.e., the union of all attributes of all entities of a class) as a baseline. This strategy was adopted in some works, such as Christodoulou, Paton and Fernandes (2015). The value of RR and RS for this universal schema are equal to 1. Our goal is to provide a summarized class schema without losing relevant information. For that, we minimize the RS index while keeping the RR value as close as possible to 1.

The presented results are consolidated, i.e., it is the average of the results obtained in each class of the domain. When comparing CoFFee with the universal attribute set, the RR index varies between 0.74 (Tourism) and 0.81 (Oil and gas). Meanwhile, the RS index falls between 0.27 (Formula one) and 0.23 (Oil and gas) - see Figure 36. Table 8 presents the difference obtained between CoFFee's performance and the baseline (Universal). Note that the difference in performance measured by the RR index is smaller than the RS index. In summary, the number of attributes selected by CoFFee is 75% lower than the baseline on average, see the RS index. Nonetheless, the RR index is significant. CoFFee averaged 78% for RR, achieving a difference of just 0.22 (see RR index) against the baseline. In other words, CoFFee is able to select relevant attributes to offer a more summarized description of the class entities while preserving the recall.

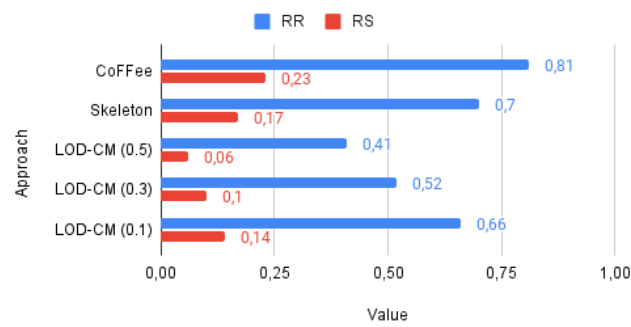
Table 8 – Difference between CoFFee and Universal approaches

Domain	Universal attribute set	
	RR = 1	RS = 1
Oil and gas	0.19	0.77
Formula one	0.22	0.73
Martial arts	0.21	0.75
Tourism	0.26	0.75
<b>Source:</b> Created by the author		

Looking at the metrics for Skeleton, we observed that this approach also manages to provide a summarized schema while keeping the RR index relatively high. Specifically, comparing CoFFee x Skeleton, we noticed an average gain in the RR index (ranging from 6 to 14%). In other words, our approach selects more relevant attributes. Consequently, by increasing the number of selected attributes, the RS index grows (cost). However, the

Figure 36 – Performance (effectiveness) of the approaches to summarize the class schema.

## Oil and gas



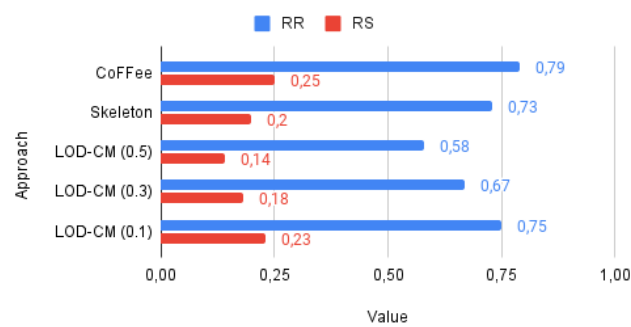
(a) Oil and gas

## Formula one



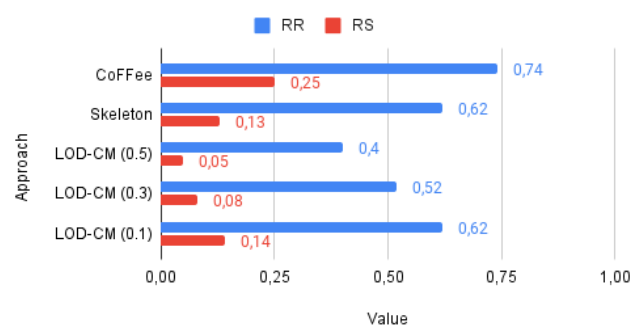
(b) Formula one

## Martial arts



(c) Martial arts

## Tourism



(d) Tourism

Source: Created by author

trade-off between RR and RS is positive, indicating that CoFFee selects attributes that contribute to increasing the retrieval rate.

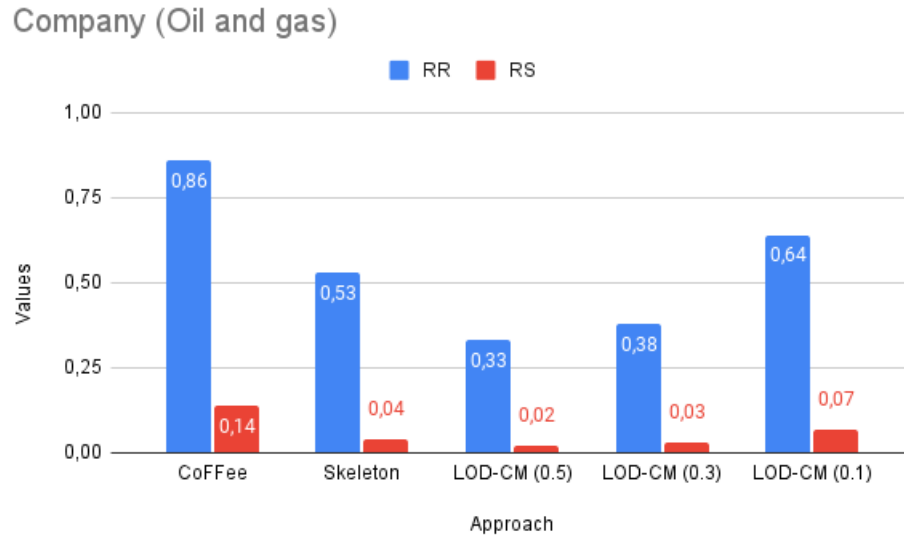
To understand and explore the particularity of each approach, we discuss individual results for some classes. Figure 37 shows results for company class (Oil and gas domain). Overall, this class has entities that describe oil companies. Note that there is a difference in the RR index between CoFFee and Skeleton. The set of attributes selected by each approach explains this difference: CoFFee selected 25 attributes, while Skeleton only 8. We noticed that Skeleton failed to consider some discriminative attributes for this class, e.g., **numEmployees**, **revenue**, and **owner**. When compare the attributes **hqLocation** (selected by both approaches) and **numEmployees** (selected only by CoFFee), we verify that they have similar frequencies ( $0.49 \times 0.45$ , respectively). Skeleton selects an attribute according to how often it occurs in equivalent schemas. For this approach, two schemas are equivalent if they share the same set of attributes. We observed the frequency of these attributes in equivalent schemas and identified that are lower concerning the selected attributes by Skeleton. This justifies the exclusion of these attributes in the selection process carried out by Skeleton.

On the other hand, our approach seeks to find a trade-off between the frequency and co-occurrence of an attribute to measure its relevance concerning a set of entity schemas. Although the previously mentioned attributes are less frequent in the set of entity schemas, they have a high co-occurrence with frequent attributes, e.g., **name**, **industry**, and **type**. This is one of the reasons why CoFFee selected a broader set of attributes concerning Skeleton. When we look at the RS index, we see that CoFFee is balanced. Note that out of the 171 distinct attributes present in the set of entity schemas of the company class, our approach selected 25 (equivalent to 14% of this total) and kept the RS index at 0.86 - see Figure 37. This demonstrates that we filter out irrelevant and/or less discriminative attributes for the class without losing recall.

We also observed that in some classes, our approach produced results similar to Skeleton, e.g., *field* (oil and gas), *driver*, and *race* (formula one) classes. We find two reasons that justify this. First, we noticed that Skeleton generally gets a high retrieval rate (RR) when the set of entity schemas is less heterogeneous. For example, in the *field* and *driver* classes, the entity schemas heterogeneity ratio is 0.60 and 0.56, respectively. In other words, Skeleton manages to leverage a greater number of attributes concerning a more heterogeneous scenario (e.g., company class) since its heuristic is based on schema equivalence. It is important to highlight that CoFFee proved to be stable in both scenarios since the core of its approach depends on the relationship of attributes and not on the equivalence between the entities' schemas.

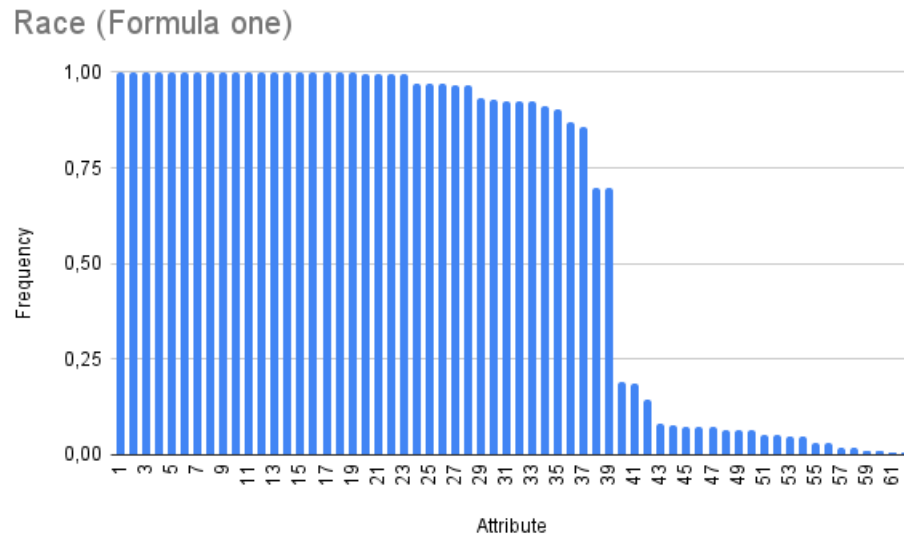
Another aspect we observed was the frequency distribution between the attributes. In the *race* class, all approaches produced the same result. Usually, in semi-structured datasets, the frequency of attributes follows a long-tail distribution. Specifically, in this

Figure 37 – Effectiveness of approaches to summarize the company class schema.



Source: Created by the author

Figure 38 – Attribute frequency (Race class).



Source: Created by the author

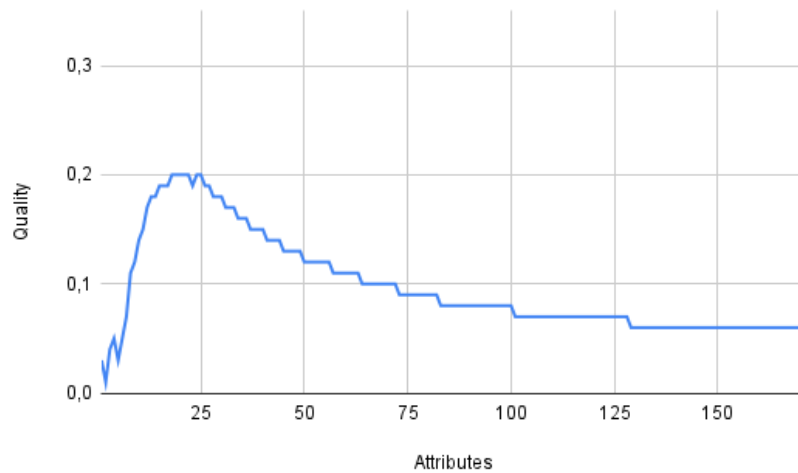
class, there was an inverse phenomenon (see Figure 38). Note that 39 out of 62 attributes appear in at least 69% of the schemas. In this case, the output is the same regardless of the adopted strategy (among those considered in this experiment). It is important to highlight that this phenomenon is not usually recurrent.

Returning to the consolidated results (Figure 36) and observing the results obtained by the LOD-CM approach, it is possible to verify that it is the approach that provides a more summarized schema, i.e., it has a low RS index, but the RR index value is also low. In other words, this approach fails to consider relevant attributes.

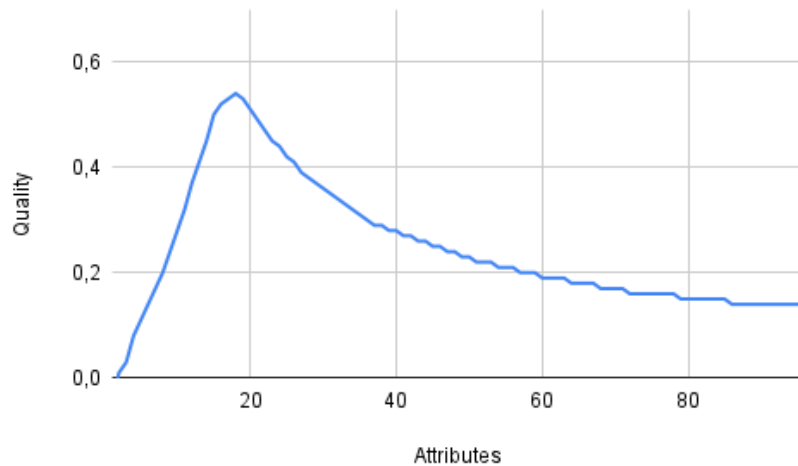
LOD-CM depends on a parameter to find a set of attributes that co-occur under this

threshold. In our experimental evaluation, we defined three values (0.5, 0.3, and 0.1). In this case, a low threshold implies an increased coverage of attributes. The selection of an attribute is conditioned on the existence of a pattern that satisfies the defined threshold. In this sense, this approach can limit the number of selected attributes depending on the value assigned to the parameter and the frequency distribution of the attributes, especially in scenarios where this distribution is long-tail. Manually setting is challenging when the user has no prior knowledge of the dataset. It is important to note that the CoFFee and Skeleton approaches are parameter-free.

Figure 39 – Quality of the class schema ordered by the relevance of the attributes.



(a) Company (Oil and gas)



(b) Driver (Formula one)

**Source:** Created by the author

We selected two previously mentioned classes (company and driver) to illustrate the behavior of the heuristic used by CoFFee to find the class schema (see Section 4.3.3). CoFFee uses a heuristic to build a class schema based on a quality metric that balances

gain and cost, weighted by the attribute relevance score (Equation 4.6). Note that in both classes the schema’s quality decreasing as we add less relevant attributes to it (Figure 39). For the Company class, CoFFee selected 25 attributes (out of 171 distinct attributes), and for the Driver class, it selected 18 attributes (out of 96 distinct attributes). In both cases, the number of selected attributes was considerably reduced, keeping the information retrieval rate above 80%.

In summary, these results answer some questions addressed at the beginning of Section 5.3. Specifically, CoFFee proved to be a competitive approach for the schema summarization task compared to other state-of-the-art approaches (Q1). Furthermore, we observed that CoFFee managed to minimize the size of the class schema, discarding attributes not relevant to the class (Q2). Next, we analyze the quality of the schema produced by the approaches concerning a reference schema.

## Experiment 2

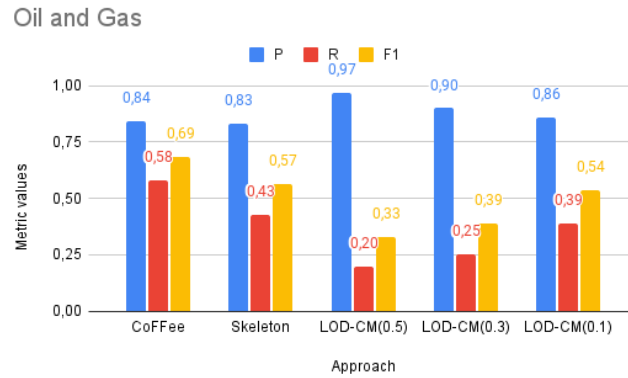
Figure 40 presents the result of the quality of the schema generated by the evaluated approaches concerning the reference schema. The results are consolidated by domain, i.e., the average results of the values observed in each class of the domain. We present the result in this way to facilitate the presentation and general understanding of the data.

Overall, we observed that the approaches have a high value for the precision metric (above 0.80). This means that most of the attributes selected by the approaches are present in the reference schema. Furthermore, regarding the F1 metric, we observed that CoFFee outperforms all approaches in the evaluated domains. The main reason for this is that CoFFee can select more attributes for the class schema and, consequently, get a higher value for the recall metric.

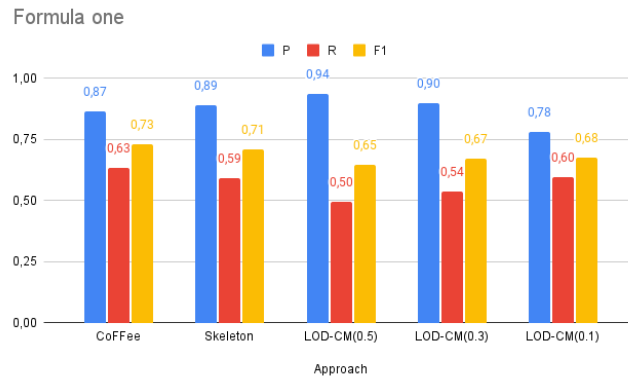
To deepen the discussion, we present in Table 9 the individual result obtained in each class of the domain *oil and gas*. We chose this domain because it has a smaller number of classes, and the results obtained have similar characteristics to the other domains. In this sense, it is possible to generalize the discussion and understand the consolidated results.

Regarding the precision metric, some attributes selected by the approaches were not present in the reference schema. The lowest value of this metric was observed in the *company* class. In this class, CoFFee obtained a precision of 0.64. Specifically, this class uses the short version of the INFOBOX\_COMPANY template as a reference schema (as described in Appendix B). Our approach selected attributes such as **services** and **equity**, which are company-specific attributes, but which were not present in the reference schema. Note that in this same class, other approaches also had a lower value for this metric, e.g., LOD-CM (0.1). A similar situation also occurs in the *businessperson* class, which uses a generic template for a person as a reference. In the other classes, the value for the precision metric is close to 1 for all approaches.

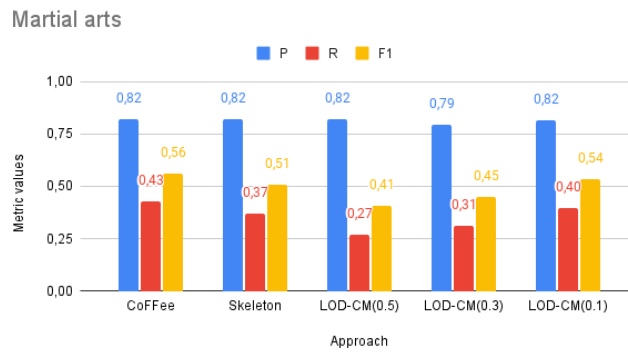
Figure 40 – Class schema quality compared to the reference schema.



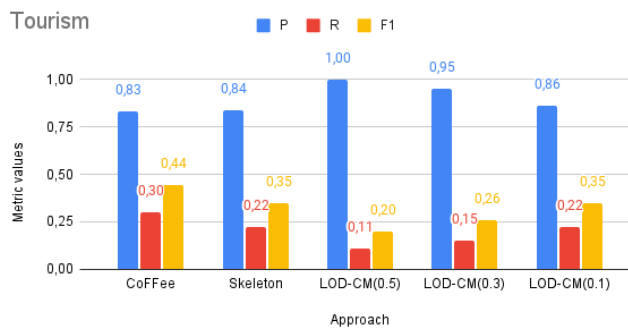
(a) Oil and gas



(b) Formula one



(c) Martial arts



(d) Tourism

Source: Created by the author

Table 9 – Class schema quality compared to the reference schema - Oil and gas domain.

Class	Approach	P	R	F1
Company	CoFFee	0.64	0.84	<b>0.72</b>
	Skeleton	0.87	0.36	0.51
	LOD-CM (0.5)	1.00	0.21	0.34
	LOD-CM (0.3)	1.00	0.26	0.41
	LOD-CM (0.1)	0.61	0.42	0.50
Field	CoFFee	1.00	0.30	<b>0.46</b>
	Skeleton	1.00	0.30	<b>0.46</b>
	LOD-CM (0.5)	1.00	0.19	0.31
	LOD-CM (0.3)	1.00	0.27	0.42
	LOD-CM (0.1)	1.00	0.30	<b>0.46</b>
Pipeline	CoFFee	0.88	0.46	<b>0.61</b>
	Skeleton	0.84	0.34	0.48
	LOD-CM (0.5)	0.83	0.15	0.26
	LOD-CM (0.3)	0.75	0.18	0.30
	LOD-CM (0.1)	0.81	0.28	0.41
Businessperson	CoFFee	0.75	0.75	0.75
	Skeleton	0.81	0.75	<b>0.78</b>
	LOD-CM (0.5)	1.00	0.33	0.55
	LOD-CM (0.3)	0.66	0.33	0.44
	LOD-CM (0.1)	0.77	0.58	0.66
Refinery	CoFFee	0.90	0.64	<b>0.75</b>
	Skeleton	0.72	0.57	0.64
	LOD-CM (0.5)	1.00	0.28	0.44
	LOD-CM (0.3)	1.00	0.42	0.60
	LOD-CM (0.1)	1.00	0.64	0.78
Ship	CoFFee	0.87	0.54	<b>0.66</b>
	Skeleton	0.75	0.28	0.41
	LOD-CM (0.5)	1.00	0.06	0.12
	LOD-CM (0.3)	1.00	0.09	0.17
	LOD-CM (0.1)	1.00	0.15	0.27

**Source:** Created by the author

We observe that the reference schema has more attributes (see Table 14) concerning the output produced by the approaches. However, some attributes contained in the reference schema are not used by entities of the class. Entity schemas are flexible, which means that entities within the same class have a set of distinct attributes and do not necessarily have all the attributes suggested for that class. Consequently, attributes less used by entities of a class may exist in the reference schema. In Moreira, Costa-Neto and Barbosa (2021), the authors observed the correlation between the quantity of attributes suggested in a template and the number of attributes used by the entities. They notice that the bigger the suggested infobox template, the lower the attributes coverage on entity infoboxes, and the smaller the suggested infobox template, the bigger the attributes coverage on entity infoboxes. In this sense, this justifies the values obtained by all approaches for the recall

metric.

To illustrate this situation, observe that in the *company* and *businessperson* classes, CoFFee obtained a high value for the recall metric. In the opposite direction, the recall value for classes *field* and *pipeline* was lower. It is important to highlight that we expect this situation since we are dealing with entities defined without schema constraints. Thus, in this scenario, is common a certain level of incompleteness. Some papers address this problem, for example, Moreira and Barbosa (2021), which seeks to improve the coverage of a KB, filling in values for missing attributes for existing entities.

Regarding the F1 metric, CoFFee outperforms the other approaches in 4 of 6 classes and gets a similar result in 1 class (oil and gas domain). The reason for this is that CoFFee achieves a high recall value. In the domain in question, the biggest difference comes from the *ship* class. In this class, CoFFee obtained a difference in recall of 0.26 points for Skeleton and 0.48, 0.45, and 0.39 for LOD-CM with parameters 0.5, 0.3, and 0.1, respectively. We consider attributes like *shipSpeed* and *shipCapacity*, which are relevant attributes for a ship. As mentioned earlier, CoFFee selects less frequent attributes considering their occurrence with core attributes (more frequent), making the number of attributes selected higher most of the time.

Overall, it was possible to verify that CoFFee provides a good quality schema to represent an entity class. The schema generated by CoFFee is in line with the reference schema (high precision) and compared with the other approaches for summarizing schemas, our approach covered the highest number of selected relevant attributes (highest recall and F1). Finally, based on the analysis of this experiment, it was possible to answer question Q3, addressed at the beginning of this section.

## 5.4 CONSIDERATIONS

In this chapter, we present and discuss the experimental results related to the pipeline proposed in this thesis. Specifically, we split our assessment into two parts.

First, we evaluate our strategy for entity representation using entity-category mappings. The results showed that our approach is feasible to identify classes of similar entities in a specific domain. Our approach obtained superior results, surpassing representations based only on the attributes of entities. In this experiment, we confirmed all the hypotheses raised, showing that the model to generate the entity representations works well as a feature extractor and can be used in different clustering algorithms: K-means++, DBSCAN, and HAC.

Finally, we evaluate the behavior of CoFFee, our approach to class schema discovery. In the experiments carried out, we verified that CoFFee is effective in selecting the most significant attributes for a set of entities within the same class. In addition, it manages to filter less relevant attributes, reducing the set of attributes and, consequently, the complexity of the class schema. CoFFee proved to be competitive concerning other state-

of-the-art approaches for schema summarization. It surpasses them concerning the recall of selected attributes since the core of its approach is based on the frequency and co-occurrence of attributes.

In the next chapter, we present the final considerations of this thesis work and address the contributions achieved, as well as future directions.

## 6 CONCLUSION

In this chapter, we present the final considerations regarding this thesis work. Initially, we present an overview of this work and answer the addressed research questions (Section 6.1). Subsequently, we list the contributions achieved (Section 6.2) and point out the limitations and future directions (Section 6.3). Finally, we list the papers written during the doctoral course (Section 6.4). Some are published, whereas others are under peer review.

### 6.1 OVERVIEW

In this thesis, we address the problem of specific domain schema discovery from KBs. As mentioned earlier, KBs are valuable datasets and have high semantic coverage. Furthermore, general-purpose KB stores a large volume of knowledge within multiple domains. This has motivated the development of many applications.

One way to consume data from KBs is by exploring their content through a schematic description. However, schema-related statements are not mandatory and may be missing or provided incompletely. To overcome this challenge, we seek to develop some solutions related to the schema discovery task, e.g., domain discovery, class identification, and class schema discovery.

In this thesis work, we propose an end-to-end pipeline (ANCHOR). It works in three steps. Each step seeks to answer a research question (addressed in the Introduction), which we discuss below.

#### ***RQ1: How to extract a specific domain from KB?***

To answer this research question, we analyzed the main general-purpose KBs, e.g., DBpedia and YAGO, and found that they use Wikipedia categories to organize their content thematically. Specifically, in this thesis, we use the DBpedia KB. We have observed that Wikipedia’s categories form a large taxonomy. This organization allows the exploration of a set of related subjects from a high-level category. Thus, we use an approach that exploits category-category mappings to extract a specific domain. In summary, given a seed category, the strategy explores these mappings towards a set of subcategories representing a domain of interest. From this, it was possible to extract a dataset with entities within a specific domain and use it as input for other tasks related to schema discovery.

***RQ2: How to identify similar entities in a KB, and which entity characteristics are the most effective for performing the class identification task?***

The class identification task is a primary task in the schema discovery problem. One of our contributions in this thesis was using entity-category mapping to create an entity representation applying a node embedding algorithm. We observe that entities within the same class generally share categories.

In the literature, we observed that many papers address this task by applying approaches based on entity attributes. This is an interesting strategy when entity attributes are less heterogeneous or when classes have low overlap between their attributes. Otherwise, this approach tends to underperform. So, to fill this gap, we investigated another way to represent entities.

Specifically, we take advantage of the entity-category mappings provided by the KB to model them as a bipartite graph and, from it, create an entity graph, in which entities that share categories have an edge. We apply node2vec on the entity graph to learn entity embeddings by exploring its neighborhood. Overall, node2vec contributed to making closer, in a representation space, entities that share categories.

We performed an experimental evaluation in four distinct DBpedia domains to evaluate the behavior of our entity representation strategy. The experiments showed that when classical clustering algorithms used our entity representations as the input, they obtained good results, e.g., identifying groups of similar entities. We compare the results obtained using our strategy against traditional (e.g., attribute-based) and embedding-based baselines. Experimental results confirm our observation regarding attribute-based approaches. In summary, we outperformed this strategy in domains in which there was a high overlap between class attributes, e.g., Martial arts e Tourism domain. Furthermore, our approach also obtained good results in the domains where the attribute-based strategy performed well.

In this sense, the observations addressed above answer the second research question. In short, we conclude that our entity representation strategy is a viable way to tackle the task of class identification. Furthermore, KB categories proved to be good characteristics in this task.

***RQ3: How to select which attributes are relevant to represent a set of entities within an entity class?***

To answer this research question, we developed CoFFee, an approach to select relevant attributes for an entity class. This approach is based on attribute frequency and co-occurrence. An attribute is considered relevant if it is used frequently by entities within a class. KBs do not use schema-based constraints to define an entity. Therefore, entities

within the same class are heterogeneous, i.e., they are described by distinct sets of attributes. In this sense, CoFFee uses a heuristic that weights attributes frequency based on co-occurrences. The idea was to balance frequency and co-occurrence to select the most significant attributes.

The experiments carried out point to the viability of CoFFee. Specifically, compared with two state-of-the-art approaches, which were more related to the objective of the work, CoFFee presented good results, making it a competitive approach for this end task. In summary, the results showed that CoFFee selects a set of relevant attributes for a class keeping its retrieval rate high. In other words, it proved to be effective in filtering out less relevant attributes. Furthermore, the output produced by CoFFee is in line with the reference schema (high precision). Compared with the other approaches for summarizing schemas, it covers the highest number of selected relevant attributes (highest recall and F1).

## 6.2 CONTRIBUTIONS

The elaboration of this thesis work generated some contributions to the state-of-the-art, which we list below:

- We created ANCHOR, an end-to-end pipeline for specific domain schema discovery from KB.
- We explore the Wikipedia category graph to extract specific domain from General-purpose KB.
- We specify an approach to entity representation using the entity-category mappings provided by the general-purpose KB.
- We conducted an experimental evaluation on four domains extracted from DBpedia to evaluate our entity representation strategy against traditional and embedding-based baselines.
- We created CoFFee, an approach that combines frequency and co-occurrence of attributes to select the most relevant attributes for a set of similar entities.
- We performed experiments to evaluate the effectiveness and quality of the class schema generated by CoFFee. We compared our approach with two state-of-the-art baselines. The experiments showed the viability of CoFFee.

## 6.3 LIMITATIONS AND FUTURE WORK

In this section, we list some limitations of this work. In addition, we address some future directions that can be explored from this thesis.

## **Limitations**

- **Choice of seed category.** ANCHOR relies on a seed category to extract a specific domain. To do this, it is necessary to know the structure of the Wikipedia category graph. Choosing a fine-grained category can restrict the domain. Therefore, the seed category must be representative of the domain of interest.
- **Pruning threshold.** ANCHOR uses a pruning strategy to avoid cycles and the addition of irrelevant categories in the domain extraction step. Currently, it uses Jaccard similarity and depends on a threshold that is manually specified by the user.
- **Link among classes.** In the class identification task, we identify only the classes within the dataset. However, semantic links among classes can exist. Thus, enriching the schema by extracting semantic links is an open issue in this work.
- **Manual labeling.** We could not find any benchmarking dataset on a specific-domain for our class identification task. A significant amount of DBpedia entities have missing `rdf:type` statements. Additionally, DBpedia ontology is cross-domain, and sometimes it does not cover the particularities of a specific domain. Therefore, we used a manually labeled gold standard to evaluate our approach to this task.
- **Domain Discovery Assessment.** In this work, we performed experiments only to evaluate the Class Identification and Class Schema Discovery tasks. For the Domain Discovery task, we only performed an empirical evaluation to analyze the adequacy of the selected categories for each domain.

## **Future Work**

- **Multi-view clustering approach.** The experiments showed that in some scenarios relying on entity attributes is a viable strategy for class identification task. So, one possibility is to investigate whether improving the class identification task is possible by applying a multi-view clustering approach. The idea is to combine different perspectives of entities, using entity-category mappings and attributes to create a unique representation for entities. The intuition is that a hybrid view can help create more homogeneous clusters.
- **Weighted entity graphs.** Verify if assigning weights in the entity graph contributes to improving the representations generated by the node embedding algorithm. The idea is to define higher weight edges between entities with more categories in common.

- **Enrich the schema.** Increment the class description with other schema-related information, such as attribute data type and range of values.
- **Perform a case study.** Evaluate ANCHOR in a domain-specific application, e.g., query formulation or extraction information applications. The idea is to verify if the schema provided by ANCHOR contributes to helping the tasks performed by the application.
- **Incremental approach.** General-purpose KBs are datasets that deal with both the insertion and the deletion of entities. The dataset schema can change from these modifications. In this sense, adapting the solution to consider the incremental aspect of the dataset can be an interesting way.
- **Identify seed category.** As mentioned, ANCHOR relies on a seed category to extract a specific domain. In this direction, a study that identifies (based on user requirements) candidate seed categories within a theme can benefit potential pipeline users.

## 6.4 PUBLICATIONS

During the doctoral course, we developed some papers, directly or indirectly, related to this thesis. These papers are listed below.

1. **Costa Neto, E.**, Moreira J., Barbosa, L., Salgado, A.C. "Domain-Specific Schema Discovery from General-Purpose Knowledge Base". *Int. J. Metadata Semantics and Ontologies* (Accepted for publication).
2. **Costa Neto, E.**, Moreira, J., Barbosa, L., Salgado, A.C. "CoFFee: A Co occurrence and Frequency-Based Approach to Schema Mining". In *Anais do XXXVII Simpósio Brasileiro de Bancos de Dados*, (pp. 52-64). Porto Alegre: SBC. (2022) doi:10.5753/sbbd.2022.224190
3. **Costa Neto, E.**, Moreira, J., Barbosa, L. A., Salgado A. C. "Class Schema Discovery for Semi-Structured Data". *Journal of Information and Data Management* (Accepted for publication).
4. Moreira, J., **Costa Neto, E.** and Barbosa, L. "Analysis of structured data on Wikipedia". *Int. J. Metadata Semantics and Ontologies*, Vol. 15, No. 1, pp.71–86. (2021) DOI: <http://dx.doi.org/10.1504/IJMSO.2021.117108>
5. Moreira, J., **Costa Neto, E.**, Barbosa, L. A. "Índices de Infoboxes para Recuperação de Informação Estruturada de Entidades da Wikipédia". In: *Brazilian Symposium on Databases, 2019, Fortaleza. 34th Brazilian Symposium on Databases - Dataset Showcase Workshop*, 2019.

## REFERENCES

- ADOLPHS, P.; THEOBALD, M.; SCHAFER, U.; USZKOREIT, H.; WEIKUM, G. Yago-qa: Answering questions by structured knowledge queries. In: *2011 IEEE Fifth International Conference on Semantic Computing*. [S.l.: s.n.], 2011. p. 158–161.
- ALI, W.; SALEEM, M.; YAO, B.; HOGAN, A.; NGOMO, A.-C. N. A survey of rdf stores & sparql engines for querying knowledge graphs. *The VLDB Journal*, Springer, p. 1–26, 2022.
- AOUICHA, M. B.; TAIEB, M. A. H.; EZZEDDINE, M. Derivation of “is a” taxonomy from wikipedia category graph. *Engineering Applications of Artificial Intelligence*, Elsevier, v. 50, p. 265–286, 2016.
- AUER, S.; BIZER, C.; KOBILAROV, G.; LEHMANN, J.; CYGANIAK, R.; IVES, Z. Dbpedia: A nucleus for a web of open data. In: SPRINGER. *The Semantic Web: 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007+ ASWC 2007, Busan, Korea, November 11-15, 2007. Proceedings*. [S.l.], 2007. p. 722–735.
- BENGIO, Y.; COURVILLE, A.; VINCENT, P. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 35, n. 8, p. 1798–1828, 2013.
- BIZER, C.; LEHMANN, J.; KOBILAROV, G.; AUER, S.; BECKER, C.; CYGANIAK, R.; HELLMANN, S. Dbpedia-a crystallization point for the web of data. *Journal of web semantics*, Elsevier, v. 7, n. 3, p. 154–165, 2009.
- BOLDI, P.; MONTI, C. Cleansing wikipedia categories using centrality. In: . Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2016. (WWW '16 Companion), p. 969–974. ISBN 9781450341448. Available at: <<https://doi.org/10.1145/2872518.2891111>>.
- BORDES, A.; USUNIER, N.; GARCIA-DURAN, A.; WESTON, J.; YAKHNENKO, O. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, v. 26, 2013.
- BOUHAMOUM, R.; KEDAD, Z.; LOPES, S. Scalable schema discovery for rdf data. In: *Transactions on Large-Scale Data-and Knowledge-Centered Systems XLVI*. [S.l.]: Springer, 2020. p. 91–120.
- BOUHAMOUM, R.; KEDAD, Z.; LOPES, S. Incremental schema discovery at scale for rdf data. In: SPRINGER. *European Semantic Web Conference*. [S.l.], 2021. p. 195–211.
- BOUHAMOUM, R.; KEDAD, Z.; LOPES, S. Incremental schema generation for large and evolving rdf sources. In: *Transactions on Large-Scale Data-and Knowledge-Centered Systems LI*. [S.l.]: Springer, 2022. p. 28–63.
- BOUHAMOUM, R.; KELLOU-MENOUE, K.; LOPES, S.; KEDAD, Z. Scaling up schema discovery for rdf datasets. In: *2018 IEEE 34th International Conference on Data Engineering Workshops (ICDEW)*. [S.l.: s.n.], 2018. p. 84–89.

Cai, H.; Zheng, V. W.; Chang, K. C. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Transactions on Knowledge and Data Engineering*, v. 30, n. 9, p. 1616–1637, 2018.

CAMPINAS, S.; PERRY, T. E.; CECCARELLI, D.; DELBRU, R.; TUMMARELLO, G. Introducing rdf graph summary with application to assisted sparql formulation. In: *2012 23rd International Workshop on Database and Expert Systems Applications*. [S.l.: s.n.], 2012. p. 261–266.

CHRISTODOULOU, K.; PATON, N. W.; FERNANDES, A. A. A. Structure inference for linked data sources using clustering. In: \_\_\_\_\_. *Transactions on Large-Scale Data- and Knowledge-Centered Systems XIX: Special Issue on Big Data and Open Data*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015. p. 1–25. ISBN 978-3-662-46562-2. Available at: <[https://doi.org/10.1007/978-3-662-46562-2\\_1](https://doi.org/10.1007/978-3-662-46562-2_1)>.

COSTA-NETO, E.; MOREIRA, J.; BARBOSA, L.; SALGADO, A. C. Coffee: A co-occurrence and frequency-based approach to schema mining. In: SBC. *Anais do XXXVII Simpósio Brasileiro de Bancos de Dados*. [S.l.], 2022. p. 52–64.

DALMIA, A.; J, G.; GUPTA, M. Towards interpretation of node embeddings. In: *Companion Proceedings of the The Web Conference 2018*. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2018. (WWW '18), p. 945–952. ISBN 9781450356404. Available at: <<https://doi.org/10.1145/3184558.3191523>>.

DONG, X. L.; SRIVASTAVA, D. Schema alignment. In: \_\_\_\_\_. *Big Data Integration*. Cham: Springer International Publishing, 2015. p. 31–61. ISBN 978-3-031-01853-4. Available at: <[https://doi.org/10.1007/978-3-031-01853-4\\_2](https://doi.org/10.1007/978-3-031-01853-4_2)>.

ESTER, M.; KRIEGEL, H.-P.; SANDER, J.; XU, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In: . [S.l.]: AAAI Press, 1996. (KDD'96), p. 226–231.

FAHAD, A.; ALSHATRI, N.; TARI, Z.; ALAMRI, A.; KHALIL, I.; ZOMAYA, A. Y.; FOUFOU, S.; BOURAS, A. A survey of clustering algorithms for big data: Taxonomy and empirical analysis. *IEEE transactions on emerging topics in computing*, IEEE, v. 2, n. 3, p. 267–279, 2014.

FENSEL, D.; ŞİMŞEK, U.; ANGELE, K.; HUAMAN, E.; KÄRLE, E.; PANASIUK, O.; TOMA, I.; UMBRICH, J.; WAHLER, A.; FENSEL, D. et al. Introduction: what is a knowledge graph? *Knowledge graphs: Methodology, tools and selected use cases*, Springer, p. 1–10, 2020.

FONT, L.; ZOUAQ, A.; GAGNON, M. Assessing the quality of domain concepts descriptions in dbpedia. In: IEEE. *2015 11th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*. [S.l.], 2015. p. 254–261.

GÓMEZ, S. N.; ETCHEVERRY, L.; MAROTTA, A.; CONSENS, M. P. Findings from two decades of research on schema discovery using a systematic literature review. *AMW*, 2018.

- GROVER, A.; LESKOVEC, J. Node2vec: Scalable feature learning for networks. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2016. (KDD '16), p. 855–864. ISBN 9781450342322. Available at: <<https://doi.org/10.1145/2939672.2939754>>.
- HAHN, R.; BIZER, C.; SAHNWALDT, C.; HERTA, C.; ROBINSON, S.; BÜRGLE, M.; DÜWIGER, H.; SCHEEL, U. Faceted wikipedia search. In: ABRAMOWICZ, W.; TOLKSDORF, R. (Ed.). *Business Information Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. p. 1–11. ISBN 978-3-642-12814-1.
- HAN, L.; FININ, T.; JOSHI, A. Gorelations: An intuitive query system for dbpedia. In: SPRINGER. *Joint International Semantic Technology Conference*. [S.l.], 2011. p. 334–341.
- HEIST, N.; PAULHEIM, H. Uncovering the semantics of wikipedia categories. In: GHIDINI, C.; HARTIG, O.; MALESHKOVA, M.; SVÁTEK, V.; CRUZ, I.; HOGAN, A.; SONG, J.; LEFRANÇOIS, M.; GANDON, F. (Ed.). *The Semantic Web – ISWC 2019*. Cham: Springer International Publishing, 2019. p. 219–236. ISBN 978-3-030-30793-6.
- HINTON, G. E.; ROWEIS, S. T. Stochastic neighbor embedding. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2003. p. 857–864.
- HOFFART, J.; SUCHANEK, F. M.; BERBERICH, K.; WEIKUM, G. Yago2: A spatially and temporally enhanced knowledge base from wikipedia. *Artificial intelligence*, Elsevier, v. 194, p. 28–61, 2013.
- HUBERT, L.; ARABIE, P. Comparing partitions. *Journal of classification*, Springer, v. 2, n. 1, p. 193–218, 1985.
- ISSA, S.; PARIS, P.-H.; HAMDI, F.; CHERFI, S. S.-S. Revealing the conceptual schemas of rdf datasets. In: SPRINGER. *International conference on advanced information systems engineering*. [S.l.], 2019. p. 312–327.
- JACCARD, P. Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bull Soc Vaudoise Sci Nat*, v. 37, p. 547–579, 1901.
- KASSAMBARA, A. *Practical guide to cluster analysis in R: Unsupervised machine learning*. [S.l.]: Sthda, 2017.
- KAUFMAN, L.; ROUSSEEUW, P. J. *Finding groups in data: an introduction to cluster analysis*. [S.l.]: John Wiley & Sons, 1989.
- KELLOU-MENOUER, K.; KARDOULAKIS, N.; TROULLINO, G.; KEDAD, Z.; PLEXOUSAKIS, D.; KONDYLAkis, H. A survey on semantic schema discovery. *The VLDB Journal*, Springer, v. 31, n. 4, p. 675–710, 2022.
- KELLOU-MENOUER, K.; KEDAD, Z. Schema discovery in rdf data sources. In: JOHANNESSEN, P.; LEE, M. L.; LIDDLE, S. W.; OPDAHL, A. L.; LÓPEZ, Ó. P. (Ed.). *Conceptual Modeling*. Cham: Springer International Publishing, 2015. p. 481–495. ISBN 978-3-319-25264-3.

KOTLERMAN, L.; AVITAL, Z.; DAGAN, I.; LOTAN, A.; WEINTRAUB, O. A support tool for deriving domain taxonomies from wikipedia. In: *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*. [S.l.: s.n.], 2011. p. 503–508.

KUZEY, E.; WEIKUM, G. Extraction of temporal facts and events from wikipedia. In: *Proceedings of the 2Nd Temporal Web Analytics Workshop*. New York, NY, USA: ACM, 2012. (TempWeb '12), p. 25–32. ISBN 978-1-4503-1188-5. Available at: <<http://doi.acm.org/10.1145/2169095.2169101>>.

LALITHSENA, S.; PERERA, S.; KAPANIPATHI, P.; SHETH, A. Domain-specific hierarchical subgraph extraction: A recommendation use case. In: *2017 IEEE International Conference on Big Data (Big Data)*. [S.l.: s.n.], 2017. p. 666–675.

LAN, Y.; HE, G.; JIANG, J.; JIANG, J.; ZHAO, W. X.; WEN, J.-R. Complex knowledge base question answering: A survey. *IEEE Transactions on Knowledge and Data Engineering*, IEEE, 2022.

LANGE, D.; BOHM, C.; NAUMANN, F. Extracting structured information from wikipedia articles to populate infoboxes. In: *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*. New York, NY, USA: ACM, 2010. (CIKM '10), p. 1661–1664. ISBN 978-1-4503-0099-5. Available at: <<http://doi.acm.org/10.1145/1871437.1871698>>.

LEHMANN, J.; ISELE, R.; JAKOB, M.; JENTZSCH, A.; KONTOKOSTAS, D.; MENDES, P. N.; HELLMANN, S.; MORSEY, M.; KLEEF, P. van; AUER, S.; BIZER, C. DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, v. 6, n. 2, p. 167–195, 2015. Available at: <[http://jens-lehmann.org/files/2015/swj\\_dbpedia.pdf](http://jens-lehmann.org/files/2015/swj_dbpedia.pdf)>.

LI, Z.; XU, Q.; ZHANG, W.; ZHANG, T. An approach and implementation for knowledge graph construction and q&a system. In: IEEE. *2021 3rd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI)*. [S.l.], 2021. p. 425–429.

LIU, Y.; LI, Z.; XIONG, H.; GAO, X.; WU, J. Understanding of internal clustering validation measures. In: IEEE. *2010 IEEE international conference on data mining*. [S.l.], 2010. p. 911–916.

LIU, Z.; LIN, Y.; SUN, M. Word representation. In: \_\_\_\_\_. *Representation Learning for Natural Language Processing*. Singapore: Springer Singapore, 2020. p. 13–41. ISBN 978-981-15-5573-2. Available at: <[https://doi.org/10.1007/978-981-15-5573-2\\_2](https://doi.org/10.1007/978-981-15-5573-2_2)>.

MA, C.; MOLNÁR, B.; TARCSI, Á.; BENCZÚR, A. Knowledge enriched schema matching framework for heterogeneous data integration. In: IEEE. *2022 IEEE 2nd Conference on Information Technology and Data Science (CITDS)*. [S.l.], 2022. p. 183–188.

MACQUEEN, J. Some methods for classification and analysis of multivariate observations. In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*. Berkeley, Calif.: University of California Press, 1967. p. 281–297. Available at: <<https://projecteuclid.org/euclid.bsmisp/1200512992>>.

- MAHDISOLTANI, F.; BIEGA, J.; SUCHANEK, F. Yago3: A knowledge base from multilingual wikipe-dias. In: CIDR CONFERENCE. *7th biennial conference on innovative data systems research*. [S.l.], 2014.
- MEHRI, R.; VALTCHEV, P. Mining schema knowledge from linked data on the web. In: LI, G.; GE, Y.; ZHANG, Z.; JIN, Z.; BLUMENSTEIN, M. (Ed.). *Knowledge Science, Engineering and Management*. Cham: Springer International Publishing, 2017. p. 261–273. ISBN 978-3-319-63558-3.
- MEIMARIS, M. Visualizing implicit rdf schema with the help of embeddings. In: *EDBT/ICDT Workshops*. [S.l.: s.n.], 2019.
- MIKOLOV, T.; CHEN, K.; CORRADO, G.; DEAN, J. Efficient Estimation of Word Representations in Vector Space. In: . [s.n.], 2013. p. 1–12. Available at: <<http://arxiv.org/abs/1301.3781>>.
- MIKOLOV, T.; SUTSKEVER, I.; CHEN, K.; CORRADO, G.; DEAN, J. Distributed representations of words and phrases and their compositionality. In: *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*. USA: Curran Associates Inc., 2013. (NIPS’13), p. 3111–3119. Available at: <<http://dl.acm.org/citation.cfm?id=2999792.2999959>>.
- MIRYLENKA, D.; PASSERINI, A.; SERAFINI, L. Bootstrapping domain ontologies from wikipedia: A uniform approach. In: *Twenty-Fourth International Joint Conference on Artificial Intelligence*. [S.l.: s.n.], 2015.
- MONATH, N.; DUBEY, K. A.; GURUGANESH, G.; ZAHEER, M.; AHMED, A.; MCCALLUM, A.; MERGEN, G.; NAJORK, M.; TERZIHAN, M.; TJANAKA, B. et al. Scalable hierarchical agglomerative clustering. In: *Proceedings of the 27th ACM SIGKDD Conference on knowledge discovery & data mining*. [S.l.: s.n.], 2021. p. 1245–1255.
- MONTGOMERY, D. C.; RUNGER, G. C. *Applied statistics and probability for engineers*. [S.l.]: John wiley & sons, 2010.
- MORALES, A.; PREMTOON, V.; AVERY, C.; FELSHIN, S.; KATZ, B. Learning to answer questions from Wikipedia infoboxes. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, 2016. p. 1930–1935. Available at: <<https://www.aclweb.org/anthology/D16-1199>>.
- MOREIRA, J.; BARBOSA, L. Deepex: A robust weak supervision system for knowledge base augmentation. *Journal on Data Semantics*, Springer, v. 10, n. 3, p. 309–325, 2021.
- MOREIRA, J.; COSTA-NETO, E.; BARBOSA, L. Índices de infoboxes para recuperação de informação estruturada de entidades da wikipédia. In: *SBBD Datasets Showcase*. [S.l.: s.n.], 2019. (SBBD’19).
- MOREIRA, J.; COSTA-NETO, E.; BARBOSA, L. Analysis of structured data on wikipedia. *International Journal of Metadata, Semantics and Ontologies*, Inderscience Publishers (IEL), v. 15, n. 1, p. 71–86, 2021.
- NASTASE, V.; STRUBE, M. Decoding wikipedia categories for knowledge acquisition. In: *AAAI*. [S.l.: s.n.], 2008. v. 8, p. 1219–1224.

- NGUYEN, T.; MOREIRA, V.; NGUYEN, H.; NGUYEN, H.; FREIRE, J. Multilingual schema matching for wikipedia infoboxes. *Proc. VLDB Endow.*, VLDB Endowment, v. 5, n. 2, p. 133–144, Oct. 2011. ISSN 2150-8097. Available at: <<http://dx.doi.org/10.14778/2078324.2078329>>.
- NGUYEN, T. H.; NGUYEN, H. D.; MOREIRA, V.; FREIRE, J. Clustering wikipedia infoboxes to discover their types. In: *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*. New York, NY, USA: Association for Computing Machinery, 2012. (CIKM '12), p. 2134–2138. ISBN 9781450311564. Available at: <<https://doi.org/10.1145/2396761.2398588>>.
- PAN, J. Z.; PAVLOVA, S.; LI, C.; LI, N.; LI, Y.; LIU, J. Content based fake news detection using knowledge graphs. In: SPRINGER. *The Semantic Web–ISWC 2018: 17th International Semantic Web Conference, Monterey, CA, USA, October 8–12, 2018, Proceedings, Part I 17*. [S.l.], 2018. p. 669–683.
- PEROZZI, B.; AL-RFOU, R.; SKIENA, S. Deepwalk: Online learning of social representations. In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2014. (KDD '14), p. 701–710. ISBN 9781450329569. Available at: <<https://doi.org/10.1145/2623330.2623732>>.
- RAYNAUD, T.; SUBERCAZE, J.; LAFOREST, F. Fouilla: Navigating dbpedia by topic. In: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. New York, NY, USA: Association for Computing Machinery, 2018. (CIKM '18), p. 1907–1910. ISBN 9781450360142. Available at: <<https://doi.org/10.1145/3269206.3269210>>.
- REBELE, T.; SUCHANEK, F.; HOFFART, J.; BIEGA, J.; KUZEY, E.; WEIKUM, G. Yago: A multilingual knowledge base from wikipedia, wordnet, and geonames. In: GROTH, P.; SIMPERL, E.; GRAY, A.; SABOU, M.; KRÖTZSCH, M.; LECUE, F.; FLÖCK, F.; GIL, Y. (Ed.). *The Semantic Web – ISWC 2016*. Cham: Springer International Publishing, 2016. p. 177–185. ISBN 978-3-319-46547-0.
- RISTOSKI, P.; PAULHEIM, H. Rdf2vec: Rdf graph embeddings for data mining. In: *International Semantic Web Conference*. [S.l.: s.n.], 2016.
- ROCHA, O. R.; ZUCKER, C. F.; GIBOIN, A. Extraction of relevant resources and questions from dbpedia to automatically generate quizzes on specific domains. In: NKAMBOU, R.; AZEVEDO, R.; VASSILEVA, J. (Ed.). *Intelligent Tutoring Systems*. Cham: Springer International Publishing, 2018. p. 380–385. ISBN 978-3-319-91464-0.
- ROSENBERG, A.; HIRSCHBERG, J. V-measure: A conditional entropy-based external cluster evaluation measure. In: *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. Prague, Czech Republic: Association for Computational Linguistics, 2007. p. 410–420. Available at: <<https://www.aclweb.org/anthology/D07-1043>>.
- ROUSSEEUW, P. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, Elsevier Science Publishers B. V., NLD, v. 20, n. 1, p. 53–65, Nov. 1987. ISSN 0377-0427. Available at: <[https://doi.org/10.1016/0377-0427\(87\)90125-7](https://doi.org/10.1016/0377-0427(87)90125-7)>.

- SáEZ, T.; HOGAN, A. Automatically generating wikipedia info-boxes from wikidata. In: *Companion Proceedings of the The Web Conference 2018*. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2018. (WWW '18), p. 1823–1830. ISBN 978-1-4503-5640-4. Available at: <<https://doi.org/10.1145/3184558.3191647>>.
- SANDER, J. Density-based clustering. In: \_\_\_\_\_. *Encyclopedia of Machine Learning*. Boston, MA: Springer US, 2010. p. 270–273. ISBN 978-0-387-30164-8. Available at: <[https://doi.org/10.1007/978-0-387-30164-8\\_211](https://doi.org/10.1007/978-0-387-30164-8_211)>.
- SANTOS, J. M.; EMBRECHTS, M. On the use of the adjusted rand index as a metric for evaluating supervised classification. In: ALIPPI, C.; POLYCARPOU, M.; PANAYIOTOU, C.; ELLINAS, G. (Ed.). *Artificial Neural Networks – ICANN 2009*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. p. 175–184. ISBN 978-3-642-04277-5.
- SATER, A. *Tocando Em Frente*. 1992. CD.
- SENOUSSI, H. Using dbpedia categories to evaluate and explain similarity in linked open data. In: *KEOD*. [S.l.: s.n.], 2018. p. 115–125.
- SERRA, E.; CORTEZ, E.; SILVA, A. S. d.; MOURA, E. S. On using wikipedia to build knowledge bases for information extraction by text segmentation. *JIDM*, v. 2, n. 3, p. 259–272, 2011. Available at: <<http://seer.lcc.ufmg.br/index.php/jidm/article/view/140>>.
- SILVA, D. N. R. da; ZIVIANI, A.; PORTO, F. Aprendizado de máquina e inferência em grafos de conhecimento. In: \_\_\_\_\_. [S.l.]: TÓPICOS EM GERENCIAMENTO DE DADOS E INFORMAÇÕES, 2019. p. 1–30.
- SOFRONOVA, R.; BISWAS, R.; ALAM, M.; SACK, H. Entity typing based on rdf2vec using supervised and unsupervised methods. In: SPRINGER. *The Semantic Web: ESWC 2020 Satellite Events: ESWC 2020 Satellite Events, Heraklion, Crete, Greece, May 31–June 4, 2020, Revised Selected Papers 17*. [S.l.], 2020. p. 203–207.
- STEINBACH, M.; KUMAR, V.; TAN, P. Cluster analysis: basic concepts and algorithms. *Introduction to data mining, 1st edn*. Pearson Addison Wesley, 2005.
- SUCHANEK, F. M.; KASNECI, G.; WEIKUM, G. Yago: A core of semantic knowledge. In: *Proceedings of the 16th International Conference on World Wide Web*. New York, NY, USA: Association for Computing Machinery, 2007. (WWW '07), p. 697–706. ISBN 9781595936547. Available at: <<https://doi.org/10.1145/1242572.1242667>>.
- TANON, T. P.; WEIKUM, G.; SUCHANEK, F. Yago 4: A reason-able knowledge base. In: SPRINGER. *The Semantic Web: 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31–June 4, 2020, Proceedings 17*. [S.l.], 2020. p. 583–596.
- TITZE, G.; BRYL, V.; ZIRN, C.; PONZETTO, S. P. DBpedia domains: augmenting DBpedia with domain information. In: *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. Reykjavik, Iceland: European Language Resources Association (ELRA), 2014. p. 1438–1442. Available at: <[http://www.lrec-conf.org/proceedings/lrec2014/pdf/233\\_Paper.pdf](http://www.lrec-conf.org/proceedings/lrec2014/pdf/233_Paper.pdf)>.
- VIVALDI, J.; RODRÍGUEZ, H. Finding domain terms using wikipedia. In: *LREC*. [S.l.: s.n.], 2010.

- VRANDEČIĆ, D.; KRÖTZSCH, M. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, ACM New York, NY, USA, v. 57, n. 10, p. 78–85, 2014.
- WANG, L.; ZHANG, S.; SHI, J.; JIAO, L.; HASSANZADEH, O.; ZOU, J.; WANGZ, C. Schema management for document stores. *Proc. VLDB Endow.*, VLDB Endowment, v. 8, n. 9, p. 922–933, May 2015. ISSN 2150-8097. Available at: <<https://doi.org/10.14778/2777598.2777601>>.
- WECHEL, K.; LEWONIEWSKI, W. Modelling the quality of attributes in wikipedia infoboxes. In: ABRAMOWICZ, W. (Ed.). *Business Information Systems Workshops*. Cham: Springer International Publishing, 2015. p. 308–320.
- WEISE, M.; LOHMANN, S.; HAAG, F. Ld-vowl: Extracting and visualizing schema information for linked data. In: *2nd international workshop on visualization and interaction for ontologies and linked data*. [S.l.: s.n.], 2016. p. 120–127.
- WIEMER-HASTINGS, P. *Latent Semantic Analysis*. 2006.
- WU, F.; WELD, D. S. Automatically refining the wikipedia infobox ontology. In: *Proceedings of the 17th International Conference on World Wide Web*. New York, NY, USA: Association for Computing Machinery, 2008. (WWW '08), p. 635–644. ISBN 9781605580852. Available at: <<https://doi.org/10.1145/1367497.1367583>>.
- XU, F.; LI, M.; WU, S.; HUANG, Q.; YAN, K.; WANG, M. Exploring hierarchical language knowledge in graph neural networks for fake news detection. In: IEEE. *2022 International Conference on Computer Engineering and Artificial Intelligence (ICCEAI)*. [S.l.], 2022. p. 646–650.
- XU, W.; GAO, X.; SHENG, Y.; CHEN, G. Recommendation system with reasoning path based on dqn and knowledge graph. In: IEEE. *2021 15th International Conference on Ubiquitous Information Management and Communication (IMCOM)*. [S.l.], 2021. p. 1–8.
- Xu, X.; Li, Q.; Dong, Y.; Ding, Y. Dynamically constructing a global schema for web entities. In: *2010 Seventh Web Information Systems and Applications Conference*. [S.l.: s.n.], 2010. p. 127–131.
- YAMADA, I.; SHINDO, H.; TAKEDA, H.; TAKEFUJI, Y. *Joint Learning of the Embedding of Words and Entities for Named Entity Disambiguation*. 2016.
- YAO, C.; YU, Y.; SHOU, S.; LI, X. Towards a global schema for web entities. In: *Proceedings of the 17th International Conference on World Wide Web*. New York, NY, USA: Association for Computing Machinery, 2008. (WWW '08), p. 999–1008. ISBN 9781605580852. Available at: <<https://doi.org/10.1145/1367497.1367632>>.
- YIH, S. W.-t.; CHANG, M.-W.; HE, X.; GAO, J. Semantic parsing via staged query graph generation: Question answering with knowledge base. 2015.
- ZEPEDA-MENDOZA, M. L.; RESENDIS-ANTONIO, O. Hierarchical agglomerative clustering. In: \_\_\_\_\_. *Encyclopedia of Systems Biology*. New York, NY: Springer New York, 2013. p. 886–887. ISBN 978-1-4419-9863-7. Available at: <[https://doi.org/10.1007/978-1-4419-9863-7\\_1371](https://doi.org/10.1007/978-1-4419-9863-7_1371)>.

ZHANG, J.; LUO, Y. Degree centrality, betweenness centrality, and closeness centrality in social network. In: *Proceedings of the 2017 2nd International Conference on Modelling, Simulation and Applied Mathematics (MSAM2017)*. [S.l.: s.n.], 2017. v. 132, p. 300–303.

ZHANG, X.; MENG, M.; SUN, X.; BAI, Y. Factqa: question answering over domain knowledge graph based on two-level query expansion. *Data Technologies and Applications*, Emerald Publishing Limited, v. 54, n. 1, p. 34–63, 2020.

ZHOU, J.; LIU, L.; WEI, W.; FAN, J. Network representation learning: From preprocessing, feature extraction to node embedding. *ACM Comput. Surv.*, Association for Computing Machinery, New York, NY, USA, v. 55, n. 2, jan 2022. ISSN 0360-0300. Available at: <<https://doi.org/10.1145/3491206>>.

ZOUAQ, A.; MARTEL, F. What is the schema of your knowledge graph? leveraging knowledge graph embeddings and clustering for expressive taxonomy learning. In: *Proceedings of The International Workshop on Semantic Big Data*. New York, NY, USA: Association for Computing Machinery, 2020. (SBD '20). ISBN 9781450379748. Available at: <<https://doi.org/10.1145/3391274.3393637>>.

## APPENDIX A – DATASET STATISTICS

Tables 10, 11, 12 and 13 present statistics of the data used in class schema discovery experiment (see Section 5.3). The Entity Schemas column indicates the number of entities (and schemas) belonging to each class. The Attributes column shows the number of distinct attributes contained in the entities’ schemas.

Table 10 – Dataset statistics - Oil and gas domain.

Class	Entity Schemas	Attributes
Company	304	171
Field	196	55
Pipeline	45	43
Businessperson	138	105
Refinery	64	59
Ship	95	99
<b>Source:</b> Created by the author		

Table 11 – Dataset statistics - Formula one domain.

Class	Entity Schemas	Attributes
People	98	105
Race	168	62
Game	78	33
Driver	115	96
Circuit	64	62
Team	43	127
Manufactures	32	90
Car	146	55
<b>Source:</b> Created by the author		

Table 12 – Dataset statistics - Martial arts domain.

Class	Entity Schemas	Attributes
Boxer	96	51
Event	70	164
Fencer	86	26
Film	83	49
Game	45	17
Judoka	64	50
Mixedmartial	122	162
Serie	53	133
Taekwondo	33	36
Wrestler	78	90
Sumo	52	42

**Source:** Created by the author

Table 13 – Dataset statistics - Tourism domain.

Class	Entity Schemas	Attributes
Festival	54	63
Art Gallery	46	34
Historic District	127	155
Lighthouse	40	25
Skyscrapers	69	79
Museum	57	35
Protected Area	98	61
Mall	60	38
Zoo	24	16
Sport Venue	105	298
Palace	47	74
Theatre	45	58

**Source:** Created by the author

## APPENDIX B – SCHEMA REFERENCE

Tables 14, 15, 16 and 17 present information about the reference schema used in experiment 2 (see Section 5.3). The Template column indicates the used template’s name, and the Attribute column shows the number of attributes contained in the template. Infobox templates are available at [<https://en.wikipedia.org/wiki/Template:\(name\)>](https://en.wikipedia.org/wiki/Template:(name)), replacing (name) with the template name.

Table 14 – Schema reference information - Oil and gas domain (\*short version).

Class	Template	Attributes
Company	Infobox_company*	19
Field	Infobox_oilfield	37
Pipeline	Infobox_pipeline	32
Businessperson	Infobox_person*	12
Refinery	Infobox_oilrefinery	14
Ship	Infobox_ship	63

**Source:** Created by the author

Table 15 – Schema reference information - Formula one domain.

Class	Template	Attributes
People	Infobox_person	12
Race	Infobox_grand_prix_race_report	35
Game	Infobox_videogame	17
Driver	Infobox_F1_driver	23
Circuit	Infobox_motorsport_venue	27
Team	Infobox_F1_team	16
Manufactures	Infobox_F1_engine_manufacturer	17
Car	Infobox_racing_car	46

**Source:** Created by the author

Table 16 – Schema reference information - Martial arts domain.

<b>Class</b>	<b>Template</b>	<b>Attributes</b>
Boxer	Infobox_boxer	20
Event	Infobox MMA event	12
Fencer	Infobox_fencer	27
Film	Infobox_film	20
Game	Infobox_videogame	17
Judoka	Infobox_judoka	50
Mixedmartial	Infobox_martial_artist	64
Serie	Infobox television	44
Taekwondo	Infobox_sportperson	60
Wrestler	Infobox_professional_wrestler	22
Sumo	Infobox_sumo_wrestler	21

**Source:** Created by the author

Table 17 – Schema reference information - Tourism domain.

<b>Class</b>	<b>Template</b>	<b>Attributes</b>
Festival	Infobox_festival_film	31
Art Gallery	Infobox_museum	27
Historic District	Infobox_NRHP	13
Lighthouse	Infobox_lighthouse	25
Skyscrapers	Infobox_building	77
Museum	Infobox_museum	27
Protected Area	Infobox_protected_area_(basic)	22
Mall	Infobox_shopping_mall	17
Zoo	Infobox_zoo	21
Sport Venue	Infobox_venue	47
Palace	Infobox building	77
Theatre	Infobox_theatre	47

**Source:** Created by the author