mathematics

MDPI

Article

Knowledge Graph-Based Framework for Decision Making Process with Limited Interaction

Sivan Albagli-Kim [1,2] and Dizza Beimel [1,2,*]

[1] Department of Computer and Information Sciences, Ruppin Academic Center, Emek Hefer 4025000, Israel
[2] Dror (Imri) Aloni Center for Health Informatics, Ruppin Academic Center, Emek Hefer 4025000, Israel
* Correspondence: dizzab@ruppin.ac.il

Abstract: In this work, we present an algorithmic framework that supports a decision process in which an end user is assisted by a domain expert to solve a problem. In addition, the communication between the end user and the domain expert is characterized by a limited number of questions and answers. The framework we have developed helps the domain expert to pinpoint a small number of questions to the end user to increase the likelihood of their insights being correct. The proposed framework is based on the domain expert's knowledge and includes an interaction with both the domain expert and the end user. The domain expert's knowledge is represented by a knowledge graph, and the end user's information related to the problem is entered into the graph as evidence. This triggers the inference algorithm in the graph, which suggests to the domain expert the next question for the end user. The paper presents a detailed proposed framework in a medical diagnostic domain; however, it can be adapted to additional domains with a similar setup. The software framework we have developed makes the decision-making process accessible in an interactive and explainable manner, which includes the use of semantic technology and is, therefore, innovative.

Keywords: knowledge graph; semantic reasoning; medical diagnostic; decision support systems

MSC: 68T35

Citation: Albagli-Kim, S.; Beimel, D. Knowledge Graph-Based Framework for Decision Making Process with Limited Interaction. *Mathematics* **2022**, *10*, 3981. https://doi.org/10.3390/math10213981

Academic Editors: Ripon Kumar Chakrabortty and Zhao Kang

Received: 18 August 2022
Accepted: 22 October 2022
Published: 26 October 2022

1. Introduction

In recent years, the world of "big data" has gained significant momentum and continues to generate opportunities and challenges [1,2]. The various uses of big data have penetrated almost every field of the technological world. We are interested in the challenge of integrating big data in the technological realm dealing with decision-making processes in order to leverage these processes.

These processes can be found in a wide variety of content worlds (medicine, commerce, education, etc.) and require an understanding of situation awareness, data modeling, and algorithms for delivering intelligent insights. However, these processes provide different answers to different needs; thus, there are several types of decision-making processes, each with a suitable setup [3,4].

In this work, we focus on decision-making processes with the following setup: (a) the process involves two entities: an end user and a domain expert, (b) the end user initiates the process, (c) between the two entities there is an interaction that includes questions (of the domain expert) and answers (of the end user), (d) the interaction between the two entities is limited as possible (in time, the number of questions, money, etc.).

Given the above setup, the purpose of the presented work is to provide a framework based on semantic technology that enables integrating big data to assist the domain expert during the process of decision making, by suggesting to them a set of questions (inferred from the data) for the end user, that will reduce the cycles of questions and answers.

Consider the following two examples of domains whose processes are naturally suitable for such a setup: medical diagnosis [5] and appliance repairs [6] (Table 1):

Table 1. Examples for domains with limited interaction.

	Appliance Repairs	**Medical Diagnosis**
Domain expert	Service center representative	Clinician
End user	Customer	Patient
Interaction	Limited, as the representative has a small number of questions for the end user. Using the end user's answers, the representative must identify the type of fault, and, on this basis, the treatment will be determined	Limited, as the clinician has about 10 minutes per patient during which they (a) ask the patient a small number of questions (symptoms), and (b) decide on a limited number of tests

As noted, the two mentioned domains contain a two-sided limited interaction. The limitation can be expressed in terms of time, the number of questions, etc. Note that both domains, the medical and the appliance repairs are broad domains that can be specialized into specific subdomains. For example, the domain of appliance repairs can be specialized into construction service, internet service, household faults service, etc. The same goes for the medical domain. It may also contain subdomains, such as medical counseling in various fields (e.g., psychology), treatment of urgent medical calls, etc.

The suggested framework includes two main components: (a) a formal representation of the relevant domain expert's knowledge using semantic technology, specifically a *knowledge graph*, and (b) an interactive set of algorithms that begins with a set of initial domain values (i.e., prior knowledge of the end user), then, based on this prior knowledge and the knowledge graph representation, it will suggest specific questions to the end user. Answers to these questions will advance the domain expert in the decision-making process and become input for the next iteration. The iterations will continue until the domain expert is satisfied and a decision is made.

We were motivated to represent the expert's knowledge via a knowledge graph as graphs have emerged as a natural way of representing connected data [7]. Efforts during the last decade have organized large amounts of data as collections of nodes and edges, especially in recommendation systems, search engine optimization, and decision-making processes [8–10]. The resulting flexible structure, called a knowledge graph, allows quick adaptation of complex data and connections through relationships. Their inherent interconnectivity enables the use of graph algorithms to reveal hidden patterns and infer new knowledge [11–14]. Furthermore, knowledge graphs are computationally efficient and scale to very large sizes as exemplified by social graphs analysis [15,16].

Our framework was inspired by the perception presented by Musen and his colleagues [17], who are well-known researchers in the field of biomedical informatics, regarding information technology that assists with clinical decision support (CDS). Musen et al. [17] present the guiding principles for systems that provide CDS: their discourse is about communication rather than retrieving information, recommendations rather than producing reports, and assisting domain experts to develop more informed judgments. Respectively, the concept that led us in developing our framework is to provide the domain expert with recommendations inferred from the analysis of relevant data represented by a graph and enable him to make an informed decision. Nevertheless, an additional leading concept was to carry it out with a limited number of iterations. Our framework can be extended to additional domains.

In the presented work we have introduced a new approach for an interactive framework addressed to support decision-making processes characterized by a limited number of interactions. The framework is innovative by being generic, using a graph data model, graph algorithms, and semantic technology. We run our algorithms on a real data set and demonstrate the framework feasibility in a possible realistic scenario. Hence, we provide a proof of concept to our framework.

To illustrate the proposed framework, we begin by reviewing knowledge graphs and decision-making processes (Section 2). We then define the framework's terminology and its algorithms (Section 3). Following this, we demonstrate the framework in the medical diagnostic domain, using a data set consisting of diseases and patient symptoms (Section 4). Finally, we summarize and consider potential future directions (Section 5).

2. Background and Prior Works

2.1. Background

In this subsection, we review semantic technologies, and, in particular, knowledge graphs (KG). Then, we describe the algorithms we used on top of the KG within our framework.

2.1.1. Knowledge Graphs

A knowledge graph encodes data in the form of graph structures by capturing relationships between entities in a flexible manner. Knowledge graphs, or representations of information as semantic graphs, have attracted widespread attention in both the industrial and the academic worlds. Their property of providing semantically structured information has realized important solutions for many tasks, including question answering [18], recommendation systems [8], and information retrieval [19]. Knowledge graphs are also considered to offer great promise for building more intelligent machines.

2.1.2. Community Detection

With respect to graphs, a community can be defined as a subset of nodes densely connected to each other and loosely connected to nodes in the other communities in the same graph. Detecting communities in graphs is an important algorithmic challenge in the process of data understanding. Many methods have been devised over the last few years within different scientific disciplines, such as physics, biology, computer science, and social science. Recent studies show that by combining graph topology and node properties, we can better understand community structures in complex graphs [20]. Common algorithms for community detection in large graphs are the Louvain method and modularity optimization, as described below.

2.1.3. Louvain Method

The Louvain method is an algorithm for the detection of communities in large graphs [21]. For each community, the algorithm maximizes a modularity score. The modularity quantifies the quality of an assignment of nodes to communities. Namely, this provides an evaluation of how much more densely connected the nodes within a community are, compared to what could be expected in a random graph. The Louvain algorithm is a hierarchical clustering algorithm that recursively merges communities into a single node and executes the modularity clustering on the condensed graphs, and it stops when a local maxima of modularity is obtained. The method is a greedy optimization method, easy to implement and efficient, which appears to run in time $O(nlogn)$ where n is the number of nodes in the graph.

2.1.4. Modularity Optimization

The modularity optimization is an algorithm for the detection of communities in the graph based on their modularity [22]. Modularity is a measure of the graph structure. It measures the density of connections within a module or community. Namely, graphs that have a high modularity score are those that have many connections within a community, but only a few connections to other communities. The algorithm then tries to optimize the modularity score for each of the nodes; namely, it determines whether the modularity score of a node might increase if it would change its community to one of its connected nodes.

2.2. Prior Work

In this subsection, we review prior work in the context of decision support frameworks and then we focus on frameworks based on KG.

2.2.1. Clinical Decision Support Frameworks

According to Osheroff and his colleagues, clinical decision support (CDS) is the process that "provides clinicians, staff, patients, or other individuals with knowledge and person-specific information, intelligently filtered or presented at appropriate times, to enhance health and health care" [23]. Moreover, they claim that "a clinical decision support system (CDSS) is intended to improve healthcare delivery by enhancing medical decisions with targeted clinical knowledge, patient information, and other health information" [24]. CDSSs are used to assist and empower clinicians in their complex decision-making processes [25]. Musen and his colleagues [17] pinpoint the definition of CDSS and clarify that these systems assist not only by retrieving relevant data but by also considering the specific clinical context and thereby suggesting recommendations for the particular situation. Musen et al. also emphasize that CDSSs do not themselves make clinical decisions, but assist the decision makers (e.g., clinicians, patients, and healthcare organizations) in producing more informed judgments by providing relevant knowledge and analyses.

The range of functions provided by CDSS is wide and includes alarm systems, diagnostics, disease management, and prescription and drug control, among others [26]. They can be implemented in several ways, such as computerized alerts and reminders, or clinical workflow tools and computerized clinical guidelines, where patient data are taken into consideration. This last example involves developing a guideline-based point-of-care decision support system. To develop such systems, it is necessary to first create computer interpretable representations of the clinical knowledge contained in clinical guidelines [5].

Constructing CDS systems requires the most effort in creating the reasoning engine and in specifying the knowledge on which the reasoning engine operates. There are many strategies for accomplishing this, each addressing different requirements, including infobuttons [27], probabilistic systems [28], rule-based approaches [29], ontology-driven CDS systems [30], etc.

2.2.2. Knowledge Graph-Based Applications Including Decision Support Frameworks

Quoting Sprague from 1980 [31], the definition for decision support system is: "interactive computer-based systems, which help decision makers utilize data and models to solve unstructured problems". One of the main challenges in designing efficient decision support frameworks is knowledge acquisition, especially in complicated and uncertain decision contexts [32]. Knowledge graphs have emerged as a dynamic, scalable, and domain-independent form of knowledge representation, as Abu-Salih [33] claims "Knowledge Graphs have made a qualitative leap and effected a real revolution in knowledge representation". Abu-Salih continues to argue that the underlying structure of the KG enables better comprehension, reasoning, and interpretation of knowledge for both humans and machines. These features attract more and more researchers in recent years to use KG as the main means to deal with real-life problems in various fields, such as threat detections [34], interactive recommendations [35], healthcare and medical consultations [36–38], service system developments [39], designing decision support systems [32], and more. We elaborate on KG usages, which are similar to our framework, in the following paragraphs.

In recent years, KG penetrated the domain of interactive recommender systems (IRS), which elicit the dynamic preferences of users and take actions based on their current needs through real-time multi-turn interactions. Zhou et al. [35] investigated the potential of leveraging KG to provide rich side information for recommendations of decision-making. Yet, this system does not focus on restricting the interactions with the end user, as our framework does.

Huang et al. [37] introduced a framework for an AI-based medical consultation system with knowledge graph embedding. Their framework implementation leverages knowledge

organized as a graph to have diagnosis according to evidence collected from patients recurrently and dynamically. This system, similar to our framework, assists the domain expert. However, while Huang et al.'s system only serves the domain expert, our suggested framework addresses a situation in which the domain expert conducts a real-time question and answer interaction with the end user, whose answers are used as input for our algorithms.

Elnagar and Weistroffer [32] were the first to introduce KG to DSS Design. In their study, they explored how KGs can enhance the decision-making process in DSSs, by presenting a framework to integrate a KG into the DSS design. They claimed that using KG may assist in addressing the limitations of varied, unstructured, and dynamic sources of data that exist among most organizations. They stated that knowledge graphs can support the decision-making of enterprises by enhancing the efficacy of all data integration steps and allowing real-time analysis. However, they use a KG for designing the DSS, i.e., without changing the main structure of the DSS design, while our work uses the KG as the main platform and infrastructure for knowledge acquisition, management, and inference.

To summarize: the framework we suggest for a decision-making process has the following characteristics and is unique to the best of our knowledge: (a) is grounded on a knowledge graph, (b) enables dynamic and optimal management and inference of big data, (c) supports an interactive scenario consisting of questions and answers in real-time between two entities: a domain expert and an end user, and (d) aims to produce a limited interaction, i.e., to reduce the number of questions in the scenario.

3. Framework and Algorithms

In this section, we introduce the proposed framework, which includes a collection of algorithms and the flow between them.

We aim for interaction-based decision-making processes. The interaction is between a domain expert and an end user, and results in a limited number of iterations, consisting of questions that the framework suggests the domain expert ask the end user. The decision-making process will progress according to the end-user's answers.

When we analyzed these types of processes, we concluded that they can be generically modeled as a collection of *symptoms* and *diseases*. Eventually, the process goal is to assist the domain expert to decide on a *diagnosis* (i.e., provide an explanation for a given set of symptoms based on analyzing available data). Musen described the diagnostic process as being about deciding which questions to ask, which tests to order, or which procedures to perform [7,17]. *Questions* that may arise during the diagnosis process are of the type: *Does the end-user have a particular symptom?*

The above terms (i.e., symptoms, diseases, questions, and diagnoses) produce a jargon that can naturally be used in the medical diagnostic domain, yet it is also suitable for other domains, such as appliance repairs: the symptom represents a problem, the disease represents a malfunction, the diagnosis is a fault identification, and a typical question can be: *Does the end-user have a particular problem with his appliance?*

When using this jargon in the context of the proposed framework, we replace the term diagnosis with the term *hypothesis*, as the framework does not provide the domain expert with diagnoses, but rather with possible hypotheses. Each hypothesis is in fact a potential disease, and it is accompanied by a question, which is a symptom that indicates the disease (hypothesis). Therefore, the jargon we used throughout the paper to describe the framework and its various algorithms include the terms: *symptoms*, *diseases*, *questions*, and *hypotheses*. In particular, the framework infers hypotheses along with their related questions and submits them to the domain expert, who decides whether to use (or not) the questions to confirm (or not) the hypotheses (diseases).

In the rest of this section, we describe the framework along with its algorithms, first in general, then in detail.

In general, we start with building a knowledge graph from raw data, which will assist in exploring the relationships between diseases and symptoms. Following this, we use

the Louvain hierarchical clustering [21] on the KG (Algorithm 1) to find *communities* (i.e., clusters of diseases that have similar symptoms). Then, given the symptoms reported by the end user (called *evidence symptoms*), we find the possible diseases that are compatible with the evidence symptoms using inference on the KG (Algorithm 2). At this point, we infer the most probable community to include the end user disease and suggest to the domain expert a question (symptom) that indicates this community (Algorithm 3). Lastly, we find the best hypotheses to suggest to the domain expert (Algorithm 4), i.e., we suggest to the domain expert diseases and symptoms that the end user might have, to address the improvement of the diagnostic process.

The whole framework is divided into two main parts: the first part, the pre-processing part, is carried out once the framework is launched; while the second part, the processing part, is carried out each time a new request arrives in the framework. The pre-processing part consists of two steps and one algorithm (Algorithm 1), while the processing part consists of three steps and three algorithms (Algorithms 2–4), as we describe below.

The data structures we use include the structure for representing the KG (the default is an adjacency list) and additional structures required for running the algorithms. In the following paragraphs describing the algorithms, we detail these structures and their use.

Pre-processing part:

Input: A list of diseases and their symptoms
Step 1: Construct a knowledge graph (KG) of diseases and symptoms (see Section 3.1).
Step 2: Cluster the diseases into groups (called communities) according to their symptoms, i.e., diseases with similar symptoms will be in the same community (Algorithm 1).
Output: (1) each disease is associated with a community within the KG; (2) additional data structure, called a symptoms community matrix (SCM), represents the associations between groups of diseases and the various symptoms

Processing part:

Input: k evidence symptoms
Step 1: Find the most probable diseases, i.e., the possible diseases that are compatible with the evidence symptoms (Algorithm 2).
Step 2: Infer and suggest to the domain expert (repeatedly as required) a question (symptom) that indicates the most probable community to include the end user disease (Algorithm 3).
Step 3: Infer and suggest to the domain expert a list of hypotheses (diseases the end user might have) and their related questions (symptoms), sorted by relevance (Algorithm 4).

See Figure 1 for a high-level view of the whole suggested framework. In the following subsections, we elaborate on each of the above four algorithms in detail.

3.1. Building the Knowledge Graph

In this subsection, we describe the construction of the graph. In addition, we define framework-specific terminology used to describe the algorithms.

Let $KG = (V, E)$ be a directed graph, which is defined as follows. Let $V = D \bigcup S$ be the set of nodes, where D is the set of diseases and S is the set of symptoms. The edges of the graph are defined as follows: E=$\{(s, d) \in E \mid$ symptom $s \in S$ indicates disease $d \in D\}$, that is, there is an edge from a symptom s to disease d if s might indicate d.

We demonstrate the graph construction and the four algorithms on a simple KG (named *toy problem*), which is presented in Figure 2. The toy problem includes five diseases (represented by the nodes: d_1, d_2, d_3, d_4, d_5) and ten symptoms (represented by the nodes: $s_1 \ldots s_{10}$), so symptom 1 indicates disease 1, symptoms 2 and 3 indicate diseases 1 and 2, etc.

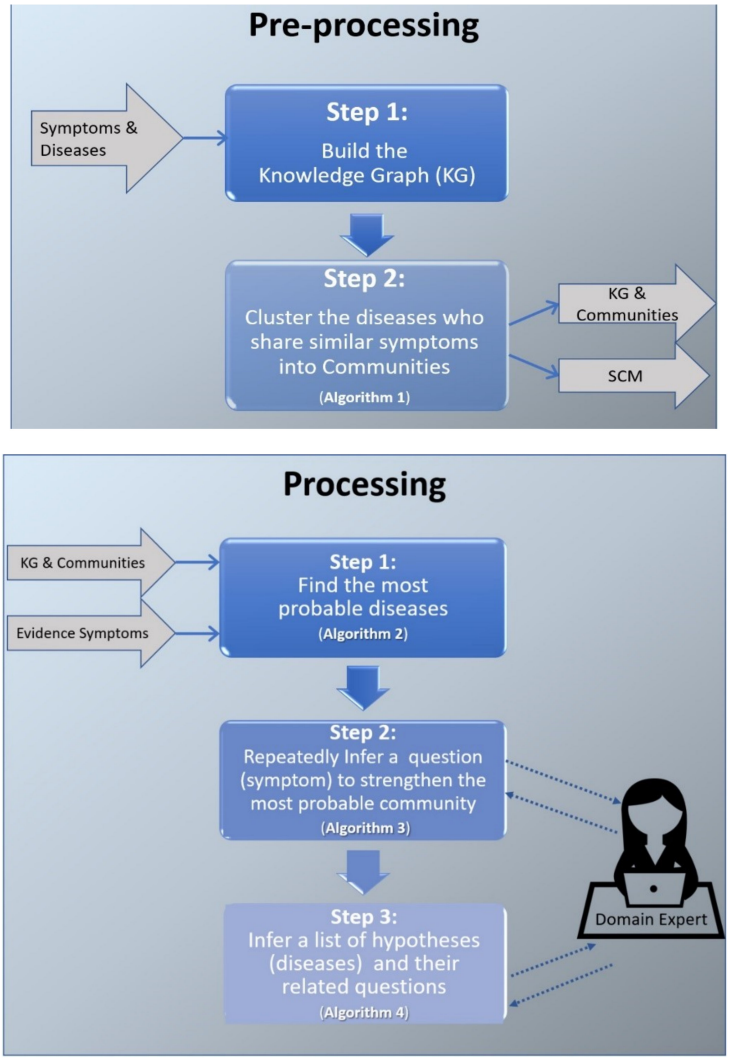

Figure 1. A high-level view of the framework. On the top, we demonstrate the pre-processing part, on the bottom the processing part.

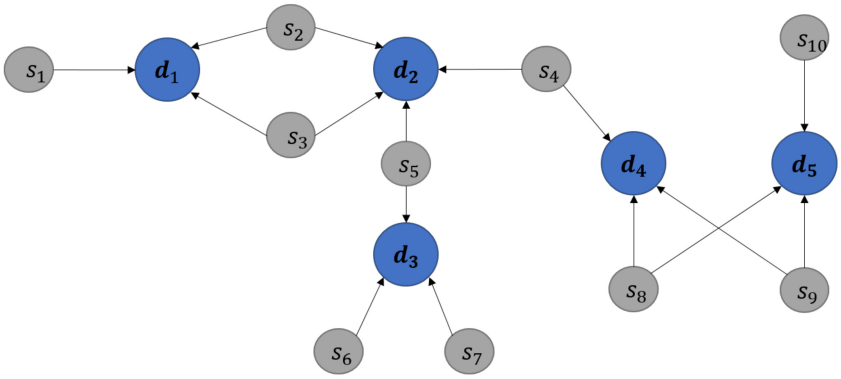

Figure 2. The toy problem KG.

3.2. Framework-Specific Terminology

The following (Table 2) is the terminology that we use to describe the algorithms.

Table 2. Definition of terms used in our algorithms.

Term	Definition
D	*The set of diseases nodes*
S	*The set of symptoms nodes*
ES	*The set of evidence symptoms (i.e., the symptoms indicated by the patient)*
C	*The set of communities*
$\mid c \mid$	*The size of a single community* $c \in C$. *Defined by the number of diseases that belongs to c*
R^c(s,c)	*The symptom's community rank of a given* $s \in S$ *and* $c \in C$. *Defined by the number of edges that point from s to c*
LinD(c)	*The local-in-degree of a given* $c \in C$. *Defined by the number of edges that point to diseases of c, by ES, hence, it is the sum of* R^c*(s,c), for each* $s \in ES$ *and the given c*
PD's communities	*The set of communities* $c \in C$ *with a positive LinD(c), hence, a community in which at least one edge from* $s \in ES$ *points to c*
R^s*(s,c)*	$R^c(s,c)$- $\left(\sum_{c \neq c' \in PD} R^c(s,c')\right)$ *Defined by the number of edges from symptom s to community c, less the number of edges from s to some other community c′. The outcome indicates how this symptom characterizes c.*
cs	*Community symptom.Defines a symptom indicating a high number of diseases in the community c and indicating a low number of diseases out of c. Hence, given a community c, it is the symptom s with the highest* R^s*(s,c)*
R^d(d)	*The disease's symptoms rank. Defined by the number of symptoms the patient has that indicate D*

3.3. The Framework Algorithms

In this subsection, we describe the algorithms that we developed as part of our framework.

Algorithm 1: Cluster the Diseases

To create the communities, we used the Louvain method [12] (see more details in Section 2.1). You can see below the pseudo-code of Algorithm 1.

Algorithm 1: Disease Community Detection

Input: Knowledge Graph $KG = (D \bigcup S, E)$.
Output: (1) For every $d \in D$, add a property named *community*, which determines the community d belongs to. (2) Symptoms community matrix (SCM), which is exhibited in Table 3.
Algorithm:

1. (preprocessing): for every two diseases $d1,\ d2 \in D$ such that $(s, d1) \in E,\ (s, d2) \in E$, add $e = (d1, d2) \in E$. At the end of this process, the number of edges between d1 and d2 is the number of symptoms they share.
2. Apply the Louvain method for community detection on the resulting graph accepted in Step 1.
3. Construct the SCM: an $|S| \times |C|$ matrix such that SCM[s, c] = R^c(s,c).

Table 3. The symptoms community matrix (SCM) derived from the toy problem KG. The evidence symptoms are highlighted for the future calculation of the LinD for each community.

	c_1	c_2	c_3
s_1	1	0	0
s_2	2	0	0
s_3	2	0	0
s_4	1	0	1
s_5	1	1	0
s_6	0	1	0
s_7	0	1	0
s_8	0	0	2
s_9	0	0	2
s_{10}	0	0	1

Given the toy problem KG represented in Figure 2, we present the communities that were found on that KG in Figure 3. The respective SCM is presented in Table 3, for instance, SCM[$s1$][$c1$] = 1, since there is one edge pointing from $s1$ to $c1$.

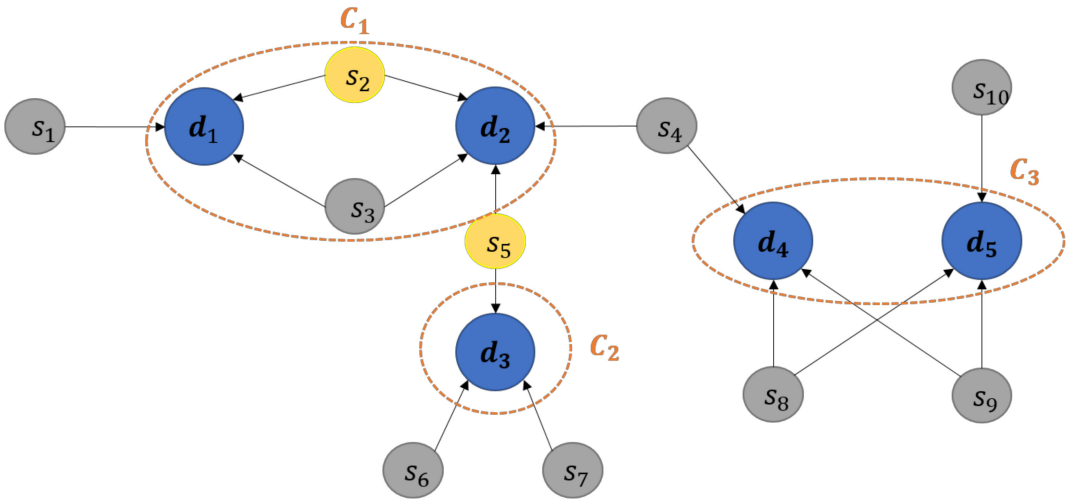

Figure 3. The toy problem KG, including the communities and evidence symptoms (yellow nodes).

Algorithm 2: Find the Most Probable Diseases

Algorithm 2 receives the evidence symptoms and uses the KG to infer which diseases explain these evidence symptoms and outputs them. You can see below the pseudo-code of Algorithm 2.

Algorithm 2: Find the most probable diseases

Input: Knowledge graph $KG = (D \bigcup S, E)$, evidence symptoms $ES \subseteq S$.
Output: $PD \subseteq D$- set of possible diseases.
Algorithm:
1. For every symptom s in ES:
  1.1 For every disease d such that $(s,d) \in E$
  1.1.1 Add d to PD.
  2. Return PD

Based on the given toy problem graph (presented in Figure 2), and on a set of given evidence symptoms (recall in our example they are d_2, d_5), the output of Algorithm 2 is $PD = \{d_1, d_2, d_3\}$, thus, the PD's communities are $c1$ and $c2$.

Algorithm 3: Find the Most Probable Community

Algorithm 3 receives the most probable diseases found by Algorithm 2 and uses the SCM to infer which community (i.e., group of diseases) is more likely to include the

end-user disease. To determine whether the inferred community is relevant, the algorithm outputs a symptom (which is a question for the end user) named *community symptom* (cs). The answer to this question will help to determine whether the patient disease is one of the community diseases or not.

If the end user indicates having the symptom, the framework will proceed to Algorithm 4. Otherwise, Algorithm 3 will iteratively continue to search for the next community with the highest potential to contain the user's disease and accordingly offer the next relevant cs.

You can see below the pseudo-code of Algorithm 3.

Algorithm 3: Find the most probable community

Input: possible diseases PD, symptoms community matrix (SCM).
Output: cs and the community it indicates (presented as a question to the domain expert) or null if it does not exist.
Algorithm:

1. Let C be the list of PD's communities, sorted by their LinD property, in a decreasing order.
2. Let $c \in C$ be the current community in the order.
3. For every symptom $s \notin ES$ in SCM(_,c), calculate R^s (s,c).
4. Let $s' = argmax_{s' \notin ES} R^s(s',c)$. If $R^s(s',c) > 0$, return s' (i.e., cs) and c. Otherwise, return to step 2.
5. Return null.

Based on the given $PD = \{d_1, d_2, d_3\}$ (output by Algorithm 2) that resulted with $c1$ and $c2$ as the PD's communities, the respective $LinD(c1)$ is 3 and $LinD(c2)$ is 1 (i.e., $LinD(c1)$ is the sum SCM[s2,c1] + SCM[s5,c1], as it is presented in Table 3). As c1 has the highest LinD, we calculate R^s for each symptom with respect to $c1$ and compared to $c2$. $s3$ has the highest R^s, as $R^c(s3, c1) - R^c(s3, c2)$ yields the maximum value (=2). Thus, the algorithm outputs c1 and $s3$ as its respective *cs* and presents them to the domain expert.

Algorithm 4: Find Disease Symptoms

Algorithm 4 receives the evidence symptoms and a community c and uses the *SCM* to infer which diseases in c are more likely to explain the patient's symptoms. The output of the algorithm is a list of ordered pairs R. Each pair consists of a hypothesis (disease) and its related question (symptom), the answers to which might help the diagnosis process. You can see below the pseudo-code of Algorithm 4.

We define an order between hypotheses in the community c as follows:

(i) Let h1 and h2 be two hypotheses with the same number of evidence symptoms indicating them (that is, $R^d(h1) = R^d(h2)$) and let $s1$ and $s2$ be two symptoms that strengthen them, respectively. Then, hypothesis h1 is before h2 in the order if $R^c(s1) \leq R^c(s2)$.

(ii) Let h1 and h2 be two different hypotheses such that $R^d(h1) < R^d(h2)$. Then, h1 is before h2 in the order.

Algorithm 4: Find Disease Symptom

Input: Community c, evidence symptoms ES, symptoms community matrix (SCM).
Output: a list (R) consisting of ordered pairs. Each pair consists of a hypothesis (disease) and its related question (symptom). The pairs are sorted by their relevance defined above.
Algorithm:

1. Let R be an empty list.
2. Let D be the list of diseases in c, sorted in a decreasing order by their R^d.
3. Let S = SCM(_,c)\ES be the list of symptoms in community c, without the evidence symptoms, sorted in an increasing order by their R^c.
4. For each $d \in D$: 4.1. for each s' in S such that (d,s') $\in E$, add (d,s') to R.
5. Return R ordered by relevance.

Based on the previous output, let us consider that $s3$ is a symptom that the end user indicates they have. Thus, we assume that $c1$ is more likely to include the end user

disease. At that point, the algorithm calculates for each disease in $c1$ their R^d: $R^d(d1) = 2$ and $R^d(d2) = 3$. Thus, the sorted list D includes $(d1, d2)$. Then, the algorithm sorts the symptoms of $c1$ (excluding the evidence symptoms, in our case $s2$, $s3$, *and* $s5$), by their R^c: $R^c(s1, c1) = 1$ and $R^c(s4, c1) = 1$. Thus, the sorted list S includes $(s1, s4)$. Finally, the algorithm returns the sorted list R, which includes the following pairs: [($d2$, $s4$),($d1$, $s1$)].

At that point, R is presented to the domain expert for further consideration.

4. Case Study Scenario

To examine the proposed framework, particularly the use of the algorithms listed in Section 3, we used a data set composed of patients' records that were taken from Kaggle (https://www.kaggle.com/ (accessed on 25/October/2022)) (described in Section 4.1). We ran a sample scenario on the given data set, which is presented in Section 4.2, followed by the results of the algorithms run using the sample scenario (in Section 4.3).

4.1. Data Set Description

The data set contained a total of 410 patient records. Each record referred to one patient and included the name of the disease and the symptoms the patient was experiencing. The data set included a total of 41 different diseases and most of the known symptoms that can characterize the specific disease.

The number of disease symptoms ranges from 4 to 17. The data set included a total of 130 different symptoms. Some of the symptoms were unique and characterized one specific disease, while others were quite common and characterized various diseases.

4.2. Knowledge Graph Construction and Community Detection

In this paragraph, we demonstrate the pre-processing part, that is, the knowledge graph construction and the communities' detection.

The knowledge graph was implemented using Neo4j and constructed as follows: we created a node for each of the 41 diseases and 130 symptoms. We created an edge between a symptom node and a disease node if that symptom characterized the disease. Some of the symptom nodes characterize multiple diseases, and thus have multiple connections.

After building the graph, we ran Algorithm 1 for community detection (if you recall, we used the Louvain method). This part was implemented using Neo4j Graph Data Science library (https://neo4j.com/docs/graph-data-science/current/algorithms/ (accessed on 17 August 2022)). Four communities were identified. Figure 4 exhibits the knowledge graph along with the detected communities. For clarity, each community is represented by a distinct color.

4.3. Scenario Description

Let us consider the following scenario: A patient arrives with the following two symptoms: yellowish skin and itching. These are our evidence symptoms.

Figure 5 depicts a sub-graph derived from the KG, including the evidence symptoms (in green) and the relations of the symptoms (i.e., the diseases that these symptoms characterize).

For display clarity, we present only some of the relations. In addition, Figure 5 presents two communities that were found by the community detection algorithm (Algorithm 1). The first community is colored in yellow and includes drug reaction and chickenpox, while the second community is colored in gray and includes hepatitis A–E and jaundice.

Running Algorithm 2 outputs the most probable diseases. In our case, they are six gray nodes that belong to the gray community and two yellow nodes that belong to the yellow community.

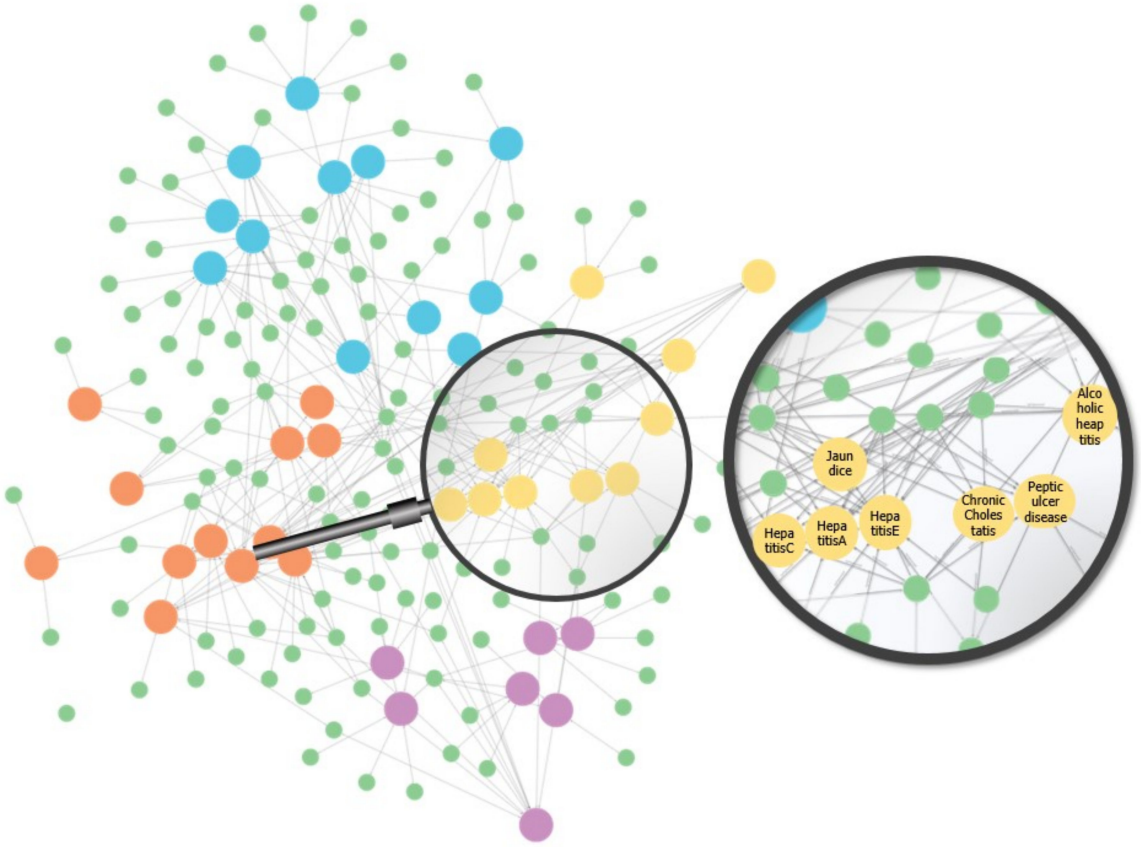

Figure 4. The detected communities on the knowledge graph (to be uploaded).

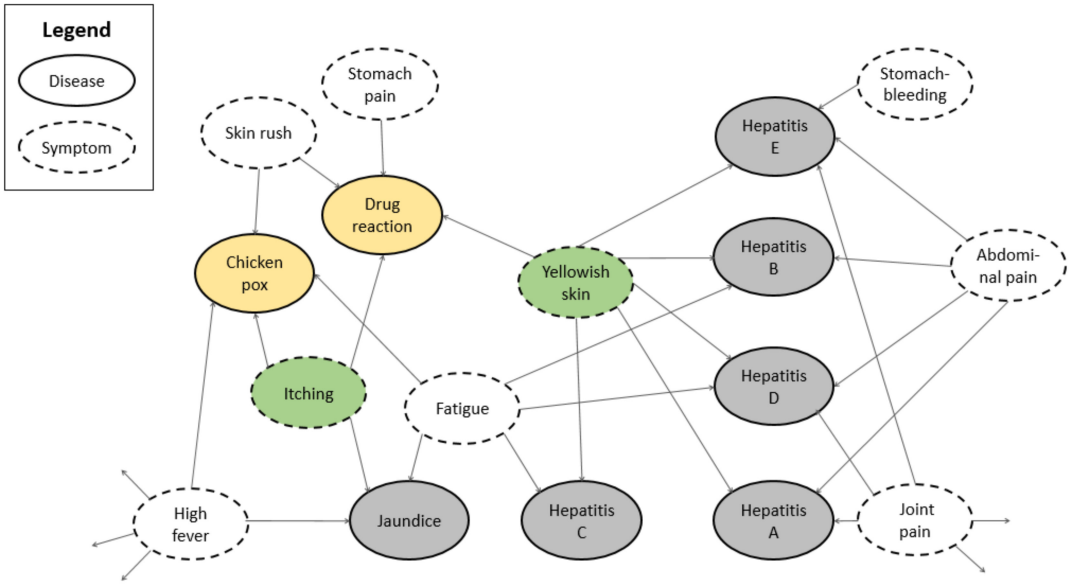

Figure 5. The sub-graph derived from the KG.

Algorithm 3 first finds the most probable community, which, as explained in the previous section, is the community with the highest *LinD*. As mentioned, in our case we have two communities: the gray community and the yellow community. The *LinD* of the yellow community is three, since three edges are pointing from the evidence symptoms (the green nodes) to the diseases of the yellow community (yellow nodes): itching pointing to two yellow nodes (chickenpox and drug reaction) and yellowish skin to one yellow

node (drug reaction). Similarly, the *LinD* of the gray community is six: there are six edges connecting the evidence symptoms with the diseases of the gray community. Thus, the gray community has the highest *LinD*.

At this point, Algorithm 3 examines the community with the highest *LinD* (in our case, the gray community) to suggest *cs*: a symptom that is best indicative of this community. In fact, *cs* is the symptom with the highest $R^s(s,c)$, given *c* is the gray community and compared to the other PD's communities (in our case the yellow community). Thus, to find the respective *cs*, the algorithm calculates R^s for each of the symptoms concerning the gray community, as can be seen in Figure 6 (table a). We can see that the symptom with the highest R^s relative to the gray community is abdominal pain. As so, Algorithm 3 outputs the gray community and its respective *cs*.

table a

Symptom	R^c(s,gray)	R^c(s,yellow)	R^s(s,gray)
Skin rush	0	2	−2
Stomach pain	0	1	−1
Fatigue	4	1	3
Joint pain	3	0	3
Abdominal pain	**4**	**0**	**4**
Stomach bleeding	1	0	1
High fever	1	1	0
Itching	1	2	−1
Yellowish skin	4	1	3

table b

Diseases	R^d(d)
Hepatitis A	2
Hepatitis B	2
Hepatitis D	2
Hepatitis E	2
Hepatitis C	1
Jaundice	1

table c

Symptom	R^c(s,gray)
High fever	1
Stomach bleeding	1
Joint pain	3
Fatigue	4

Figure 6. Table A—the Symptom's R^c for each PD's community along with the R^s for the gray community; Table B—the diseases in the gray community with their R^d; Table C—the symptoms indicating the gray diseases with their R^c.

In the presented scenario, the patient has this symptom, and, therefore, the hypothesis that the gray community contains one of the patient's diseases is strengthened. We can now continue to the last step and run Algorithm 4. Otherwise, if you recall, Algorithm 3 infers (repeatedly as required) the next question (symptom) to indicate the next most probable community.

Algorithm 4 returns *R*, which is a list of sorted pairs (disease, symptom), such that the symptom indicates the disease. The gray diseases are sorted in a decreasing manner according to their R^d and listed in Figure 6, Table B. In addition, the symptoms indicating these diseases are sorted increasingly according to their R^c and listed in Figure 6, Table C. In our case study, the algorithm returns the sorted list *R* that includes the pairs as they appear in the following table (Table 4):

Table 4. The sorted list of hypotheses with their related questions returned by Algorithm 4.

Order	Hypothesis (Disease)	Question (Symptom)
1	Hepatitis E	Stomach bleeding
2	Hepatitis A	Joint pain
3	Hepatitis D	Joint pain
4	Hepatitis E	Joint pain
5	Hepatitis B	Fatigue
6	Hepatitis D	Fatigue
7	Jaundice	High fever
8	Jaundice	Fatigue
9	Hepatitis C	Fatigue

5. Conclusions and Discussion

5.1. Summary

Decision-making processes are found in almost every area of our lives, and thus the realm of decision making is constantly evolving and receiving a lot of research attention. When we come to analyze, model, and implement systems that support these processes, we are required to focus on a specific sub-domain, since it is almost impossible to provide one selected solution for all the different requirements of decision-making processes, which arise from different content worlds.

In the current work, we focus on a sub-domain of decision-making processes characterized by the following characteristics: (a) the trigger for the procedure is an end user's request, (b) a domain expert is present, and (c) these two entities have an interaction, in a real-time scenario, consisting of questions (asked by the domain expert) and answers (given by the end user) that are limited in nature, i.e., the number of questions the domain expert addresses to the end user and the answers they receive must be limited.

This sub-domain (we named it "decision-making process with limited interaction") includes specific processes, such as an encounter between a physician and a patient, a contact between a service provider and a customer, an urgent call from a person in need of help to an assisting party, etc. All the examples below illustrate why the interaction needs to be limited. As such, one of our goals is to distill the necessary questions, asked by the domain expert, to assist them in reaching the right decision efficiently.

5.2. Contribution

In the literature review we performed, we found few references to the described process configuration. Therefore, we believe that our work will provide a contribution to addressing such a configuration of decision-making processes.

As noted, the algorithmic framework we developed aims to help the domain experts to pinpoint their questions to the end user. The proposed framework is based on the knowledge of the domain expert and their interaction with the end users.

The algorithmic framework consists of two parts. In the first part, a knowledge graph is constructed that characterizes the domain expert's knowledge. In the second part, as part of the interaction with the end user, the answers they provide are entered in the graph as evidence properties and generate a trigger for the inference algorithm in the graph.

As stated, this study aims to provide a generic framework that helps to refine the work processes with the characteristics mentioned earlier. At the same time, we want to present a possible use of the framework, and to that end, we chose the medical world as a case study. Specifically, we focused on the classic problem of medical diagnostics, which is part of a wide range of clinical decisions [7,17]. Medical diagnostics is a challenge that in recent decades has led to the development of methodologies and systems to support clinical decisions [40]. In this chosen case study, the end user is a patient, the domain expert is a physician, and the interaction is the encounter between them that aims to diagnose the patient's disease.

In this work, we propose a new approach for an interactive framework addressed to support decision-making processes characterized by a limited number of interactions. The innovation of the work stems from the use of semantic technologies, including a graphical data model, combined with unique algorithms. In addition, we tried to provide a generic framework that can be adapted to other domains beyond the medical domain.

5.3. Limitations

Many researchers are passionate about exploring the potential of artificial intelligence to support decision making, particularly within the clinical domain [41]. Nevertheless, there are still complexities researchers are trying to address. For instance, one of the challenges is to evaluate the improvement, if any, that such systems provide. Vasey and colleagues argue that "little is known about the outcomes of these systems when used as adjuncts to human decision making (human vs. human with)". Via systematic review, they explored the association between the interactive use of machine learning (ML)-based diagnostic CDSSs and clinician performance and reported that there is minimal evidence to suggest that using ML-based CDSSs is associated with improved physician diagnostic performance, since most studies had a small number of participants [42].

Besides the innovation and uniqueness of our work, there are also several limitations, which we discuss in the next paragraphs:

- In the case study we presented, we demonstrated the feasibility of the framework but did not compare its performance to another system or to a real doctor–patient situation. In other words, we did not carry out a comprehensive evaluation.
- We encountered a difficulty in estimating the complexity of the Algorithms 2–4, as their activation depends on the data existing in the KG, and on the number of iterations that will be performed in the interaction between the domain expert and the end user.
- As mentioned, the KG was constructed based on the data of diseases and symptoms taken from the Kaggle website. The KG was used for examining the case study we presented, as a proof of concept. Yet, the existing KG cannot be considered as big data. Knowledge graphs are designed to handle large volumes of data, and in our future work, we might test the scalability of the framework on a larger scale.
- The data we used did not contain information on the extent to which a symptom is related to a disease. Therefore, the KG did not have weights on the edges. This issue will be addressed in our future work, as well as statistical aspects.

5.4. Future Work

The framework we have developed makes the decision-making process accessible in an interactive and explainable manner, which includes the use of semantic technology and is, therefore, innovative.

Following our current work, we will aim to produce a comparative analysis of the suggested framework. The following are potential future directions:

- Using ontologies to enrich semantic reasoning.
- Using a weighted knowledge graph for representing the cost of each question.

In addition, we plan to combine the knowledge graph with medical ontologies having semantic and verbal data that supplement and/or expand the medical information. Furthermore, integration with specific medical information about patients (test results, medical background, etc.) can also increase the accuracy of the medical diagnosis.

Author Contributions: Conceptualization, S.A.-K. and D.B.; methodology, S.A.-K. and D.B.; software, S.A.-K. and D.B.; validation, S.A.-K. and D.B.; formal analysis, S.A.-K. and D.B.; investigation, S.A.-K. and D.B.; resources, S.A.-K. and D.B.; data curation, S.A.-K. and D.B.; writing—original draft preparation, S.A.-K. and D.B.; writing—review and editing, S.A.-K. and D.B.; visualization, S.A.-K. and D.B.; supervision, S.A.-K. and D.B.; project administration, S.A.-K. and D.B.; funding acquisition, S.A.-K. and D.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: This study is based on an anonymized, publicly available database. The study was conducted according to the guidelines of the Ruppin Academic Center Research.

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

References

1. Hashem, I.A.T.; Yaqoob, I.; Anuar, N.B.; Mokhtar, S.; Gani, A.; Khan, S.U. The rise of "big data" on cloud computing: Review and open research issues. *Inf. Syst.* **2015**, *47*, 98–115. [CrossRef]
2. Sivarajah, U.; Kamal, M.M.; Irani, Z.; Weerakkody, V. Critical analysis of Big Data challenges and analytical methods. *J. Bus. Res.* **2017**, *70*, 263–286. [CrossRef]
3. Power, D.J. *Decision Support Systems: Concepts and Resources for Managers*; Greenwood Publishing Group: Westport, CT, USA, 2002.
4. Power, D.J. A Brief History of Decision Support Systems. 2007. p. 3. Available online: DSSResources.com (accessed on 17 August 2022).
5. Kumar, D.S.; Sathyadevi, G.; Sivanesh, S. Decision support system for medical diagnosis using data mining. *Int. J. Comput. Sci. Issues (IJCSI)* **2011**, *8*, 147.
6. Hossayni, H.; Khan, I.; Aazam, M.; Taleghani-Isfahani, A.; Crespi, N. SemKoRe: Improving machine maintenance in industrial iot with semantic knowledge graphs. *Appl. Sci.* **2020**, *10*, 6325. [CrossRef]
7. Robinson, I.; Webber, J.; Eifrem, E. *Graph Databases: New Opportunities for Connected Data*; O'Reilly Media, Inc.: Middlesex County, MA, USA, 2015.
8. Guo, Q.; Zhuang, F.; Qin, C.; Zhu, H.; Xie, X.; Xiong, H.; He, Q. A survey on knowledge graph-based recommender systems. *IEEE Trans. Knowl. Data Eng.* **2020**, *34*, 3549–3568. [CrossRef]
9. Xiong, C.; Power, R.; Callan, J. Explicit semantic ranking for academic search via knowledge graph embedding. In Proceedings of the 26th International Conference on World Wide Web 2017, Perth, Australia, 3–7 April 2017; pp. 1271–1279.
10. Rotmensch, M.; Halpern, Y.; Tlimat, A.; Horng, S.; Sontag, D. Learning a health knowledge graph from electronic medical records. *Sci. Rep.* **2017**, *7*, 1–11. [CrossRef]
11. Lbath, H.; Bonifati, A.; Harmer, R. Schema inference for property graphs. In Proceedings of the EDBT 2021-24th International Conference on Extending Database Technology, Nicosia, Cyprus, 23–26 March 2021; pp. 499–504.
12. Das, M.; Wu, Y.; Khot, T.; Kersting, K.; Natarajan, S. Scaling lifted probabilistic inference and learning via graph databases. In Proceedings of the 2016 SIAM International Conference on Data Mining, Miami, FL, USA, 5–7 May 2016; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 2016; pp. 738–746.
13. Ma, Y.; Crook, P.A.; Sarikaya, R.; Fosler-Lussier, E. Knowledge graph inference for spoken dialog systems. In Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2015, South Brisbane, QLD, Australia, 19–24 April 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 5346–5350.
14. Ji, S.; Pan, S.; Cambria, E.; Marttinen, P.; Philip, S.Y. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *33*, 494–514. [CrossRef]
15. Sandryhaila, A.; Moura, J.M. Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure. *IEEE Signal Process. Mag.* **2014**, *31*, 80–90. [CrossRef]
16. Rashidy, R.A.H.E.; Hughes, P.; Figueres-Esteban, M.; Harrison, C.; Van Gulijk, C. A big data modeling approach with graph databases for SPAD risk. *Saf. Sci.* **2018**, *110*, 75–79. [CrossRef]
17. Musen, M.A.; Middleton, B.; Greenes, R.A. Clinical decision-support systems. In *Biomedical Informatics*; Springer: Cham, Switzerland, 2021; pp. 795–840.
18. Gashkov, A.; Perevalov, A.; Eltsova, M.; Both, A. *Improving Question Answering Quality through Language Feature-Based SPARQL Query Candidate Validation*; The Semantic Web. ESWC 2022. Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2022; Volume 13261. [CrossRef]
19. Dietz, L.; Kotov, A.; Meij, E. Utilizing knowledge graphs for text-centric information retrieval. In Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, Ann Arbor, MI, USA, 8–12 July 2018; pp. 1387–1390.
20. Bhatt, S.; Padhee, S.; Sheth, A.; Chen, K.; Shalin, V.; Doran, D.; Minnery, B. Knowledge graph enhanced community detection and characterization. In Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, Melbourne, VIC, Australia, 11–15 February 2019; pp. 51–59.
21. Lu, H.; Halappanavar, M.; Kalyanaraman, A. Parallel heuristics for scalable community detection. *Parallel Comput.* **2015**, *47*, 19–37. [CrossRef]
22. Newman, M.E.J.; Girvan, M. Finding and evaluating community structure in networks. *Phys. Rev. E* **2004**, *69*, 026113. [CrossRef] [PubMed]
23. Osheroff, J.A.; Teich, J.M.; Middleton, B.; Steen, E.B.; Wright, A.; Detmer, D.E. A roadmap for national action on clinical decision support. *J. Am. Med. Inform. Assoc.* **2007**, *14*, 141–145. [CrossRef]
24. Osheroff, J.A.; Teich, J.M.; Levick, D.; Saldana, L.; Velasco, F.T.; Sittig, D.F.; Jenders, R.A. *Improving Outcomes with Clinical Decision Support: An Implementer's Guide*; Himss Publishing: Chicago, IL, USA, 2012.

25. Sutton, R.T.; Pincock, D.; Baumgart, D.C.; Sadowski, D.C.; Fedorak, R.N.; Kroeker, K.I. An overview of clinical decision support systems: Benefits, risks, and strategies for success. *NPJ Digit. Med.* **2020**, *3*, 1–10. [CrossRef] [PubMed]
26. Omididan, Z.; Hadianfar, A.M. The role of clinical decision support systems in healthcare (1980–2010): A systematic review study. *Jentashapir Sci. -Res. Q.* **2011**, *2*, 125–134.
27. Cimino, J.J.; Patel, V.L.; Kushniruk, A.W. The patient clinical information system (PatCIS): Technical solutions for and experience with giving patients access to their electronic medical records. *Int. J. Med. Inform.* **2002**, *68*, 113–127. [CrossRef]
28. Saria, S.; Koller, D.; Penn, A. Learning individual and population level traits from clinical temporal data. In Proceedings of the Neural Information Processing Systems 2010, Vancouver, BC, Canada, 6–9 December 2010; pp. 1–9.
29. Buchanan, B.G.; Shortliffe, E.H. Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project. 1984. Available online: http://papers.cumincad.org/cgi-bin/works/Show&_id=caadria2010_044/paper/ec87 (accessed on 17 August 2022).
30. De Clercq, P.A.; Blom, J.A.; Korsten, H.H.; Hasman, A. Approaches for creating computer-interpretable guidelines that facilitate decision support. *Artif. Intell. Med.* **2004**, *31*, 1–27. [CrossRef]
31. Sprague, R.H., Jr. A framework for the development of decision support systems. *MIS Q.* **1980**, *4*, 1–26. [CrossRef]
32. Elnagar, S.; Weistroffer, H.R. Introducing Knowledge Graphs to Decision Support Systems Design. In *Information Systems: Research, Development, Applications*; Wrycza, S., Maślankowski, J., Eds.; SIGSAND/PLAIS. Lecture Notes in Business Information Processing; Springer: Cham, Switzerland, 2019; Volume 359. [CrossRef]
33. Abu-Salih, B. Domain-specific knowledge graphs: A survey. *J. Netw. Comput. Appl.* **2021**, *185*, 103076. [CrossRef]
34. Kurniawan, K.; Ekelhart, A.; Kiesling, E.; Quirchmayr, G.; Tjoa, A.M. KRYSTAL: Knowledge graph-based framework for tactical attack discovery in audit data. *Comput. Secur.* **2022**, *121*, 102828. [CrossRef]
35. Zhou, S.; Dai, X.; Chen, H.; Zhang, W.; Ren, K.; Tang, R.; Yu, Y. Interactive recommender system via knowledge graph-enhanced reinforcement learning. In Proceedings of the 43rd International ACM Sigir Conference on Research and Development in Information Retrieval 2020, virtual event, 25–30 July 2020; pp. 179–188.
36. Malik, K.M.; Krishnamurthy, M.; Alobaidi, M.; Hussain, M.; Alam, F.; Malik, G. Automated domain-specific healthcare knowledge graph curation framework: Subarachnoid hemorrhage as phenotype. *Expert Syst. Appl.* **2020**, *145*, 113120. [CrossRef]
37. Huang, Y.; Chen, M.; Tang, K. Training like Playing: A Reinforcement Learning And Knowledge Graph-based framework for building Automatic Consultation System in Medical Field. *arXiv* **2021**, arXiv:2106.07502.
38. Xiang, X.; Wang, Z.; Jia, Y.; Fang, B. Knowledge Graph-Based Clinical Decision Support System Reasoning: A Survey. In Proceedings of the IEEE Fourth International Conference on Data Science in Cyberspace (DSC) 2019, Hangzhou, China, 23–25 June 2019; pp. 373–380. [CrossRef]
39. Li, X.; Chen, C.H.; Zheng, P.; Wang, Z.; Jiang, Z.; Jiang, Z. A knowledge graph-aided concept–knowledge approach for evolutionary smart product–service system development. *J. Mech. Des.* **2020**, *142*, 101403. [CrossRef]
40. Berg, M.; Berg, P.A.M. *Rationalizing Medical Work: Decision-Support Techniques and Medical Practices*; MIT Press: Cambridge, MA, USA, 1997.
41. Shortliffe, E.H.; Sepúlveda, M.J. Clinical decision support in the era of artificial intelligence. *JAMA* **2018**, *320*, 2199–2200. [CrossRef]
42. Vasey, B.; Ursprung, S.; Beddoe, B.; Taylor, E.H.; Marlow, N.; Bilbro, N.; McCulloch, P. Association of clinician diagnostic performance with machine learning–based decision support systems: A systematic review. *JAMA Netw. Open* **2021**, *4*, e211276. [CrossRef]