Extraction of Knowledge Representations for Reasoning from Medical Questionnaires

Emir Mujić*
Alexander Perko
Franz Wotawa
emir.mujic@tugraz.at
alexander.perko@tugraz.at
wotawa@tugraz.at
Graz University of Technology, Institute of Software Engineering and Artificial Intelligence
Graz, Austria

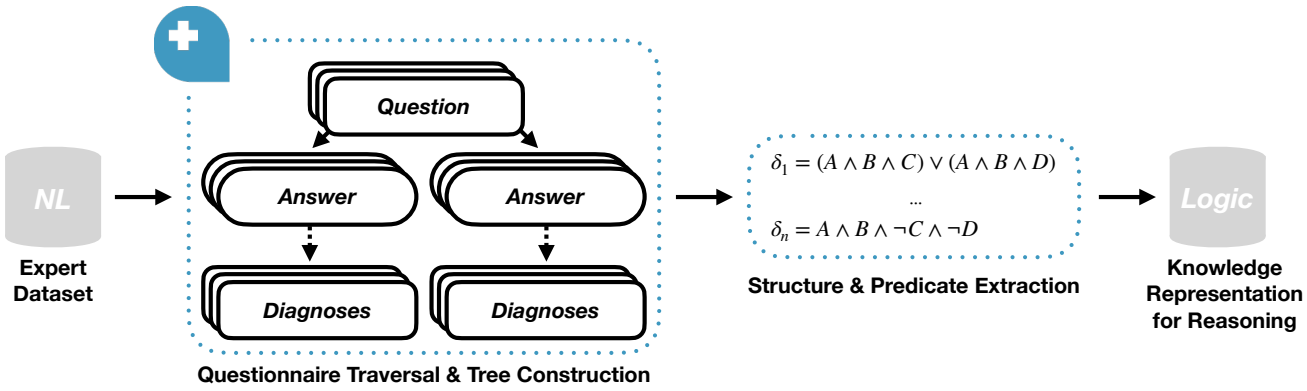


Figure 1: Overview of Knowledge Extraction from the Medical Expert Dataset through Questionnaire Traversal

Abstract

Knowledge representations supporting reasoning are versatile and enable automated use cases such as testing and verification. In contrast to purely data-driven approaches to AI, logical reasoning is explainable. Logic for encoding knowledge yields tremendous potential because of a strong theoretical foundation, and there exist efficient solvers. However, within medicine, we do not find a publicly accessible corpus of expert knowledge encoded in logic. Construction of such a corpus usually requires manual effort and experts in the field, as well as in formal methods. In this work, we contribute by describing a methodology for the automated extraction of logical formulae through interacting with a questionnaire, which is based on a database curated by medical professionals. We propose to use tree traversal and automated predicate extraction from question/answer-nodes comprising natural language. The proposed methods are already established in graph theory, natural language processing, and autoformalization. Hence, we use synergies from different research domains to enable the creation of a logical corpus of medical expert knowledge. With this concept paper, we lay the basis for future work and hope to contribute to use cases, such as rigorous testing of large language models and other medical expert systems.

*Authors are listed in alphabetical order. All authors contributed equally to this research.

Information Society 2025, 6–10 October 2025, Ljubljana, Slovenia

https://doi.org/10.70314/is.2025.gptzdravje.7

Keywords

knowledge representation, reasoning, decision trees, natural language processing, medical questionnaires

1 Introduction & Related Work

Logical formalisms, like First-Order Logic (FOL), or the Answer Set Programming paradigm (ASP) [12], can be used to encode knowledge enabling reasoning through theorem provers/solvers, such as Prover9 [15] or Clingo [11]. Having a logical knowledge base, one can easily query existing facts, check statements for consistency, and infer new knowledge. Consider now a medical knowledge base KB, where symptoms are mapped to diagnoses such that one can infer a set of diagnoses given a set of facts about a person, and a set of symptoms. Given a proper user interface, this can be directly used as an expert system. What is more, it can be used as a test oracle for comparisons with other medical expert systems, providing a transparent view of how diagnoses are made. Even more interestingly, we can evaluate large language models (LLMs) tasked with diagnosing a person given the same input, which we already demonstrated in earlier work [20, 19]. Although there exist benchmarks & datasets for question answering [14] and natural language inference [22] in medicine, we do not find a dataset that fulfills the described properties and is publicly available. Hence, our goal is to build such a knowledge base. As manually creating a gold standard dataset requires expert knowledge and is costly, we propose the automated enrichment of an existing database, which can be accessed through a questionnaire. More specifically, we show how to extract logical formulae from NetDoktor's „Symptom-Checker" questionnaire (SCQ) [21], which is curated by medical professionals and is based on the AMBOSS dataset [1]. Our methodology aims for

automated formalization, i.e., autoformalization of knowledge encoded in natural language. Furthermore, we contribute by elaborating how to leverage the fact that tree representations can be converted into logical formulae [2]. Vice versa, tree structures can be created from logical sentences [5]. A benefit of having a decision tree from a knowledge base is being able to exactly compute bias in the diagnoses (and the knowledge base), as well as the sufficient and necessary reasons behind decisions [9, 2], even in cases of trees with non-binary features (multiple choice questions) [13]. That said, this work directly builds upon our earlier work [20], where we outline the concept of representing a medical questionnaire as a decision tree.

At this point, it makes sense to introduce medical questionnaires & similar systems, such as chatbots: The main idea is to provide answers to a user given symptomatic and/or other information about a person. They are used by the general public and medical professionals alike, and their application varies from general health assessment, over risk calculators to medical triage [16]. These systems often use different combinations of rule-based and data-driven approaches [3, 7]. Most recently, general purpose, as well as domain-specific LLMs, are heavily utilized as well [17, 23, 6], which increases the demand for testing them. We argue that it makes sense to rely on an evaluation methodology that is fully understandable, deterministic, and finite to test non-deterministic, black box systems, such as LLMs. You can find a pilot evaluation of ChatGPT [18] using SCQ in our earlier work [20]. This brings us back to medical questionnaires in the classical sense, from which we will extract a logical knowledge base. Questions within a medical questionnaire can be distinguished in several ways. Namely, we distinguish by:

- Question format:
  - Open-ended questions (Type 1).
  - Closed-ended questions (Type 2).
- Fact permanence:
  - Questions about what a person *is*, which yield permanent facts about a person.
  - Questions about what a person *has*, which yield temporary facts about a person, i.e. symptoms.
- Question requirement:
  - Obligatory questions.
  - Optional questions, with an option to skip.
- Answer types:
  - Predefined options to answer.
  - Freeform answers (not present in SCQ).

Note that these categories are mutually exclusive within but not across distinguishable dimensions, e.g., in principle, it is possible to either have obligatory or optional questions that are open-ended, as well as closed-ended. Having introduced the general problem and domain, we will now proceed with describing a methodology for the enrichment of an expert dataset, with logical representations through tree traversal & basic semantic parsing.

2 Methodology

This work aims to automatically extract logical formulae from knowledge encoded in structured, natural language. Thus, there are three parts to the proposed methodology:

(1) Construction of the tree structure, through filling out SCQ.
(2) Extraction of predicate names from natural language.
(3) Aggregation of formulae, through tree traversal.

While our methods are universally applicable to extracting knowledge from any questionnaire of a similar form, we base all elaborations on SCQ.

2.1 Tree Representations of Questionnaires

In this work, we represent medical questionnaires as decision trees. We first look at creating a simple tree T from SCQ, which corresponds to a session a user might have with the tool:

The root node $r(T)$ is always a question with which every new session is started: *Um wen geht es?* (Who is this about?). From this root node $r(T)$, the tree branches down in a depth-first manner, starting with obligatory questions of Type 1, and followed by optional Type 2 questions. The leaf node(s) $l(T)$ represent a set Δ of diagnoses proposed by SCQ.

Given a tree T with a root node $r(T)$, any number of regular nodes $n_i(T), i = 1, ..., N-1$ and leaf nodes $l(T)$, a walk[1] [10] defines a "Tree Path Structure" from the root to any other node, including the leaf node i.e. the diagnosis possible within the system. Since we know that we can treat trees as graphical representations of logical formulae in disjunctive normal form (DNF), we can write that any tree path structure represents a world w that satisfies at least one diagnosis δ, $w \models \delta$. In other words, models of any diagnosis δ , $Mods(\delta)$ is *any* set of variable assignments that lead to that diagnosis. In most cases, there will be more than one diagnosis given for a world w, we denote this as $w \models \Delta, \delta \in \Delta$, where Δ is a subset of all possible diagnoses, $\Delta \subseteq \mathcal{D}$ [2]. The set of all diagnoses $\mathcal{D}$ is satisfied by the union of worlds of all diagnoses: $Mods(\mathcal{D}) = \bigcup_{j=0}^{M} w_j$, where M is the number of possible diagnoses.

We show a simple example: A diagnosis δ_1 (acute gastroenteritis) is given as a result if a patient has nausea (A) and stomach ache (B) and either fever (C) or diarrhea (D). Another diagnosis δ_2 (gastritis) is a result if a patient has nausea (A) and stomach ache (B) without fever ($\neg C$) and diarrhea ($\neg D$). We can write this as a set of formulae in DNF as:

$$\delta_1 = (A \wedge B \wedge C) \vee (A \wedge B \wedge D), \quad (1)$$
$$\delta_2 = A \wedge B \wedge \neg C \wedge \neg D,$$

which we can represent as a decision tree shown in Figure 2.

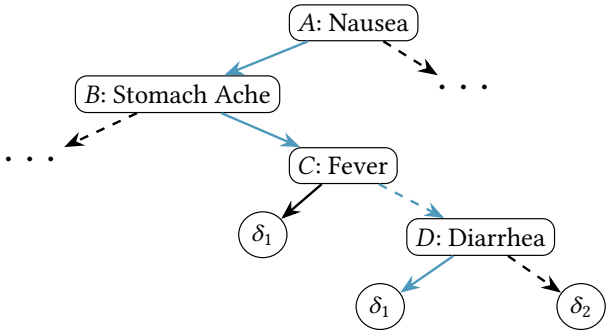


Figure 2: Example 1 as Decision Tree

In Figure 2, a full edge between any two variables represents a truth assignment to the upper variable in the tree based on which the lower variable follows. The dashed edge between represents

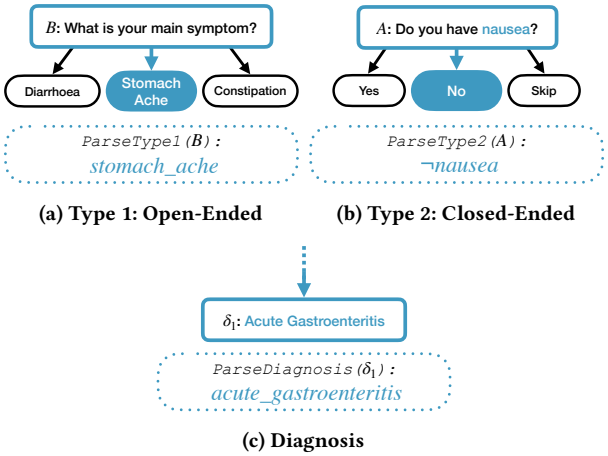
[1] A walk in this context refers to its graph-theoretical definition: In a graph $(V, E) : G, E \subseteq [V]^2$, a walk is a sequence $v_0 e_1 v_2, ..., e_{n-1} v_n$ of alternating vertices and edges such that $\forall_i : e_i = \{v_{i-1}, v_i\}$.
[2] In general: $\Delta \subset \mathcal{D}$. However, $\Delta \subseteq \mathcal{D}$ *iff* $w = \{\emptyset\}$.

a false assignment to the upper variable from which the lower variable follows. The walk highlighted in blue represents one possible instantiation of symptoms where the patient has nausea, a stomach ache, and diarrhea without fever. The three dots ("···") in Figure 2 denote that there are parts of the tree not shown in the example but may exist in the complete tree representation. We would also like to point out that there may exist multiple walks to any single node in the tree, including the leaf nodes ($w_i, w_j \models \Delta, w_i \neq w_i, i \neq j$), something that is excluded in the example in Figure 2 for clarity.

Finally, we summarize how to extract a complete tree out of SCQ, following a depth-first-search methodology: Opening the first session with the questionnaire corresponds to creating a root node. This is followed by answering questions systematically, remembering all questions and answers, and adding corresponding nodes to the tree. At the end of one session, we are presented with a set of diagnoses, which represent the leaf nodes in the tree. This procedure is repeated until we have traversed the entire search space. For further explanations, we refer the interested reader to our previous work [20], which provides elaborations on SCQ, and extracted tree nodes. Due to space limitations of visually representing large trees, we provide examples separately, which can be downloaded at Zenodo[3].

2.2 Predicate Extraction

For now, we assumed the nodes of the constructed tree representation to be directly usable as predicates. However, as the nodes correspond to statements (e.g., sentences, words, or noun phrases) in natural language (NL), we first have to extract predicates. Moreover, in order to enable more than two answers per question, we extend the simplified tree structure from above by the inclusion of separate answer nodes. Thus, we have three types of NL nodes: Questions, corresponding answers, and diagnoses. Furthermore, we assume to remember the relation of questions to their answers and a basic classification of question types into "Type 1", i.e., open-ended, and "Type 2", i.e., closed-ended questions. This distinction can also be seen in Figure 3.



(a) Type 1: Open-Ended (b) Type 2: Closed-Ended

(c) Diagnosis

Figure 3: Predicate Extraction through Parsing Functions for Different Question Types, & Diagnoses

We define three node-level parsing functions: 1) `ParseType1`, 2) `ParseType2`, and 3) `ParseDiagnosis`, which are explained visually in Figure 3. We can simplify the step of autoformalization, as the NL statements found in SCQ show a very limited linguistic complexity. Therefore, we propose to either use naive semantic parsing or LLM-based predicate extraction. For the naive approach, one would simply return the object of a sentence (i.e., singular word or whole noun phrase), modified for the formal language in question. ASP, as used in Clingo, for instance, demands predicates to be written in lower case and allows underscores for separating words in predicate names, which can be seen in Figure 3. Table 1 shows further examples for predicate extractions.

[3]https://doi.org/10.5281/zenodo.17058631

2.3 Formula Aggregation

Continuing with the aggregation of the extracted predicates into logical formulae, we propose a simple algorithm, which can be seen in Algorithm 1. The input is the (extended) tree T, or rather its root node $r(T)$, and the output is a list of formulae, corresponding to all paths in the tree, each comprising a persona and its symptomatic (which we subsume by "symptoms"), as well as corresponding diagnoses.

Algorithm 1 SCQ Tree Traversal for Formula Aggregation

Input: Root node $r(T)$ (assumed to be the first question)
Output: A list of all paths, corresponding to formulae:
(i) a list of symptoms, and
(ii) a list of diagnoses.

```
1:  function TreeTraversal(r(T))
2:      Formulae ← [ ]          ▷ Final list of aggregated formulae
3:      Visit(r(T), [ ], [ ], Formulae)
4:      return Formulae
5:  end function
6:  function Visit(node, Symptoms, Diagnoses, Formulae)
7:      if node.type = "Leaf Node" then
8:          NewPredicates ← ParseDiagnosis(node)
9:          Diagnoses ← Diagnoses ∪ {NewPredicates}
10:         append (Symptoms, Diagnoses) to Formulae
11:         return
12:     end if
13:     if node.type = "Question" then
14:         for each child in node.children do
15:             if node.subtype = "Type1" then
16:                 NewPredicates ← ParseType1(child)
17:             else if node.subtype = "Type2" then
18:                 NewPredicates ← ParseType2(node, child)
19:             end if
20:             Symptoms ← Symptoms ∪ {NewPredicates}
21:             Visit(child, Symptoms, Diagnoses, Formulae)
22:         end for
23:     end if
24: end function
```

As can be seen in Lines 1-5 of Algorithm 1, the depth-first search is started by calling the TreeTraversal function with $r(T)$. Next, a Visit function (Lines 6-24) is called recursively, visiting all nodes on a path until it reaches the/each leaf node (Line 7). In the final list of formulae, which represents all paths, symptoms are assumed to be conjunctions whereas diagnoses are assumed to be disjunctions. Both comprise parsed predicates, and can now be joined to form strings, depending on the target formalism and solver/theorem prover.

ID	Type	Tree Node		Predicate
		Question	Selected Answer	
1	1	Geht es um eine Frau oder einen Mann? *Is it about a woman or a man?*	Weiblich *Female*	female
2	1	Wo treten die Beschwerden auf? *Where do the symptoms occur?*	Kopf *Head*	head
3	1	Wähle dein wichtigstes Symptom *Select your most important symptom*	Schnarchen *Snoring*	snoring
4	2	Leidet die Person unter Schnupfen oder laufender Nase? *Does the person have a cold or runny nose?*	Ja *Yes*	cold ∨ runny_nose
5	2	Ist die Haut (stellenweise) gerötet? *Is the skin reddened (in places)?*	Nein *No*	– reddened_skin
6	2	Hattest du schon einmal eine Allergie? *Have you ever had an allergy?*	Überspringen *Skip*	×

Table 1: Exemplary Predicates by ID, Extracted from Question- & Answer-Tree-Nodes. For Type 1 questions, predicates are extracted from answers. Type 2 questions yield predicates directly, while (potential) negations are extracted from answers.

3 Conclusion & Future Work

In summary, we propose a methodology for constructing & traversing trees from medical questionnaires for the extraction of logical formulae. We describe how to leverage this to construct a medical knowledge base, which can be used for reasoning and enables future work, such as testing LLMs. Future work on decision trees extracted from medical questionnaires will include dealing with multiple paths to the same diagnosis, the intersection of structured tree paths, redundant trees, as well as transforming the large trees into different structures that allow for more efficient computation of certain properties. These include ordered binary decision diagrams [4] and deterministic decomposable negation normal form (d-DNNF) circuits [8], offering the possibility of model counting (asking what diagnoses are possible for any subset of symptoms), reasoning about the biases in the knowledge base by analyzing the decisions made, giving us a complete reason behind diagnoses from which we can compute the sufficient reason (the reason why that diagnosis was chosen) and the necessary reason (why any other diagnosis was not chosen) [9, 13, 2]. With these analyses, we hope to gain further insights into the knowledge base of SCQ and find new and interesting ways of using its logically enriched form. Ultimately we hope to enable new testing strategies of AI-based systems in medicine, particularly LLMs.

Acknowledgements

The work presented in this paper was partially funded by the European Union under Grant 101159214 – ChatMED. Views and opinions expressed are, however, those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

References

[1] AMBOSS GmbH. 2025. Amboss. www.amboss.com. Accessed: 2025-09-03. (2025).

[2] Gilles Audemard, Steve Bellart, Louenas Bounia, Frédéric Koriche, Jean-Marie Lagniez, and Pierre Marquis. 2021. On the explanatory power of decision trees. *arXiv preprint arXiv:2108.05266*.

[3] Ahmad Taher Azar and Shereen M El-Metwally. 2013. Decision tree classifiers for automated medical diagnosis. *Neural Computing and Applications*, 23, 7, 2387–2403.

[4] Randal E Bryant. 1992. Symbolic boolean manipulation with ordered binary-decision diagrams. *ACM Computing Surveys (CSUR)*, 24, 3, 293–318.

[5] Chin-Liang Chang and Richard Char-Tung Lee. 1973. *Symbolic logic and mechanical theorem proving*. Academic press.

[6] Zeming Chen et al. 2023. Meditron-70b: scaling medical pretraining for large language models. (2023). eprint: 2311.16079.

[7] Dillon Chrimes. 2023. Using decision trees as an expert system for clinical decision support for covid-19. *Interact J Med Res*, 12, (Jan. 2023), e42540. DOI: 10.2196/42540.

[8] Adnan Darwiche. 2001. Decomposable negation normal form. *Journal of the ACM (JACM)*, 48, 4, 608–647.

[9] Adnan Darwiche and Auguste Hirth. 2023. On the (complete) reasons behind decisions. *Journal of Logic, Language and Information*, 32, 1, 63–88.

[10] Reinhard Diestel. 2025. *Graph theory*. Vol. 173. Springer Nature.

[11] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub. 2018. Multi-shot ASP solving with clingo. (Mar. 2018). arXiv: 1705.09811 [cs]. DOI: 10.48550/arXiv.1705.09811.

[12] Michael Gelfond and Vladimir Lifschitz. 1988. The stable model semantics for logic programming. In *Proceedings International Logic Programming Conference and Symposium*. MIT Press, Cambridge, MA, USA, 1070–1080.

[13] Chunxi Ji and Adnan Darwiche. 2023. A new class of explanations. In *Logics in Artificial Intelligence: 18th European Conference, JELIA 2023, Dresden, Germany, September 20–22, 2023, Proceedings*. Vol. 14281. Springer Nature, 106.

[14] Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2020. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *arXiv preprint arXiv:2009.13081*.

[15] W. McCune. 2005–2010. Prover9 and Mace4. (2005–2010).

[16] Bilal A Naved and Yuan Luo. 2024. Contrasting rule and machine learning based digital self triage systems in the usa. *NPJ digital medicine*, 7, 1, 381.

[17] Harsha Nori, Nicholas King, Scott Mayer McKinney, Dean Carignan, and Eric Horvitz. 2023. Capabilities of gpt-4 on medical challenge problems. (2023). https://arxiv.org/abs/2303.13375 arXiv: 2303.13375 [cs.CL].

[18] OpenAI. 2023. ChatGPT. (2023). chat.openai.com/chat.

[19] Alexander Perko, Iulia Nica, and Franz Wotawa. 2024. Using Combinatorial Testing for Prompt Engineering of Large Language Models in Medicine. In *Proceedings of the 27th International Multiconference Information Society – IS 2024*. Ljubljana, Slovenia. DOI: 10.70314/is.2024.chtm.10.

[20] Alexander Perko and Franz Wotawa. 2024. Testing ChatGPT's Performance on Medical Diagnostic Tasks. In *Proceedings of the 27th International Multiconference Information Society – IS 2024*. Ljubljana, Slovenia. DOI: 10.70314/is.2024.chtm.7.

[21] Jens Richter, Hans-Richard Demel, Florian Tiefenböck, Luise Heine, and Martina Feichter. 2025. Symptom-checker. www.netdoktor.at/symptom-checker/. Accessed: 2025-09-03. (2025).

[22] Alexey Romanov and Chaitanya Shivade. 2018. Lessons from natural language inference in the clinical domain. *arXiv:1808.06752 [cs]*, (Aug. 21, 2018). Retrieved Aug. 27, 2018 from arXiv: 1808.06752.

[23] Khaled Saab et al. 2024. Capabilities of gemini models in medicine. (2024). https://arxiv.org/abs/2404.18416 arXiv: 2404.18416 [cs.AI].