applied sciences

MDPI

Article

Understanding Horizon 2020 Data: A Knowledge Graph-Based Approach

Nikolaos Giarelis * and Nikos Karacapilidis

Industrial Management and Information Systems Lab, MEAD, University of Patras, 26504 Patras, Greece; karacap@upatras.gr

* Correspondence: giarelis@ceid.upatras.gr

Abstract: This paper aims to meaningfully analyse the Horizon 2020 data existing in the CORDIS repository of EU, and accordingly offer evidence and insights to aid organizations in the formulation of consortia that will prepare and submit winning research proposals to forthcoming calls. The analysis is performed on aggregated data concerning 32,090 funded projects, 34,295 organizations participated in them, and 87,067 public deliverables produced. The modelling of data is performed through a knowledge graph-based approach, aiming to semantically capture existing relationships and reveal hidden information. The main contribution of this work lies in the proper utilization and orchestration of keyphrase extraction and named entity recognition models, together with meaningful graph analytics on top of an efficient graph database. The proposed approach enables users to ask complex questions about the interconnection of various entities related to previously funded research projects. A set of representative queries demonstrating our data representation and analysis approach are given at the end of the paper.

Keywords: Horizon 2020; Natural Language Processing; text mining; knowledge graph; graph analytics; unsupervised learning; keyphrase extraction; named entity recognition

Citation: Giarelis, N.; Karacapilidis, N. Understanding Horizon 2020 Data: A Knowledge Graph-Based Approach. *Appl. Sci.* **2021**, *11*, 11425. https://doi.org/10.3390/app112311425

Academic Editors: Agostino Forestiero and Federico Divina

Received: 15 September 2021
Accepted: 30 November 2021
Published: 2 December 2021

1. Introduction

Horizon 2020 (H2020) is a recently completed EU funding programme for research and innovation, which was running from 2014 to 2020 with a €80 billion budget (https://ec.europa.eu/programmes/horizon2020/ (accessed on 27 November 2021)). Its main aim was to ensure research and innovation funding for multi-national collaboration projects as well as for individual researchers and SMEs. A wealth of data about funded H2020 projects can be retrieved through CORDIS (Community Research and Development Information Service, https://cordis.europa.eu (accessed on 27 November 2021)), the major public repository and portal of European Commission, which offers diverse information on EU-funded research projects and their results. Specifically, CORDIS provides valuable information about the projects' factsheets and results, public reports and deliverables, communication and exploitation material, as well as links to related external sources (such as websites and open access content).

The problem of finding the most appropriate call to prepare and eventually submit a research proposal, as well as that of finding the most promising partners to collaborate with in such endeavors, while also taking into account the required complementarity of competences, certainly requires a large amount of manual inspection of both the aforementioned wealth of data and the hundreds of associated documents corresponding to call descriptions and deliverables already produced in the context of the H2020 programme. To properly address this issue, we perform a deep analysis of all these data and develop an approach that enables users to obtain answers and insights about the issues under consideration. Specifically, we represent, analyze, and understand the H2020 data existing in the CORDIS repository (https://data.europa.eu/data/datasets/cordish2020projects (accessed on 27 November 2021)), and accordingly build on robust Machine Learning (ML),

Natural Language Processing (NLP) methods and tools, as well as a graph database, to perform state-of-the-art text mining and graph data analytics.

The proposed methodology seamlessly integrates text mining techniques with graph theory to enable users to ask questions about the underlying data. A comparative assessment of the proposed methodology against existing ones, which is presented in detail in Section 2.4, reveals that it does not merely provide visual statistical aggregations of data, or keyphrases extracted from multiple documents; instead, it builds on prominent text mining techniques to retrieve information about the topics of H2020 projects, thus providing insights about latest research trends and potential collaborators, it constructs and exploits a semantically rich knowledge graph that meaningfully models all the connections between data and enables the users to perform diverse types of queries, it utilizes hybrid keyphrase extraction techniques to capture more context through extracting keyphrases consisting of multiple terms, and it is fully unsupervised.

Motivated by the fact that the preparation of a winning research proposal is a complex and intense task, our aim was to extract meaningful new knowledge and insights from the H2020 related data, and accordingly provide the evidence that organizations need in this task. Specifically, during the preparation of their future research proposals (for instance, proposals to be submitted in the recently launched calls of the Horizon Europe funding programme), organizations need to follow specific steps including the assessment of the innovative character of the idea and research to be funded, the identification of the most relevant call/programme within a funding framework, the check of diverse eligibility criteria, and the building of the appropriate consortium by making clear the role and capacity of each participant.

For instance, when an ICT company specializing in AI-enhanced solutions, such as chatbots, etc., considers the preparation of a research proposal that aims to develop a beyond the state-of-the-art personal assistant for medical decision-making issues, it needs to obtain answers to queries such as:

- *What are the top-10 organizations that have coordinated (or participated in) the most health-related projects of H2020?*
- *What are the topics of interest/expertise for each of these top-10 organizations that have coordinated (or participated in) the most health-related projects of H2020?*
- *For a given list of topics (e.g., Medical Novision Making, Natural Language Processing, Chatbot, Machine Learning, Medical Guidelines, Policy Making), which deliverables produced by these top-10 organizations were related to pilot applications?*
- *For the list of topics above, what types of deliverables have been produced by these top-10 organizations?*
- *Which people from these top-10 organizations have been working on this list of topics (by preparing the corresponding deliverables)? Have they also been working on additional ones?*

Our analysis is performed on aggregated data concerning 32,090 funded projects, 34,295 organizations participated in them, and 87,067 public deliverables produced by these organizations. The source of these data is two CORDIS files, namely *cordis-h2020projects.csv* and *cordis-h2020projectDeliverables.csv*. The first file contains the following information for each project: Record Control Number (RCN), the project ID (grant agreement number), acronym, status, start and end date, and its objective and total cost; additionally, it contains information about the following: funding scheme and programme, topic, EC contribution, call ID, coordinator, and participants (with their countries). The second file contains information about the deliverables, including a public URL to access the full content of each of them.

The remainder of this paper is organized as follows. Section 2 introduces related work aiming to give the reader some basic information about the components of the proposed approach. Section 3 presents in detail the proposed pipeline towards meaningfully representing and analyzing the Horizon 2020 data; a set of representative queries is used to demonstrate the corresponding data visualization features. Finally, Section 4 presents preliminary evaluation results and sketches future work directions.

2. Background

The approach described in this paper builds on tools and methods traditionally met in the fields of knowledge graphs, unsupervised keyphrase extraction (KE), and named entity recognition (NER).

2.1. Knowledge Graphs

In recent years, we have witnessed an increasing popularization of knowledge graphs, which are used for a variety of tasks ranging from semantic parsing to question answering. Knowledge graph databases include Freebase [1] and DBPedia [2], which focus on data and facts extracted from online public datasets and Wikipedia, respectively.

Despite their popularity, there is no universal definition for knowledge graphs. The authors in [3] gathered multiple definitions appearing in the literature and clarified a set of terminology issues that stem from these different definitions. For the purpose of this paper, we adopted the knowledge graph definition given in [4]; according to it, a knowledge graph (i) mainly describes real world entities and their interrelations, organized in a graph; (ii) defines possible classes and relations of entities in a schema; (iii) allows for potentially interrelating arbitrary entities with each other, and (iv) covers various topical domains.

2.2. Keyphrase Extraction

YAKE! [5] is a statistical unsupervised KE approach. It splits a document into distinct terms, where for each term t a score $S(t)$ is calculated. $S(t)$ is computed using five term related metrics: *Trel* (term relatedness to context, which measures the number of distinct terms that appear on the left and right side of t.), *Tpos* (the positional of t, which assigns a higher score to terms appearing closer to the beginning of the document.), *Tcase* (casing aspect of t, which favours uppercase and terms having their first letter capitalized, minus those located at the starting point of a sentence.), *TFnorm* (term frequency normalization), *Trel* (term relatedness to context, which computes the number of different terms that occur on the left and right side of the term), and *Tdifsent* (which measures the number of times t appears in different sentences). Formally $S(t)$ is defined as:

$$S(t) = \frac{T_{rel} * T_{pos}}{T_{case} + \frac{TF_{norm}}{T_{rel}} + \frac{T_{difsent}}{T_{rel}}} \quad (1)$$

After the calculation of each score for every term, n-grams of candidate keyphrases (*ck*) are formed. For each *ck*, a score $S(ck)$ is computed. The lower the value of this score, the higher the quality of the *ck* is.

$$S(ck) = \frac{\prod_{t \in ck} S(t)}{TF(ck) * (1 + \sum_{t \in ck} S(t))} \quad (2)$$

TextRank [6] is one of the most notable graph-based KE approaches. For each term in the document, it assigns part-of-speech (POS) tags, as to only include nouns and adjectives for its list of candidate keyphrases. For the document, a graph is initialized. Each *ck* is represented as a node in the graph. Edges are formed between terms that co-occur in a sliding window of N. If the edges are undirected and unweighted, the *TextRank* score $S(vi)$ for each node *vi* is calculated by the following formula:

$$S(vi) = (1 - d) + d \times \sum_{v_j \in \Gamma(v_i)} \frac{1}{|\Gamma(v_j)|} S(vj) \quad (3)$$

where d is the damping factor, usually set to 0.85 as indicated in [7] and $\Gamma(vj)$ is the set of neighbouring nodes of *vj*. When formula (3) converges, the nodes are sorted in a descending order of their calculated scores, and the top-n *ck* are returned.

SingleRank [8] is a similar approach to *TextRank*, being different in three key aspects [7]. First, the weighted version of *TextRank* uses the same pre-defined weight for each edge,

whereas *SingleRank* uses the number of times the connected terms co-occurred together in the sliding window of *N* terms. Second, *SingleRank* considers both highest ranking and lowest ranking terms in the *ck* forming process, while *TextRank* only employs the former. This has the effect of a *ck* being ranked as a sum of the ranking of all terms in it. The resulting score is then used in descending order to obtain the top-N highest scored *cks*. Third, *SingleRank* uses a larger window size (e.g., 10), in contrast to a smaller window sizes employed by *TextRank* (with 2 being the minimum one).

The mathematical formulation of the weighted *SingleRank* score $WS(v_i)$ is nearly the same as the weighted *TextRank* score. The two major differences are: (i) the weight of an edge between two nodes v_i and v_j is set to the number of cooccurrences c_{ij}; (ii) instead of dividing the weight of an edge by $\left|\Gamma(v_j)\right|$, it is instead divided by $\sum_{v_k \in \Gamma(v_j)} c_{jk}$ which is the sum of the number of co-occurrences between v_j and each of its neighboring nodes $\Gamma(v_j)$. The weighted SingleRank score $WS(v_i)$ is calculated as:

$$WS(vi) = (1-d) + d \times \sum_{v_j \in \Gamma(v_i)} \frac{c_{ij}}{\sum_{v_k \in \Gamma(v_j)} c_{jk}} WS(vj) \quad (4)$$

As stated in [9], graph-based approaches such as *TextRank* and *SingleRank* are performing well in terms of extracting important keyphrases for short descriptive texts, such as descriptions of projects, which is rather useful for the context of examining H2020 projects. In a few cases, we manually observed that the statistical approach of *YAKE!* is also producing some interesting keyphrases, despite being a statistical approach, which relies on co-occurrence of terms found in longer texts, in order to produce accurate keyphrases.

Recent literature reviews [10,11] have deeply analyzed the pros and cons of the aforementioned approaches, while also including several other state-of-the-art unsupervised keyphrase methods that apart from statistical or graph-based measures rely on external knowledge bases (e.g., Wikipedia) to extract topics or even utilize word embedding vectors.

2.3. Named Entity Recognition

Named Entity Recognition (NER) is a fundamental subtask of NLP. By definition, it enables the extraction of phrases called '*entities*' from unstructured textual data, which are detected and classified into specific pre-determined categories (e.g., '*Person*', '*Location*', '*Organization*', etc.). Many NLP libraries have been developed so far to address this subtask. A few notable examples include: GATE [12], NLTK [13], spaCy (https://spacy.io (accessed on 27 November 2021)), StanfordNLP [14], Flair [15] and Stanza [16]. In terms of employed approaches, the libraries in question vary from rule-based (GATE, NLTK), statistical neural network based (spaCy), to deep learning based ones (Stanza, Flair, spaCy-v3-trf). Rule-based approaches, rely on carefully hand-crafted grammar-based rules, and a lot of work done by human linguistic experts, whereas statistical ones require large numbers of manually annotated data [17].

The authors in [18] compare notable NLP frameworks from each approach, excluding the deep learning ones. In their experiments, it is shown that for the subtask of NER, StanfordNLP achieves the best performance in terms of F1 score, whereas spaCy-v2 has the second best performance. After this study was published, the Stanford NLP Research group released Stanza, and the authors of spaCy released the third version of their software; both libraries came with better pretrained models to the best of our knowledge, compared to their predecessors.

The authors in [16] evaluated deep learning approaches (Stanza, Flair) that achieve higher accuracies than existing methods; however, those suffer from severely increased execution runtime, compared to the pretrained convolutional neural network approach of spaCy-v3. These results are also confirmed by the authors of spaCy in their benchmarks (https://spacy.io/usage/facts-figures#benchmarks (accessed on 27 November 2021)). Recently, the creators of spaCy introduced a new pretrained transformer English model (*en_core_web_trf*), which reaches the state-of-the-art accuracy of deep learning approaches.

From the aforementioned two studies, we observed that there is an execution runtime versus accuracy of results trade-off, which influences the choice between different NLP libraries.

2.4. Analyzing CORDIS and Other Textual Datasets

We have identified a few recent works in the literature that adopt alternative approaches to analyze the CORDIS dataset. In the work described in [19], authors apply graph social networks analysis using various measures to discover insights about countries/organizations that have acquired funding from the European Commission for projects related to green energy and sustainable technologies. Although their approach showcases interesting graph measures and results, it does not utilize any NLP techniques to retrieve information about the topics of these projects, as is the case in the approach described in this paper; this information would, for instance, provide insights about latest research trends and potential collaborators.

By also using CORDIS data, the work presented in [20] builds on a known statistical keyphrase extraction algorithm for candidate keyphrases, along with sentence similarity word embeddings, to provide a list of insights and challenges upon the user input. However, this approach does not allow one to find similar organizations or deliverables or persons to collaborate by considering a list of topics specified by the user. On the contrary, the approach proposed in this paper builds on a semantically rich knowledge graph, which meaningfully models all the connections between data, and enables users to pose such questions.

The Corpus Viewer System [21] adopts an approach that is very similar to the work described in this paper; it utilizes various NLP practices to construct a knowledge graph of projects funded by the European Commission and their associated topics. Nonetheless, there are some major differences: (i) their approach relies on topic modelling, where each topic consists of a single term, whereas our approach utilizes keyphrase extraction to capture more context through extracting keyphrases consisting of multiple terms; (ii) their approach is not fully unsupervised, since it relies on pre-supplied taxonomies as training data, in order to perform text classification among the documents found in the dataset; (iii) apart from graph visualizations, their tool offers geographical visualizations for some pre-defined questions, which rely on aggregating statistical results from the CORDIS dataset.

The work described in [22] proposes a document summarization and visualization framework based on both statistical and semantic analysis of textual and visual contents, aiming to highlight relevant terms in a document using some features (such as font size, color, etc.) showing the importance of a term compared to other ones. This framework builds on the combination of textual and visual features, which are stored in the Neo4j graph database, to improve the user knowledge acquisition by means of a synthesized visualization. This work demonstrates that with the help of semantic analysis and the combination of textual and visual features it is possible to improve the user knowledge acquisition by means of a synthesized visualization. However, compared to our approach, it does not build a knowledge graph to enable the user performing diverse types of queries and visualizing their results.

OLGAVIS (On-Line Graph Analysis and Visualization for Bibliographic Information Network) [23] is an approach for visualizing connections in research bibliography. It builds a research collaboration knowledge graph where publications, venues, authors, and their affiliations are modelled using the Neo4j graph database. This approach also analyzes the data using various graph queries, and then forwards these results into the visualization layer. This layer allows users to ask questions about the data, and then filter or group or aggregate results. Compared to our approach, OLGAVIS provides more sophisticated visualization functionalities; however, it does not utilize any text mining method to extract topics related to the various entities of the knowledge graph.

Finally, Metaphactory [24] is an enterprise knowledge graph platform, which enables users to build and visualize knowledge graphs from multiple sources. These sources

include various databases, analytics platforms, and open standards (e.g., RDF, SPARQL etc.) To store the knowledge graph, it utilizes a graph database. This platform offers multiple views of the knowledge graphs, as well as diverse graph filtering functionalities. Compared to our approach, Metaphactory provides a richer and easier customizable user interface; however, as is the case with OLGAVIS, it does not utilize any text mining method to extract topics related to the various entities of the knowledge graph.

Table 1 summarizes and provides a comparative assessment of the key characteristics of the aforementioned works against the one described in this paper.

Table 1. Overview of key characteristics of various approaches.

Approach	Graph Database	Knowledge Graph	Text Mining Method	Word Embeddings	Fully Unsupervised	Data Visualization
[19]	-	-	-	-	✓	✓
[20]	-	-	Keyphrase Extraction	✓	-	-
[21]	-	✓	Topic Modelling	-	-	✓
[22]	✓	-	Topic Modelling	-	-	✓
[23]	✓	✓	-	-	✓	✓
[24]	✓	✓	-	-	✓	✓
The proposed approach	✓	✓	Keyphrase Extraction	-	✓	✓

3. The Proposed Approach

This section describes in detail the proposed approach towards representing and analyzing the Horizon 2020 data. The novelty of our work lies in the utilization and orchestration of keyphrase extraction and named entity recognition models, together with meaningful graph analytics on top of a highly performant graph database, to inform the process of preparation of research proposals for funding acquisition. The proposed approach enables users to ask complex questions about the organizations, projects, deliverables, topics, and people involved in already funded research projects, as well as about the way that all these entities are meaningfully interconnected. This is an advantage over previous approaches that merely analyze projects or organizations via a list of topics or via a statistical distribution of funds or projects over countries or research topics.

As mentioned in prominent works (e.g., [25–27]), there are multiple reasons to opt for a graph database over a relational database (e.g., SQL). The two basic ones are that: (i) joins are expensive, and therefore, reasoning about a graph's paths becomes very costly; (ii) global properties are no easy to compute with classical queries based on logical methods [25]. In addition, graph databases allow our graph model to be stored natively. Other analytic tools such as Apache Spark store graph representations in a relational schema by converting the graph model into tables connected by multiple foreign keys. After this complex graph representation is built, these tools employ multiple JOIN statements, which lead to memory bottlenecks and poor performance. On the contrary, graph databases, and in our case Neo4j, provide a scalable solution that allows multiple hops across billions of nodes and edges with increased performance. Furthermore, thanks to the flexible schema nature of graph databases, we can store heterogeneous node types containing different types of data, thus unifying diverse data sources into a common model. Finally, graph databases permit us to easily adapt the model if and whenever needed, without the additional re-modelling required in traditional approaches.

As illustrated in Figure 1, we propose a pipeline approach comprising four basic tasks, namely (i) data collection (Section 3.2), (ii) unsupervised keyphrase extraction and person extraction (Section 3.3), (iii) knowledge graph construction (Section 3.4), and (iv) graph data analysis and visualization (Section 3.5). It is noted here that our overall approach is presented in pseudocode form in Appendix A, Algorithm A1.

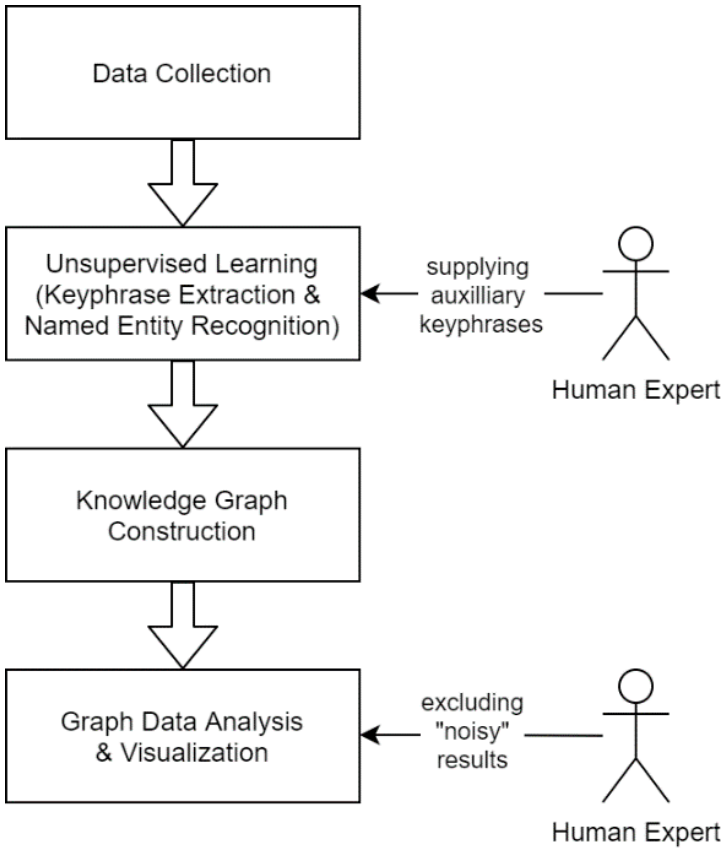

Figure 1. Overview of the proposed approach.

3.1. Technical Information

This subsection contains information about the software and hardware specifications that were utilized in the work described in this paper. Regarding our hardware specifications, we utilized a PC with an Intel Core i9 CPU (20 threads), and a base and max clock speed of 3.70 Ghz and 5.00 Ghz, respectively. In terms of RAM, we used 2 dual in-line memory modules, each having a size of 32 GB.

For the implementation of the proposed approach, we used the Python programming language and the Neo4j database to store our knowledge graph. We also used the following keyphrase extraction libraries: *YAKE!* (https://github.com/LIAAD/yake (accessed on 27 November 2021)), *TextRank* (https://github.com/DerwenAI/pytextrank (accessed on 27 November 2021)) and *SingleRank* (https://github.com/boudinfl/pke (accessed on 27 November 2021)). To create sublists of overlapping keyphrases obtained from the above approaches, we used the *get_close_matches*() method of the *difflib* built-in python library (https://docs.python.org/3/library/difflib.html#difflib.get_close_matches (accessed on 27 November 2021). For the implementation of the suffix tree, which is used to extract the longest common substring from each list of overlapping keyphrases, we used the *suffix-trees* library (https://pypi.org/project/suffix-trees/ (accessed on 27 November 2021)). For the NER task of extracting authors from text, we used spaCy v3 (https://spacy.io/ (accessed on 27 November 2021)) with a large pre-trained English model titled *en_core_web_lg* (https://spacy.io/models/en#en_core_web_lg (accessed on 27 November 2021)).

3.2. Data Collection

We retrieve the required data from the CORDIS repository (https://data.europa.eu/euodp/en/data/dataset/cordisH2020projects (accessed on 27 November 2021)). We mainly use two .csv files, which contain information about the accepted (funded) projects and their deliverables (the *cordis-h2020projects.csv* and *cordis-h2020projectDeliverables.csv*

files). These files are also used to generate some additional .csv files pertaining to our analysis, which contain keyphrases for the respective projects and deliverables. The second file contains URLs for the publicly available deliverables, which are large .pdf files containing mostly text. We automated the process of downloading these .pdf files, which are then converted to .txt files.

The text conversion process was successful for 87,067 out of 89,644 deliverables, mainly due to the existence of corrupted files. We considered only the first five pages of each .pdf file, by assuming that the information being sought about the authors of each deliverable can be found there (it is noted that this information is not included in the original .csv files considered). This assumption is justified by an extensive manual inspection of the format adopted by diverse projects in the context of Horizon 2020. After the five pages of each deliverable are extracted and saved in a .txt file, we use NER to extract the authors of the deliverables. The accuracy score of the NER task of the spaCy v3 model is 85.5% (as stated in https://spacy.io/usage/facts-figures#benchmarks (accessed on 27 November 2021)). Finally, from the unstructured text fields (e.g., fields about the description of a project or deliverable), we extract important keyphrases using a hybrid unsupervised keyphrase approach, described in the next section.

3.3. Unsupervised Learning

We propose a hybrid unsupervised approach called *HybridRank*, which combines several well-established unsupervised keyphrase extraction approaches, a list of auxiliary keyphrases given by human experts, and a suffix tree to reduce the duplication of keyphrases in the final list. We then utilize spaCy's NER component using the large pretrained English model to produce a .csv that contains the names of the authors of a certain deliverable.

Taking into account the particularities of previous keyphrase extraction approaches (outlined in Section 2.2), we propose a hybrid one that takes the union of the list of keyphrases produced by individual approaches, and returns an enriched list of keyphrases. To the best of our knowledge, no other works in the literature combine the lists of keyphrases produced by different approaches, in a meaningful way. Specifically, in the context of this work, we developed a keyphrase extraction approach that produces keyphrases in an unsupervised manner; we combine the top-10 keyphrases from each individual keyphrase extraction approach into a single list of keyphrases, which produces a maximum of top-30 keyphrases (in some cases, there were partial duplicates).

Consider an example of a document which mostly refers to the topic of Machine Learning. The final list could contain the following keyphrases: '*advanced machine learning approaches*', '*machine learning approaches*', and '*machine learning methods*'. Out of the similar keyphrases, our goal is to extract the common longest substring, which in this case would be '*machine learning*'. This is done in an attempt to summarize similar keywords in a single one, thus reducing the resulting keyphrase list.

From the list of keywords, we generate sub-lists of overlapping keyphrases, by utilizing the Gestalt pattern matching algorithm [28]. After the sublists are produced, we extract the longest common substring representation with the use of the homonym method of a suffix tree. As illustrated in Figure 2, a suffix tree can extract a common string representation from k strings inserted in the tree in $\Theta(n)$ time, where n is the number of nodes in the tree [29].

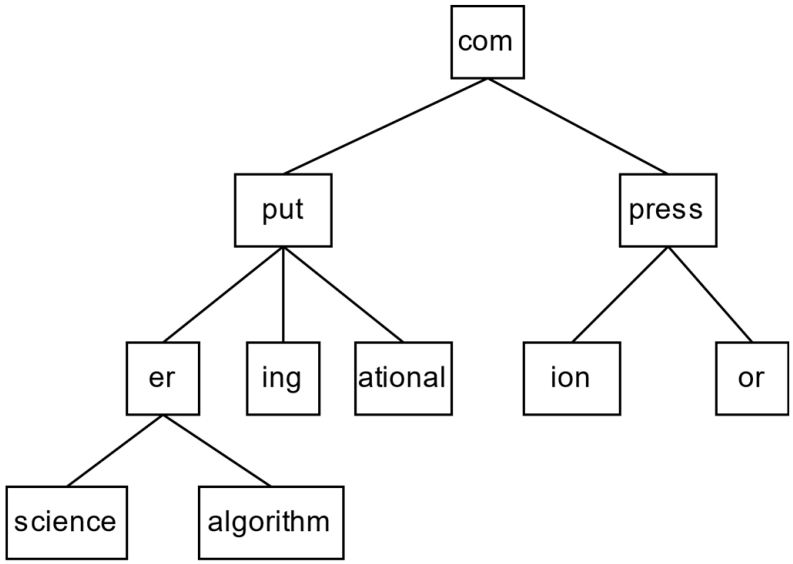

Figure 2. An example of a suffix tree.

Our overall approach is fully unsupervised; however, there are some problems with it. Specifically, some keywords are not extracted properly, or at-all, thus we came up with the idea of using an auxiliary list of keyphrases that are extracted as-is from the text. This list of keyphrases is manually supplied each time (upon the interest of the user) via a .txt file, which is written by human experts before the execution.

3.4. Knowledge Graph Construction

From the entities described in the previously mentioned .csv files, we devise the schema of our knowledge graph, which is shown in Figure 3. There are various types of nodes, such as '*Project*', '*Deliverable*', '*Person*', '*Organization*' and '*Keyphrase*'. Each '*Organization*' coordinates or participates in a certain '*Project*' as extracted from the '*coordinators*' and '*participants*' fields of the .csv. Each '*Project*' and '*Deliverable*' node include a '*Keyphrase*' node, which is going to be used to group similar nodes. Finally, a '*Deliverable*' belongs to a '*Project*' node, and a '*Person*' node writes a '*Deliverable*'. This allows us to associate the deliverables with the projects and the authors involved.

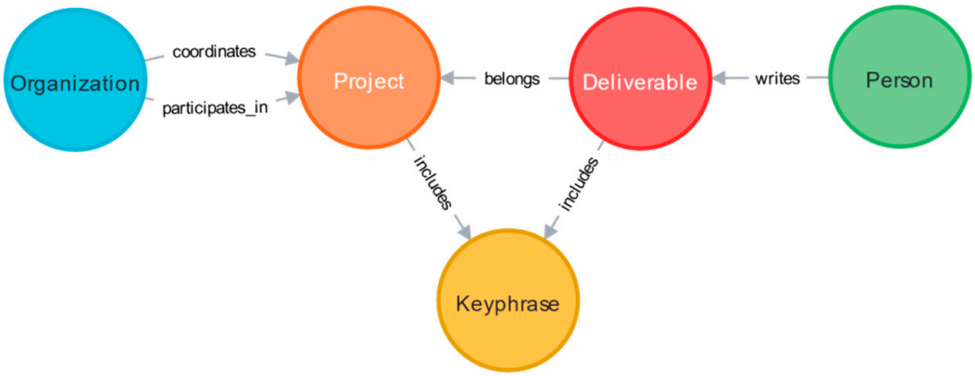

Figure 3. Schema of the knowledge graph.

These connections were extracted by the .csv files of the original dataset; in the case of authors writing a certain deliverable, they were extracted through text mining approaches from 87,067 deliverables. In terms of features, the nodes of our knowledge graph contain the features appearing in the .csv (their type, funding scheme, etc.). The only exception to this is the '*Keyphrase*' node, which only contains the keyphrase itself as a label. Each

keyphrase is modelled as a node, in order to facilitate the comparison of other nodes that are directly or indirectly linked to it.

3.5. Graph Data Analysis and Visualization

This section demonstrates features related to the analysis and visualization of our graph data through the following three representative queries:

- *Q1: What are the topics of interest for each of the top-n organizations that have coordinated (or participated in) the most projects of H2020?*
- *Q2: For a given list of topics, which deliverables were produced by the top-n organizations that have coordinated (or participated in) the most projects of H2020 were related to pilot applications?*
- *Q3: Which people have been working on a given list of topics? Have they been working on additional ones?*

To answer such questions in the Neo4j graph database, we employ the *Cypher* graph query language, due to its expressiveness and native support. The cypher queries that we implemented can be found in Appendix B. To visualize these queries, we can use the *Neo4j Browser tool*, due to its ease of use and nice visualization features; it is noted that the graph database can also be accessed programmatically, using a driver plugin from multiple programming languages (https://neo4j.com/developer/language-guides/ (accessed on 27 November 2021)). For large queries that return thousands of nodes and edges, we have also implemented a custom visualization tool, building on top of the neovis.js (accessed on 27 November 2021) graph visualization library, which has been also included in our GitHub repository (https://github.com/NC0DER/CORDISKG (accessed on 27 November 2021)). This tool bypasses common memory limitations imposed by the *Neo4j Browser tool* and has been used in the instances shown in Figures 4–6.

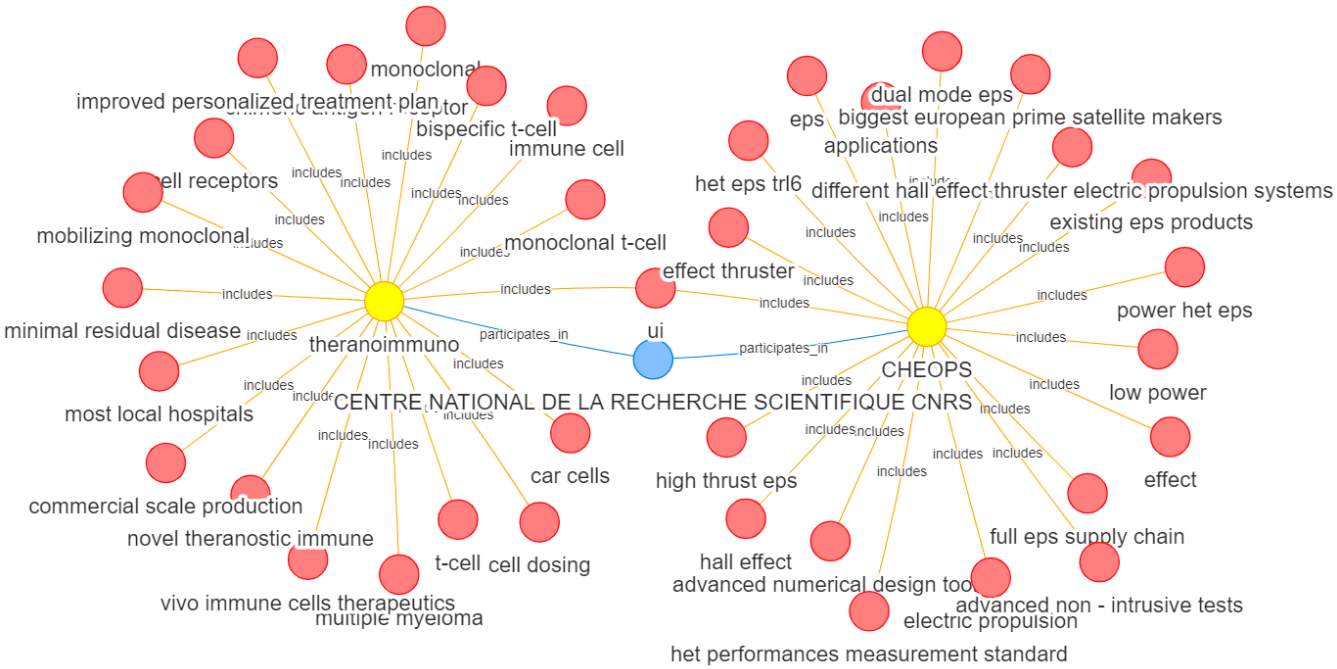

Figure 4. Visualization of a specific organization and some of its projects and respective keyphrases.

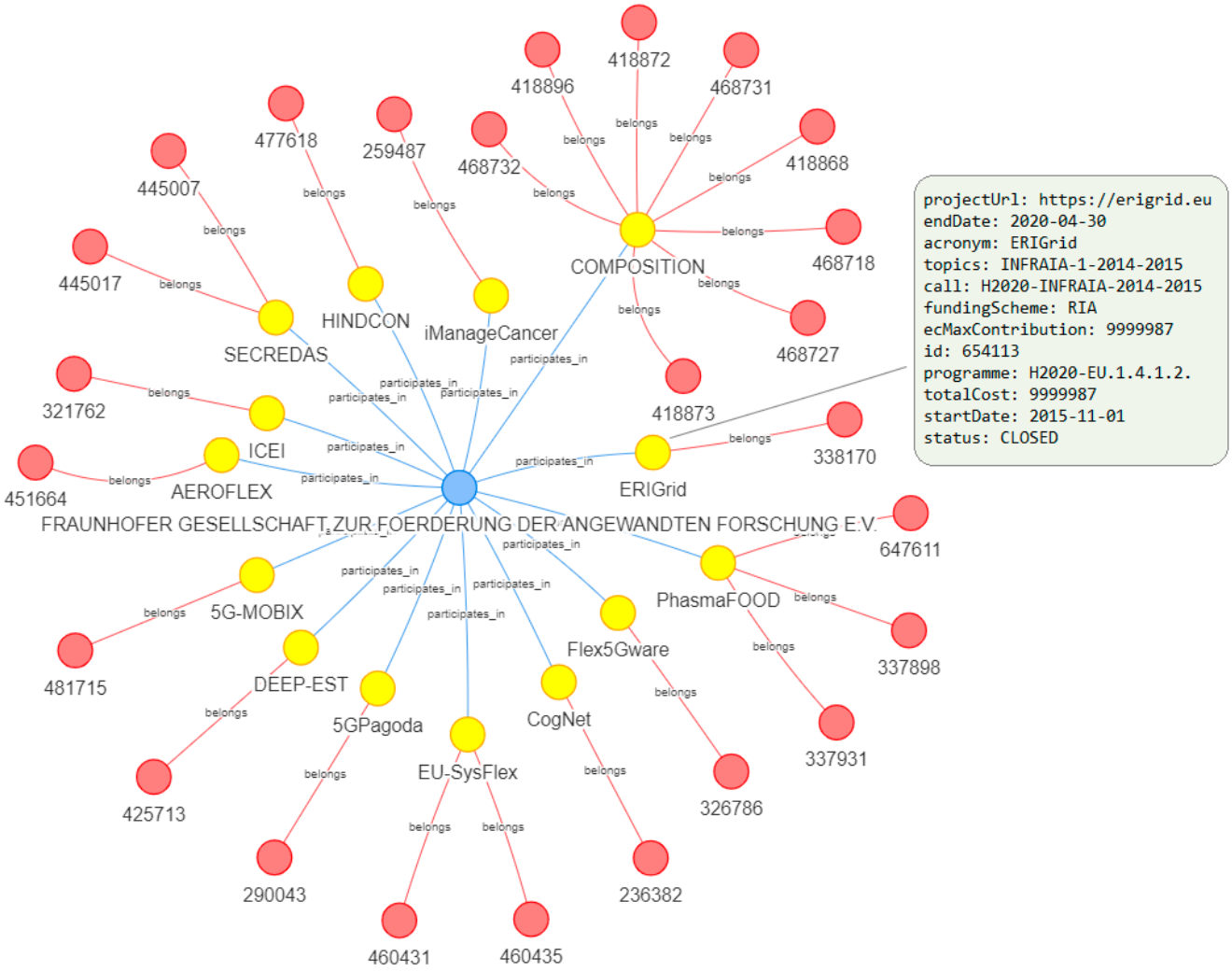

Figure 5. Visualization of a specific organization and some of its projects and respective deliverables.

Q1 is a common query raised when an organization is about to shape a project proposal and identify potential partners (especially, "big players") to collaborate with. It is noted that the results of a graph query can either be a subgraph of the original graph or a table of aggregated results formed from the matched graph path. Figure 4 illustrates an instance of the subgraph returned by queries of Q1 type, visually representing the topics that the top organization (i.e., the one that has coordinated or participated in the most projects) has been working on, together with the associated projects. It is noted that, due to the very large number of projects this organization has been working on, the graph of Figure 4 is just a small part of the one returned by Q1.

Q2 attempts to reveal information about the type of deliverables that were produced by organizations, which have participated in projects elaborating a certain list of topics. Particularly, Q2 focuses on deliverables reporting on the implementation (or evaluation) of a project's pilot applications (use cases). Figure 5 illustrates an instance of the subgraph returned by queries of Q2 type, visually representing the projects that one of the top organizations has been working on, together with the associated deliverables. As shown, whenever the user hovers over a node, detailed information about its properties and values appears. As above, due to space limitations, only a small part of the graph returned by Q2 is shown in this instance.

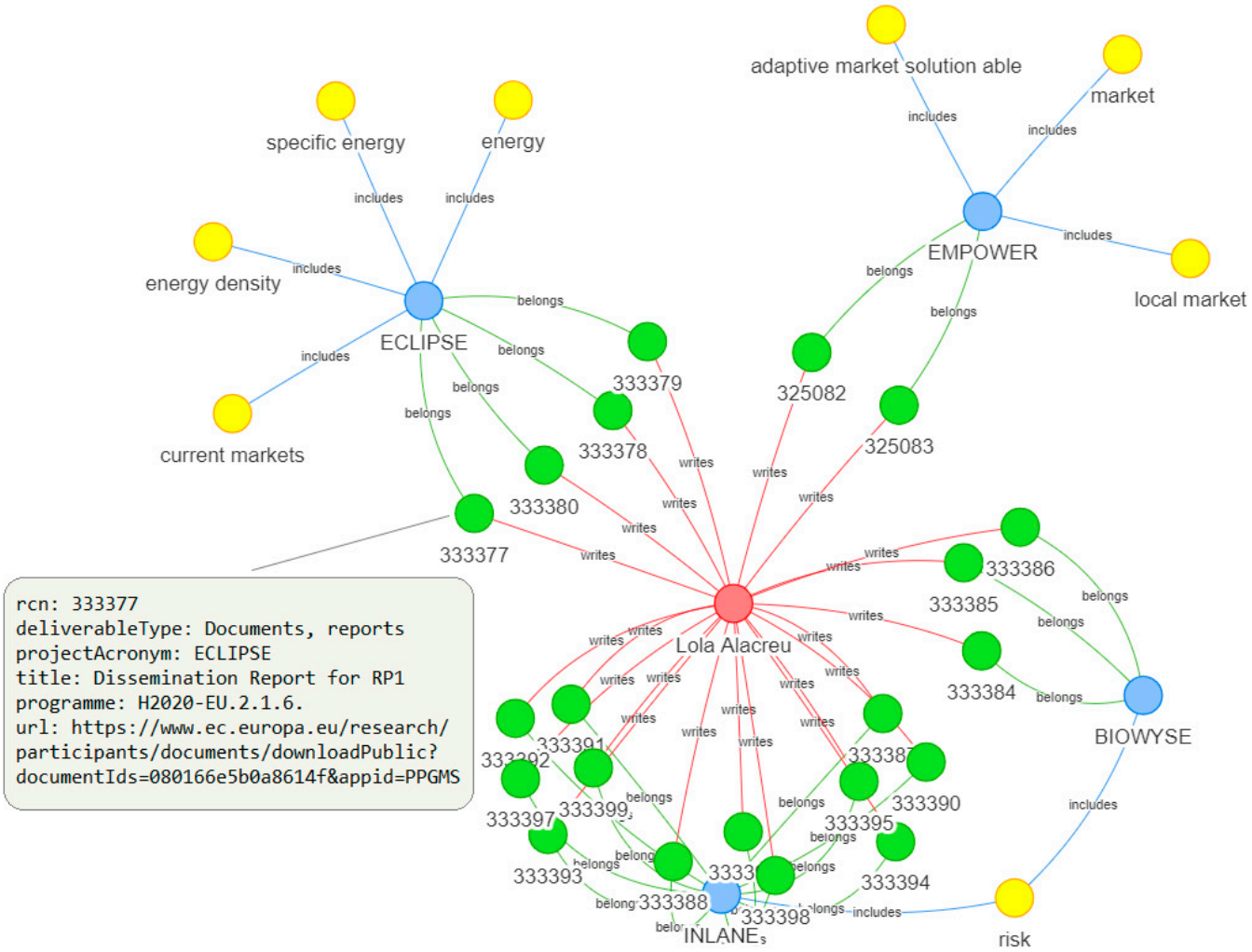

Figure 6. Visualization of a specific person and some of the associated projects, deliverables, and keyphrases.

Finally, Q3 is also a frequently asked question during the preparation of a research proposal, raised when one wants to identify colleagues with particular skills and competences that cover the needs of a call. Figure 6 illustrates an instance of the subgraph returned by queries of Q3 type, visually representing the deliverables that a person has been working on, together with the associated projects and topics. Once again, we have imposed a limitation on the number of nodes to be returned by the specific query.

4. Conclusions

Adopting a knowledge graph-based approach, the work described in this paper builds on prominent ML and NLP tools and methods to analyse the H2020 dataset appearing in the CORDIS repository. As described in detail in the previous section, the main contribution of this work lies in the proper utilization and orchestration of keyphrase extraction and named entity recognition models, together with meaningful graph analytics on top of an efficient graph database. The proposed approach enables users to ask complex questions about the interconnection of various entities related to previously funded research projects.

We argue that the proposed approach can reveal hidden knowledge, trigger insights, and accordingly extract evidence to aid organisations in the highly complex and knowledge-intense task of the formulation of future research project proposals and related consortia. As elaborated in detail in [25], the proposed approach builds on the dual character of graphs, i.e., on one hand, providing a simple, flexible, and extensible data structure, and on the other, being one of the most deep-rooted form of representing human knowledge. In addition, the proposed approach contributes to an important line of research that has to do with the interplay of graph analytics and machine learning, or computing statistics over graphs [26].

Nowadays, the proposed approach can be very useful for organisations that are getting prepared to submit their proposals to the recently launched calls of the Horizon Europe funding programme. It has already undergone a preliminary assessment and validation round, with the aid of five experienced practitioners in project proposals' building (from three different European organisations). The feedback collected was highly positive in terms of usefulness and ease-of-use. In any case, a future work direction concerns a carefully planned assessment, also aiming to capture additional queries of major importance. A second work direction concerns the elaboration of a deep learning approach, which would maximize the accuracy of the proposed approach's NER subtask and require less amount of post-processing effort by human experts. In this direction, we plan to assess and eventually integrate in our approach word embeddings-based clustering techniques that incorporate the textual content with latent feature vector representations of words appearing in the text to improve the quality of topic detection [30]. Aiming to augment the visualization of the generated results, a third work direction concerns the development of alternative views, such as the hierarchical tree-based view elaborated in [31] and the combined table-tree view of aggregated graph features described in [32]. Finally, a fourth work direction concerns the potential incorporation of approaches that aim to decrease the number of necessary queries and consequently lower the search time in the context under consideration [33].

Author Contributions: Conceptualization, N.G. and N.K.; methodology, N.G. and N.K.; software, N.G.; investigation, N.G.; resources, N.K.; data curation, N.G.; writing—original draft preparation, N.G.; writing—review and editing, N.G. and N.K.; visualization, N.G.; supervision, N.K.; project administration, N.K.; funding acquisition, N.K. All authors have read and agreed to the published version of the manuscript.

Funding: The work presented in this paper is supported by the inPOINT project (https://inpoint-project.eu/ (accessed on 27 November 2021)), which is co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH—CREATE—INNOVATE (Project id: T2EDK-04389).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The employed datasets and the code repository are available at https://drive.google.com/drive/folders/15mSjwYhsNQyrpTGUlxmBvXdsWEqEprvc?usp=sharing (accessed on 27 November 2021) and https://github.com/NC0DER/CORDISKG (accessed on 27 November 2021).

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Algorithm A1. Pseudocode of the Proposed Approach

Input: CORDIS H2020 Data
Output: Knowledge graph (Neo4j), Visualized Graph Data (Figures)

// Section 3.2: Data collection step.
pdfs ← download_pdfs_from(CORDIS H2020 Data)
save_pdfs_to_txts(pdfs, convert_pdfs_to_raw_text ← *True*, max_num_of_pages ← *5*,
output_dir ← *'deliverables'*)

people ← extract_people_using_NER_from(*'deliverables'*, entity_type ← *'Person'*)

// Section 3.3: Unsupervised learning step.
for row **in** CORDIS H2020 Data **do**
text ← row.description
results ← []

// Extract keyphrases from each method and auxiliary keyphrases.
keyphrases ← extract_keyphrases_from(
methods ← [*'YAKE'*, *'TextRank'*, *'SingleRank'*],
parameters ← [text, top_n ← *10*],
use_expert_ auxilliary_keyphrases ← *True*)

// For each keyphrase get a list of overlapping keyphrases.
for keyphrase **in** keyphrases **do**
overlapping ← get_close_matches(keyphrases, *'gestalt_pattern_matching'*)

// If there are overlapping keyphrases,
// then extract the longest common substring using the lcs() method.
if overlapping.length() > *1* **do**
tree ← Suffix_Tree.initialize_from_set({overlapping} ∪ {keyphrase})
results.append(tree.lcs())

// Section 3.4. Knowledge Graph Construction step.
database ← create_neo4j_database_instance(username, password)
create_project_subgraph(database)
create_deliverables_subgraph(database)
create_keyphrase_subgraph(database)
create_people_subgraph(database)

// Section 3.5. Graph Data Analysis and Visualization step.
visualize_subgraph_data_from_cypher(queries, output_path ← *'figures'*)

Appendix B

This appendix lists the Cypher queries that correspond to the questions Q1, Q2, and Q3 mentioned in Section 3.5. In the queries below, *top_n* (organizations, people etc.) is set to 100, *top_m* (topics) is set to 25, the number of deliverables is set to 10, and *profile* is supplied by the user as a list of topics (e.g., ['sustainability, 'carbon emissions', 'green technologies']). We used a profile of 86 topics related to e-government, banking, sustainable technologies, and cryptocurrency.

Q1: What are the topics of interest for each of the top-n organizations that have coordinated (or participated in) the most projects of H2020?

```
MATCH (o:Organization)-[:participates_in | coordinates]->(p:Project)
    WITH o.name AS Organization, COUNT(p) AS Amount
    ORDER BY Amount DESC LIMIT 100
    WITH COLLECT(Organization) AS orgs
```

UNWIND orgs *AS* org
MATCH (o:Organization {name: org})-[:participates_in | coordinates]->(p:Project)-[:includes]-(k:Keyphrase)
WHERE NOT k.name *IN* ['report', 'deliverable', 'task', 'project', 'document', 'available', 'terms']
WITH o.name *AS* Organization, k.name *AS* Topic, *COUNT(k) AS* NumberOfTopics
ORDER BY NumberOfTopics *DESC*
RETURN Organization, *COLLECT*([Topic, NumberOfTopics])[..25]

Performance: Execution time: 1279 ms, Memory used: 253.69 MB, Database hits: 5,373,808. For hardware specifications see Section 3.1.

Q2: For a given list of topics, which deliverables were produced by the top-n organizations that have coordinated (or participated in) the most projects of H2020 were related to pilot applications?

UNWIND profile *AS* search_term
MATCH (o:Organization)-[:participates_in | coordinates]->(p:Project)-[:includes]->(k:Keyphrase)
WHERE k.name *CONTAINS*(search_term)
WITH o.name *AS* Organization, *COUNT*(p) *AS* ProjectCount
ORDER BY ProjectCount *DESC LIMIT* 100
UNWIND ['use case', 'pilot', 'application'] *AS* del_type
MATCH (o:Organization {name: Organization})-[:participates_in | coordinates]->(p:Project)<-[:belongs]-(d:Deliverable)-[:includes]->(k:Keyphrase)
WHERE d.title *CONTAINS* (del_type)
RETURN o.name AS Organization, *COLLECT*(DISTINCT d.title)[..10]

Performance: Execution time: 5919 ms, Memory used: 5.35 MB, Database hits: 10,529,877. For hardware specifications see Section 3.1.

Q3: Which people have been working on a given list of topics? Have they been working on additional ones?

UNWIND profile *AS* search_term
MATCH (a:Person)-[:writes]->(d:Deliverable)-[:belongs]->(:Project)-[:includes]->(k:Keyphrase)
WHERE k.name *CONTAINS*(search_term) *AND NOT* k.name *IN* ['report', 'deliverable', 'task', 'project', 'document', 'available', 'terms']
WITH a.name *AS* Person, k.name *AS* Topic, *COUNT*(k) *AS* NumberOfTopics
ORDER BY NumberOfTopics *DESC*
RETURN Person, *COLLECT*([Topic, NumberOfTopics])[..25] *LIMIT* 100

Performance: Execution time: 7575 ms, Memory used: 423.29 MB, Database hits: 12,069,554. For hardware specifications see Section 3.1.

References

1. Bollacker, K.; Evans, C.; Paritosh, P.; Sturge, T.; Taylor, J. Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge. In Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, Vancouver, BC, Canada, 9 June 2008; Association for Computing Machinery: Vancouver, BC, Canada, 2008; pp. 1247–1250.
2. Auer, S.; Bizer, C.; Kobilarov, G.; Lehmann, J.; Cyganiak, R.; Ives, Z. DBpedia: A Nucleus for a Web of Open Data. In Proceedings of the Semantic Web; Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., et al., Eds.; Springer: Berlin/Heidelberg, Germany, 2007; pp. 722–735.
3. Ehrlinger, L.; Wöß, W. Towards a Definition of Knowledge Graphs. In Proceedings of the Joint Posters and Demos Track of 12th International Conference on Semantic Systems—SEMANTiCS2016 and 1st International Workshop on Semantic Change & Evolving Semantics (SuCCESS16): Posters and Demos Track, Leipzig, Germany, 13–14 September 2016.
4. Paulheim, H. Knowledge Graph Refinement: A Survey of Approaches and Evaluation Methods. *Semant. Web* **2017**, *8*, 489–508. [CrossRef]
5. Campos, R.; Mangaravite, V.; Pasquali, A.; Jorge, A.; Nunes, C.; Jatowt, A. YAKE! Keyword Extraction from Single Documents Using Multiple Local Features. *Inf. Sci.* **2020**, *509*, 257–289. [CrossRef]

6. Mihalcea, R.; Tarau, P. TextRank: Bringing Order into Text. In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain, 25–26 July 2004; Association for Computational Linguistics: Barcelona, Spain, 2004; pp. 404–411.
7. Hasan, K.S.; Ng, V. Automatic Keyphrase Extraction: A Survey of the State of the Art. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Baltimore, MD, USA, 22–27 June 2014; Association for Computational Linguistics: Baltimore, MD, USA, 2014; pp. 1262–1273.
8. Wan, X.; Xiao, J. Single Document Keyphrase Extraction Using Neighborhood Knowledge. In Proceedings of the 23rd National Conference on Artificial Intelligence, Chicago, IL, USA, 13 July 2008; AAAI Press: Chicago, IL, USA, 2008; Volume 2, pp. 855–860.
9. Giarelis, N.; Kanakaris, N.; Karacapilidis, N. A Comparative Assessment of State-Of-The-Art Methods for Multilingual Unsupervised Keyphrase Extraction. In *Artificial Intelligence Applications and Innovations*; Maglogiannis, I., Macintyre, J., Iliadis, L., Eds.; Springer International Publishing: Cham, Switzerland, 2021; Volume 627, pp. 635–645. ISBN 9783030791490.
10. Alami Merrouni, Z.; Frikh, B.; Ouhbi, B. Automatic Keyphrase Extraction: A Survey and Trends. *J. Intell. Inf. Syst.* **2020**, *54*, 391–424. [CrossRef]
11. Papagiannopoulou, E.; Tsoumakas, G. A Review of Keyphrase Extraction. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2020**, *10*, e1339. [CrossRef]
12. Cunningham, H.; Wilks, Y.; Gaizauskas, R.J. GATE-a General Architecture for Text Engineering. In Proceedings of the 16th International Conference on Computational Linguistics, Copenhagen, Denmark, 5–9 August 1996; Association for Computational Linguistics: Stroudsburg, PA, USA, 1996; Volume 2, pp. 1057–1060.
13. Bird, S.; Klein, E.; Loper, E.; Baldridge, J. Multidisciplinary Instruction with the Natural Language Toolkit. In Proceedings of the Third Workshop on Issues in Teaching Computational Linguistics, Columbus, OH, USA, 19–20 June 2008; Association for Computational Linguistics: Columbus, OH, USA, 2008; pp. 62–70.
14. Manning, C.; Surdeanu, M.; Bauer, J.; Finkel, J.; Bethard, S.; McClosky, D. The Stanford CoreNLP Natural Language Processing Toolkit. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Baltimore, MD, USA, 22–27 June 2014; Association for Computational Linguistics: Baltimore, MD, USA, 2014; pp. 55–60.
15. Akbik, A.; Bergmann, T.; Blythe, D.; Rasul, K.; Schweter, S.; Vollgraf, R. Flair: An Easy-to-Use Framework for State-of-the-Art Nlp. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations), Minneapolis, MN, USA, 2–7 June 2019; Association for Computational Linguistics: Minneapolis, MN, USA, 2019; pp. 54–59.
16. Qi, P.; Zhang, Y.; Zhang, Y.; Bolton, J.; Manning, C.D. Stanza: A Python Natural Language Processing Toolkit for Many Human Languages. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Online, 5–10 July 2020; Association for Computational Linguistics: Stroudsburg, PA, USA, 2020; pp. 101–108.
17. Kapetanios, E.; Tatar, D.; Sacarea, C. *Natural Language Processing: Semantic Aspects*; CRC Press: Boca Raton, FL, USA, 2013; ISBN 9781466584969.
18. Schmitt, X.; Kubler, S.; Robert, J.; Papadakis, M.; LeTraon, Y. A Replicable Comparison Study of NER Software: StanfordNLP, NLTK, OpenNLP, SpaCy, Gate. In Proceedings of the 2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS), Granada, Spain, 22–25 October 2019; pp. 338–343.
19. Calignano, G.; Trippl, M. Innovation-Driven or Challenge-Driven Participation in International Energy Innovation Networks? Empirical Evidence from the H2020 Programme. *Sustainability* **2020**, *12*, 4696. [CrossRef]
20. Malloci, F.M.; Penadés, L.P.; Boratto, L.; Fenu, G. A Text Mining Approach to Extract and Rank Innovation Insights from Research Projects. In Proceedings of the Web Information Systems Engineering—WISE 2020, Leiden, The Netherlands, 20–23 October 2020; Huang, Z., Beek, W., Wang, H., Zhou, R., Zhang, Y., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 143–154.
21. Pérez-Fernández, D.; Arenas-García, J.; Samy, D.; Padilla-Soler, A.; Gómez-Verdejo, V. Corpus Viewer: NLP and ML-based Platform for Public Policy Making and Implementation. *Proces. Leng. Nat.* **2019**, *63*, 193–196.
22. Rinaldi, A.M.; Russo, C. Using a Multimedia Semantic Graph for Web Document Visualization and Summarization. *Multimed. Tools Appl.* **2021**, *80*, 3885–3925. [CrossRef]
23. Jo, S.; Park, B.; Lee, S.; Kim, J. OLGAVis: On-Line Graph Analysis and Visualization for Bibliographic Information Network. *Appl. Sci.* **2021**, *11*, 3862. [CrossRef]
24. Haase, P.; Herzig, D.M.; Kozlov, A.; Nikolov, A.; Trame, J. Metaphactory: A Platform for Knowledge Graph Management. *Semant. Web* **2019**, *10*, 1109–1125. [CrossRef]
25. Arenas, M.; Gutierrez, C.; Sequeda, J.F. Querying in the Age of Graph Databases and Knowledge Graphs. In Proceedings of the 2021 International Conference on Management of Data (SIGMOD/PODS '21), Xi'an, China, 20–25 June 2021; Association for Computing Machinery: New York, NY, USA, 2021; pp. 2821–2828. [CrossRef]
26. Angles, R.; Arenas, M.; Barceló, P.; Hogan, A.; Reutter, J.; Vrgoč, D. Foundations of Modern Query Languages for Graph Databases. *ACM Comput. Surv.* **2017**, *50*, 68. [CrossRef]
27. Gong, F.; Ma, Y.; Gong, W.; Li, X.; Li, C.; Yuan, X. Neo4j graph database realizes efficient storage performance of oilfield ontology. *PLoS ONE* **2018**, *13*, e0207595. [CrossRef] [PubMed]
28. Ratcliff, J.W.; Metzener, D.E. Pattern-Matching-the Gestalt Approach. *Dr Dobbs J.* **1988**, *13*, 46.

29. Gusfield, D. *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*; Cambridge University Press: Cambridge, UK, 1997; ISBN 9780521585194.
30. Comito, C.; Forestiero, A.; Pizzuti, C. Word Embedding based Clustering to Detect Topics in Social Media. In Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI '19), Thessaloniki, Greece, 14–17 October 2019; Association for Computing Machinery: New York, NY, USA, 2019; pp. 192–199. [CrossRef]
31. Sheng, S.; Zhou, P.; Wu, X. CEPV: A Tree Structure Information Extraction and Visualization Tool for Big Knowledge Graph. In Proceedings of the 2019 IEEE International Conference on Big Knowledge (ICBK), Beijing, China, 10–127 November 2019; pp. 221–228.
32. Nobre, C.; Streit, M.; Lex, A. Juniper: A Tree+Table Approach to Multivariate Graph Visualization. *IEEE Trans. Vis. Comput. Graph.* **2019**, *25*, 544–554. [CrossRef] [PubMed]
33. Forestiero, A.; Mastroianni, C.; Papuzzo, G.; Spezzano, G. A Proximity-Based Self-Organizing Framework for Service Composition and Discovery. In Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, Melbourne, VIC, Australia, 17–20 May 2010; IEEE Computer Society: Washington, DC, USA, 2010; pp. 428–437. [CrossRef]