Inference over Knowledge Representations Automatically Generated from Medical Texts with Applications in Healthcare Robotics

Laura Brannan
University of Utah

UUCS-21-005

School of Computing
University of Utah
Salt Lake City, UT 84112 USA

20 April 2021

Abstract

Many nations across the world, including the United States, face an impending shortage of trained medical professionals and personnel. The development of a robotic healthcare assistant would help alleviate this ongoing shortage in healthcare workers. For a robotic healthcare assistant to be useful, it must facilitate human-like interactions and maintain contextual understanding of its environment. In this work, we take steps toward endowing healthcare assistant robots with the ability to anticipate the equipment needs of healthcare providers without being explicitly asked. We utilize an automatically formulated knowledge representation from web-based knowledge bases paired with a traversal algorithm to achieve these objectives. Equipped with a proper knowledge base and rule-based traversal algorithm, our robot will have the ability to retrieve relevant related information given a medical condition or symptom.

INFERENCE OVER KNOWLEDGE REPRESENTATIONS AUTOMATICALLY GENERATED FROM MEDICAL TEXTS WITH APPLICATIONS IN HEALTHCARE ROBOTICS

by

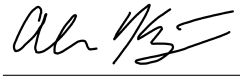
Laura Brannan

A senior thesis submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree

Bachelor of Computer Science

School of Computing
The University of Utah
May 2021

Approved:

	4/20/2021
Alan Kuntz Thesis Advisor	Date
H. James de St. Germain Director of Undergradute Studies	Date
Mary Hall Director of School of Computing	Date

This work was supported by funding from the Undergraduate Research Opportunities Program at the University of Utah awarded to Laura Brannan

ABSTRACT

Many nations across the world, including the United States, face an impending shortage of trained medical professionals and personnel. The development of a robotic healthcare assistant would help alleviate this ongoing shortage in healthcare workers. For a robotic healthcare assistant to be useful, it must facilitate human-like interactions and maintain contextual understanding of its environment. In this work, we take steps toward endowing healthcare assistant robots with the ability to anticipate the equipment needs of healthcare providers without being explicitly asked. We utilize an automatically formulated knowledge representation from web-based knowledge bases paired with a traversal algorithm to achieve these objectives. Equipped with a proper knowledge base and rule-based traversal algorithm, our robot will have the ability to retrieve relevant related information given a medical condition or symptom.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

1.1 INTRODUCTION

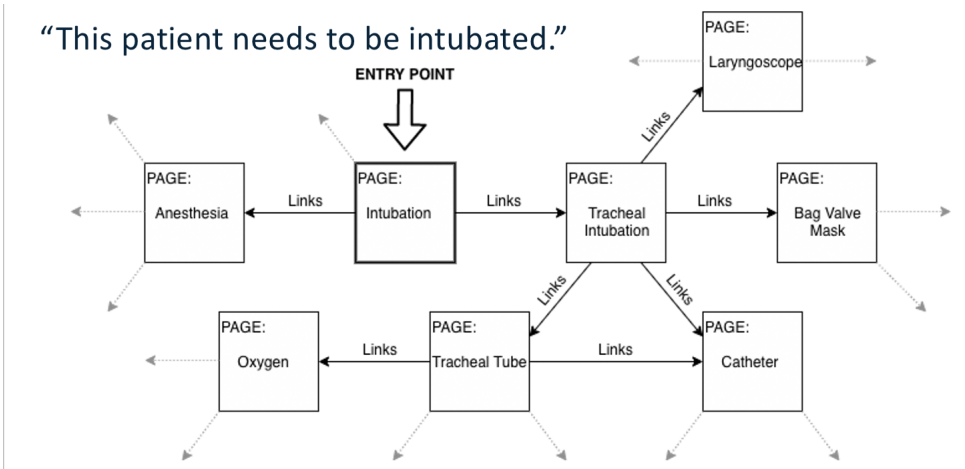
A crisis is fast approaching in the United States due to a shortage of nurses and other healthcare providers. It is estimated that the number of registered nurses entering the workforce needs to double in 2020-2021 to address a predicted shortage of 1.1 million nurses in 2022 [19]. Recent advances in artificial intelligence and robotics may open the door for robots to help mitigate this crisis by acting as assistants to healthcare providers, reducing the providers' per-patient workloads, and enabling them to help more patients. However, due to the complex, ever-changing nature of the medical field, it is infeasible to pre-program a robot with the ability to properly handle every possible medical scenario. Consequently, an effective robot healthcare assistant must be well-informed and adaptive.

Human healthcare assistants adapt to unique scenarios by using context and accumulated knowledge to act intelligently. However, robots do not have the same ability to apply semantic knowledge and contextual details to make inferences. For example, if a healthcare provider says "This patient needs an IV.", their assistant would know this implies they need to go retrieve the proper supplies to do so. This includes knowing both what equipment is involved in placing an IV and where these supplies are located. While this is obvious to a human assistant, it is not at all obvious to a robot. For a robot to behave in such a way it would need to be able to meaningfully and intelligently reason over some computational representation of medical knowledge. Such a computational representation of knowledge can be effectively formulated from existing structured content on the internet and then efficiently reasoned over, enabling a robotic healthcare assistant to respond and operate in a more applicable and intelligent manner. We make meaningful progress toward this goal in two ways: (i) by leveraging existing web-based knowledge systems as computational

representations called knowledge graphs and (ii) developing an algorithm for rule-based inference on the knowledge graph to identify medical supplies needed by a human health-care professional as they treat a patient, so that a robot assistant may proactively retrieve them.

The first step is providing the robot with a knowledge base that includes information about the healthcare domain in which it is working. There is an incredibly vast, and growing, online literature base relating to medical procedures, diagnoses, and treatments. In order to utilize this information to its fullest extent, the construction of an effective knowledge graph must be automated. A knowledge graph is composed of nodes, or concrete concepts, and edges, the thematic relations between two concepts. A knowledge graph is an ideal technical tool as it facilitates knowledge retrieval and recommendation. In this work, we leverage existing online knowledge systems, such as Wikipedia, as an implicit knowledge graph, wherein the pages represent nodes and links between pages represent edges and encode a close conceptual relationship.

The next step is utilizing this implicit knowledge graph to perform contextual reasoning. An efficient healthcare assistant robot must be able to traverse the graph it is equipped with in order to obtain relevant information. In this work we use the knowledge graph to identify specific medical equipment required by a healthcare provider. With the returned set of medical supplies, the robot would then be capable of retrieving this equipment from a fixed location without the healthcare provider ever explicitly asking. Our robot takes a single procedure or condition as input and determines what equipment is relevant to performing the procedure or treating the condition. A robot equipped with a substantial knowledge graph and a meaningful traversal algorithm has the means to act as an assistant



Available Supplies		
Antiseptic	Catheter	Albuterol
Cannula	Epinephrine	Anesthesia
Tracheal Tube	...	Oxygen
Oxygen Tubing	Syringe	Bag Valve Mask
Blood Thinners	Surgical Masks	Bandages

Figure 1.1: An example of a scenario a robotic healthcare assistant may encounter and a visualization of the information to be gained through our proposed method. In this case, "Intubation" is used as the initial procedure and entry point into the conceptual knowledge graph. The robot's traversal from this page can be visualized by following edges in the above graph. After this traversal, the touched pages are mapped to a set of available supplies. The table shows a portion of these available supplies, and highlights any matches found in green.

to a healthcare provider.

Consider a patient experiencing respiratory distress. A provider treating said patient might address this symptom, depending on the severity, by saying, "This patient needs to be intubated." Our robot will recognize that "intubation" is the procedure to be performed on the patient and identify the associated node in the knowledge graph. As seen in Figure 1, the intubation page will then be used as the entry point into the graph. Figure 1 also depicts a local portion of the knowledge graph around the intubation page, along with the equipment

that might be retrieved based on the exploration of these pages. The algorithm explores pages until it reaches a specified number of steps away from the entry page. Looking at Figure 1, we would say that the anesethesia page is one step away from the intubation page, and the catheter page could be two or three steps away depending on the path taken. We also assign a score to each page to indicate relevance. While traversing, any links that have a score greater than a set threshold will be queued to be visited. On each step the algorithm adds the link occurrences from new pages to the current scores and checks for any that need to be explored. A number of links with the top scores are then mapped to a set of equipment available to the robot. Figure 1 shows a table with an example result of such a mapping that identified a tracheal tube, anaesthesia, catheter, and bag valve mask as relevant equipment for intubation. By automatically generating medically focused knowledge graphs from web-based sources and developing pertinent traversal algorithms for knowledge retrieval, this project takes steps toward making an intelligent and adaptive robot healthcare assistant a reality.

2.2 BACKGROUND

Autonomous robotic assistants have been an area of interest for research since the early 2000's due to their extensive range of potential applications. In as early as 2001 there was work being done on a surgical robotic assistant that had the capacity to work manually, partly autonomously with shared control, or fully autonomously under supervision of a surgeon [10]. There has also been work done towards a socially interactive robotic assistant that would autonomously provide reminders and guidance for elderly residents of nursing homes [16]. In the medical field these robotic assistants are often used to assist and improve the precision of minimally invasive surgeries [11, 14, 17]. Robotic assistants have also been envisioned at the patient's bedside, listening for explicitly requested equipment from a healthcare provider [3]. These previous works demonstrate the value of an effective autonomous healthcare assistant, however in their current state they are limited in their ability to automatically reason about medical context. This motivates further work in automating the construction of a knowledge base that facilitates contextual reasoning for a robotic healthcare assistant in dynamic medical scenarios.

Equipping a robotic healthcare assistant with a meaningful knowledge representation is a key part of enabling knowledge retrieval and recommendation. A particularly promising knowledge representation is the knowledge graph. There have been many different approaches to knowledge graph generation [2, 6, 15, 20, 21]. The generation of knowledge graphs should be automated due to the vast amount of electronic medical information available and the shortage of medical experts to manually construct the graph. One of the challenges faced when attempting to perform this knowledge aggregation automatically is that related resources are frequently scattered and disconnected, creating what is known as

"knowledge islands." Constructing a knowledge graph is the perfect solution to knowledge islands as it can capture and present the intricate relationships between entities, even across information systems. In the field of Traditional Chinese Medicine (TCM), Yu et al. perform an initial, manual population of a database into a knowledge graph and use an automated text-mining tool to extract entities and relations through semantic analysis on texts—though these extractions must be validated by an expert [20]. This approach requires significant human expert effort not only in validating the extractions, but also in rigorously examining and clarifying data sources and entering them into the database for the initial construction of the graph.

The construction of a knowledge graph to achieve things like knowledge retrieval, question-answering, and knowledge recommendation has been studied in many other medical domains. One approach to simplify this problem is to construct the knowledge graph over a specific disease or topic, such as chronic obstructive pulmonary disease (COPD) or knee osteoarthritis [6, 12]. The approach in [6] constructs the knowledge graph fully automatically and then performs feature selection to select the most optimal subset of these COPD specific relations. The resulting COPD diagnosis tool performed well in testing, however, significant human effort was required on top of the constructed knowledge graph to make it useful and reliable. Similarly, the approach in [12] performs well, but requires a manually labelling and construction of a domain ontology of knee osteoarthritis from which to generate the knowledge graph.

Constructing knowledge graphs over a broader range of topics or diseases is inherently a more difficult task. Incorrect, irrelevant, and duplicate or synonymous relationships are frequently extracted. Some works have utilized a semi-automatic approach, augmenting the

efficiency of a machine with the accuracy of experts in the field. This helps to overcome issues with natural language processing; such as complex concepts and relationships, heterogeneous data structures, poor data quality, etc. The Health Knowledge Graph Builder uses this human-augmented technique, and quantifies the clinician effort to construct a knowledge graph over just the cardiovascular domain to be about 2,158 days [21]. There have even been human curated health knowledge graphs constructed for the purpose of performing diagnosis or classification [1, 5, 9, 13]. Knowledge graphs constructed with human validation and annotations are typically more accurate, however, they are often too costly as they have to rely on a valuable and scarce resource—human experts.

The use of knowledge graphs extends to non-medical fields as well. IBM's Watson uses a knowledge graph, which enabled Watson to win at Jeopardy! by retaining enormous amounts of information, simulating strong language skills, understanding what is being asked, and accurately determining the likelihood of an answer [8]. Another application is in accident investigations. One study constructed a knowledge graph by reusing the reasoning knowledge from documents written by human experts on past accidents to extract causality for some event. This was achieved using manually constructed rules and patterns that specifically target and extract causal relationships from text [18]. This solution may be a good approach over narrow or restricted topics, but by design is limited in its scalability by the availability of both human experts and detailed analysis reports authored by experts.

Regardless of how the graph is constructed, an algorithm is needed to facilitate traversal and information retrieval. It is challenging to traverse knowledge graphs due to difficulties such as ambiguity in traversal queries, language considerations such as synonyms and variation in dialect, inherent computational complexity, and a lack of prior knowledge of

the underlying graph data. There are a variety of different techniques to refine a knowledge graph, such as association rule mining for predicting relations and statistical methods to identify potential errors [15]. Refining a knowledge graph generates more relevant results, allowing for a less intelligent traversal algorithm to still perform well. Rules can also be added to the traversal algorithm in order to provide knowledge recommendations, or fill in missing pieces of information in the graph. Inferring relationships is very important in facilitating more human-like interactions with robots. One approach achieves inference by parsing instructions into verb frames, finding the incomplete frames, and completing them by finding the highest probability among all possible combinations [4]. This results in a much more robust representation of knowledge for the robot and the ability to act in more potential scenarios. There are many different methods to construct a knowledge graph, and even more methods for traversal and inference. It is no trivial task to provide a robot with knowledge and the ability to reason as a human would, and there are many different avenues in achieving this feat left to be explored.

In this work, we build upon these existing concepts by first formulating existing online knowledge repositories as implicit knowledge graphs. We then present an algorithm that traverses the implicit graph to infer related medical equipment in anticipation of healthcare providers' needs.

3.3 METHODS

We bypass the need to explicitly construct a knowledge graph by noting the knowledge graph structure implied by the page connectivity of existing web-based knowledge systems such as Wikipedia, i.e., its pages and how they internally link to each other. We note that in this formulation, each page represents a node in an implicit graph with links between pages representing edges. The resulting implicit knowledge graph provides context for our robot healthcare assistant, enabling it to identify relevant medical equipment based on a given medical procedure or condition. Our method takes an individual medical procedure or condition as input and uses this as an entry point into the knowledge graph. We use defining characteristics of the structure of the knowledge graph to formulate a traversal algorithm that returns a set of objects related to the entry node. For example, since the knowledge graph structure is based on linking directly related concepts, we can use the distance between nodes to depict the strength of their relatedness. This is known in linguistics as the phenomenon of co-occurrence. We also provide our robot with a dictionary of available equipment and their locations. Then, after the traversal has been completed and a set of pertinent objects has been returned, it is just a matter of mapping these to the set of available equipment to see which are available for our robot to fetch. See Fig. 1.1 for an overview.

Algorithm 1 shows the pseudo-code for the driver method, `GetEquipment`. This function takes as input four parameters that are used to shape the traversal, $entry\_page$, num_steps, $threshold$, and $scoring\_func$. The $entry\_page$ is a single condition or procedure that corresponds to a Wikipedia page. The `TakeSteps` and `Traverse` functions are then to explore breadth-first from the $entry\_page$ a set number of steps by following internal

Algorithm 1: Pseudo-code for the driver method that initiates the traversal and maps the top 100 results to the set of available equipment.

```
1 Function GetEquipment(entry_page, num_steps, threshold,
    scoring_func):
      Input : The Wikipedia page on which to start the traversal, entry_page, the
              num_steps to take from that page,the threshold at which to explore
              pages, and the scoring_func to be used. These are manually set by
              the user.
      Output: A list of potentially relevant, and available, medical equipment.
2     results = TakeSteps({}, num_steps, 0, threshold, scoring_func);
3     top_100 = results.sort()[:100];
      // MapToSupplies reads, parses, & compares each line of the
         supply closet with the input list, returning matches
4     return MapToSupplies(top_100);
```

links to other pages. Every link followed counts as a step, and the traversal ends after num_steps calls to `Traverse`. The $threshold$ determines how many pages are explored in each step by setting a value that the page score must exceed. The $scoring\_func$ is used to weight pages differently at different points in the traversal. This enables, for example, weighting the links seen on directly related entity pages higher than links seen on pages 2 steps away. In order to achieve this, we could modify the scoring function from $scoring\_func = 1 * curr\_score$, to something like $scoring\_func = \frac{1}{curr\_step} * curr\_score$. This would help to reduce the impact of irrelevant links seen on further away pages, and boost the scores of close pages that are more likely to be relevant. We utilize $curr\_step$ as a scaling parameter in $scoring\_func$ because the number of steps away from the entry page directly relates to a decrease in relevance, making it a crucial factor in generating a page's score. This parameter is passed along to the `TakeSteps` and `Traverse` function, along with the num_steps and $threshold$.

Algorithm 2 shows the pseudo-code for a recursive helper function that drives the traversal.

Algorithm 2: Pseudo-code for a recursive method that will take num_steps into the graph, determining traversal order by a page's score.

```
1 Function TakeSteps(page_scores, num_steps, curr_step, threshold,
    score_func):
      Input  : The current set of pages and their scores, page_scores, as well as the
               total num_steps desired and the curr_step being performed.
      Output: The set of pages and their updated scores after the full traversal has
               been performed.
2     if curr_step > num_steps then
3         return page_scores;
4     else
5         updated_scores = Traverse(page_scores, threshold, score_func);
6         return TakeSteps(page_scores, num_steps, curr_step + 1, threshold,
                scoring_func);
7     end
```

This function will check that another step needs to be taken, and if so, makes the call to `Traverse` which visits all input pages with a score greater than or equal to the $threshold$. The specifics of the `Traverse` function are shown in Algorithm 3.

Algorithm 3 shows the pseudo-code for the traversal function. `TakeSteps` passes all of the necessary information to the function, including the dictionary of $page\_scores$ from the previous step, the $threshold$ at which to visit pages, and the $scoring\_func$ to be used. The list of potential pages to explore in this step is found by selecting any pages from $page\_scores$ with a score that is greater than or equal to the threshold. The $threshold$ is used in order to maintain the relevance of the results being explored, since irrelevant pages may be linked once on a Wikipedia page, but are not going to be linked multiple places. These high-scoring pages are explored one by one, but only if they have not yet been visited. A page is considered to have been "visited" once our algorithm follows the link to the page and scrapes it for link elements. This is performed in the call made to

Algorithm 3: Pseudo-code for the Traversal function. Given a set of pages and their scores, a page scoring function, and a threshold to guide the traversal path, returns a dictionary of the pages seen and their scores. Called recursively by `TakeSteps`.

```
1  Function Traverse(page_scores, threshold, scoring_func) :
       Input  : A dictionary of pages and their current scores page_scores, the
                threshold at which to explore pages, and the scoring_func to be
                used.
       Output: A dictionary of potentially relevant pages (medical equipment) and
                their scores, updated_scores
2      toExplore = page_scores where score ≥ threshold;
3      updated_scores = {};
4      for current in toExplore do
5          if !current.visited then
6              curr_scores = current.getLinkedPages(scoring_func);
7              updated_scores.addOrUpdate(current, curr_scores);
8              visited.add(current);
9          end
           // Do nothing if the page has already been visited
10     end
11     return updated_scores;
```

`getLinkedPages`, which takes the $scoring\_func$ as input. In `getLinkedPages`, before the dictionary of results is returned, the $scoring\_func$ is applied to the page scores. The resulting page score dictionary (see $curr\_scores$ on line 6 of Alg. 3) contains all the links found on the $current$ page as well as its score. If the $scoring\_func = 1$ then the page score is entirely based on the occurrences of a link on the page. A single link on any given page is typically found somewhere between 0 and 5 times. We visit the Wikipedia page, find all link elements, aggregate the number of occurrences of each link into a score, and finally apply the $scoring\_func$ for the dictionary of $page\_scores$ for the current page. These scores are added into the current working page-score dictionary for the traversal. If the page does not yet have a score, it is simply added. If the page already has a score, the two are summed for the new score. The page-score dictionary for the traversal is then

returned—to be used for the next step or to be mapped to available supplies.

The number of steps, $threshold$, and scoring function parameters allow for a fine tuning of our algorithm and the results it produces—regardless of the knowledge base it's equipped with. The pages that are seen and referenced the most across the traversed pages will have the greatest scores, and should be the most relevant to the entry page.

4.4 RESULTS

Our method achieved multiple promising results on different example procedures/conditions. For each entry point into our graph we mapped the results to a supply closet generated from the "Comprehensive List of Basic Medical Supplies" by the Bureau of Industry and Security [7]. A set of available equipment was parsed from this document and manually revised so that groups of equipment are separated into individual pieces of equipment. The resulting set of equipment, used as a conceptual supply closet, is shown in Table 4.1. The output of this mapping is the list of supplies that our robotic healthcare assistant would be able to fetch. Table 4.2 and 4.3 show items successfully identified by our method as potentially relevant for four different conditions or procedures using different sets of parameter values. In Table 4.2, results were achieved with only 2 or 5 steps, and a simple implementation of a score decrementing function where the scores of links seen on the entry page are doubled. That is, on the entry page links are weighted as $2 * link\_occurrences$ while links seen on other steps contribute $link\_occurrences$ to a links overall score. This heavy weighting of links on closer pages, paired with a threshold that increases as the distance from the entry page increases, ensures that we do not explore too far breadth-wise into the knowledge graph and obfuscate our results. After each traversal in Table 4.2 and 4.3, the pages with the top 100 scores were searched for in the set of available supplies to yield the results recorded below.

As shown in Table 4.2, taking 2 or 5 steps into the graph yields the same results for anaphylaxis, intubation, and IV. Examining the path taken for each traversal revealed that for both anaphylaxis and intubation there were no additional pages visited in steps 3, 4 or 5. Both traversals starting at the anaphylaxis page visited a total of 12 pages and returned

Table 4.1: Comprehensive List of Basic Medical Supplies

Syringe	Ear Plugs	Bandages	IV Stand
Cannula	Ear Muffs	Gauze	Instrument Stand
Needle	Otology Sponges	Medical Tape	Heating Pad
Catheter	Ear Syringes	Surgical Sutures	Ice Pack
Catheter Kit	Clinical Swabs	Surgical Staples	Nitrile Gloves
Coils	Clinical Applicators	Suture Removal Kit	Surgical Mask
Guidwire	Specimen Collector	Staple Removal Kit	Apron
Medical Tubing	Urine Container	Tourniquet	Medical Adhesive
Laryngoscope	Antiseptic Wipes	IV Tubing	Adhesive Remover
Laproscope	Iodine Wipes	IV Sugar Solution	Capnograph
Sinuscope	Splint	Tegaderm dressing	CPAP Machine
Blood Pressure Gauge	Cane	Lactated Ringers	Medical Flowmeter
Blood Pressure Cuff	Crutches	Saline	Ventilator, Adult
Glucose monitor	Wheelchair	Antiseptic	Non-rebreather mask
Defibrillator	Walker	Anesthesia	Oxygen
Stethoscope	Pillow	Topical Anesthetic	Oxygen Tank
Speculum	Blanket	Thermometer	Oxygen Tubing
Medical Scissors	Sheets	Bedpan	Pulse Oximeter
Forceps	Gown	Emesis Bag	Spirometer
Blood Lancets	Scrubs	Syringe Aspirator	Nebulizer
Endotracheal tube	Patient Socks	Bladder Scanner	Ventilator, pediatric
Tracheal Tube	Surgical Shoe Covers	Urostomy Pouch	Scoliometer
Nasal Cannula	Bag valve mask	Pads	Goniometer
Nasogastric Tube	Ventilation Face Mask	Tampons	Hand Sanitizer
ECG machine	PEEP Valve	Enema Set	Medical Penlight
EKG machine	Stents	Stockinettes	Scalpel
12-lead Wires	Lubricant	Surgical clean-up kit	Clamps
4-lead Wires	Epinephrine	Benadryl	Surgical Clips
Cardiac Pacemaker	EPI Autoinjector	Albuterol	Ranitidine
Cardiac Monitor	Norepinephrine	Famotidine	Zantec
Ant-acids	Aspirin	Morphine	Antidepressants
Acid Suppressants	Blood Thinners	Nitroglycerin	Cortisone

3 relevant pieces of equipment. The intubation page, however, only visited one page in both traversals—Tracheal Intubation. Despite only visiting one additional page, 6 of the 7 medical supplies returned were relevant to intubating a patient. The 5 step traversal from

Condition	Anaphylaxis		Chest Pain	
Steps	2	5	2	5
Supplies to Fetch	Epinephrine EPI Autoinjector Norepinephrine	Epinephrine EPI Autoinjector Norepinephrine	Electrocardiogram Aspirin	Electrocardiogram
Condition	Intravenous (IV) Therapy		Intubation	
Steps	2	5	2	5
Supplies to Fetch	IV Sugar Solution Saline Cannula Oxygen Nasal Cannula	IV Sugar Solution Saline Cannula Oxygen Nasal Cannula	Catheter Anesthesia Tracheal Tube Topical Anesthetic Bag Valve Mask Oxygen Epinephrine	Catheter Anesthesia Tracheal Tube Topical Anesthetic Bag Valve Mask Oxygen Epinephrine

Table 4.2: Equipment correctly identified by our method with a linear score function. The above performance was achieved using 2 or 5 steps, a page score of $2 * link\_occurrences$ on directly related pages, and a threshold of 5 for the first step and $(steps\_away * 5) + 5$ for subsequent steps. Pages with the top 100 scores were mapped to the set of retrievable supplies in all of the above scenarios.

the IV page visited just one additional page than the 2 step traversal, for a total of 24 pages. Again, there were no additional pages visited in steps 4 or 5 for this 5 step traversal. The IV traversals both returned 5 pieces of equipment, of which 3 were relevant. The difference in the 2 step and 5 step traversal from the chest pain page was much larger than the other scenarios, visiting an additional 246 pages in steps 3, 4 and 5 for a total of 289. Taking 2 steps from the chest pain page resulted in 2 pieces of relevant equipment, while taking 5 steps only resulted in 1 piece of relevant equipment.

Table 4.3 shows results for 2 and 5 step traversals of the same set of 4 conditions and procedures as Table 4.2. However, to achieve these results the threshold was $2 + curr\_step^2$ and the scoring function $link\_occurrences * (1/curr\_step)$. Removing the heavy weighting of

Condition	Anaphylaxis		Chest Pain	
Steps	2	5	2	5
Supplies to Fetch	Norepinephrine Epinephrine Oxygen	Norepinephrine Epinephrine Cardiac Pacemaker	Aspirin Electro-cardiogram	Aspirin
Condition	Intravenous (IV) Therapy		Intubation	
Steps	2	5	2	5
Supplies to Fetch	IV Sugar Solution Saline Oxygen Cannula		Catheter Tracheal Tube Anesthesia Topical Anesthetic Bag Valve Mask Oxygen Epinephrine	Catheter Tracheal Tube Anesthesia Topical Anesthetic Bag Valve Mask Oxygen Epinephrine

Table 4.3: Equipment correctly identified by our method with a nonlinear score function. The above performance was achieved using 2 or 5 steps, a page score of $link\_occurrences*(1/curr\_step)$ for all pages, and a threshold of $2 + curr\_step^2$. Pages with the top 100 scores were mapped to the set of retrievable supplies in all of the above scenarios.

directly related entities and lowering the threshold allows our algorithm to explore further into our conceptual graph. While this did not change the results for the intubation page, we saw small variations in the anaphylaxis and chest pain equipment and a significant change in results from the IV page. The intubation page, which contains much less content than the others, takes the same traversal in this trial as the trial from Table 4.2 and yields the same set of relevant results. With these new parameters the anaphylaxis page only resulted in 2 relevant pieces of equipment from the 3 returned. Taking 2 steps returned oxygen rather than an epinephrine auto-injector as in Table 4.2, and when taking 5 steps a cardiac pacemaker is instead returned. The results for chest pain in Table 4.3 are very similar to those in Table 4.2, however, in 4.3 the 5 step traversal explores hundreds of more pages on pharmaceutical drugs that were not seen in 4.2. Using this set of parameters to

traverse from the IV page was much less successful with more steps. With only 2 steps there were 3 relevant pieces of equipment in the 4 returned. When the traversal expands to 5 steps we explore around 1,500 additional pages than in the 2 step traversal. The resulting top 100 pages had become completely irrelevant, including “Freediving,” “Scuba Schools International,” “Diving Bell,” and “Sheck Exley”.

Every trial, except taking 5 steps for IV in Table 4.3, was able to identify at least one piece of relevant equipment. There were cases in which we failed to return some pieces of equipment that would be expected by a healthcare provider. For example, to place an IV you need IV tubing and a syringe. Neither of these supplies were included in the results for either set of parameters. Additionally, there were cases in which pieces of equipment were returned that would not be needed by a healthcare provider in that situation. For example, the results in both tables returned epinephrine as standard equipment for performing intubation when it is not. In Table 4.2, a nasal cannula is returned for both IV trials but is not necessary to place an IV.

5.5 CONCLUSION

In this thesis we propose that equipping a robot with a relevant knowledge graph and traversal algorithm can provide context for a robotic healthcare assistant in healthcare scenarios. Healthcare providers go through extensive training to become experts in their fields and their time is very valuable. An efficient and autonomous robotic healthcare assistant could help to alleviate the shortage in healthcare workers currently seen in the United States. To help achieve this, we propose endowing a robot with the ability to reason about equipment necessary for medical procedures by equipping it with the proper knowledge system. Our work uses web-based knowledge bases as an automatically formulated implicit knowledge graph, leveraging internal page linking systems and the phenomenon of co-occurrence. The achieved results show that our algorithm is able to successfully extract one or more pieces of relevant equipment that can be retrieved given a medical condition or procedure as input.

It is clear from the trials completed in this work that the content used to create the knowledge graph quickly becomes a limiting factor in the relevancy of results. The parameters used to compute the results in Table 4.1 and 4.2 produced noticeably different traversal paths, but extremely similar results. When examining the top 100 pages that are mapped to the set of available supplies, it shows that the actual results of the traversal differ more than is immediately apparent. However, because we are using texts from a general knowledge source instead of a medically focused knowledge base, there are inherently fewer nodes representative of the medical equipment we are interested in. For example, if a patient is experiencing chest pain a healthcare provider would ensure that they have a way to measure and monitor their blood pressure. In all of the above trials for chest pain, “blood pressure” was one of the pages being mapped to the supply closet. While our set of supplies

includes “blood pressure monitor” and “blood pressure gauge,” our robot was only made aware of the idea of blood pressure, and is not equipped with the ability to reason about the results returned from the traversal—just to search them for equipment the robot has access to. Additionally, due to the structure of Wikipedia, some pages contain far less information and link very few pages. This makes it more difficult to tune a set of parameters (number of steps, threshold, and scoring function) that will generalize to all entry pages. The intubation page is like this, containing only 40 conceptual edges to other related pages. On the other hand, the IV page links 322 related pages and the tracheal intubation page explored in the intubation traversal links 590. When replacing the threshold in Table 4.2 with $2+curr\_step$, so that the intubation traversal is able to explore more than one page, taking 5 steps from the IV pages explores over 21,000 pages.

In the future, we intend to build upon the methods outlined in this paper by augmenting the knowledge graph with more medically focused texts and encoding more complex relationships between entities in the graph. Additionally, our method currently is provided with the condition or procedure as input, and in the future we envision automatically identifying this by listening to the providers spoken words. Augmenting our knowledge base with medically focused texts will help to eliminate the limiting factors discussed above. Parsing more complex relationships from these texts and encoding them into our knowledge graph will allow for a more intelligent and intentional traversal of the graph. To achieve this we could, for example, use a natural language parser to perform relationship extraction and semantic parsing to retrieve a more detailed encoding of the relationship between two objects. Our knowledge graph would then have different relationships encoded into edges, more than just the “is related to” relationship that can be inferred in our implicit web-based knowledge graph. Then as the graph is being traversed we would have additional information about

the relationship they share. Consider a graph that is connected based on the relatedness of two objects without regard for the context in which they are related. Entering the graph at "anaphylaxis" could reasonably return the following set of entities: [allergy, epinephrine, hypotension, vasodilation, penicillin]. Our robot assistant would detect that it has access to epinephrine and penicillin and retrieve them both. However, penicillin has shown up here because it is a common trigger for anaphylaxis, not a treatment. To avoid this behavior, a traversal algorithm could prioritize following edges that encode relationships like "is a treatment for," "supply with," "is treated by," etc., rather than following edges based on how many times they have been seen thus far. Taking into account the relationship an edge encodes when traversing is comparable to how graph search algorithms use associated edge weights to make more educated decisions. A robotic assistant with or without this additional context embedded into its knowledge graph will have the ability to assist a human provider in a way that has not been accomplished before.

The work presented in this thesis makes meaningful progress toward endowing an autonomous robotic healthcare assistant with the ability to anticipate equipment needs of healthcare providers. This may allow experts in the healthcare field more time with their patients, enhancing the quality of care they provide and improving patient outcomes.

REFERENCES

[1] G. Octo Barnett, James J. Cimino, Jon A. Hupp, and Edward P. Hoffer. DXplain: An evolving diagnostic decision-support system. *JAMA*, 258(1):67–74, 1987.

[2] Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. COMET: Commonsense transformers for automatic knowledge graph construction. *ACL*, 2019.

[3] E. Carpintero, C. Pérez, R. Morales, N. García, A. Candela, and J. Azorín. Development of a robotic scrub nurse for the operating theatre. *3rd IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics*, pages 504–509, 2010.

[4] Haonan Chen, Hao Tan, Alan Kuntz, Mohit Bansal, and Ron Alterovitz. Enabling robots to understand incomplete natural language instructions using commonsense reasoning. *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[5] Irene Y. Chen, Monica Agrawal, Steven Horng, and David Sontag. Robustly extracting medical knowledge from EHRs: A case study of learning a health knowledge graph. *Pacific Symposium on Biocomputing*, 2019.

[6] Y. Fang, H. Wang, L. Wang, R. Di, and Y. Song. Diagnosis of COPD based on a knowledge graph and integrated model. *IEEE Access*, 7:46004–46013, 2019.

[7] Million Gebreyesus. Comprehensive medical supplies updated list - 2014, 2014.

[8] Alfio Gliozzo, Or Biran, Siddharth Patwardhan, and Kathleen McKeown. Semantic technologies in IBM watson. In *Proceedings of the Fourth Workshop on Teaching NLP and CL*, pages 85–92, 2013.

[9] D. Hou, Z. Zhao, and S. Hu. Multi-label learning with visual-semantic embedded knowledge graph for diagnosis of radiology imaging. *IEEE Access*, 9:15720–15730, 2021.

[10] Hyosig Kang and J. T. Wen. EndoBot: a robotic assistant in minimally invasive surgeries. *IEEE International Conference on Robotics and Automation (ICRA)*, 2:2031–2036 vol.2, 2001.

[11] Hyosig Kang and J. T. Wen. Robotic knot tying in minimally invasive surgeries. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2:1421–1426 vol.2, 2002.

[12] Xin Li, Haoyang Liu, Xu Zhao, Guigang Zhang, and Chunxiao Xing. Automatic approach for constructing a knowledge graph of knee osteoarthritis in chinese. *Health Information Science and Systems*, 8(1):12, 2020.

[13] Xuedong Li, Yue Wang, Dongwu Wang, Walter Yuan, Dezhong Peng, and Qiaozhu Mei. Improving rare disease classification using imperfect knowledge graph. *BMC Medical Informatics and Decision Making*, 19(5):238, 2019.

[14] G. P. Moustris, S. C. Hiridis, K. M. Deliparaschos, and K. M. Konstantinidis. Evolution of autonomous and semi-autonomous robotic surgical systems: a review of the literature. *The International Journal of Medical Robotics and Computer Assisted Surgery*, 7(4):375–392, 2011.

[15] Heiko Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web*, 8(3):489–508, 2017.

[16] Joelle Pineau, Michael Montemerlo, Martha Pollack, Nicholas Roy, and Sebastian Thrun. Towards robotic assistants in nursing homes: Challenges and results. *Robotics and Autonomous Systems*, 42(3):271–281, 2002.

[17] Carol Reiley, Erion Plaku, and Gregory Hager. Motion generation of robotic surgical tasks: Learning from expert demonstrations. *IEEE Engineering in Medicine and Biology Society*, 2010:967–970.

[18] Gleb Sizov, Pinar Ozturk, and Jozefštyrák Jozefštyrák. Acquisition and reuse of reasoning knowledge from textual cases for automated analysis. *Lecture Notes in Computer Science*, 8765:465–479, 2014.

[19] Darrell Spurlock. The nursing shortage and the future of nursing education is in our hands. *Journal of Nursing Education*, 59(6):303–304, 2020.

[20] Tong Yu, Jinghua Li, Qi Yu, Ye Tian, Xiaofeng Shun, Lili Xu, Ling Zhu, and Hongjie Gao. Knowledge graph for TCM health preservation: Design, construction, and applications. *Artificial Intelligence in Medicine*, 77:48–52, 2017.

[21] Yong Zhang, Ming Sheng, Rui Zhou, Ye Wang, Guangjie Han, Han Zhang, Chunxiao Xing, and Jing Dong. HKGB: An inclusive, extensible, intelligent, semi-auto-constructed knowledge graph framework for healthcare with clinicians' expertise incorporated. *Information Processing & Management*, 57(6):102324, 2020.