Article

Knowledge Graph-based Framework for Decision-making Process with Limited Interaction

|  |
| --- |
| **Publisher’s Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.  A picture containing text, clipart  Description automatically generated  **Copyright:** © 2021 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/). |

Sivan Albagli-Kim 1, Dizza Beimel 1

1 Computer Science Department, Ruppin Academic Center, Emek Hefer 4025000, Israel

**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 so that their insights will be correct. The proposed framework is based on the domain expert’s knowledge and includes 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 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

1. Introduction

In recent years, the world of “big data” has gained significant momentum, and its various uses have penetrated almost every area of ​​our lives. Still, vast amount of data accumulating in domains remains unutilized (Hashem, Yaqoob, Anuar, Mokhtar, Gani & Khan, 2015; Sivarajah, Kamal, Irani & Weerakkody, 2017), and researchers continue to offer value-added applications that improve business processes in many varied domains. We focus on processes designed to assist in decision-making, using the massive volume of data available to domain experts.

There are several approaches for designing and implementing decision support systems (Power, 2002; Power, 2007). For example, focusing on clinical guidelines in the domain of medical diagnosis, several studies offer various approaches for modeling and computing general treatment protocols considering the patient’s specific information (Peleg, Tu, Bury, Ciccarese, Fox, Greenes, Hall, Johnson, Jones, Kumar, Miksch, 2003). In the domain of appliance repairs, the focus is on building an architecture that covers the customers’ varying needs and improves their existing maintenance process (Hossayni, Khan, Aazam, Taleghani-Isfahani & Crespi, 2020).

In this paper, we focus on decision-making processes that involve a domain expert and an end-user, with limited communication between them. Accordingly, we propose an interactive decision support framework for a domain expert who is required to conduct limited interaction with an end-user. Consider the following two examples (Table 1) for such interactions from different domains:

**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 |

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 algorithmic framework that begins with a set of initial domain values (i.e., prior knowledge of the end-user) then, based on prior knowledge, and the knowledge graph representation, will suggest specific questions to the end-user. Answers to these questions will advance the domain expert in the decision-making process and become inputs for the next iteration. The iterations will continue until the domain expert is satisfied and a decision is made.

Our system was inspired by Musen and his colleagues’ study of clinical decision support systems. However, this can be extended to additional domains. Their discourse is about communication rather than retrieving information, recommendations rather than producing reports, and assisting domain experts to develop more informed judgments (Musen, Middleton & Greenes, 2021).

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

1. Background and Prior Works
   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.

**Knowledge Graph**

A knowledge graph encodes data in the form of graph structures by capturing relationships between entities in a flexible manner. Knowledge graphs, or representation of information as a semantic graph, have caused wide concern in both the industrial and the academic world. Their property of providing semantically structured information has realized important solutions for many tasks, including question answering (Gashkov, Perevalov, Eltsova & Both, 2022), recommendation systems (Guo, Zhuang, Qin, Zhu, Xie, Xiong & He, 2020) and information retrieval (Dietz, Kotov & Meij, 2018). Knowledge graphs are also considered to offer great promise for building more intelligent machines.

**Community Detection**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 (Bhatt, S., Padhee, S., Sheth, A., Chen, K., Shalin, V., Doran, D., & Minnery, B.,2019). Common algorithms for community detection in large graphs are the Louvain method and modularity optimization as described below.

**Louvain Method**

The Louvain method is an algorithm for detecting communities in large networks (Lu, Mahantesh & Ananth, 2015). It maximizes a modularity score for each community, where the modularity quantifies the quality of an assignment of nodes to communities. This means evaluating how much more densely connected the nodes within a community are, compared to how connected they would be in a random network. 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.

**Modularity**

The modularity optimization algorithm tries to detect communities in the graph based on their modularity [Newman and Girvan, 2004]. Modularity is a measure of the structure of a graph, measuring the density of connections within a module or community. Graphs with a high modularity score will have many connections within a community but only a few pointing outwards to other communities. The algorithm will explore every node to determine if its modularity score might increase if it changes its community to one of its neighboring nodes.

* 1. 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.

**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” (Osheroff, Teich, Middleton, Steen, Wright & Detmer, 2007). 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” (Osheroff, Teich, Levick, Saldana, Velasco, Sittig & Jenders, 2012). CDSSs are used to assist and empower clinicians in their complex decision-making processes (Sutton, Pincock, Baumgart, Sadowski, Fedorak & Kroeker, 2020). Musen and his colleagues (Musen et al. 2021) 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 (Omididan & Hadianfar, 2011). 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 (Peleg et al., 2003).

Constructing CDS systems requires the bulk of the 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 (Cimino, Patel & Kushniruk, 2002), probabilistic systems (Saria, Koller & Penn, 2010), rule-based approaches (Buchanan & Shortliffe, 1984), ontology-driven CDS systems (De Clercq, Blom, Korsten & Hasman, 2004), etc.

**Knowledge Graph-based Decision Support Frameworks**

One of the main challenges in designing efficient decision support frameworks is knowledge acquisition, especially in complicated and uncertain decision contexts. Knowledge representation plays an important role in finding solutions to problems. Knowledge graphs have emerged as a dynamic, scalable, and domain-independent form of knowledge representation. Therefore, it is natural to integrate the KG into the decision support framework design (Elnagar & Weistroffer, 2019; Malik, Krishnamurthy, Alobaidi, Hussain, Alam & Malik, 2020; Xiang, Wang, Jia & Fang, 2019; Li, Chen, Zheng, Wang, Jiang & Jiang, 2020).

**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 focus on the end-user’s answers.

When we analyzed these processes, we concluded that they can be generically modeled as a collection of s*ymptoms* and *diseases*. Furthermore, the process goal is to decide on a *diagnosis* (i.e., analyzing available data to determine the explanation for given symptoms). Musen described the diagnostic process as being about deciding which questions to ask, which tests to order, or which procedures to perform (Musen et al. 2021). The questions arising during the framework iterations are of the type: *Does the end-user have a particular symptom?* The use of this terminology (i.e., symptoms, diseases, and diagnoses) is common in the medical diagnostic domain, yet analogous terminology 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?*

Therefore, the terminology we used throughout the paper to describe the framework and its various algorithms includes the terms s*ymptoms,* *diseases,* and *diagnoses.* Moreover, the framework’s output is a collection of *hypotheses* and their *questions*, so that each hypothesis (disease) is accompanied by a question (a symptom characterizing the disease) that interrogates the hypothesis. This collection is suggested to the domain expert to assist them while deciding on the diagnosis.

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 (Lu, et. al., 2015] 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 strengthens 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 additional diseases and symptoms that the end-user might have, in order to improve 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, while the processing part consists of three steps, as we describe below.

The data structures we use include the structure for representing the KG (usually 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 of diseases and symptoms (see subsection 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; (2) a data structure, called a symptoms community matrix (SCM), is representing the associations between groups of diseases and the various symptoms.*

Processing part:

*Input: k evidence symptoms*

**Step 1:** Find the most probable diseases (**Algorithm 2**).

**Step 2:** Infer (repeatedly as required) a question to strengthen the most probable community (**Algorithm 3**).

**Step 3:** Infer a list of hypotheses (diseases) and related questions (symptoms) sorted by relevance (**Algorithm 4**).

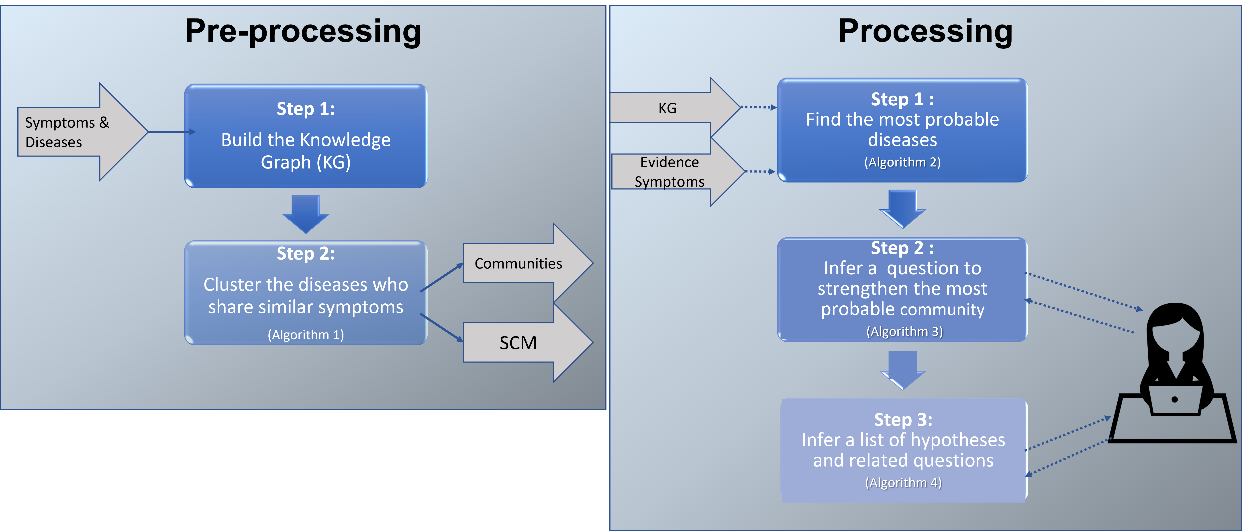
See Figure 1 for a High-level view of the whole suggested framework. In the following subsections, we elaborate in detail on each of the above algorithms.

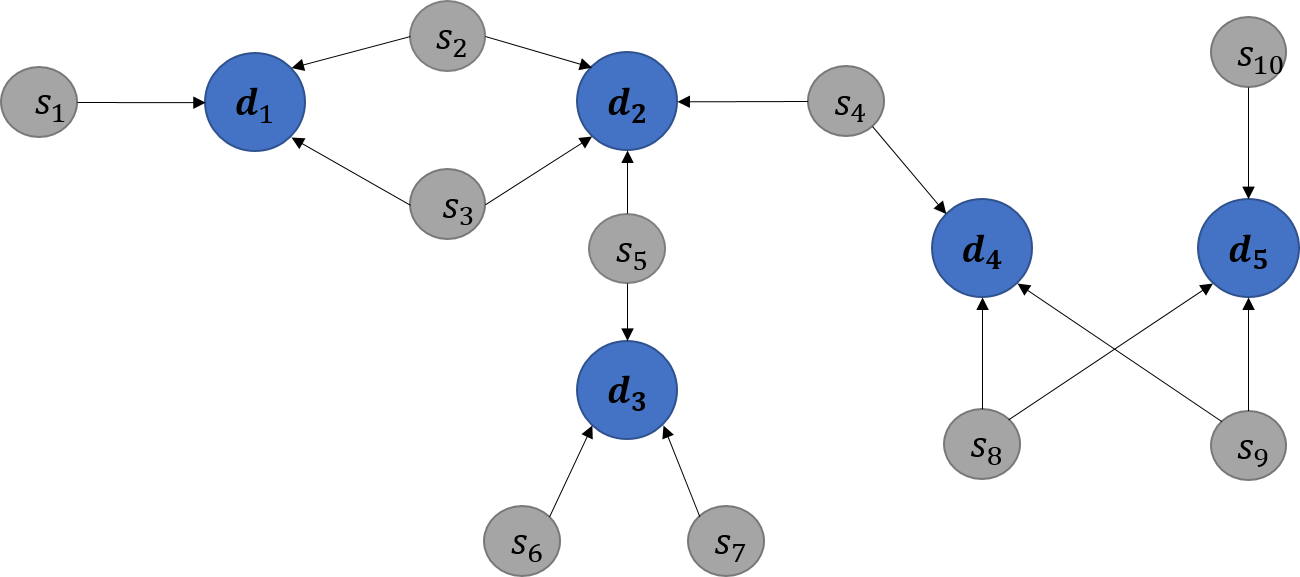
Figure 1: A high-level view of the framework. On the left, we demonstrate the pre-processing part, on the right side the processing part.

3.1 Building the Knowledge Graph

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

Let be a directed graph, which is defined as follows. Let be the set of nodes, where is the set of diseases and S is the set of symptoms. The edges of the graph are defined as follows: E=, 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 algorithms on a simple KG (named *toy problem*), which is presented in Figure 2. The *toy problem* includes three diseases (represented by the nodes: ) and ten symptoms (represented by the nodes: ), so symptom 1 indicates disease 1, symptoms 2 and 3 indicate diseases 1 and 2, symptom 4 indicates diseases 2 and 4, symptom 5 indicates diseases 2 and 3, symptoms 6 and 7 indicate disease 3, symptoms 8 and 9 indicate diseases 4 and 5, and symptom 10 indicates disease 5.

Figure : The toy problem KG

3.2 Terminology

The following is the terminology that we use to describe the algorithms.

**Table 2:** definition of terms used in our algorithms

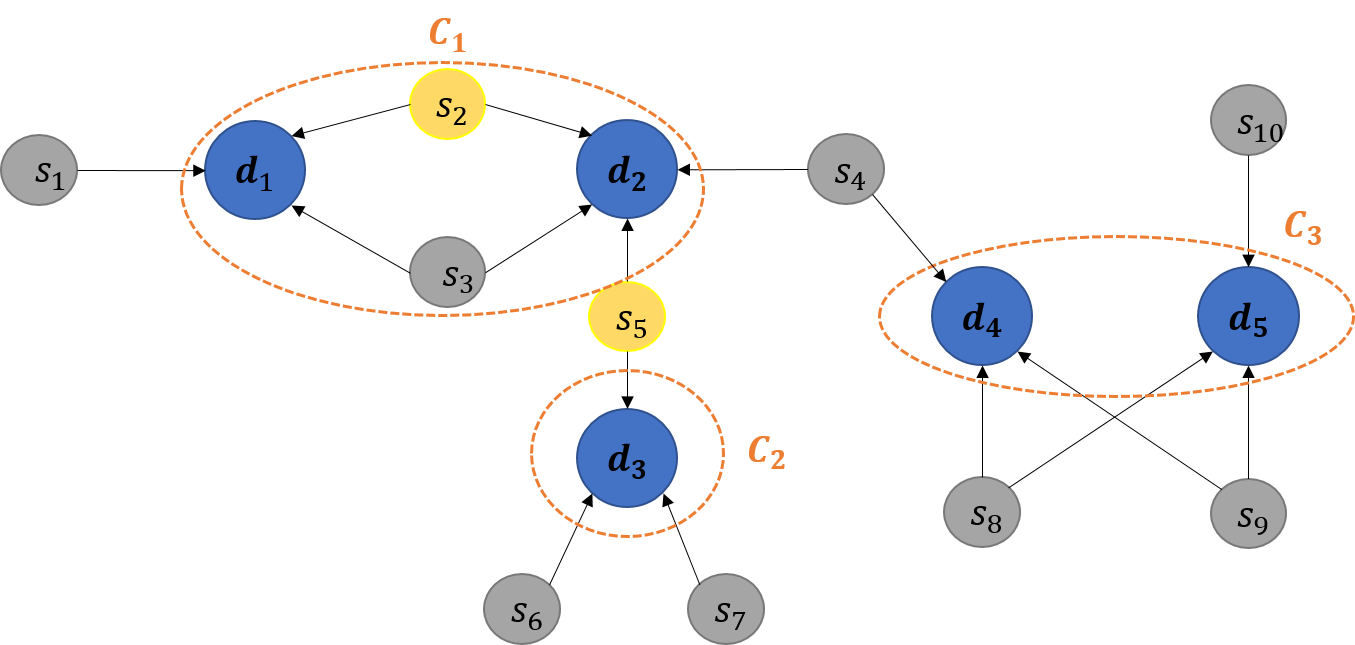
|  |  |
| --- | --- |
| **Term** | **Definition** |
| D | The set of diseases nodes |
| S | The set of symptoms nodes |
| ES | The set of evidence symptoms (i.e., the symptoms indicated by the patient) |
| C | The set of communities |
| |c| | The size of a single community.  Defined by the number of diseases that belongs to c |
| (s,c) | The symptom’s community rank of a given and  Defined by the number of edges that point from s to c |
| LinD(c) | The Local-in-Degree of a given .  Defined by the number of edges that point to diseases of c, by ES, hence, it is the sum of (s,c), for each and the given |
| PD's communities | The set of communities with a positive LinD(c), hence, a community in which at least one edge from points to c |
| (s,c) | (s,c)- |
| scs | A strengthened 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(s,c) |
| (d) | The Disease’s Symptoms Rank.  Defined by the number of symptoms the patient has that indicate D |

3.3 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 [Lu, et. al., 2015] (see more details in subsection 2.1). You can see below the pseudo-code of Algorithm 1.

Given the toy problem KG, represented in Figure 2, we present in Figure 3 the communities that were found on that KG. The respective SCM is presented in figure 4, for instance, SCM[][] = 1, since there is one edge pointing from to .

Algorithm 1: Disease Community Detection

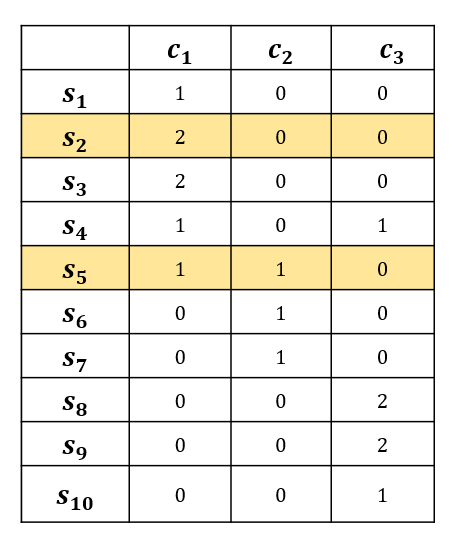
Input: Knowledge Graph .

Output: (1) For every , add a property named *community*, which determines the community d belongs to. (2) Symptoms community matrix (SCM), which is exhibited in Figure 4.

Algorithm:   
1. (preprocessing): for every two diseases such that , add . 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| matrix such that SCM[s,c]=(s,c).

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

**Figure 4**: 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.

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 , evidence symptoms .

Output: - set of possible diseases.

Algorithm:   
1. For every symptom s in *ES*:  
1.1 For every disease d such that *(s,d)E*:  
1.1.1 Add d to *PD*.  
2. Return

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 , the output of Algorithm 2 is , thus the *PD's* communities are and .

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 *strengthened community symptom* (scs). The answer to this question will help to determine whether the patient disease is one of the community diseases or not. You can see below the pseudo-code of Algorithm 3.

Based on the given (output by Algorithm 2) that resulted with and as the PD's communities, the respective is 3 and is 1 (i.e., is the sum of SCM[s2,c1]+SCM[s5,c1] as it is presented in figure 4). As c1 has the highest LinD, we calculate for each symptom with respect to and compared to . has the highest , as yields the maximum value (=2). Thus, the algorithm outputs c1 and as its respective scs and presents them to the domain expert.

Algorithm 3: Find the most probable community

Input: possible diseases , symptoms community matrix (SCM).

Output: scs 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 be the current community in the order.   
3. For every symptom in SCM(\_,c), calculate (s,c).   
4. Let (s’,c). If (s’,c)>0, return s' (i.e., scs) and c.   
Otherwise, return to step 2.  
5. Return null.

Algorithm 4: Find Strengthen 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 symptoms (questions), 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 indicating them (that is, ), and let s1 and s2 be two symptoms that strengthen them respectively. Then hypothesis h1 is before h2 in the order if .

(ii) Let h1 and h2 be two different hypotheses such that . 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 be an empty list.

2. Let be the list of diseases in c, sorted in a decreasing order by their .

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 .  
4. For each :  
 4.1. for each s' in S such that (d,s’) , add (d,s’) to R.  
5. Return R ordered by relevance.

Based on the previous output, let’s consider that 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 : (d1)=2 and (d2)=3. Thus, the sorted list includes . Then, the algorithm sorts the symptoms of c1 (excluding the evidence symptoms, in our case ), by their : and . Thus, the sorted list includes . Finally, the algorithm returns the sorted list , which includes the following pairs: [(s4,d2),(s1,d1)].

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

1. Case Study Scenario

To examine the proposed framework, in particular the use of the algorithms listed in section ‎3, we used a data set composed of patients’ records that were taken from Kaggle (described in subsection 4.1). We ran a sample scenario on the given data set, which is presented in subsection 4.2, followed by the results of the algorithms run using the sample scenario (in subsection 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 all the symptoms that can characterize the specific disease.

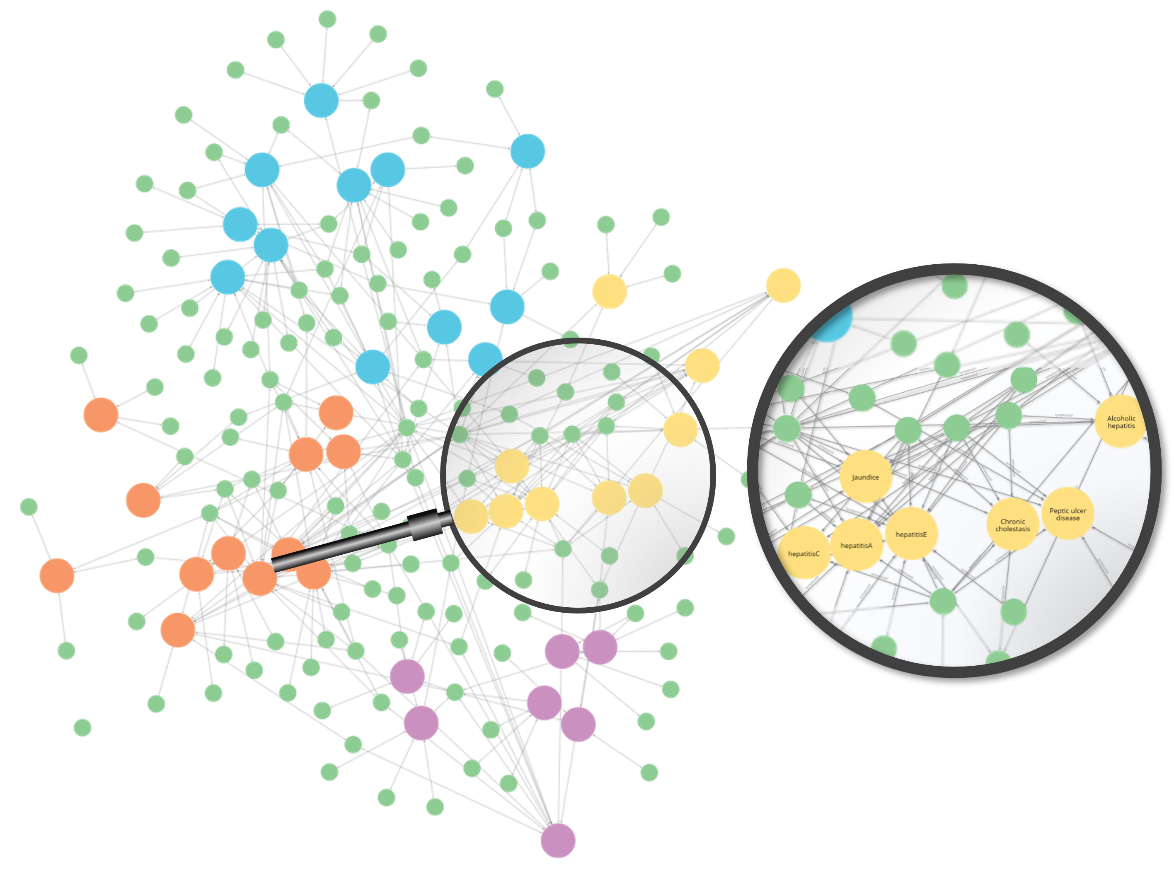
The number of disease symptoms ranges from 4 to 17. The data set included a total of 130 different symptoms. Some of the symptoms were unique and characterized one specific disease, while others were quite common and characterized various diseases.

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.

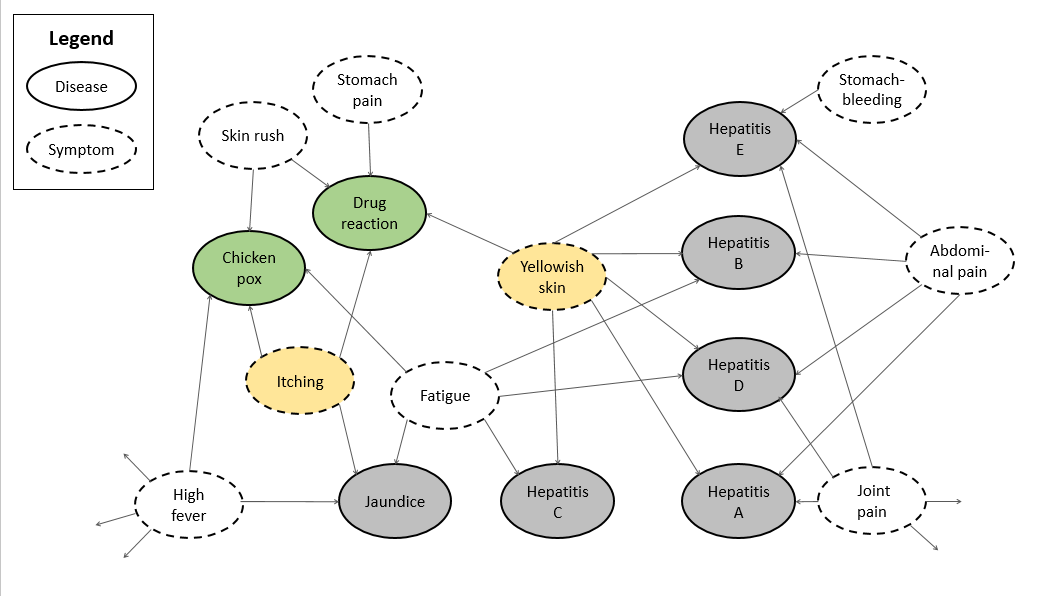
The knowledge graph was constructed by creating a node for each of the 41 diseases and 130 symptoms. We created an edge between a symptom node and a disease node if that symptom characterized the disease. Some of the symptom nodes characterize multiple diseases and thus have multiple connections.

After building the graph, we ran Algorithm 1, for community detection (recall, we used the Louvain method). Four communities were identified. Figure 5 exhibits the knowledge graph along with the detected communities. For clarity, each community is represented by a distinct color.

******Figure 5:** The detected communities on the knowledge graph (to be uploaded).

4.3 Scenario Description

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

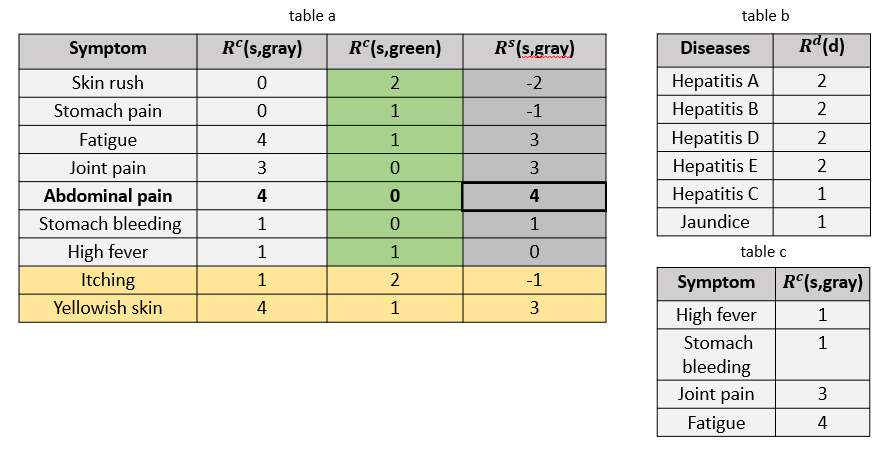
Figure 6 depicts a sub-graph derived from the KG, including the evidence symptoms (in yellow), and the relations of the symptoms (i.e., the diseases that these symptoms characterize). For display clarity, we present only some of the relations. In addition, Figure 6 presents two communities that were found by the community detection algorithm (Algorithm 1). The first community is colored in green and includes drug reaction and chickenpox, while the second community is colored in gray and includes hepatitis A-E and Jaundice.

**Figure 6**: The sub-graph derived from the KG.

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

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 green community. The LinD of the green community is 3 since three edges are pointing from the evidence symptoms (the yellow nodes) to the diseases of the green community (green nodes): itching pointing to two green nodes (chickenpox and drug reaction) and yellowish skin to one green node (drug reaction). Similarly, the LinD of the gray community is 6: 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) in order to suggest a symptom that best indicates this community. In other words, the algorithm searches for a strengthened community symptom. In fact, the scs is the symptom with the highest (s,c), given c is the gray community and compared to the other PD’s communities (in our case the green community). Thus, to find the respective scs, the algorithm calculates for each of the symptoms with respect to the gray community, as can be seen in Figure 7 (table a). We can see that the symptom with the highest relative to the gray community is abdominal pain. As so, Algorithm 3 outputs the gray community and its respective scs.

In the presented scenario, the patient has this symptom, and therefore, the gray community is strengthened. We can now continue to the last step and run

*Figure 7: Table A - the Symptom’s for each PD’s community along with the for the gray community; Table B - the diseases in the gray community with their ; Table C – the symptoms indicating the gray diseases with their*

Algorithm 4.

Algorithm 4 returns R, which is a list of sorted pairs (symptom, disease), such that the symptom indicates the disease and strengthens it. The gray diseases are sorted in a decreasing manner according to their and listed in Figure 7, Table B. In addition, the symptoms indicating these diseases are sorted increasingly according to their and listed in Figure 7, Table C. In our case study, the algorithm returns the sorted list that includes the pairs as they appear in the following table:

**Table 3**: The sorted list of hypotheses 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 |

1. Discussion

The world of decision-making is a world that is constantly evolving. This includes the sophistication to automate the decision-making process. This capability stems on the one hand from a collection in an organization that is constantly growing, and on the other hand from the technological development of machining. Within this evolving world, the present study examines decision-making processes that have 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 that is 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.

Hence, we can know these questions, thus refining the insights of the domain expert so that they can reach a decision.

Driven by this motivation, we developed an algorithmic framework that aims to help the domain expert 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-user.

The algorithmic framework consists of two parts. In the first part, a knowledge graph is constructed that characterizes the world of content. 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 pinpoint 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 (Musen et al. 2021). Medical diagnosis is a challenge that in recent decades has led to the development of methodologies and systems to support clinical decisions (Berg & Berg, 1997). 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.

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. In addition, compared to an exhaustive (naive) search in the knowledge graph, the proposed framework will, at best, return a fixed number of questions that do not depend on the number of testimonies and the size of the graph.

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

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

In addition, we would like 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.

Many researchers are passionate about exploring the potential of artificial intelligence to support decision-making, particularly within the clinical domain (Shortliffe & Sepúlveda, 2018). 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. Vasily 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 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 (Vasey, Ursprung, Beddoe, Taylor, Marlow, Bilbro & McCulloch, 2021).

1. References

Berg, M., & Berg, P. A. M. (1997). Rationalizing medical work: decision-support techniques and medical practices. MIT Press.

Bhatt, S., Padhee, S., Sheth, A., Chen, K., Shalin, V., Doran, D., & Minnery, B. (2019, January). Knowledge graph enhanced community detection and characterization. In Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining (pp. 51-59).

Buchanan, B. G., & Shortliffe, E. H. (1984). Rule-based expert systems: the MYCIN experiments of the Stanford Heuristic Programming Project.

Cimino, J. J., Patel, V. L., & Kushniruk, A. W. (2002). The patient clinical information system (PatCIS): technical solutions for and experience with giving patients access to their electronic medical records. International journal of medical informatics, 68(1-3), 113-127.

De Clercq, P. A., Blom, J. A., Korsten, H. H., & Hasman, A. (2004). Approaches for creating computer-interpretable guidelines that facilitate decision support. Artificial intelligence in medicine, 31(1), 1-27.

Dietz, L., Kotov, A., & Meij, E. (2018, June). Utilizing knowledge graphs for text-centric information retrieval. In The 41st international ACM SIGIR conference on research & development in information retrieval (pp. 1387-1390).

Elnagar, S., & Weistroffer, H.R. (2019). Introducing Knowledge Graphs to Decision Support Systems Design. SIGSAND/PLAIS.

Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., & Khan, S. U. (2015). The rise of “big data” on cloud computing: Review and open research issues. Information systems, 47, 98-115.

Hossayni, H., Khan, I., Aazam, M., Taleghani-Isfahani, A., & Crespi, N. (2020). SemKoRe: improving machine maintenance in industrial iot with semantic knowledge graphs. Applied Sciences, 10(18), 6325.

Gashkov, A., Perevalov, A., Eltsova, M., Both, A. (2022). Improving Question Answering Quality Through Language Feature-Based SPARQL Query Candidate Validation. In: , et al. The Semantic Web. ESWC 2022. Lecture Notes in Computer Science, vol 13261. Springer, Cham. <https://doi.org/10.1007/978-3-031-06981-9_13>

Guo, Q., Zhuang, F., Qin, C., Zhu, H., Xie, X., Xiong, H., & He, Q. (2020). A survey on knowledge graph-based recommender systems. IEEE Transactions on Knowledge and Data Engineering.

Li, X., Chen, C. H., Zheng, P., Wang, Z., Jiang, Z., & Jiang, Z. (2020). A knowledge graph-aided concept–knowledge approach for evolutionary smart product–service system development. Journal of Mechanical Design, 142(10), 101403.

Lu, Hao, Mahantesh Halappanavar, and Ananth Kalyanaraman. "Parallel heuristics for scalable community detection." Parallel Computing 47 (2015): 19-37.‏

Malik, K. M., Krishnamurthy, M., Alobaidi, M., Hussain, M., Alam, F., & Malik, G. (2020). Automated domain-specific healthcare knowledge graph curation framework: Subarachnoid hemorrhage as phenotype. Expert Systems with Applications, 145, 113120.

Musen, M. A., Middleton, B., & Greenes, R. A. (2021). Clinical decision-support systems. In Biomedical informatics (pp. 795-840). Springer, Cham.

Newman, Mark EJ, and Michelle Girvan. "Finding and evaluating community structure in networks." Physical review E 69.2 (2004): 026113.‏

Peleg, M., Tu, S., Bury, J., Ciccarese, P., Fox, J., Greenes, R. A., ... & Stefanelli, M. (2003). Comparing computer-interpretable guideline models: a case-study approach. Journal of the American Medical Informatics Association, 10(1), 52-68.

Omididan, Z., & Hadianfar, A. M. (2011). The role of clinical decision support systems in healthcare (1980-2010): A systematic review study. Jentashapir Scientific-Research Quarterly, 2(3), 125-34.

Osheroff, J. A., Teich, J. M., Middleton, B., Steen, E. B., Wright, A., & Detmer, D. E. (2007). A roadmap for national action on clinical decision support. Journal of the American medical informatics association, 14(2), 141-145.

Osheroff, J. A., Teich, J. M., Levick, D., Saldana, L., Velasco, F. T., Sittig, D. F., ... & Jenders, R. A. (2012). Improving outcomes with clinical decision support: an implementer’s guide. Himss Publishing.

Power, D. J. (2002). Decision support systems: concepts and resources for managers. Greenwood Publishing Group.

Power, D. J. (2007). A brief history of decision support systems. DSSResources.com, 3.

Saria, S., Koller, D., & Penn, A. (2010). Learning individual and population level traits from clinical temporal data. In Proceedings of Neural Information Processing Systems (pp. 1-9).

Sivarajah, U., Kamal, M. M., Irani, Z., & Weerakkody, V. (2017). Critical analysis of Big Data challenges and analytical methods. Journal of business research, 70, 263-286.

Shortliffe, E. H., & Sepúlveda, M. J. (2018). Clinical decision support in the era of artificial intelligence. Jama, 320(21), 2199-2200.

Sutton, R. T., Pincock, D., Baumgart, D. C., Sadowski, D. C., Fedorak, R. N., & Kroeker, K. I. (2020). An overview of clinical decision support systems: benefits, risks, and strategies for success. NPJ digital medicine, 3(1), 1-10.

Vasey, B., Ursprung, S., Beddoe, B., Taylor, E. H., Marlow, N., Bilbro, N., ... & McCulloch, P. (2021). Association of clinician diagnostic performance with machine learning–based decision support systems: a systematic review. JAMA network open, 4(3), e211276-e211276.‏

X. Xiang, Z. Wang, Y. Jia and B. Fang. (*2019)* "Knowledge Graph-Based Clinical Decision Support System Reasoning: A Survey,"  *IEEE Fourth International Conference on Data Science in Cyberspace (DSC)*, 2019, pp. 373-380, doi: 10.1109/DSC.2019.00063.