Dissecting NetDoktor's Online Questionnaire with Entropy, Tree Simplification & Logical Abduction

Alexander Perko
Institute of Software Engineering and Artificial Intelligence, Graz University of Technology, Austria

Emir Mujić
Institute of Software Engineering and Artificial Intelligence, Graz University of Technology, Austria

Franz Wotawa
Institute of Software Engineering and Artificial Intelligence, Graz University of Technology, Austria

Abstract

Medical diagnosis and its accuracy are dependent on the available information. Hence, it is in the interest of both the patient and the doctor to provide the necessary information and ask the right questions in an efficient manner. The same can be said about medical questionnaires, which resemble the conversation during a doctor's appointment. NetDoktor's SymptomChecker is a freely accessible, expert-curated questionnaire in the German language. It can be traversed like a large decision tree, linking symptoms to diseases. However, does it find an optimal conversation path to acquire the necessary information from its users? Here, our study surfaces structural deficiencies and ways to optimise the questionnaire paths. In particular, we model the process of acquiring information and refining prognoses as an abductive reasoning problem and provide a corresponding algorithm. In our experiments, we apply simple tree transformations and optimisations, building upon Shannon's information entropy to expose redundant subtrees and inefficient question order. Our results contribute to a better understanding of medical questionnaires and open the door for future applications of the data provided by NetDoktor. Because SymptomChecker is validated by medical professionals, it is a prime candidate to serve as a reference for benchmarking other systems. For instance, a dataset derived from SymptomChecker can be used for testing systems such as large language models in the task of medical diagnosis.

2012 ACM Subject Classification Information systems → Information extraction; Computing methodologies → Causal reasoning and diagnostics; Computing methodologies → Logic programming and answer set programming; Applied computing → Health care information systems

Keywords and phrases Medical Questionnaire, Diagnosis, Abductive Reasoning, Information Entropy

Digital Object Identifier 10.4230/OASIcs.DX.2026.3

Supplementary Material *Software (Replication Package)*: `https://zenodo.org/records/20425820`

Funding 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.

1 Introduction

Medical questionnaires (i.e., expert systems with a questionnaire user interface; distinct from questionnaires used in scientific studies) are modelled to resemble conversations a patient might have with a doctor. The goal is an accurate diagnosis, where the accuracy is dependent on the provided information. In turn, the information flow is guided by the questions raised by the questionnaire or doctor. This means that the questions and their order matter to arrive at accurate diagnoses early. This can be described using information theory, and Shannon's information entropy [20] specifically. Entropy is used to describe information gain,

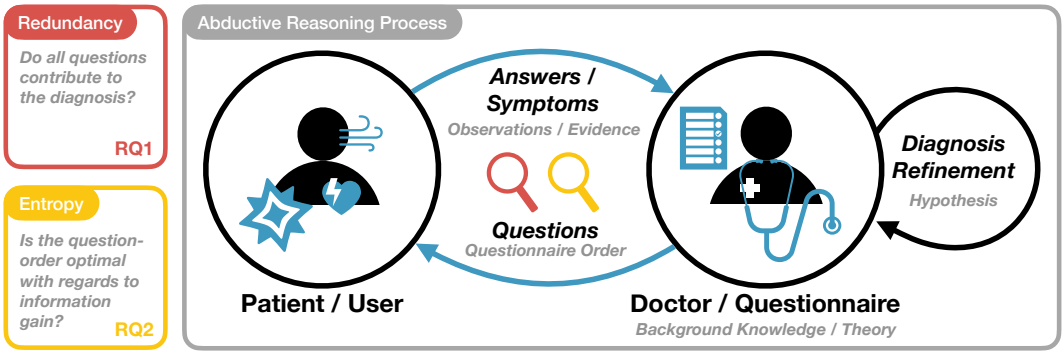


Figure 1 Doctor's Appointment ≈ Medical Questionnaire ≈ Abductive Reasoning Process.

measured in bits and given through the formula:

$$H(P) = -\sum_{\Delta} p(\Delta)\, \log_2 p(\Delta),$$

where P are all paths a conversation with a doctor may take, Δ is the diagnostic outcome a conversation path ends with, and $p(\Delta)$ is the fraction of conversation paths leading to the same outcome. This reading of Entropy yields the possibility to measure how much information is gained when asking a particular question on the conversation path, if paths and possible outcomes are known. Moreover, we can model the retrieval of a diagnosis in the context of a medical questionnaire as an abductive reasoning problem, where abduction was originally coined by Peirce [13] and is nowadays understood as inference to the best explanation.

Figure 1 shows this framing of a medical questionnaire as an abduction problem with the goal of finding a diagnosis (i.e., a hypothesis), which explains symptoms (i.e., observations / evidence) under the consideration of background knowledge (i.e., a theory). If now an observation does not change anything with regards to the hypothesis, we say it is redundant. Again, this ties-in with entropy, as a redundant observation also does not yield any information gain with respect to the hypothesis. Having now prepared our operating room, we can turn to the subject we want to dissect. NetDoktor [19] provides a freely accessible questionnaire, which is validated by medical professionals. It is called SymptomChecker, as it asks users consecutive questions about symptoms and provides possible diagnoses based on the given answers. In our previous work [15, 11], we demonstrated how to traverse SymptomChecker like a large decision tree and used the connection between tree and logical representations to transform the tree into logical formulae [11]. More specifically, we represent all questionnaire paths - from start to diagnosis - as conjunctions of the question-answer pairs, which are interpreted as propositions. The whole tree (i.e., questionnaire) can be represented as a disjunction of all questionnaire paths. In this work the main focus is a concrete subtree of SymptomChecker, where we are interested in its structure, information content and usability for logical abduction, in particular. Our work is guided by the main research question of whether NetDoktor's SymptomChecker is structured in an efficient manner, which we can further separate into the two sub-criteria:

- **RQ1**: Do all questions in SymptomChecker contribute to the outcome (i.e., the diagnoses) or is there redundancy?
- **RQ2**: Is SymptomChecker's question order optimal with respect to the information gained at each step?

Here, we first structurally analyse a dataset aggregated from over 90000 individual SymptomChecker sessions, which is first compiled as part of this work, to find redundant subtrees (RQ1) and sub-optimal question order (RQ2) after which we use the resulting structural variants for a comparative analysis in an abductive reasoning setting. The latter aims to unveil differences when performing reasoning. Both questions are answered based on the subset of SymptomChecker seen in the recorded sessions. That said, this paper contributes by combining and applying a framework of methodologies - all of which are rooted in well-known theories - to investigate and transform a tree-like questionnaire. Furthermore, this work can be seen as a cornerstone for using NetDoktor's data for benchmarking other systems in the future. This paper is organised as follows: We start by tapping into related research. Then, we introduce the subject of our investigation: NetDoktor's SymptomChecker, where we highlight the concrete subtree (i.e., dataset) aggregated through web scraping. After that, we introduce the used methodologies and experimental setup comprising a structural analysis and an abductive reasoning experiment. Figure 5 shows this and contextualises the other figures, tables, and algorithms. Finally, after presenting and discussing the results, we conclude this paper.

2 Related Work

Medical diagnosis can be viewed as a reasoning problem in which observed symptoms, answers, or test results have to be explained by one or more possible diagnoses. This is closely related to abductive reasoning [13], where observations are explained by selecting suitable abducible hypotheses subject to integrity constraints [8]. This can be easily translated into medical questionnaires where patients' answers can be treated as observations and candidate diagnoses can be treated as hypotheses. The issue is that unrestricted abductive reasoning is computationally inefficient [3]. The complexity motivates us to use more structured representations such as decision trees to restrict the search space.

Another way we can view medical diagnosis is using model-based diagnosis [18]. An approach to hypothesis classification using model-based abduction was done by Friedrich *et al.* [4]; they distinguish between different roles that a hypothesis can play as necessary, possible, or irrelevant. This is important because diagnostic reasoning is not only concerned with finding any compatible diagnosis, but also with identifying which hypotheses are actually needed to explain the observations.

Abduction has been applied directly in medicine; Lucas [10] used abduction for checking the quality of medical guidelines. The work showed how abductive reasoning can support the analysis of whether guideline recommendations satisfy broader medical requirements. Pukancová and Homola [16] studied medical diagnosis using description-logic abduction, focusing on diabetes as a use case and arguing that medical diagnosis often requires abductive reasoning as it eliminates diagnostic hypotheses that do not explain all the symptoms. More recently, Xu *et al.* [24] proposed a medical dialogue system that explicitly separates abductive and deductive components, using abduction to generate candidate diagnostic hypotheses and deduction to verify them against the dialogue context. Unlike classical logical abduction/deduction, they leverage black-box models to achieve the results. These works show that abduction remains relevant for modern medical decision-support systems.

Decision trees have been widely used as transparent classifiers that recursively select informative attributes and produce human-readable decision paths [17]. In a medical questionnaire, such paths naturally correspond to sequences of questions and answers, which, in turn, resemble a conversation with a doctor. From a more formal perspective, tree

Table 1 Exemplary SCQ Questions. German Original (DE). English Translation (EN).

ID	Question (DE)	Question (EN)
1	Um wen geht es?	Who is this about?
2	Bist du männlich oder weiblich?	Are you male or female?
3	Zu welcher Altersgruppe gehörst du?	Which age group do you belong to?
4	In welcher Körperregion hast du Beschwerden?	In which body region do you have complaints?
5	Wo treten die Beschwerden auf?	Where do the complaints occur?
6	Welches Symptom belastet dich am stärksten?	Which symptom troubles you the most?
7	Ist deine Haut stellenweise oder überall gerötet?	Is your skin reddened in places or all over?
12	Hast du eine nässende, wunde Stelle an der Haut?	Do you have a weeping, sore spot on the skin?
13	Hattest du schon einmal eine Allergie?	Have you ever had an allergy?
14	Spürst du irgendwo ein Brennen?	Do you feel a burning sensation anywhere?
17	Fühlt sich ein Körperteil besonders warm an?	Does a body part feel particularly warm?
32	Hast du offene Verletzungen an der Schleimhaut?	Do you have open lesions on a mucous membrane?
33	Hattest du ungeschützten Sex oder häufig wechselnde Partner?	Have you had unprotected sex or frequently changing partners?

automata provide a mathematical framework for reasoning about tree-shaped structures and their accepted paths; Comon *et al.* [2] describe the correspondence between regular tree languages, finite tree automata, and logical characterisations of trees. Tree-based methods have also been used in clinical decision support and questionnaire reduction.

Hannöver *et al.* [7] applied classification trees to data from the German multicenter study on eating disorders, showing how tree models can support clinical decision-making in an interpretable way. Strobl *et al.* [21] provide an especially relevant medical example: using data from the German DizzyReg patient registry, they identified a small set of eight key questions that helped classify common vestibular disorders, to be used as triage before conducting a clinical examination. This work is close in spirit to questionnaire-based diagnosis, since it demonstrates how a larger clinical history can be reduced to a compact and interpretable set of discriminative questions.

3 NetDoktor's SymptomChecker Questionnaire

NetDoktor SymptomChecker [19] is a freely accessible[1], medical expert system with a questionnaire UI that retrieves likely diagnoses from a person's reported symptoms. SymptomChecker itself is based on a proprietary, expert-curated database called AMBOSS [1]. The questionnaire is available in German and combines static questions about a person such as sex, age (referred to as "persona" in our work) and the affected body region and main symptom, with dynamically selected follow-up questions. The dynamic part consists of closed-ended questions with possible answers "Yes", "No", and "Skip". We do not use "Skip" in our experiments and only consider positive and negative answers, which is why we refer to these questions as binary. We treat SymptomChecker as a deterministic reference model and traverse the questionnaire systematically, storing the resulting interactions as a tree, which we explained in our previous work [15, 11] and is illustrated in Figure 2.

[1] SymptomChecker Questionnaire: `www.netdoktor.at/symptom-checker`

3.1 Questionnaire Traversal & Tree Representation

The extraction procedure consists of two main steps: ID extraction and the questionnaire traversal. Since the questionnaire is meant for humans, the questions are written in natural language. It would be inefficient to save all question-answer (QA) pairs in such a format, so we set an ID integer value for each unique question and each unique answer. Then each question and answer is saved as a QA-pair. For example, a QA-pair $[13, 7]$ represents a decision node in a tree where a question "Hattest du schon einmal eine Allergie?" (Have you ever had an allergy?) is answered by "Yes". A QA-pair $[13, 8]$ represents a negative answer to the same question. We follow a depth-first search to extract the tree structure. This way, the questionnaire is unfolded into a finite decision tree T where each internal node corresponds to a question, and each edge corresponds to an answer. A complete root-to-leaf path p therefore represents one possible questionnaire session and can be interpreted as a conjunction of extracted predicates, each corresponding to a QA-pair. The sets of diseases at the leaf are the diagnostic conclusions associated with this conjunction. We denote a set of potential diseases as diagnosis Δ. In turn, all distinct diseases in the whole tree are referred to by $\mathcal{D}$. Furthermore, we denote a collection of paths (i.e., a disjunction of conjunctions) as $p \in P$ and assume that a tree constructed in the way we described (or a subtree thereof) can be represented as such a collection.

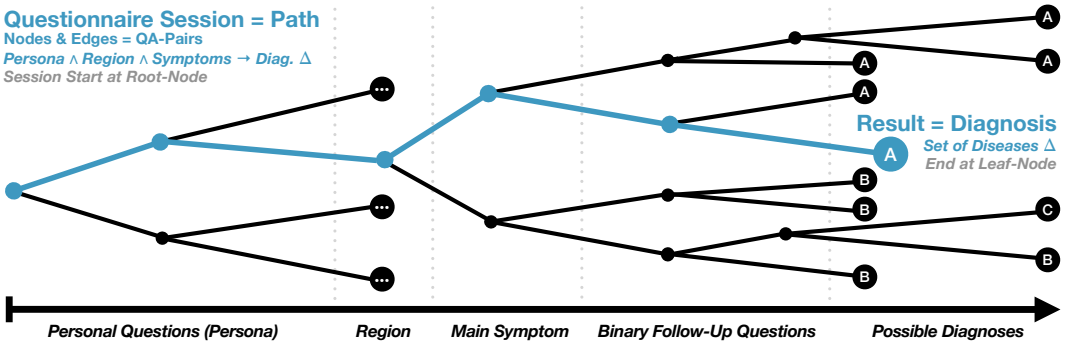


Figure 2 Tree Representation of NetDoktor's SymptomChecker Questionnaire.

3.2 SCQ Dataset

In this subsection, we describe the dataset, which we refer to as SCQ, for SymptomChecker Questionnaire, which we derived by traversing SymptomChecker. It forms the basis for all of our experiments. In particular, the dataset we extract is a subset of the complete questionnaire tree. For our extraction, we balanced the search between depth and breadth of the constructed trees. This was done since there is no public information on the true size of SCQ, or its algorithm, and the search doesn't spend too much time only looking into a single set of symptoms, but explores other branches as well. Furthermore, for simplicity, we focus on a single persona description, an adult male, and on the sub-region skin (German: "Haut" and "Kopfhaut"). We selected skin, as this sub-region appears in all but two of the main regions of the full tree. The first row of Table 2 shows the statistics for the SCQ dataset used in our experiments. From the table, we can see that the total number of complete paths extracted from SymptomChecker is $95,318$. A path is a single possible sequence of answered questions from the beginning of the questionnaire to the terminal nodes (diagnosis sets Δ). The total number of diseases $\mid \mathcal{D} \mid$ is 129. There are 635 unique diagnoses (i.e., sets of

diseases). The average depth per questionnaire path is 20.4, meaning that it takes, on average, 20.4 questions to obtain a set of diseases. The shortest path only has 6 question-answer pairs while the longest has 24.

4 Methodology

In this section, we introduce the theoretical basis of our experiments, which comprises naïve tree simplification, entropy-based tree reconstruction, and Horn Clause Abduction Problems.

4.1 Naïve Tree Simplification

In this section, we describe a naïve tree simplification method, which we will use to identify and remove redundant subtrees. In our context, this translates to finding QA pairs which do not change the outcome and can be removed from the questionnaire. The basic idea of the algorithm is shown in Figure 3: On the left-hand side is the binary input tree, and on the right-hand side the simplified output tree. In both trees, the same diagnoses (leaves) are reachable. The output tree, however, has the same or a smaller amount of paths leading there. We define the following recursive algorithm: For each binary, follow-up subtree, preceded by

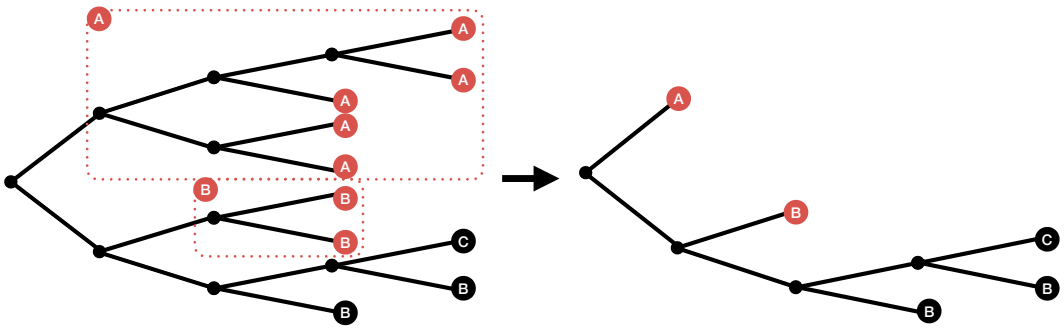


Figure 3 Tree Simplification: Identification & Removal of Redundant Questions.

a main symptom node, we walk the paths from leaf upwards. At each QA-node, we compare the two children and collapse them, i.e., replace the current QA-node with the child-nodes' diagnoses, iff both child-nodes are leaves and carry the same diagnoses. We repeat this collapsing procedure until there are no QA-nodes satisfying the criterion.

4.2 Entropy-Based Tree (Re-)Construction

Following our previous work, where we introduced an entropy-based algorithm and demonstrated it within a chatbot-powered questionnaire [12, 22], we employ a similar strategy here. The core idea of the algorithm follows information-gain-based decision tree induction [17]: at each step, the algorithm selects the eligible question that produces the greatest reduction in entropy over the possible answers. However, the previously introduced algorithm was built with user interaction in mind and modelled as a constraint satisfaction problem. While the core principle remains the same for this implementation, we apply the algorithm without user interaction and to reorder an existing diagnostic tree. Figure 4 shows an exemplary input tree on the left-hand side and a reconstructed output tree on the right-hand side. The questions in the output tree are always ordered to maximise the information gain by selecting the question to come next with the highest discriminative power given the previous

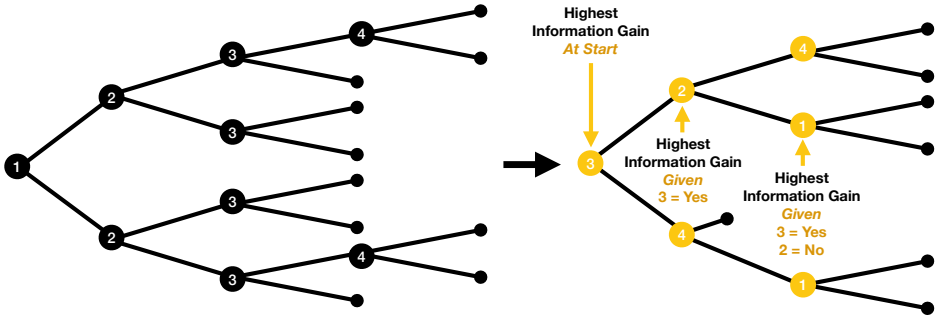


Figure 4 Entropy-Based Tree (Re-)Construction / Question Reordering.

QA-nodes. We define a recursive algorithm: For each binary, follow-up subtree, preceded by a main symptom node, per path we collect the set of diseases it ends in and the QA-pairs (i.e., questions, which were answered in this path and the answer), which we write as a pair (QA, Δ). Then we start recursively constructing a new tree from the subtree-root by calling the function EntropyTree, which is introduced in Algorithm 1, and first called with the set of paths of a binary follow-up subtree P and an empty set of used questions U. We start by determining question eligibility in Line 2, where a question q is considered if it is on every path in P, i.e., every $(QA, \Delta) \in P$ contains a pair (q, a), and is not part of the set of questions, which have been used already U. Eligible questions are collected in E, where we abort and return one terminal node $\text{Leaf}(\Delta)$ per distinct diagnosis in P, i.e., one per $\Delta \in \text{Diagnoses}(P) = \{\, \Delta \mid (QA, \Delta) \in P \,\}$, when E stays empty (i.e., when there are no eligible questions) in Lines 3-4. Then, H is used to compute the information gain IG for all eligible questions, which can be seen in Lines 5-6, where $P_a^q = \{\, (QA, \Delta) \in P \mid (q, a) \in QA \,\}$ refers to all paths in P whose answer to q is a, and where the sum runs over $\text{Answers}(q, P)$, the answers to q observed in P. H in this context refers to the information entropy after Shannon [20]. Specifically, we use it as $H(P) = -\sum_{\Delta \in \text{Diagnoses}(P)} p(\Delta) \log_2 p(\Delta)$, where $p(\Delta)$ is the fraction of paths in P ending in diagnosis Δ. This information is used to select the question with the highest information gain q^*, in Line 7. We stop and return one leaf per distinct diagnosis in P, if there is no information to be gained, i.e., even the highest information gain is (close to) zero $IG(q^*) \leq \varepsilon \approx 0$, in Lines 8-9. Finally, in Lines 10-13, we reconstruct the tree by attaching the result of a recursive call of EntropyTree below a branch node $\text{Branch}(q^*, a, S)$, which carries the question-answer-pair (q^*, a) and the recursively constructed subtree S, for every answer a (i.e., branch) to question q^* that is observed in P. The recursive call per branch is invoked using the subset of paths $P_a^{q^*}$ to which the answer to q^* is the branching answer a, and with U updated by adding q^*. Ultimately, the length of the new, reconstructed paths is guided by the availability of eligible questions and their information gain, while it is ensured that all diagnoses / diseases are preserved.

4.3 Horn Clause Abduction Problems

In this paper, and this Subsection in particular, the definitions are based on those of Propositional Horn Clause Abduction Problems (PHCAP) by Friedrich et al. [4], which we adapt for our application domain and modelling using the Answer Set Programming (ASP) paradigm [6, 9].

▶ **Definition 1.** *$HCAP = \langle Var, Hyp, SD, Obs \rangle$. Horn Clause Abduction Problem.*

Algorithm 1 Entropy-Based Tree Re-Construction.

Require: Paths P of one binary follow-up subtree; used questions U. Initial call per subtree: EntropyTree(P, $\emptyset$)

Ensure: The reconstructed subtree below the current QA-pair (below the main symptom node on the initial call)

function EntropyTree(P, U)
// Eligible Questions: Answered by Every Path in P, Not Yet Used
  $E \leftarrow \{\, q \mid \text{every } (QA, \Delta) \in P \text{ contains a pair } (q, a) \,\} \setminus U$
  if $E = \emptyset$ **then**
    return $\{\, \text{Leaf}(\Delta) \mid \Delta \in \text{Diagnoses}(P) \,\}$
// Score Every Eligible Question by Information Gain
  for all $q \in E$ **do**
    $\text{IG}(q) \leftarrow H(P) - \sum_{a \in \text{Answers}(q,P)} \frac{|P_a^q|}{|P|} H(P_a^q)$
// Select the Most Informative Question; Stop If None Discriminates
  $q^* \leftarrow \arg\max_{q \in E} \text{IG}(q)$
  if $\text{IG}(q^*) \leq \varepsilon$ **then**
    return $\{\, \text{Leaf}(\Delta) \mid \Delta \in \text{Diagnoses}(P) \,\}$
// Construct Subtree by Recursing Once per Answer of q^* Observed in P
  nodes $\leftarrow \emptyset$
  for all $a \in \text{Answers}(q^*, P)$ **do**
    nodes $\leftarrow$ *nodes* $\cup \{\, \text{Branch}(q^*, a, \text{EntropyTree}(P_a^{q^*}, U \cup \{q^*\})) \,\}$
  return *nodes*

▶ **Definition 2.** *Var. All possible questions (1...244), answers (1...25) and diseases (1...129) on the paths of SCQ.*

▶ **Definition 3.** *Hyp* $\subset$ *Var. The abducible hypotheses. In SCQ, Hyp* $= \mathcal{D}$.

▶ **Definition 4.** *SD. The System Description encoded via Horn Clauses (ASP rules).*

▶ **Definition 5.** *Obs* $\subset$ *Var. Observations, i.e., QA-pairs (ASP facts).*

4.3.1 From Questionnaire Tree to Horn Clauses

In this section, we introduce Algorithm 2 to obtain a system description SD from a questionnaire tree T. In particular, we use this algorithm to derive ASP [6, 9] models to be used with Clingo [5] as a solver. It has to be stated that this work does not aim at nor claim to derive a perfect SD, but to use abductive reasoning as a mere vehicle to investigate SCQ. Hence, the main goal of this algorithm is the straightforward, automatic extraction of an SD given a tree in the format of SCQ. That said, Algorithm 2 produces rules (i.e., Horn Clauses) of two types, which are based on the polarity of the QA pair and form subsets of SD: SD^+ and SD^-. Together with a predefined set of base rules SD_{base} (see Subsection 5.1.2), the two subsets of polarity-based rule types are unified to form the full SD:

$$\text{SD} = \text{SD}_{\text{base}} \cup \text{SD}^+ \cup \text{SD}^-$$

In order to have some control over the added rules, we introduce confidence thresholds τ^+ and τ^- for the two polarity-specific subsets, respectively. In tandem with SD_{base} as used in our experiments, rules in SD^+ enable diagnoses to cover observations and those in SD^-

eliminate a diagnosis, if an observation is "Ja" for the same symptom. Looking now at Algorithm 2, we see in Lines 1-6 the initialisation of the co-occurrence matrices CO_{cat}, for categorical questions, i.e., questions 1-6 (see Table 1), and for all binary, follow-up questions, i.e., CO_{bin}, CO^{+}_{bin}, CO^{-}_{bin} as well as the total counter. Next, in Lines 7-18, the co-occurrence matrices are filled by iterating over the questionnaire paths and increasing the corresponding co-occurrence counter for every pair of diagnosis and question. Finally, the SD subsets are aggregated in Lines 19-29 based on the confidence thresholds and at the end unified with $\mathrm{SD}_{\mathrm{base}}$.

Algorithm 2 Extract System Description SD from Questionnaire Tree T.

Require: Questionnaire Tree T; Base Rules $\mathrm{SD}_{\mathrm{base}}$; Confidence Thresholds τ^{+}, τ^{-}
Ensure: $\mathrm{SD} = \mathrm{SD}_{\mathrm{base}} \cup \mathrm{SD}^{+} \cup \mathrm{SD}^{-}$

1: **function** ExtractSystemDescription(T, $\mathrm{SD}_{\mathrm{base}}$, τ^{+}, τ^{-})
// Init Co-Occurrence Matrices for Diagnoses and Questions
2: **for all** $\Delta \in \mathrm{Diagnoses}(T)$ **do**
3: $total[\Delta] \leftarrow 0$
4: **for all** $s \in \mathrm{Atoms}_{\mathrm{cat}}(T)$ **do**
5: $\mathrm{CO}_{\mathrm{cat}}[\Delta][s] \leftarrow 0$
6: **for all** $q \in \mathrm{Questions}_{\mathrm{bin}}(T)$ **do**
7: $\mathrm{CO}_{\mathrm{bin}}[\Delta][q] \leftarrow 0$; $\mathrm{CO}^{+}_{\mathrm{bin}}[\Delta][q] \leftarrow 0$; $\mathrm{CO}^{-}_{\mathrm{bin}}[\Delta][q] \leftarrow 0$
// Fill Co-Occurrence Matrices by Iterating over Questionnaire Paths
8: **for all** $p \in \mathrm{Paths}(T)$ **do**
9: $S_{\mathrm{cat}} \leftarrow \{\, (q,a) \mid (q,a) \in \mathit{Region}(p) \cup \mathit{MainSymptom}(p) \,\}$
10: **for all** $\Delta \in \mathrm{Leaves}(p)$ **do**
11: $total[\Delta] \leftarrow total[\Delta] + 1$
12: **for all** $s \in S_{\mathrm{cat}}$ **do**
13: $\mathrm{CO}_{\mathrm{cat}}[\Delta][s] \leftarrow \mathrm{CO}_{\mathrm{cat}}[\Delta][s] + 1$
14: **for all** (q, polarity) $\in \mathit{FollowUp}(p)$ **do**
15: $\mathrm{CO}_{\mathrm{bin}}[\Delta][q] \leftarrow \mathrm{CO}_{\mathrm{bin}}[\Delta][q] + 1$
16: **if** polarity is positive ("Ja") **then**
17: $\mathrm{CO}^{+}_{\mathrm{bin}}[\Delta][q] \leftarrow \mathrm{CO}^{+}_{\mathrm{bin}}[\Delta][q] + 1$
18: **if** polarity is negative ("Nein") **then**
19: $\mathrm{CO}^{-}_{\mathrm{bin}}[\Delta][q] \leftarrow \mathrm{CO}^{-}_{\mathrm{bin}}[\Delta][q] + 1$
// Construct Horn Clauses using Confidence Thresholds
20: **for all** Δ, s **do**
21: **if** $\mathrm{CO}_{\mathrm{cat}}[\Delta][s] \,/\, total[\Delta] \;\geq\; \tau^{+}$ **then**
22: $\mathrm{SD}^{+} \leftarrow \mathrm{SD}^{+} \cup \{\, s \leftarrow \Delta \,\}$
23: **for all** Δ, q **with** $\mathrm{CO}_{\mathrm{bin}}[\Delta][q] > 0$ **do**
24: $c^{+} \leftarrow \mathrm{CO}^{+}_{\mathrm{bin}}[\Delta][q] \,/\, \mathrm{CO}_{\mathrm{bin}}[\Delta][q]$; $c^{-} \leftarrow \mathrm{CO}^{-}_{\mathrm{bin}}[\Delta][q] \,/\, \mathrm{CO}_{\mathrm{bin}}[\Delta][q]$
25: **if** $c^{+} \geq \tau^{+}$ **then**
26: $\mathrm{SD}^{+} \leftarrow \mathrm{SD}^{+} \cup \{\, s^{+}(q) \leftarrow \Delta \,\}$
27: **if** $c^{-} \geq \tau^{-}$ **then**
28: $\mathrm{SD}^{-} \leftarrow \mathrm{SD}^{-} \cup \{\, s^{-}(q) \leftarrow \Delta \,\}$
29: **return** $\mathrm{SD}_{\mathrm{base}} \cup \mathrm{SD}^{+} \cup \mathrm{SD}^{-}$

5 Experimental Evaluation

In this section, we describe our experimental setup, which is modelled around the information gained during a doctor's appointment, or rather a questionnaire session. Furthermore, we introduce structural variations of SCQ and methods to produce diagnoses from the questionnaire paths. Finally, we present the experimental results and discuss them.

5.1 Experimental Setup

The core idea of our experimental setup is to simulate an SCQ session via traversals of the SCQ tree and structural variants thereof as an abductive reasoning problem. We obtain observations and system descriptions from SCQ and its variants. Then, we evaluate the reasoning performance at evidence levels, which symbolise the acquisition of information throughout the questionnaire session. This experimental setup is coupled with a statistical analysis of the SCQ dataset with a focus on prominence of diagnoses, information gain (as measured by entropy), and redundancy (as detected by tree simplification).

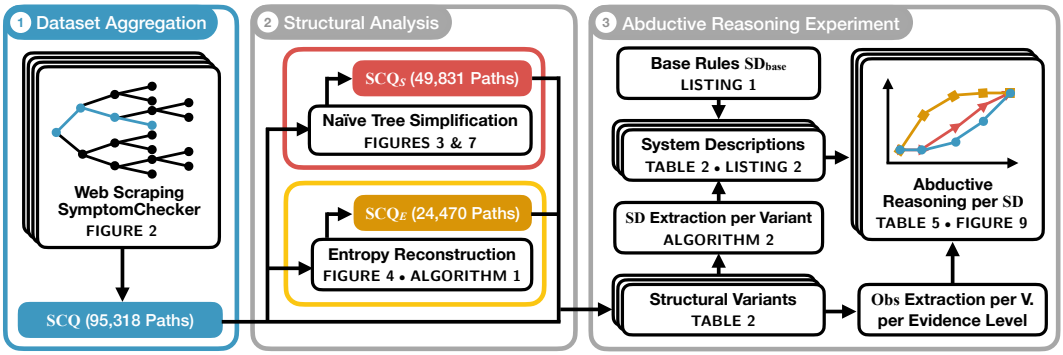


Figure 5 Experimental Setup & Overview of Figures, Tables, Algorithms, and Listings.

5.1.1 Structural Variants of SCQ

In our abductive reasoning experiments, the original SCQ tree serves as a reference. In addition, we use two structural variants of SCQ, which are in-line with our research questions. First, we apply naïve tree simplification, as described in Subsection 4.1, to explore potential redundancies. Here we aim to answer **RQ1**, as a tree with redundant question nodes would demonstrate that these questions are not contributing to the final diagnosis. Second, we use an entropy-based tree construction, as described in Subsection 4.2. The aim of this is to answer **RQ2**, as a comparison between the original and the entropy-optimised tree should have equal (or similar) results with regards to the information gained throughout the steps of the abductive reasoning process. In summary, we use the following three structural variants in our experiments: **SCQ**, $\mathbf{SCQ}_{\boldsymbol{S}}$, and $\mathbf{SCQ}_{\boldsymbol{E}}$, which are listed and explained further in Table 2. Furthermore, we use a balanced subsample of SCQ used for hyperparameter search and to ground the comparison. We call this variation **SCQ'** and balance it by limiting the number of paths per set of diseases to 8 and randomly select paths when there are more than 8 for a diagnosis. This way, dominant diagnoses do not skew the dataset as much.

Table 2 Statistics of Structural Variants of SCQ and Their Derived System Descriptions. Number of Individual Diseases (129) and Number of Distinct Sets of Diagnoses (635) are Equal Among All Variants. We use the following shorthands: Number of Paths (#P), Average (Avg).

Structural Variant		#P	Path Depth		Derived SD Rules		
			Range	Avg	SD^+	SD^-	Σ
SCQ	Original questionnaire tree.	95 318	6/24	20.4	6 322	3 132	9 454
$\mathbf{SCQ}_S$	Simplified/pruned version of SCQ.	49 831	6/24	19.7	6 083	3 044	9 127
$\mathbf{SCQ}_E$	Entropy-reconstruction of SCQ.	24 474	6/23	19.1	5 955	3 085	9 040
$\mathbf{SCQ}'$	Balanced subset of SCQ.	4 165	6/23	19.7	5 552	2 870	8 422

5.1.2 System Description

SD is obtained using Algorithm 2, which we introduced in Subsection 4.3. In this subsection, we list the input parameters used for our abductive reasoning experiment. Listing 1 shows the base rules SD_{base} as used for our experiments. In this case, the base rules comprise a rule introducing *Hyp*, a cardinality-based minimisation criterion, and the following integrity constraints:

- C1: Every positive observation must be explained by some hypothesised diagnosis.
- C2: A positive observation ("Ja") contradicts a negative.
- C3: A negative observation ("Nein") penalises a positive-predicting diagnosis.

Guided by the results of a hyperparameter search (see Subsection 5.1.4) to find a good balance between answers in SCQ and rules in the corresponding SD, we assume the following confidence thresholds for the polarity-based rules: $\tau^+ = 0$ and $\tau^- = 0.8$. Using consistent SD_{base} and confidence thresholds, we use SD(SCQ), SD(SCQ_S), and SD(SCQ_E) when referring to a complete SD derived from SCQ, SCQ_S, and SCQ_E, respectively. Listing 2 shows an excerpt of the polarity-based rules derived from SCQ, i.e., $SD^+ \cup SD^- \subset$ SD(SCQ).

Listing 1 Predefined Subset of Base Rules SD_{base} of SD.

```
possible_diagnosis(1..129).                % Hyp: Abducible Diagnoses
{ diagnosis(D) } :- possible_diagnosis(D).
:- obs_ja(Q, A), not sym_ja(Q, A).         % Integrity Constraint C1
:- obs_ja(Q, 7), sym_nein(Q).              % Integrity Constraint C2
:~ obs_nein(Q), sym_ja(Q, 7). [1@2, Q]     % Integrity Constraint C3
:~ diagnosis(D). [1@1, D]                  % Minimise Cardinality of Hyp
```

Listing 2 United Subsets $SD^+ \cup SD^-$ of SD Extracted from SCQ Using Algorithm 2.

```
sym_ja(5, 5) :- diagnosis(1). % SD+ Rule 1: conf_ja=1.00 [Fusspilz]
sym_ja(6, 6) :- diagnosis(1). % SD+ Rule 2: conf_ja=0.54 [Fusspilz]
sym_ja(7, 7) :- diagnosis(1). % SD+ Rule 3: conf_ja=0.71 [Fusspilz]
...                           % All SD+ Rules: 6322 (conf_ja >= 0)
sym_nein(17) :- diagnosis(1). % SD- Rule 1: conf_nein=0.98 [Fusspilz]
sym_nein(32) :- diagnosis(1). % SD- Rule 2: conf_nein=1.00 [Fusspilz]
sym_nein(33) :- diagnosis(1). % SD- Rule 3: conf_nein=1.00 [Fusspilz]
...                           % All SD- Rules: 3132 (conf_nein >= 0.8)
```

5.1.3 Observations & Evidence Levels

As pointed out above, SCQ structurally resembles a doctor's appointment. At first, only the patient's information according to the intake form is known, which translates to the persona in SCQ. Then, at the beginning of the examination, the general region is located, which is followed by the description of the main symptom. Afterwards, further information is acquired through follow-up questions by the doctor, which is concluded by giving a final (hopefully accurate) diagnosis. We translated this into the 5 distinct evidence levels L0-L4, at which observations are added consecutively and abduction is evaluated. L0-L4 are tightly bound to the structure of the questionnaire sessions, which is why we introduce additional percentage-based evidence levels P0-P100 for reference. Both are further described in Table 3.

Table 3 Discrete evidence levels at which observations are drawn and abduction is evaluated.

Based on Doctor's App. / Questionnaire Session		**Percentage-Based**	
L0	Persona information only.	**P0**	No information.
L1	Persona and region.	**P25**	25% of information.
L2	Persona, region and main symptom.	**P50**	50% of information.
L3	Persona, region, main symptom and 50% of follow-up.	**P75**	75% of information.
L4	All the information.	**P100**	All the information.

5.1.4 Hyperparameter Search

We performed a hyperparameter search to find values for the confidence thresholds τ^+ and τ^- controlling the polarity based SD^+ and SD^-. The whole setup is shown in Table 4. Each cell in the table corresponds to one combination of τ^+ and τ^-, where τ^+ controls positive rules, and τ^- negative ones. For every combination, we generated an SD from the balanced SCQ' and computed the F1-score at evidence level P100. Notably, we included thresholds excluding all negative rules ($\tau^- = \infty$), as well as keeping all ($\tau^- = 0$). Conversely, for positive rules we did not include a setting excluding all of them, as this would yield a degenerate SD. Overall, keeping all positive rules ($\tau^+ = 0$) and a τ^- of 0.8 performed best.

Table 4 Hyperparameter Search for Confidence Thresholds τ^- (Columns) and τ^+ (Rows): Grid search using SCQ′ and maximising F_1 at evidence level P100. Best value in **bold**.

τ^+	τ^-												
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	0.95	0.99	∞
0	0.002	0.008	0.012	0.018	0.031	0.060	0.134	0.220	**0.294**	0.246	0.190	0.145	0.052
0.05	0.000	0.006	0.010	0.018	0.031	0.060	0.129	0.212	0.274	0.245	0.215	0.195	0.120
0.1	0.000	0.006	0.010	0.018	0.032	0.060	0.133	0.219	0.293	0.274	0.257	0.235	0.139
0.2	0.000	0.005	0.007	0.014	0.024	0.046	0.095	0.160	0.213	0.220	0.216	0.202	0.127
0.3	0.000	0.004	0.007	0.013	0.020	0.036	0.080	0.133	0.158	0.181	0.185	0.186	0.127
0.5	0.001	0.005	0.006	0.012	0.019	0.034	0.051	0.065	0.082	0.095	0.099	0.102	0.140

5.1.5 Evaluation

Following a structural analysis of SCQ, we will evaluate the questionnaire sessions as an abductive reasoning process using the standard metrics Precision (P), Recall (R), and F1.

5.2 Results & Discussion

In this section, we show our experimental results and provide our interpretation. We provide a replication package including SCQ, its structural variants, as well as the corresponding SDs and code to simultaneously draw Obs from the SCQ variants and execute Clingo using our settings. We use Clingo `5.8.0` in brave mode.

5.2.1 Disease Prominence & Prior Baseline

We start discussing our results by highlighting the statistical imbalance found in the SCQ dataset. In particular, we find that there are some diseases more prominent than others to a high degree. Figure 6 shows this by comparing the most prominent with the least prominent

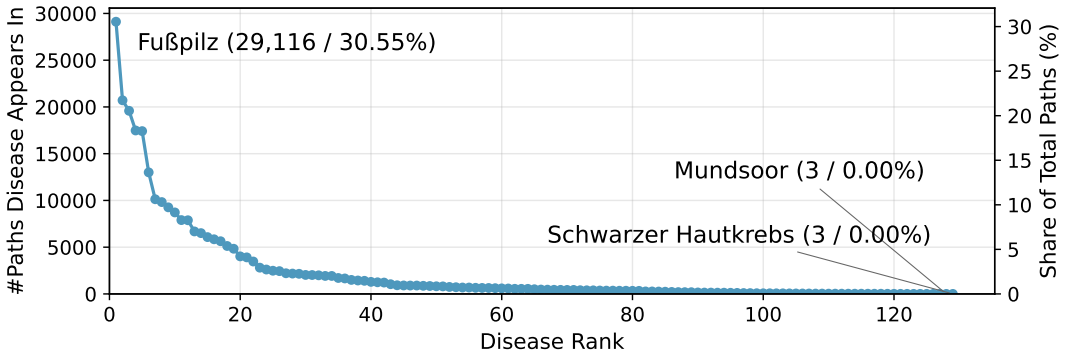


Figure 6 Disease Prominence in SCQ from Most Prominent (i.e., Rank 1 = Fußpilz/Athlete's Foot) to Least Prominent (i.e., Rank 129 = Schwarzer Hautkrebs/Melanoma & Mundsoor/Oral Thrush).

disease based on the number of paths they appear in. For instance, the most prominent disease "Athlete's foot" appears in 29116/95318 paths, which is over 30% of SCQ. As a stark contrast, the least prominent diseases, "Melanoma" and "Oral thrush", each appear in only 3/95318 paths. Overall one can easily spot the sharp peak of a few, very prominent diseases, followed by the "long tail" of the many diseases appearing in few paths in Figure 6. This, of course, leads to a skewed prior if we consider it as the relative frequency of any individual disease. The other approach for selecting a prior is using $\frac{1}{|\mathcal{D}|}$ for each disease. We include both the skewed prior and $\frac{1}{|\mathcal{D}|}$ as baselines in the experiments below, where we refer to them as B1 and B2, respectively.

5.2.2 Redundancy & Information Gain

Now, we want to shift our attention to the main research questions by graphically demonstrating 1) the redundancy present in the original SCQ and 2) the differences in information gain per question between the structural variants of SCQ. The former is visualised in Figure 7 and the latter, in Figure 8. Starting with 1), we can observe in Figure 7 that the probability for a question being redundant in SCQ rises (blue line) after question 8, which is only two questions into the binary follow-up questions. The red line, showing the pruned subtrees, also rises, but more slowly, because it counts the subtrees, not the individual questions. This shows that SCQ is not free of redundancy to begin with and that there exist indeed entire redundant subtrees (as the probability of individual questions being redundant rises faster than the count of pruned subtrees). Turning to 2), we can read from Figure 8 that there is

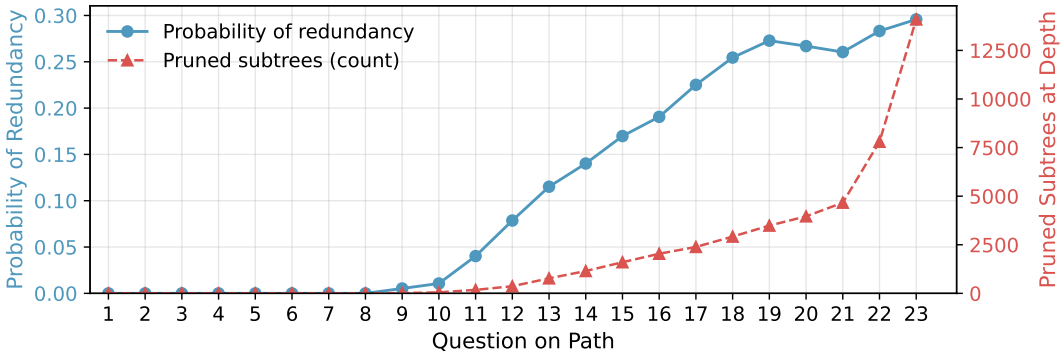


Figure 7 Probability of Redundancy by Question as Encountered in SCQ (Left Y-Axis; Blue). Number of Pruned subtrees during Simplification of SCQ to Generate SCQ_S (Right Y-Axis; Red).

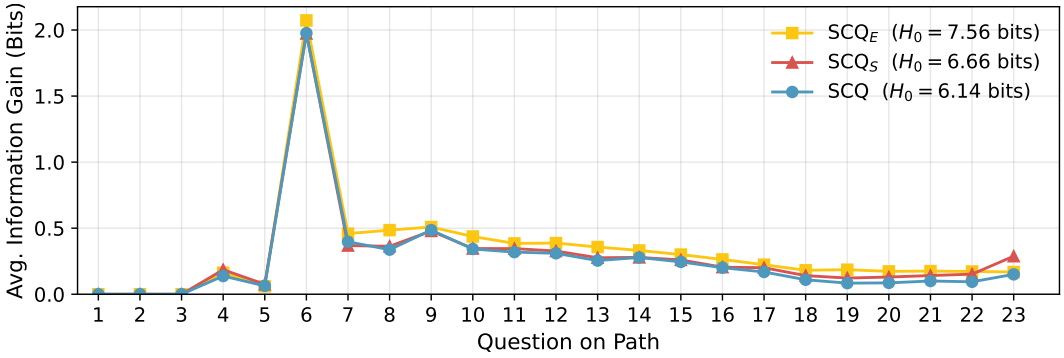


Figure 8 Average Information Gain by Question per SCQ Variant.

no information gain at the beginning of the questionnaire. This is not surprising, as we are looking at a subtree with fixed persona, which translates to the first three questions having no discriminative power at all within SCQ. Compare to Table 1, where the first 6 (among others) questions are listed. Question 4, which asks the user for the main body region the symptom occurs in, shows a first local maximum. This is explainable, as we did not fix the main region when acquiring SCQ. Question 5, on the other hand, was fixed to the two skin-related sub-regions, which shows as a drop in information gain in the plot. After this question, we encounter the global maximum at question 6, which asks for the main symptom. After this point, the binary questions related to other symptoms follow. Here, we can observe two main differences across the structural variants of SCQ: On the one hand, we see that SCQ_S gets ahead of the original SCQ after question 16, which is a sign of pruning redundant subtrees actually contributing to information gain. On the other hand, we find that SCQ_E is continuously ahead of the other two (except for the last one, where SCQ_S overtakes it). This, of course, comes as no surprise given the construction of SCQ_E. It must be noted, however, that neither of these differences is major.

5.2.3 Abductive Reasoning

Having analysed the statistics and gained insight into each of the structural variants of SCQ, we can finally discuss the derived SD and Obs, which serve as simulated questionnaire sessions. The main questions to ask are: 1) Do the derived SDs differ? And: 2) How do they cope with Obs drawn from any of the structural variants? To answer 1), we read from Table 2 that the three SDs are similarly sized, even though their origins are vastly different as measured by their number of paths. For instance, SCQ has 95318 paths and the corresponding SD,

Table 5 Precision (P), Recall (R), and F1 per System Description (SD), Observation (Obs), and Evidence Level (L0-L4 & P0-P100). Baselines are constant across levels and shown next to each SD header where: B1 = Skewed Prior, B2 = $1/\mid\mathcal{D}\mid$.

	Obs	L0	L1	L2	L3	L4	P0	P25	P50	P75	P100
SD(SCQ) · B1 (P/R/F1) = 0.224/0.221/0.222 · B2 (P/R/F1) = 0.024/0.023/0.023											
P	SCQ	0.000	0.031	0.103	**0.385**	**0.493**	0.000	**0.105**	**0.317**	**0.497**	**0.493**
	SCQ'	0.000	**0.034**	**0.108**	0.135	0.313	0.000	0.095	0.117	0.176	0.313
	SCQ_S	0.000	0.032	0.106	0.316	0.427	0.000	0.103	0.254	0.405	0.427
	SCQ_E	0.000	0.029	0.092	0.229	0.395	0.000	0.082	0.181	0.291	0.395
R	SCQ	0.000	1.000	1.000	**0.636**	**0.519**	0.000	0.770	**0.643**	**0.700**	**0.519**
	SCQ'	0.000	1.000	1.000	0.297	0.267	0.000	0.731	0.344	0.299	0.267
	SCQ_S	0.000	1.000	1.000	0.614	0.455	0.000	**0.812**	0.627	0.650	0.455
	SCQ_E	0.000	1.000	1.000	0.454	0.407	0.000	0.745	0.473	0.478	0.407
F1	SCQ	0.000	0.058	0.184	**0.455**	**0.475**	0.000	**0.178**	**0.396**	**0.561**	**0.475**
	SCQ'	0.000	**0.065**	**0.190**	0.171	0.265	0.000	0.160	0.159	0.204	0.265
	SCQ_S	0.000	0.060	0.189	0.396	0.408	0.000	0.176	0.341	0.479	0.408
	SCQ_E	0.000	0.055	0.167	0.287	0.370	0.000	0.144	0.242	0.343	0.370
SD(SCQ_S) · B1 (P/R/F1) = 0.224/0.221/0.222 · B2 (P/R/F1) = 0.024/0.023/0.023											
P	SCQ	0.000	0.031	0.103	**0.362**	**0.586**	0.000	**0.103**	**0.296**	**0.472**	**0.586**
	SCQ'	0.000	**0.034**	**0.108**	0.131	0.349	0.000	0.095	0.114	0.175	0.349
	SCQ_S	0.000	0.032	0.106	0.306	0.544	0.000	0.103	0.245	0.394	0.544
	SCQ_E	0.000	0.029	0.092	0.214	0.468	0.000	0.082	0.169	0.287	0.468
R	SCQ	0.000	1.000	1.000	**0.625**	**0.534**	0.000	0.763	**0.630**	**0.682**	**0.534**
	SCQ'	0.000	1.000	1.000	0.285	0.274	0.000	0.730	0.337	0.281	0.274
	SCQ_S	0.000	1.000	1.000	0.609	0.507	0.000	**0.810**	0.620	0.653	0.507
	SCQ_E	0.000	1.000	1.000	0.443	0.441	0.000	0.743	0.463	0.480	0.441
F1	SCQ	0.000	0.058	0.184	**0.437**	**0.514**	0.000	0.176	**0.379**	**0.536**	**0.514**
	SCQ'	0.000	**0.065**	**0.190**	0.164	0.285	0.000	0.160	0.154	0.196	0.285
	SCQ_S	0.000	0.060	0.189	0.387	0.478	0.000	**0.176**	0.332	0.472	0.478
	SCQ_E	0.000	0.055	0.167	0.273	0.416	0.000	0.144	0.230	0.341	0.416
SD(SCQ_E) · B1 (P/R/F1) = 0.224/0.221/0.222 · B2 (P/R/F1) = 0.024/0.023/0.023											
P	SCQ	0.000	0.031	0.103	**0.398**	**0.614**	0.000	**0.106**	**0.314**	**0.519**	**0.614**
	SCQ'	0.000	**0.034**	**0.108**	0.141	0.365	0.000	0.095	0.121	0.193	0.365
	SCQ_S	0.000	0.032	0.106	0.325	0.569	0.000	0.103	0.250	0.427	0.569
	SCQ_E	0.000	0.029	0.092	0.241	0.506	0.000	0.083	0.189	0.323	0.506
R	SCQ	0.000	1.000	1.000	**0.612**	**0.511**	0.000	0.771	**0.577**	**0.680**	**0.511**
	SCQ'	0.000	1.000	1.000	0.282	0.269	0.000	0.729	0.333	0.287	0.269
	SCQ_S	0.000	1.000	1.000	0.581	0.490	0.000	**0.812**	0.564	0.651	0.490
	SCQ_E	0.000	1.000	1.000	0.476	0.448	0.000	0.748	0.495	0.514	0.448
F1	SCQ	0.000	0.058	0.184	**0.455**	**0.509**	0.000	**0.180**	**0.379**	**0.559**	**0.509**
	SCQ'	0.000	**0.065**	**0.190**	0.168	0.286	0.000	0.161	0.159	0.206	0.286
	SCQ_S	0.000	0.060	0.189	0.394	0.478	0.000	0.177	0.323	0.490	0.478
	SCQ_E	0.000	0.055	0.167	0.301	0.434	0.000	0.146	0.253	0.375	0.434

SD(SCQ), comprises 9454 rules in its polarity-based subset (we will always refer to this subset from now on). SCQ_S has 49831 paths, which corresponds to roughly 52% as compared to SCQ. SD(SCQ_S) has 9127 rules, which is about 97% of SD(SCQ). Correspondingly, SCQ_E has 24474 paths (26%) and SD(SCQ_E) comprises 9040 rules (96%). This tells us that way less paths were necessary to extract SDs (i.e., ASP encodings) of similar size in the case of the structural variants. Of course, as Algorithm 2 was used for this extraction, this is tied to the selected confidence thresholds (which remained unchanged across all three SDs). Nevertheless, we interpret this as a sign of much bloat in the original SCQ. This, however, is just one part of the story, as we also have to look at the evaluated, simulated questionnaire sessions, which answer question 2):

Table 5 shows the results, where each SD, namely SD(SCQ), SD(SCQ_S), and SD(SCQ_E), is evaluated on observations Obs drawn from each structural variant of SCQ, as well as a balanced version of SCQ, which we name SCQ'. Overall, there are four sets of observations: Obs(SCQ), Obs(SCQ'), Obs(SCQ_S), and Obs(SCQ_E). These are revealed throughout the evidence levels. These results are visualised in Figure 9. We can look at the plot in multiple ways: A) comparing the scores, B) comparing the SDs, and C) comparing the Obs.

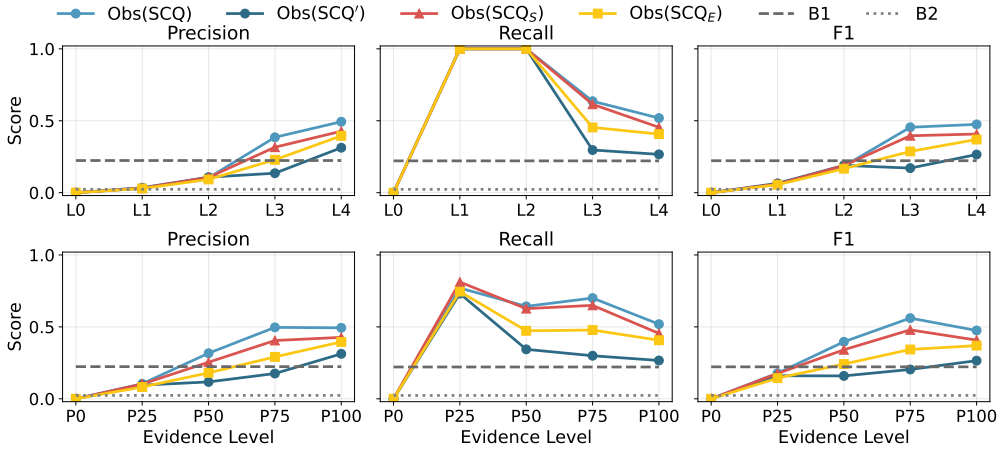
Starting with A), we can observe that Precision and F1 score generally increase throughout the evidence levels, which is expected. The increase is not monotonous in every case, however, and there are cases where both scores peak before the final evidence is presented. To be more precise, we see such an "early peak" in P75 of SD(SCQ) for the observations drawn from the original SCQ, Obs(SCQ), depicted in light blue. This peak of F1 = 0.561 is also the highest F1 score amongst all. Although not as prominent, the same "early peak" can be seen in P75 of SD(SCQ_S) and SD(SCQ_E). Another observation with respect to the Precision is the "long plateau", from L0-L2, which is also expected as the discriminating evidence is not there yet. This fits well into the picture drawn in Figure 8, where information gain is low in questions 1-5 (except 4, defining the main region). Regarding Recall, it can be easily seen that it goes up immediately after L0 and peaks at L1 / P25. This can be explained by the fact that the search space with regards to possible diagnoses is wide open and not yet narrowed down in any meaningful way.

Switching now to B) the comparison of SDs, we see similar curves across all three SDs and scores but with minor differences in spreads across Obs. In fact, F1-differences of SD(SCQ) and SD(SCQ_S) are smaller or equal to 0.07 across evidence levels and Obs. This is another strong indicator of the SDs being very similar, although their basis is very different in the amount of paths.

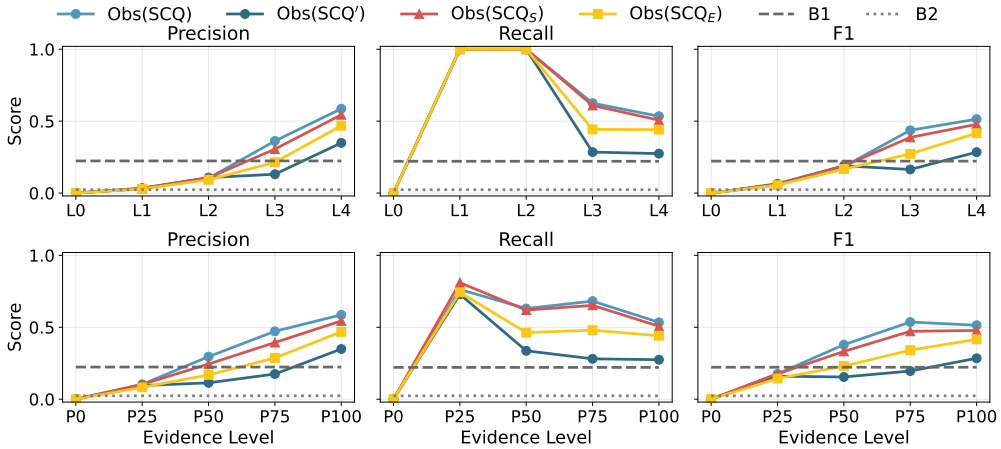
Finally, we arrived at C) comparing the observations. The aforementioned spread across varies depending on the SD. The largest spread can be seen in SD(SCQ_S)-Recall (0.205 on average across all evidence levels) and the least in SD(SCQ)-Precision (0.147). Overall, for the average spread, it can be observed that Recall > F1 > Precision (as averaged over all non-trivial evidence levels, i.e., without L0 and P0). This tells us that the Obs differ most in the simplified SD. We want to conclude this section by pointing out two additional observations: 1) Although the skewed prior baseline is strong, all SDs beat it at full evidence, across all Obs, and 2) the balanced set of observations Obs(SCQ') consistently performs worse than the regular Obs(SCQ), which further highlights the skewed distribution of diagnoses in the original SCQ.

5.2.4 Threats to Validity & Use of Generative AI

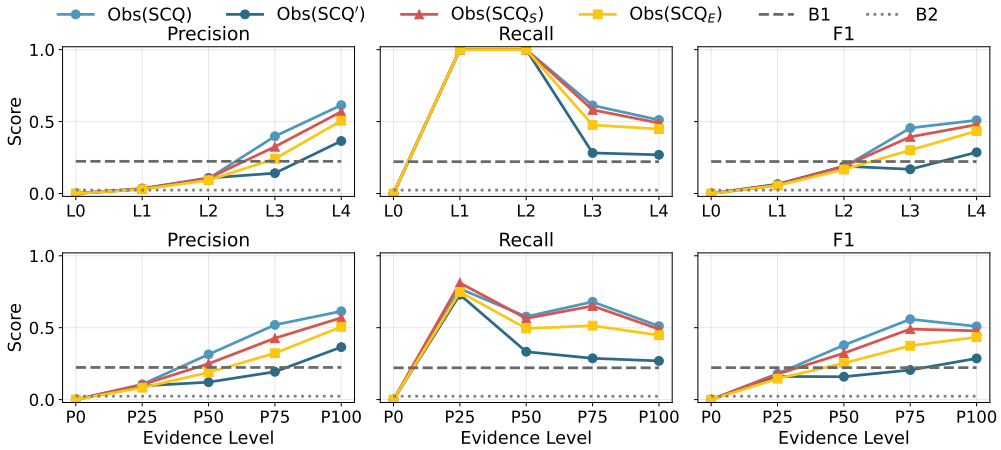
One of the base threats to validity of this study is the fact that we web-scrape the dataset we refer to as SCQ. While we check our data by hand, scraping might introduce artifacts. Furthermore, we only acquired a subtree of the full SymptomChecker questionnaire, and all assumptions are being made based on this. We are aware that our SCQ dataset is imbalanced, which we documented. Some diseases appear disproportionately more than others, causing a skewed prior if we consider it as the relative frequency of any individual disease. The other approach is using the prior as $\frac{1}{|\mathcal{D}|}$ for each disease. Neither of these numbers is representative of actual, real-world frequencies of any disease following from skin as the main affected region, and the resulting methods will be biased. Some other datasets, such as DDXPlus [23], do take this into account, while we do not. Our data is based solely on the subtree extracted from SymptomChecker, as the goal of this study is to investigate what is there before acting on it. Although we cannot make an informed



(a) SD(SCQ).



(b) SD(SCQ$_S$).



(c) SD(SCQ$_E$).

Figure 9 Abductive Reasoning Results: Comparing three system descriptions, each derived from a structural variant of SCQ using Algorithm 2. Corresponds to Table 5.

assessment by just looking at a subtree, we assume that the full SymptomChecker tree is also imbalanced. While we use the balanced subsample SCQ' of SCQ during hyperparameter search, we do not exclude the SCQ' samples from SCQ and the other structural variants when producing SDs and drawing Obs. Nor yet do we use separate sets of paths for SD and Obs in general. This decision was made as the subject of our investigation, SymptomChecker, does not operate on learned knowledge but rather complete traversals on predefined paths and because there are vanishingly few samples for some of the diagnoses. Another threat is our entropy-based optimisation strategy re-orders questions and might lead to unwanted side effects in their semantics, which is partially circumvented by only re-ordering the binary follow-up questions after persona, region, and main symptom have been covered in the original order. Similarly, simplifying the tree might remove wanted redundancies. However, here we must make a clear distinction between meticulously planned medical questionnaires as found in scientific studies, where redundancy is often a wanted feature, and the case of SymptomChecker, which is a web application for information retrieval and most likely retrieves questions and their order programmatically, based on rules and a database. Neither the simplified nor entropy-optimised variants should be understood as better replacements to the original questionnaire but rather give a contrasting view on matters which might not have been considered when originally arranging the questions. Lastly, it must be emphasised that neither the scraped SCQ dataset nor its structural variants are validated by medical experts.

Parts of this work were done with the help of AI. Namely, we used Grammarly for spellchecking, grammar correction and partial rephrasing as well as Claude Code (Opus `4.7` & `4.8`) also for parts of the implementation, including the visualisations and realising the algorithms and for preparing the replication package. We thoroughly checked all the generated code.

6 Conclusion

In this work, we analyse NetDoktor's SymptomChecker [19] questionnaire through the lens of abductive reasoning and answer questions about redundancy and optimality of question order. For this, we acquired a dataset called SCQ through the methodology described in Section 3.1, which links symptoms to diagnoses. SCQ represents a subset of the full SymptomChecker questionnaire and comprises over 90000 individual sessions with SymptomChecker. As we can represent SCQ as a decision tree, we refer to the sessions as paths and are able to apply algorithms on the basis of tree transformation. Thus, we produced structural variants of the SCQ tree, which we used in a comparative experimental evaluation. Our experiments revealed structural deficiencies within SCQ, ranging from a disproportionate representation of some diagnoses to redundant subtrees comprising questions without any contribution to the final diagnosis. Furthermore, we could demonstrate by comparison that the information gain throughout the question paths is sub-optimal and showed how to optimise the questionnaire using pruning and entropy-based reconstruction. What is more, our abductive reasoning experiments indicate, by very close F1-scores across SDs extracted from SCQ, SCQ_S, and SCQ_E (the gap being 0.070 at most / 0.019 on average), that the optimised variants of SCQ carry most of the original's meaning (97% / 96%, as compared by the size of a derived formal system description), while being much smaller (52% / 26%, as compared by the number of paths). Ultimately, we can answer our research questions:

- **RQ1**: Do all questions in SymptomChecker contribute to the outcome (i.e., the diagnoses) or is there redundancy?

Yes, there is redundancy. There are whole subtrees of QA pairs that are not contributing to the diagnoses. This means answering yes or no leads to the same outcome.

- **RQ2**: Is SymptomChecker's question order optimal with respect to the information gained at each step?
  No, the comparison between various question orders revealed that there are better ways to order questions and thus it is not optimal with regard to information gain for the observed subtree. However, this must be taken with a grain of salt considering the unknown size (amongst other properties) of the whole questionnaire tree.

Despite the mentioned caveats, NetDoktor's data has potential for being used as a reference for other systems as it originates from an expert-curated database and is evaluated through medical professionals. In the future, we want to use our methodologies for testing large language models on the task of medical diagnosis, as demonstrated in our previous work [15, 14]. Moreover, we would like to compare the data derived from NetDoktor with other datasets such as DDXPlus [23], which also incorporates real-world disease distributions and census information.

References

1 AMBOSS GmbH. AMBOSS, 2026. URL: www.amboss.com.

2 Hubert Comon, Max Dauchet, Rémi Gilleron, Florent Jacquemard, Denis Lugiez, Christof Löding, Sophie Tison, and Marc Tommasi. *Tree Automata Techniques and Applications*. HAL, 2008. URL: inria.hal.science/hal-03367725.

3 Thomas Eiter and Georg Gottlob. The Complexity of Logic-Based Abduction. *Journal of the ACM*, 42(1):3–42, January 1995. doi:10.1145/200836.200838.

4 Gerhard Friedrich, Georg Gottlob, and Wolfgang Nejdl. Hypothesis Classification, Abductive Diagnosis and Therapy. In J. Siekmann, G. Goos, J. Hartmanis, Georg Gottlob, and Wolfgang Nejdl, editors, *Expert Systems in Engineering Principles and Applications*, volume 462, pages 69–78. Springer Berlin Heidelberg, Berlin, Heidelberg, 1990. Series Title: Lecture Notes in Computer Science. doi:10.1007/3-540-53104-1_32.

5 Martin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub. Multi-shot ASP solving with clingo. *Theory and Practice of Logic Programming*, 19(1):27–82, January 2019. doi:10.1017/S1471068418000054.

6 Michael Gelfond and Vladimir Lifschitz. The Stable Model Semantics for Logic Programming. In *In Proceedings of International Logic Programming Conference and Symposium*, pages 1070–1080, Cambridge, MA, USA, 1988. MIT Press.

7 W. Hannöver, M. Richard, N.B. Hansen, Z. Martinovich, and H. Kordy. A Classification Tree Model for Decision-Making in Clinical Practice: An Application Based on the Data of the German Multicenter Study on Eating Disorders, Project TR-EAT. *Psychotherapy Research*, 12(4):445–461, December 2002. doi:10.1080/713664470.

8 Antonis C Kakas, Robert A. Kowalski, and Francesca Toni. Abductive Logic Programming. *Journal of Logic and Computation*, 2(6):719–770, 1992. doi:10.1093/logcom/2.6.719.

9 Vladimir Lifschitz. Action Languages, Answer Sets, and Planning. In *The Logic Programming Paradigm: A 25-Year Perspective*, pages 357–373. Springer Berlin Heidelberg, Berlin, Heidelberg, 1999. doi:10.1007/978-3-642-60085-2_16.

10 Peter Lucas. Quality Checking of Medical Guidelines through Logical Abduction. In *International conference on innovative techniques and applications of artificial intelligence*, pages 309–321. Springer, 2003. doi:10.1007/978-0-85729-412-8_23.

11 Emir Mujić, Alexander Perko, and Franz Wotawa. Extraction of knowledge representations for reasoning from medical questionnaires. In *Proceedings of the 28th International Multiconference Information Society*, pages 781–784, Ljubljana, Slovenia, 2025. Jožef Stefan Institute. doi:10.70314/is.2025.gptzdravje.7.

12 Iulia-Dana Nica, Franz Wotawa, and Oliver Tazl. Chatbot-Based Tourist Recommendations Using Model-Based Reasoning. In *Proceedings of the 20th International Configuration Workshop*, volume 2220 of *CEUR*, 2018. URL: ceur-ws.org/Vol-2220/05_CONFWS18_paper_31.pdf.

13 Charles Sanders Peirce. *Collected Papers of Charles Sanders Peirce*. Harvard University Press, 1931.

14 Alexander Perko, Iulia Nica, and Franz Wotawa. Using Combinatorial Testing for Prompt Engineering of Large Language Models in Medicine. In *Proceedings of the 27th International Multiconference Information Society*, pages 930–935, Ljubljana, Slovenia, 2024. Jožef Stefan Institute. doi:10.70314/is.2024.chtm.10.

15 Alexander Perko and Franz Wotawa. Testing ChatGPT's Performance on Medical Diagnostic Tasks. In *Proceedings of the 27th International Multiconference Information Society*, pages 914–919, Ljubljana, Slovenia, 2024. Jožef Stefan Institute. doi:10.70314/is.2024.chtm.7.

16 Júlia Pukancová and Martin Homola. Abductive Reasoning with Description Logics: Use Case in Medical Diagnosis. In *Proceedings of the 28th International Workshop on Description Logics*, volume 1350 of *CEUR*, 2015. URL: ceur-ws.org/Vol-1350/paper-60.pdf.

17 J. Ross Quinlan. Induction of Decision Trees. *Machine Learning*, 1(1):81–106, 1986. doi:10.1023/A:1022643204877.

18 Raymond Reiter. A Theory of Diagnosis from First Principles. *Artificial Intelligence*, 32(1):57–95, April 1987. doi:10.1016/0004-3702(87)90062-2.

19 Jens Richter, Hans-Richard Demel, Florian Tiefenböck, Luise Heine, and Martina Feichter. Symptom-Checker, 2026. URL: www.netdoktor.at/symptom-checker/.

20 Claude Elwood Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27(3):379–423, July 1948. doi:10.1002/j.1538-7305.1948.tb01338.x.

21 Ralf Strobl, Michael Grözinger, Andreas Zwergal, Doreen Huppert, Filipp Filippopulos, and Eva Grill. A Set of Eight Key Questions Helps to Classify Common Vestibular Disorders—Results From the DizzyReg Patient Registry. *Frontiers in Neurology*, 12:670944, April 2021. doi:10.3389/fneur.2021.670944.

22 Oliver A. Tazl, Alexander Perko, and Franz Wotawa. Conversational Recommendations Using Model-Based Reasoning. In *Proceedings of the 21th International Configuration Workshop*, volume 2467 of *CEUR*, 2019. URL: ceur-ws.org/Vol-2467/paper-03.pdf.

23 Arsene Fansi Tchango, Rishab Goel, Zhi Wen, Julien Martel, and Joumana Ghosn. DDXPlus: A New Dataset for Automatic Medical Diagnosis. In *Advances in Neural Information Processing Systems 35*, pages 31306–31318, New Orleans, Louisiana, USA, 2022. Neural Information Processing Systems Foundation, Inc. (NeurIPS). doi:10.52202/068431-2270.

24 Kaishuai Xu, Yi Cheng, Wenjun Hou, Qiaoyu Tan, and Wenjie Li. Reasoning Like a Doctor: Improving Medical Dialogue Systems via Diagnostic Reasoning Process Alignment. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 6796–6814, Bangkok, Thailand, 2024. ACL. doi:10.18653/v1/2024.findings-acl.406.