Application of Inference Methods and Caching for Real-time Medical Diagnosis using BN2O Networks

An investigation of exact and approximate methods, and their use for asking questions

Jens Engholm Aggerholm
Casper Skov Mathisen

Computer Science, mi1010f20, Spring Semester 2020

Master Thesis Report

Computer Science
Aalborg University
http://www.aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Title:
Application of Inference Methods and Caching for Real-time Medical Diagnosis using BN2O Networks

Theme:
Variational inference, BN2O networks, medical diagnosis

Project Period:
Spring Semester 2020

Project Group:
mi1010f20

Participants:
Jens Engholm Aggerholm
Casper Skov Mathisen

Supervisor:
Thomas D. Nielsen

Page Count: 47

Date of Completion:
June 11, 2020

Abstract:

This paper is an investigation of inference methods in graphical models applied in clinical decision support settings. We implement both exact and approximate algorithms designed for inference in $BN2O$ networks. As for the exact algorithms we use Quickscore, for which we suggest two caching strategies that can prove useful in an active diagnosis setting, where questions are asked to a patient and probabilities of interest need to be updated on the fly.
For the approximate algorithms we use variational methods that allow us to calculate upper and lower bounds on likelihoods as well as on posterior probabilities.
Finally we suggest an entropy based criteria for asking questions to a patient during active diagnosis and evaluate such methods on artificial patient cases.
We find that each of the caching methods are able to provide a significant speedup for exact inference in an active diagnosis setting. The entropy based questioning method shows some promise in certain cases, but currently requires too much time for slightly larger cases.

Preface

This report is a master thesis marking the completion of the master program in computer science at AAU. The report contains some material that has been reused from a report published earlier by the same authors [1]. This has been noted in the top of the relevant sections. We would like to extend our thanks to Thomas Dyhre Nielsen for his supervision throughout this project.

Summary

This paper is an investigation of inference methods in graphical models for application in clinical decision support systems (CDSS). CDSS aid health professionals in providing health services towards patients. This can be by guiding a clinician through the diagnose process by suggesting the next test to perform or the next question to ask the patient, as a step on the way towards the final diagnosis. We look at different ways to choose a question to ask in order to diagnose a patient, as well as how to determine the probabilities needed to both update probabilities based on the answer, and those needed to choose the questions themselves.

We implement both exact and approximate algorithms designed for inference in $BN2O$ networks, in order to investigate how useful the different inference algorithms are, and which advantages they have over each other.

The algorithm we are using to determine exact probabilities is Quickscore, which is an algorithm based on certain factorizations that have reduced the complexity of inference from being exponential in the number of diseases, to being exponential in only the number of positive findings. Based on this algorithm we suggest two caching strategies that can prove useful in an active diagnosis setting, where questions are asked to a patient and probabilities of interest need to be updated on the fly.

For the approximate algorithms we use variational methods that allow us to calculate upper and lower bounds on likelihoods as well as on posterior probabilities. We evaluate these bounds in order to help determine whether these can be useful both during the questioning process and for the final diagnosis.

Finally we suggest an entropy based criteria for asking questions to a patient during active diagnosis and evaluate such methods on artificial patient cases.

We find that each of the caching methods are able to provide a significant speedup for exact inference in an active diagnosis setting. The entropy based questioning method shows some promise in certain cases, but currently takes too long for slightly larger cases.

The project that this report documents was proposed in collaboration with Ambolt ApS, who also provided the necessary data for experiments.

Contents

Chapter 1

Introduction

section 1.1 and section 1.3 are based on [1].

Computer systems within the healthcare sector mainly serve the purpose of storing and maintaining electronic medical records of patients along with processing this information in order to enhance the medical care service provided by professionals. Computer systems that supports the clinical decision making process exists in various forms and are generally categorised as *Clinical Decision Support Systems*(CDSS). These systems can vary in different ways[2, 11]; for instance in the type of output a CDSS generates can range from simple warnings and reminders to suggestions during diagnosis and prescription. Also the timing in which support is offered can vary. Support can be provided before, during or after a diagnosis setting takes places. A CDSS can furthermore be specialised to deal with certain areas of medicine or it can be designed with the aim of supporting general clinical diagnosis. Common for all of these systems is that they require an input consisting of some form of patient medical data and a mechanism for reasoning about this data in order to provide an output for interpretation by a health professional.

The role of a CDSS can be to provide "the answer" in the form of a final diagnosis of a patient, or it can be to simply suggest the next question to ask or the following test to perform, given a patient's known health record.

A CDSS can be non-knowledge based or knowledge based[2]. Within the non-knowledge based domain, artificial neural networks (ANNs) can be mentioned as a tool. An ANN can be trained on databases containing diagnosis examples and doesn't need parameters to be specified by experts. However the results provided by these methods often lack in reliability and are difficult to explain. Furthermore these models require large amounts of data to be configured.

A knowledge based system on the other hand requires model parameters to be provided. These could be in the form of probabilistic connections between diseases and symptoms. Graphical models such as Bayesian Networks serve as candidates for this expert knowledge to be compiled into. These types of models are comparably more interpretable than their ANN counterparts.

This project was originally proposed in collaboration with Ambolt APS that is

developing IntMed. IntMed is a commercial decision support tool developed in partnership with doctors, which aims to support the process of diagnosis by suggesting question to ask a patient based on the already established knowledge base and records of the patient. IntMed uses a combination of gathered medical data statistics and expert knowledge that are processed into probabilities regarding diseases and their associations with symptoms.

1.1 Probabilistic representation

Graphical models have often been applied in the medical domain[6], in order to represent a distribution over multiple random variables and their dependencies. *Bayesian Networks*, which are a specific type of graphical model, are well suited for medical diagnosis, since these types of networks can handle the uncertainties that are inherent within this domain. They serve as tools for updating prior beliefs after taking evidence into account, such as a set of patient symptoms. The majority of CDS systems that utilise some form of probabilistic reasoning, use these belief networks as their framework for representing the domain dependencies[8].

A Bayesian network consists of a number of random variables representing an event in the domain, and directed edges representing the relation between these variables. For each variable node in the network we have a conditional probability distribution that depends on the parent variables.

The knowledge base of the IntMed system contains information about diseases, their prevalence and which findings they cause. It also contains demographic data, such as which people are more likely to suffer from certain diseases, which diseases are likely to result in other diseases, and which bodypart a symptom is related to. Some of this comes from statistical data, combined with expert knowledge regarding the relation between diseases and findings, and also between diseases.

We represent part of this data as a Bayesian network. More specifically we extract the diseases with prevalences to determine prior probabilities, along with symptoms and the relations between these and their disease parents, i.e. the chance of a symptom manifesting itself given that a specific disease is present.

Any demographic data can be compiled into the network by selecting only the data corresponding to the specific patient demographic. This results in a two-layer (bipartite) Bayesian network with diseases d_j, and symptoms, or more generally findings, f_i. The findings are modelled using noisy-OR[10] over the disease parents, which assumes that each disease works independently to cause the finding. This significantly reduces the number of probabilities that needs to be specified in the model.

The structure which we have just described is called a BN2O network and is illustrated in Figure 1.1. It means a **Bayesian Network** with **2** layers using noisy **OR**. This is similar to the structure used by the QMR-DT[7] model.

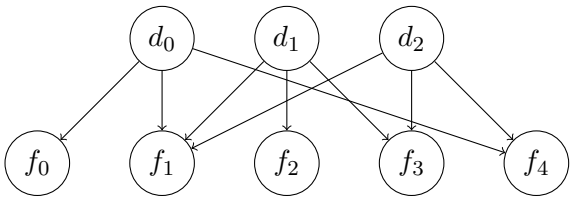


Figure 1.1: A BN2O network over the relationship between some diseases and the symptoms they cause individually

1.2 Diagnosis

In order to diagnose a patient, one first has to learn which symptoms the patient has. Some of these symptoms may be provided directly by the patient, while it might be necessary to ask about others. Since there are a lot of potential symptoms, a person performing diagnosis needs to know which questions are relevant to ask and which are best at narrowing down potential diagnoses.

One way of doing this is by assigning a value to each possible unknown symptom that serves as a score determining how good a specific question is in the current moment. The questions would then be sorted according to this score in decreasing order, and the top ones would be the best questions to ask based on the criteria the score was assigned on. In this report we suggest a entropy based criteria for the scoring of questions.

Returning to the diagnosis problem itself, there are different ways of giving a diagnosis. The simplest way is to simply provide a ranked list of individual diseases that might be causing some of the symptoms. This could potentially be expanded by allowing one or more preexisting diseases that are always present alongside the disease being looked at. But a more ambitious goal would be to search for the configuration of diseases that is most likely to be causing the symptoms. Feili Yu et al. [3, 14] takes this maximum a posteriori (MAP) configuration approach using a variety of heuristic algorithms to find these configurations that explains the observations.

1.3 Inference

A common task when working in Bayesian networks is that of inference. This is the process of updating probabilities on variables in the network, based on evidence on other variables. In the context of medical diagnosis a relevant probability to infer could be the posterior marginal $P(d_j \mid f)$; the probability of having a certain disease d_j, given a set of known findings f. This would correspond to making observations on a set of symptoms and from that adjust the belief in the prior probabilities for the possible diseases. Probabilities on this form are found through the process of marginalising out variables we are not interested in, by summing over all possible instantiations of those variables, the complexity of which depends on which order it is done.

Common inference algorithms can typically not be used in the domain of diagnosis because of the high number of connections between some diseases and symptoms. Some symptoms have a high number of disease parents, and the complexity of the common inference algorithms is so that the calculations become intractable.

Consider for example the junction tree algorithm described in [9]. It works by moralizing and triangulating the network graph, and identifying the cliques within it. Moralizing is done by adding edges between parent nodes and making the graph undirected. Triangulation makes sure that there are no cycles longer than three nodes without a chord, which is an edge that is not part of the cycle connecting two nodes in the cycle. The cliques can then be identified to form the junction tree by making a maximum spanning tree on the number of shared variables. The added edges on the graph correspond to those that would be added if using variable elimination, with each clique corresponding to a set of potentials. Inference can then be performed using a message passing algorithm on the resulting junction tree.

The problem with this method, is that the complexity of inference increases exponentially with the size of the largest clique in the moralized, triangulated graph [5]. Consider for example the findings cough and fever which have 23 and 19 causes respectively in the IntMed data set. This results in clique sizes of at least 24 and 20, i.e. the finding and its parents. It is conceivable other large cliques might form due to moralizing and triangulating shared parents.

Going back to the QMR-DT database, this becomes a problem for a lot of standard diagnosis cases. In [5] Jaakkola and Jordan work with a set of cases where the median size of the maximal clique is 151.5 nodes. This renders the junction tree algorithm intractable in this context.

One solution is to find a specialised algorithm, that reduces the complexity for the specific type of network we are working with. One such algorithm is Quickscore[4] which makes use of certain factorizations in the probabilities of findings in order to reduce the complexity to being exponential in the number of positive findings. In this paper we present two caching strategies applied to the Quickscore algorithm that becomes useful when calculating multiple posteriors $P(d_k|f)$ for $k = m...n$ and when calculating likelihoods of findings sequentially.

Another solution to the problem of doing inference is to consider approximate inference algorithms, which seek to gain speed by compromising on accuracy on the sought after probabilities. Shwe and Cooper [12] experimented with stochastic sampling methods using "likelihood-weighted sampling" and obtained reasonable results for some difficult cases for the QMR-DT network.

Vomlel and Tichavský suggest reducing the size of the inference problem, for example through methods such as tensor decomposition[13], which is able to reduce the size of the conditional probability tables (CPTs), and is shown to provide smaller CPTs than alternatives such as parent divorcing.

An alternative for approximate inference is that of variational inference. The variational approach seeks to replace certain distributions with simpler ones using *variational parameters*. This replacement of parameters renders the calculation an

approximation and the objective then becomes to find the parameters that result in the best approximation.

In [5] Jaakkola and Jordan present the idea of replacing the certain conditional probabilities of some of the positive findings with upper and lower bounds. These bounds are possible to absorb into the priors in time linear in the number of diseases, while reducing the exponent when handling the remaining findings. As we will show one of the advantages of this variational approach is that it allows us to provide bounds on the likelihoods themselves as well as on posterior probabilities. We implement this variational method and evaluate it on cases we generate based on the provided data set. All experiments will be run with data provided by Ambolt ApS.

Chapter 2

Preliminaries

section 2.1, section 2.2 and subsection 2.2.1 are based on [1]

2.1 Bayesian networks

The fundamental tool we use to represent the causal relationships within the domain is Bayesian networks. These are well suited to capture the inherent uncertainty within the medical domain in a quantitative way.

The structure of a Bayesian network consists of a set of variables X along with a set of directed edges between these variables. The structure must be so that there is no cycles. Each variable in X can take on a finite number of states that are mutually exclusive. A Baysian network specifies the joint probability over the variables in the network:

$$P(X) = \prod_{a \in X} P(a|\pi_a), \tag{2.1}$$

where π_a are the parents of variable a. Therefore all variables has to have a conditional probability table that specifies the probability over each of the states of the variables given the state of its parents.

2.2 BN2O network

One specific type of Bayesian network that is suitable for the domain of diagnosis are BN2O networks (Figure 2.1). These are bipartite graph structures with the top layer containing diseases with edges connecting to findings (symptoms) which are located in the bottom layer. This represents the causal direction in the relationship between diseases and symptoms; a given disease can cause a patient to experience one or more symptoms. These connections quantify the probability with which a disease, if present alone, causes a finding. These probabilities along with the probabilities of having each disease individually (the priors) is exactly the data that is available to us. Therefore the data fits directly into these types of models without any alteration.

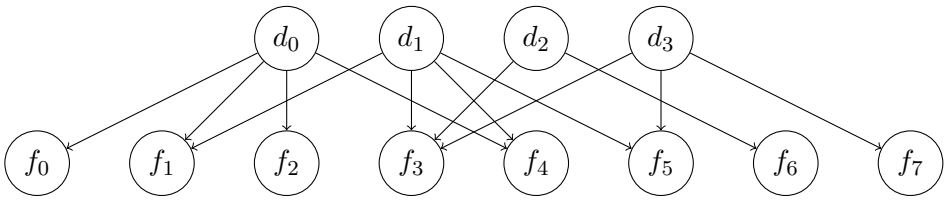


Figure 2.1: A BN2O network over the relationship between some diseases and the symptoms they cause individually

Certain observations are in order for this network. Multiple diseases can be present at the same time because they are not assumed to be mutually exclusive. The diseases are marginally independent and the findings are conditionally independent given an instantiation on the diseases.

The BN2O network specifies probabilities of the form:

$$P(f, d) = \prod_i P(f_i|d) \prod_k P(d_k), \tag{2.2}$$

where f and d are binary vectors giving the state (present/absent) of findings and diseases respectively.

2.2.1 Noisy-or

Often a finding will have multiple disease parents that can cause that finding to be present. If some finding f_i had a large number of parents, we would traditionally have to specify $p(f_i|d)$ for each configuration d of the parents π_i of the finding. This would cause the number of parameters of the model to grow exponentially with the number of parents for each finding. The number of configurations grows in $O(2^n)$, where n is the number of parent diseases, which means at just 10 disease parents we must already specify more than 1000 probabilities for that finding, and at 20 disease parents it grows to over a million.

BN2O networks seek to avoid this by assuming *noisy OR-gates* to specify the probability of a finding given the state of its disease parents. The Noisy OR-gate was originally proposed by Pearl[10] and the underlying intuition is that multiple conditions can work independently to cause a certain effect. Say for instance that a certain set of diseases are known to cause a common symptom like fever. The more of these diseases that a patient has, the more likely he would be to develop a fever. Additionally if the patient has a disease that is not associated with fever, this fact would not decrease the probability of him developing a fever.

By introducing noisy OR-gates we decrease the number of parameters needed to be specified in the Bayesian network significantly. With noisy OR we only need to specify one parameter for each disease parent of a symptom, and that is the probability that disease, when present alone, will cause the symptom.

Consider a finding f_i which can potentially be caused by two diseases d_1 and d_2. The probability of f_i being present, denoted by f_i^+, given that only one disease d_1 is

present is

$$P(f_i^+|\text{only } d_1) = q_1.$$

Conversely the probability of that finding not being present is

$$P(f_i^-|\text{only } d_1) = 1 - P(f^+|\text{only } d_1) = 1 - q_1$$

Now suppose a disease configuration (an assignment of 0 or 1 to all diseases) d is given. A symptom is not present if all of the present diseases d^+ fail to cause the symptom:

$$P(f_i^-|d) = \prod_j P(f_i^-|\text{only } d_j)^{d_j} = \prod_j (1 - q_{ij})^{d_j} \tag{2.3}$$

This is the noisy-OR model. Now the probability of a single negative finding can be found by marginalising out the disease variable from the joint probability distribution:

$$P(f_i^-) = \sum_d P(f_i^-|d)P(d). \tag{2.4}$$

Using noisy-OR and the assumption that diseases are marginally independent, this can be written as

$$P(f_i^-) = \sum_d \prod_j P(f_i^-|\text{only } d_j)^{d_j} \prod_j P(d_j) \tag{2.5}$$

The probability of multiple negative findings can similarly be found by:

$$P(f^-) = \sum_d P(f^-|d)P(d). \tag{2.6}$$

Assuming the findings are independent given a disease instance, this becomes

$$\begin{aligned} P(f^-) &= \sum_d \prod_{f_i \in f^-} p(f_i^-|d)p(d) \\ &= \sum_d \prod_{f_i \in f^-} \prod_j p(f_i^-|\text{only } d_j)^{d_j} p(d) \\ &= \sum_d \prod_{f_i \in f^-} \prod_j (1 - q_{ij})^{d_j} \prod_j p(d_j), \end{aligned} \tag{2.7}$$

where q_{ij} is the probability that disease d_j if present alone will cause finding f_i.

The corresponding probability for a positive set of findings becomes:

$$P(f^+) = \sum_d \prod_{f_i \in f^+} \left(1 - \prod_{d_j \in d^+} (1 - q_{ij})\right) \prod_{d_j \in d} p(d_j). \tag{2.8}$$

This expression does not factorise over the diseases as Equation 2.7 does, and the number of terms grows exponentially with the number of positive findings making it computationally more expensive to perform the summation over the disease configurations compared to Equation 2.7.

2.2.2 Jensen's Inequality

A function $f(x)$ is said to be convex if every cord connecting two points on the graph of f lies on or above the function. If f has the opposite property where every connecting cord lies on or below the function, then f is said to be concave. It can be shown that for any set of points x_i, the following inequality holds for a convex function f:

$$f\left(\sum_i \lambda_i x_i\right) \leq \sum_i \lambda_i f(x_i), \tag{2.9}$$

where we require that $\sum_i \lambda_i = 1$ and $\lambda_i \geq 0$ and the inequality is reversed if f is concave. This relationship is called Jensen's inequality.

Chapter 3

Theory

section 3.2 until and including subsection 3.2.1, and subsection 3.3.2 are based on [1]

3.1 Questions

Often during the diagnosis of a patient there exists a starting point, or point of origin[2], from which the diagnosis will be based on. This point of origin can be in the form of the patients medical records, previously diagnosed diseases or the symptoms that the patient is describing when seeing the physician. In practice two individual physicians can have the same point of origin, and still follow different paths consisting of different tests and questions to the patient. This difference in the steps taken by two different physicians can be due to varying experience or domain of expertise, or simply due to one of them having a hunch that the other did not have. This raises the question of what principles should guide the question apparatus that determines the next question to ask or the next test to perform, based on the already obtained knowledge.

Different criteria for determining the next questions could be justified. One guiding principle might be to ask that question which, when answered, has the potential to reduce the set of possible diseases the most. The potential benefit from such a strategy would be to rapidly reduce the size of the set of diseases under consideration.

Provided we have no method of determining which questions have the biggest impact, we would have to rely on choosing the questions to ask either randomly or in some predetermined order. If we do not otherwise limit the pool of questions, this means that we are very likely to ask about a finding that is negative, due to the limited number of findings being caused by a disease compared to the total amount of findings in the network. There is also no guarantee on the impact the findings will have on the disease posteriors.

3.1.1 Entropy based questioning

Here we present a question strategy based on entropy. Specifically we introduce a value function V, which is defined as the negative sum of the binary entropy of each disease posterior $P(d_i|f)$ of the diseases under consideration[9]:

$$\begin{aligned} V(D|f) &= -\sum_i Ent(d_i|f) = \sum_i \sum_{d_i} P(d_i|f) log P(d_i|f) \\ &= \sum_i [c_{if} log(c_{if}) + (1 - c_{if}) log(1 - c_{if})], \end{aligned} \tag{3.1}$$

where $V(D|f)$ represents the value of the current posteriors over the individual diseases given the known set of findings f up until this point, and $c_{if} = P(d_i = 1|f)$.

Since the entropy can be interpreted as the uncertainty of a random variable, we seek to minimise this quantity, and therefore maximise the introduced value function. The intuition behind the approach is, that we score questions according to how much they are able to reduce the overall uncertainty, when answered. The best next question to ask will then be the one that increases the value function the most. The answer to the next question is unknown until it is asked. Therefore we choose the next question based on the finding which has the highest expected value EV:

$$EV(f_i|f) = P(f_i^+|f)V(D|f, f_i^+) + P(f_i^-|f)V(D|f, f_i^-) \tag{3.2}$$

This quantity can be calculated for all the findings under consideration, and then sorted. That finding which then has the highest expected value, will, according to this method, serve as the best candidate for the next question.

Based on testing we have found that this method performs better if the value is shifted into the positive range, i.e. by adding approximately 0.301 to the contribution from each posterior in Equation 3.1:

$$V(D|f) = \sum_i (0.301 + [c_{if} log(c_{if}) + (1 - c_{if}) log(1 - c_{if})]), \tag{3.3}$$

3.1.2 Relevant findings

A different approach to the above is to determine which diseases and findings are relevant for the current case. If a patient goes to the doctor and describes a set of symptoms, then it is to be expected that the disease(s) to be diagnosed are among the parents of those findings. In the example given by Figure 3.1 a single symptom f_3 is provided, which is related to two diseases d_1 and d_2. These represent the possible causes for said symptom.

If we wish to determine which of the two diseases is causing the known symptom, then we will have to ask additional questions, more specifically we want to know about the findings that can be caused by the relevant diseases. Looking once again

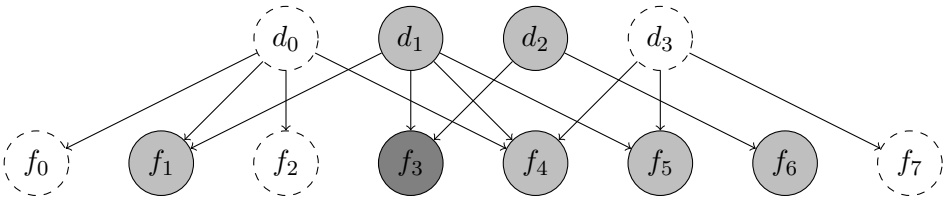


Figure 3.1: The dark gray finding f_3 is a known present symptom. The light gray diseases d_1 and d_2 are its parents and possible explanations for f_3. The light gray findings f_1, f_4, f_5, f_6 are children of those two diseases, and are therefore possibly present findings given the diseases. The dashed findings and diseases are not relevant for the finding f_3, as they are not directly contributing to the same set of diseases.

at Figure 3.1 the findings that can be caused by the two diseases are, excluding the initial finding, f_1, f_4, f_5 and f_6. The dashed remaining diseases and findings are not going to be considered initially.

This does not rule out the possibility of the complete diagnosis containing diseases that are not in the set of parents, but that would mean that the patient did not notice or find some unmentioned symptom relevant, and that the disease causing said symptom would not be able to cause any of the initial findings. If we update the set of relevant diseases after each question, then we would catch the case where the disease becomes a possible parent to any of the currently known findings.

By limiting the search scope in this way, we are able to reduce the time it takes for each step of the questioning algorithm - both in terms of determining which question to ask, and when providing the posterior probabilities$P(d_j \mid f)$ afterwards.

This method of reducing the search space can be combined with the previous methods in order to reduce the number of findings and diseases that must be iterated over. Care must be taken that we do not break them, for example if we were to calculate the entropy over a different set of diseases for each potential question.

As you learn about more and more positive findings, the number of parent diseases naturally grows. This means that the time to perform inference also grows. One way to get around this is to signify certain positive findings as being key symptoms. The potential key symptoms include anything from a small subset of the initial findings, for example more urgent symptoms, all the way to including a number of questions. With the help of these key symptoms it is possible to focus on determining the more likely causes for these findings, rather than spreading the questioning effort across all of the encountered findings and diseases. We do not focus on these key symptoms beyond limiting the relevant findings to be based on only the initial findings.

3.2 Quickscore

Quickscore is an algorithm that exploits a certain factorization for the BN2O network[4, 1], which gives a simplified expression for the probability $P(f_i^-)$ of a negative

finding f_i^- as a product over the diseases, given by:

$$P(f_i^-) = \prod_{j=1}^{n} \left[P(f_i^- | \text{only } d_j) P(d_j^+) + P(d_j^-) \right]. \tag{3.4}$$

This probability is of the same form as Equation 2.5 that requires summation over all disease configurations.

The two calculations have a significant difference in their complexity; Equation 2.5 is a sum over 2^n terms while Equation 3.4 is a product over n terms.

The probability for multiple negative findings using Quickscore becomes

$$P(f^-) = \prod_{j=1}^{n} \left[\prod_{f_i \in f^-} P(f_i^- | \text{only } d_j) P(d_j^+) + P(d_j^-) \right]. \tag{3.5}$$

This can be found by combining Equation 2.5, Equation 2.7 and the above expression for a single negative finding.

We now return to the task of calculating the probability of positive findings, starting with finding the probability of two positive findings, namely:

$$\begin{aligned} P(f_1^+, f_2^+) &= \sum_d P(f_1^+|d) p(f_2^+|d) P(d) \\ &= \sum_d [1 - P(f_1^-|d)][1 - P(f_1^-|d)] P(d) \\ = \sum_d P(d) - \sum_d P(f_1^-|d) P(d) - \sum_d P(f_2^-|d) P(d) + \sum_d P(f_1^-|d) P(f_2^-|d) P(d) \end{aligned} \tag{3.6}$$

The first term in the above expression is 1. The following three terms are just $P(f_1^-)$,$P(f_2^-)$ and $P(f_1^-, f_2^-)$ respectively, so this can be rewritten using Equation 3.5:

$$\begin{aligned} P(f_1^+, f_2^+) = 1 \\ &- \prod_{j=1}^{n} [P(f_1^- | \text{only } d_j) P(d_j^+) + P(d_j^-)] \\ &- \prod_{j=1}^{n} [P(f_2^- | \text{only } d_j) P(d_j^+) + P(d_j^-)] \\ &+ \prod_{j=1}^{n} [P(f_1^- | \text{only } d_j) P(f_2^- | \text{only } d_j) P(d_j^+) + P(d_j^-)] \end{aligned} \tag{3.7}$$

According to [4] the above expression generalises to an arbitrary set of findings by:

$$P(f^+, f^-) = \sum_{f' \in 2^{f^+}} (-1)^{|f'|} \prod_{j=1}^{n} \left(\left[\prod_{f_i \in f' \cup f^-} P(f_i^- | \text{only } d_j) \right] P(d_j^+) + P(d_j^-) \right), \tag{3.8}$$

where the negative findings f^- are also included. 2^{f^+} is the power set of the positive findings f^+.

In itself Quickscore gives the probability of a specific finding configuration $P(f) = P(f^+, f^-)$ where $f = f^+ \cup f^-$ are the present and absent findings. We can modify the algorithm slightly using the product and chain rules to calculate the posterior probability of a specific disease d_j as follows:

$$P(d_j \mid f) = \frac{P(d_j, f)}{P(f)} = \frac{P(f \mid d_j)P(d_j)}{P(f)} \tag{3.9}$$

Calculating $P(f|d_j^+)$ can be done by setting $P(d_j^+) = 1$ in the Quickscore algorithm, as stated in the Quickscore paper [4]. Similarly the probability of not having a disease can be determined by setting $P(d_j^+) = 0$.

3.2.1 Absorbing negative findings

In this section we take a closer look at how negative findings can be absorbed iteratively when calculating posteriors for the diseases of the form $P(d_k|f^-)$. Here we consider one negative finding but this could be expanded to multiple negative findings:

$$P(d_k \mid f_i^-) \propto \sum_{d \setminus d_k} P(f_i^- \mid d)P(d), \tag{3.10}$$

where the sum is over the diseases except d_k:

$$\begin{aligned}
\sum_{d \setminus d_k} P(f_i^- \mid d)P(d) &= \sum_{d \setminus d_k} \prod_{j \in \pi_i} (1 - q_{ij})^{d_j} P(d_j) \prod_{h \notin \pi_i} P(d_h) \\
&= (1 - q_{ik})^{d_k} P(d_k) \sum_{d \setminus d_k} \prod_{\substack{j \in \pi_i \\ j \neq k}} (1 - q_{ij})^{d_j} P(d_j) \prod_{h \notin \pi_i} P(d_h) \\
&\propto (1 - q_{ik})^{d_k} P(d_k),
\end{aligned} \tag{3.11}$$

where we have split the product over the disease into to those that are parents of f_i and those that are not, and then moved the factors involving k outside of the sum.

The probability $P(d_k^+|f_i^-)$ can now be found by setting $d_k = 1$ in the the above expression and normalising, which gives us:

$$P(d_k^+ \mid f_i^-) = \frac{(1 - q_{ik})P(d_k^+)}{(1 - q_{ik})P(d_k^+) + P(d_k^-)}. \tag{3.12}$$

This expression can be used iteratively to absorb negative findings, when calculating probabilites of the form $P(d_k \mid f^+, f_m^-, ..., f_n^-)$.

Say now that we want to calculate the likelihood $P(f^+, f_i^-)$ of a set of positive findings f^+ and a single negative finding f_i^-. We could write this out as:

$$P(f^+, f_i^-) = \sum_d P(f^+|d) \prod_{j \in \pi_i} (1 - q_{ij})^{d_j} P(d_j) \prod_{j \notin \pi_i} P(d_j). \tag{3.13}$$

We can move the factors regarding the negative finding as well as the prior probability into a new probability distribution $P'(d)$ given by:

$$P'(d) = \prod_{j \in \pi_i} (1 - q_{ij})^{d_j} P(d_j) \prod_{j \notin \pi_i} P(d_j), \tag{3.14}$$

which is simply obtained by multiplying $P(d_j = 1)$ with $(1 - q_{ij})$ for each disease parent d_j of finding f_i. After this update we have:

$$P(f^+, f_i^-) = \sum_d P(f^+|d)P'(d), \tag{3.15}$$

where the updated $P'(d)$ has replaced the original distribution. This absorption can be performed for an arbitrary number of negative findings.

3.2.2 Caching strategies

In this section we present two different caching strategies that can be used when calculating probabilities using the Quickscore algorithm presented in the previous section. The first caching strategy pertains to the calculation of posterior probabilities $P(d_k|f)$ for two or more different diseases d_k. This is an extension of the caching method used in [1].

The second strategy is relevant when we seek the joint probability $P(f_m, ...f_n)$ of findings in a sequential way; first $P(f_m, ..., f_n)$ then $P(f_m, ..., f_n, f_{n+1})$ and so on. This is useful both when we learn of additional findings during the questioning process, as well as when we apply the entropy based question strategy presented in subsection 3.1.1. In this setting we need to determine the probability $P(f_i|f)$ of a finding f_i given a set of already known findings $f = f^+ \cup f^-$, for a set of candidate findings $f_i \in f^c$:

$$P(f_i|f) = \frac{P(f_i, f)}{P(f)}.$$

We can also reuse the calculations of $P(f_i, f)$ to perform posterior calculations $P(d_j \mid f_i, f)$ given those same findings, which is especially of interest when calculating the entropy for each possible question.

Once a finding f_k has been chosen as the next test, it is added to the set of known findings $f \leftarrow f \cup f_k$, and it is removed from the set of candidate findings $f^c \leftarrow f^c - f_k$. The process is now ready to be repeated with the updated set of known findings and candidate findings. Since we already know the probability $P(f)$ of the known set of findings f we can exploit the sequential caching when we seek to determine $P(f, f_i)$.

We first present the strategy for calculating multiple posteriors, and then the strategy for sequential joint probabilities.

Multiple posterior caching

The brute force way of calculating $P(d_k|f)$ for m different diseases, would be to repeatedly calculate Equation 3.9 m times; once for each disease. This results in $2m$

calculations in total, since each of these requires two applications of Equation 3.8. This method fails to exploit the common finding probability $P(f)$ that could be calculated once at the outset and then used for each posterior, since this doesn't change for the different diseases. This already cuts the number of calculations in half, leaving us with $m + 1$ joint probabilities.

Let us take a look at the complexity of calculating the joint probability itself from Equation 3.8. The sum over the powerset has a complexity of $2^{|f^+|}$. Then we get to the product over all diseases, that we must compute for each of the powerset elements $f^{'}$. Let us call the number of factors n. Then we get to the inner product over finding probabilities $P(f_i^- \mid \text{only } d_j)$, for a subset of the present findings $f' \subseteq f^+$, and all of the negative findings f^-. The number of negative findings remains constant for a given calculation. The number of positive findings goes from 0 to $|f^+|$. The average cardinality of the sets in the powerset of f^+ is $|f^+|/2$. This gives a total complexity for a single joint probability of $2^{|f^+|}(n(|f^-| + \frac{|f^+|}{2}))$. Say that we want to calculate m posteriors. This would have a complexity of $(m+1)(2^{|f^+|}(n(|f^-| + \frac{|f^+|}{2})))$.

However there are further possibilities for caching optimisations that requires investigation of the Quickscore expression presented earlier in Equation 3.8 and restated here with colours indicating which calculations are being cached:

$$P(f^+, f^-) = \sum_{f' \in 2^{f^+}} (-1)^{|f'|} \prod_{j=1}^{n} \left(\left[\prod_{f_i \in f' \cup f^-} P(f_i^- | \text{only } d_j) \right] P(d_j^+) + p(d_j^-) \right). \tag{3.16}$$

We introduce two data structures A and B which hold the precalculated values that will be utilised when calculating the final posteriors of interest.

A holds the calculations marked with blue in the expression above. That is, each entry in A is calculated by taking the product of the inhibitors $P(f_i^-|\text{only } d_j)$ (which is equal to $1 - q_{ij}$) over $f^{'}$ (the positive findings associated with the current term of the summation) and f^- (the negative findings) given that only d_j is present. This means that the entries in A will be indexed by $f^{'}$ and a disease d_j so the number of entries in A will be $|2^{f^+}|N_d$, where N_d is the total number of diseases.

The second data structure B is a derivative of A, and it holds the calculation marked with red above. Each entry in B holds one instance of this calculation. That is, entries in B are indexed by a set of findings $f^{'}$ and is calculated as a product over the diseases; hence the number of entries in B is $|2^{f^+}|$.

The pseudocode for calculation of the data structures A and B is presented in

algorithm 1.

Algorithm 1: Construction of datastructures A and B for calculating multiple posterior values $P(d_k|f)$

1 **Input** : $findings f^-, f^+$;
2 **Output** : $(A, B), P(f^-, f^+)$;
3 $A = [|2^{f+}|, N_d]$; `// Initialize two dimensional array A`
4 $B = [|2^{f^+}|]$; `// Initialize one dimensional array B`
5 $P(f^-, f^+) = 0$;
6 **for** f' ***in*** 2^{f+} **do**
7 $\quad b_{f'} = 1$;
8 $\quad$ **for** j ***in*** d **do**
9 $\quad\quad a_{f'j} = 1$;
10 $\quad\quad$ **for** i ***in*** $f' \cup f^-$ **do**
11 $\quad\quad\quad a_{f'j} \leftarrow a_{f'j} \cdot (1 - q_{i,j})$;
12 $\quad\quad$ **end**
13 $\quad\quad A(f^{'}j) \leftarrow a_{f'j}$;
14 $\quad\quad b_{f'} \leftarrow b_{f'} \cdot (a_{f'j} P(d_j^+) + P(d_j^-))$
15 $\quad$ **end**
16 $\quad P(f^-, f^+) \leftarrow P(f^-, f^+) + (-1)^{|f^{'}|} \cdot b_{f'}$; `// The finding probability`
$\quad$ `// is the sum of entries in` B `with correct sign`
17 $\quad B(f^{'}) \leftarrow b_{f'}$
18 **end**

Once the dictionaries A and B have been constructed, we can use these to calculate each of the posteriors of interest, by performing one iteration through the entries in B per posterior of interest. To see this we have to consider the content of B.
First off we make the observation that the only difference in the calculation of $P(f|d_k)$ and $P(f|d_l)$ is that the corresponding prior probability $P(d_j = 1)$ is set to 1 when iterating over d_k or d_l respectively.
Since the entries in B correspond to the product marked with red for a given element of the powerset 2^{f^+} in Equation 3.16, we can calculate the finding probability simply by adding the cached terms from B while keeping the correct sign in every addition according to this equation (line 16 in algorithm 1).

However if we want the probability $P(f \mid d_k)$ of the findings f given a disease d_k, we need to consider the contribution to the product from the changed prior probability. To do this we simply need to divide out the factor $\prod_i P(f_i^- \mid \text{only } d_k)P(d_k^+)+P(d_k^-)$ pertaining to disease d_k, and multiply with the entry $A(f^{'}, d_k)$ for each sum element in B, since the effect of this is to set $P(d_k^+) = 1$ in the product over the diseases. This step essentially replaces specific factors in the elements of B with new factors that incorporate the information from the disease which we are conditioning on. Let the diseases of interest be d_Q. If we are querying $N_q = |d_Q|$

different disease posteriors, this is how many times we sweep over the elements in B.

The utilisation of A and B for calculation of the posteriors in d_Q is presented in algorithm 2.

The expected savings of using this caching method, is that we no longer have to perform the full calculations inside the sum over the powerset for each posterior we want to calculate. We recall that we originally had a complexity of $(m+1)(2^{|f^+|}(n(|f^-|+\frac{|f^+|}{2})))$ to calculate m posteriors. With the presented caching strategy it becomes possible to reduce this. First we can calculate the A and B, which is of complexity $2^{|f^+|}(n(|f^-|+\frac{|f^+|}{2}))$, where we can simultaneously calculate the likelihood $P(f)$. We then want to calculate all m posteriors. For each posterior we sum together $2^{|f^+|}$ terms each with an adjustment (a division and a multiplication) that takes constant time. These terms are what is contained in B, so performing this sweep m times has complexity $m \cdot 2^{|f^+|}$.

Putting it all together this gives us a complexity of $2^{|f^+|}(n(|f^-|+\frac{|f^+|}{2}))+m(2^{|f^+|})$. Importantly, the complexity from inside the joint probability calculation, regarding n and m, has been separated from the additional complexity of the posterior calculation regarding m. The first presented brute first strategy requires $(m+1)n$ iterations over 2^{f+} while with the suggested caching we require $m+n$.

Algorithm 2: Utilisation of datastructures A and B for calculating multiple posterior values $P(d_k|f)$

Input : dictionaries(A, B), query diseases $d_Q = \{d_i\}$;
Output : PQ;
$PQ = [|d_Q|]$; `// Initialise array of size` $|d_Q|$ `that holds the posterior of each disease of interest`
for j ***in*** d_Q **do**
    $posterior_q \leftarrow 0$;
    for $f^{'}$ ***in*** 2^{f+} **do**
        $sign \leftarrow (-1)^{|f^{'}|}$;
        $e \leftarrow \frac{B(f^{'})}{[A(f^{'},j)\cdot P(d_j^+)+P(d_j^-)]} \cdot A(f^{'}, j)$;
        $posterior_q \leftarrow posterior_q + sign \cdot e$;
    end
    $PQ(j) \leftarrow posterior_q$
end

Sequential joint probabilities

The second caching strategy we present is relevant in settings where we seek joint probabilities of findings in a sequential way. Say we want to know the probability $P(f_1)$ of a single finding f_1. Then we want to know the probability $P(f_1, f_2)$ of this

first finding and a second finding f_2. Then we add another finding and so forth. This could be achieved by repeatedly applying Equation 3.8, each time with the set of findings of interest. This method would be the brute force way to calculate these probabilities, but as in the previous section this method fails to exploit possibilities for caching, since some of the same calculations are being carried out multiple times each time a finding is added to the pool. Recall that the complexity of calculating the joint probabilities for a given set of findings is $(2^{|f^+|}(n(|f^-| + \frac{|f^+|}{2}))$.

This caching strategy is mainly concerned about updating the datastructure A. We do not explicitly mention B, but it could be updated using the results added to A. A is initialised once the first finding probability $P(f_1)$ is requested. Every time a finding f_l is added, that is going from $P(f_1, ..., f_k)$ to $P(f_1, ..., f_k, f_l)$, A is updated. The update of A happens in one of two ways determined by the new finding f_l being positive or negative.

When a new negative finding is added we simply update all of the entries in A. Noting again that A holds the calculation coloured with blue in Equation 3.16, we see that we need to multiply each entry with $P(f_l^-|\text{only } d_j)$ where f_l is the new finding and d_j the disease which we are currently iterating over. This operation requires a sweep through all of A so the finding probability can be calculated while simultaneously updating the data structure with the new information from the new finding. The algorithm that updates A with a new negative finding is presented in algorithm 3

Algorithm 3: Algorithm for updating the data structure A with the information from an added negative finding f_l

```
Input : dictionary A, new finding f_l;
Output : updated dictionary(A'), updated finding probability P_new;
P(f) ← 0 ;
for f' in 2^{f+} do
    sign ← (−1)^{|f'|} ;
    prod ← 1;
    for i in d do
        A(f', i) ← A(f', i) · (1 − q_{f_l,i});
        prod ← prod · (A(f', i)P(d_i^+) + P(d_i^−));
    end
    P(f) ← P(f) + sign · prod
end
```

When the new finding f_l to be added is positive, we need to add new entries to A. The number of new entries in A is equal to the number of existing entries, so the size of A is doubled. This is because the powerset in Equation 3.16 is over the set of positive findings which is now expanded with a new member. The already existing entries in A does not need to be altered. When expanding A to include the information from

the new finding, we simultaneously want to calculate the new finding probability. When the new finding is positive we update the finding probability by adding new terms to the old probability. This again is because adding a positive finding leads to extra terms stemming from the powerset over the positive findings. This extra contribution is saved in P_{add} in algorithm 4 which is the algorithm for updating A when adding a positive finding.

Algorithm 4: Algorithm for updating the data structure A with the information from an added positive finding f_l

```
1  Input : dictionary A, new finding f_l, old finding probability : P_old;
2  Output : updated dictionary A', updated finding probability P_new ;
3  P_add ← 0 ;
4  for f' in 2^{f+} do
5  |   sign ← (−1)^{|f'|} ;
6  |   prod ← 1;
7  |   for i in d do
8  |   |   A(f' ∪ f_l, i) ← A(f', i) · (1 − q_{f_l,i}) ;
9  |   |   prod ← prod · (A(f' ∪ f_l, i)P(d_i^+) + P(d_i^−));
10 |   end
11 |   P_add ← P_add + sign · prod ;
12 end
13 f^+ ← f^+ ∪ f_l ;
14 P_new ← P_old + P_add
```

In line 8 the calculation for the new entries in the updated A is carried out based on the already existing entries in A. In line 11 the contribution to the old probability is accumulated in the P_{add} variable before being added to the old probability in line 14.

So to reiterate, the complexity of the naive way of calculating the joint probability when a finding is added is $2^{|f^+|+1}(n(|f^-| + \frac{|f^+|+1}{2})$ if the new finding is positive and $2^{|f^+|}(n(|f^-| + 1 + \frac{|f^+|}{2})$ if this is negative. Using this caching strategy, we are able to reduce the complexity of adding both positive and negative findings. Since adding a negative finding is only updating the existing entries by multiplying the new factor onto the entries in A, the complexity is of this is $2^{|f^+|}(n)$. When adding a positive finding we only have to be concerned about the new powerset elements, for which the complexity is $2^{|f^+|}$. This caching exploits the already existing entries in A in order to reduce a complexity of $n(|f^-| + \frac{|f^+|}{2})$ into n. This means we end up with the same complexity for adding both negative and positive findings, noting that positive findings increases the complexity for adding subsequent findings exponentially.

All of the caching strategies described above, can be optimised further by only considering relevant disease parents when doing iterations such as the one in line 7 in algorithm 4. Relevant disease parents of a finding are those diseases with a connection to the finding - that is, d_j is a relevant parent of f_i if and only if $q_{ij} \neq 0$. Furthermore

in some settings it is possible to achieve speedup by absorbing the negative findings. These modifications will not be discussed here.

The experiment section of this report will demonstrate runtime performance of these methods compared to their brute force counterparts.

3.3 Variational Inference

In the previous sections we presented methods involving Quickscore as an algorithm for exact inference in BN2O networks. However some probabilities of interest quickly become intractable using only exact methods. This could for instance be the calculation of the likelihood $P(f)$ of a large number of positive findings. Previously we showed how negative findings can be dealt with in time linear in the number of associated disease parents, and how this was not the case for positive findings.

In this section we introduce variational methods that can be combined with exact inference methods in order to achieve tractable approximations to otherwise intractable probabilities. Specifically we show how introducing upper/lower bounds on single conditionals $P(f_i|d)$ provide upper/lower bounds on the likelihoods as well as posteriors.

3.3.1 Lower bound transformations

Here we introduce the transformation necessary to calculate lower bounds on joint probabilities of findings. This is done by introducing lower bounds for each of the conditional probabilities individually in order to get a lower bound on the likelihood of the findings[5]. We start out by considering the conditional probability of a positive finding in the exponential form:

$$\begin{aligned} P(f_i^+|d) &= 1 - P(f_i^-|d) = 1 - \prod_j (1 - q_{ij})^{d_j} \\ &= 1 - e^{log \prod_j (1-q_{ij})_j^d} \\ &= 1 - e^{-\sum_j d_j \theta_{ij}} \\ &= e^{\log(1-e^{-x})} = e^{f(x)}, \end{aligned} \tag{3.17}$$

where $f(x) = log(1-e^{-x})$, $x = \sum_j \theta_{ij} d_j$ and $\theta_{ij} = -\log(1 - q_{ij})$. By introducing the leak probability θ_{i0} for each finding (this could represent a finding being caused by some fault not included in the model) and applying Jensen's inequality to the

concave function f, we get:

$$\begin{aligned} & f(\theta_{i0} + \sum_j \theta_{ij} d_j) \\ & \geq \sum_j q_{j|i} f(\theta_{i0} + \frac{\theta_{ij} d_j}{q_{j|i}}) \\ & = \sum_j q_{j|i} [d_j f(\theta_{i0} + \frac{\theta_{ij}}{q_{j|i}}) + (1 - d_j) f(\theta_{i0})]. \end{aligned} \tag{3.18}$$

Here we have introduced a distribution $q_{\cdot|i}$ over the diseases for each finding. Making use of the fact that $\sum_j q_{j|i} = 1$, we can rewrite the last expression and state the inequality as:

$$f(\theta_{i0} + \sum_j \theta_{ij} d_j) \;\geq\; f(\theta_{i0}) + \sum_j q_{j|i} d_j [f(\theta_{i0} + \frac{\theta_{ij}}{q_{j|i}}) - f(\theta_{i0})] \tag{3.19}$$

Substituting this lower bound on f into Equation 3.17, we get the variational lower bound evidence:

$$\begin{aligned} P(f_i^+ | d, q_{\cdot|i}) &= e^{f(\theta_{i0}) + \sum_j q_{j|i} d_j [f(\theta_{i0} + \frac{\theta_{ij}}{q_{j|i}}) - f(\theta_{i0})]} \\ &= e^{f(\theta_{i0}) + \sum_j q_{j|i} d_j g_{ij}}, \end{aligned} \tag{3.20}$$

where we have introduced the auxiliary function $g_{ij} = f(\theta_{i0} + \frac{\theta_{ij}}{q_{j|i}}) - f(\theta_{i0})$. We now have a lower bound for the individual conditionals:

$$P(f_i^+ | d) \geq P(f_i^+ | d, q_{\cdot|i}) \tag{3.21}$$

In the Equation 3.20 the exponent is a sum over the diseases d_j. With this transformation it becomes possible to absorb the lower bound transformation of a positive finding into the priors in time linear in the number of associated diseases.

3.3.2 Upper bound transformations

It is possible to replace the exact probabilities $P(f_i^+ \mid d)$ for positive findings with corresponding upper bounds on the probabilities $P(f_i^+ \mid d, \xi)$. The way this is done is described in [5, 1], which gives us the following expression for the conditional probability of a given positive finding f_i^+:

$$P(f_i^+ \mid d) \leq P(f_i^+ \mid d, \xi_i) = e^{\xi_i \sum_j \theta_{ij} d_j - f^*(\xi_i)}. \tag{3.22}$$

where ξ_i is called a variational parameter and can be optimised to produce a tighter upper bound.

Unlike the corresponding exact probability, this upper bound, like the conditionals $P(f_i^-|d)$ of negative findings, factorises over the diseases, which means significant computational savings can be achieved just as with the lower bound transformed findings in the previous section. Once a positive finding has been transformed, i.e. variationally approximated, the joint probability is now an upper bound:

$$\begin{aligned} P(f) = \sum_d P(f^+|d)P(d) &\leq \sum_d P(f^+|d,\xi)P(d). \\ &\equiv P(f|\xi) \end{aligned} \tag{3.23}$$

Furthermore it is now possible to absorb the transformed findings into the disease priors:

$$\begin{aligned} \sum_d P(f^+|d,\xi)P(d) &= \sum_d \prod_i P(f_i^+|d,\xi) \prod_j P(d_j) \\ &= \sum_d \prod_i e^{\xi_i \sum_j \theta_{ij} d_j - f^*(\xi_i)} \prod_j P(d_j) \\ &= \sum_d \prod_i \prod_{j \in \pi_i} e^{\xi_i \theta_{ij} d_j - f^*(\xi_i)} \prod_j P(d_j), \end{aligned} \tag{3.24}$$

where we used the explicit representation of $P(f_i^+|d,\xi)$ and π_i are the disease parents of finding f_i. From Equation 3.24 we see how the transformation turns the conditional into factors that can be multiplied onto the priors instead of leading to an exponential number of terms that needs to be summed over as the exact conditionals did.
$P(f|\xi)$ is an upper bound approximation for some values of the parameters ξ. The objective is to find the assignment of values that minimises this bound. If we assume that the indices of the findings we wish to approximate is a, the upper bound becomes:

$$\begin{aligned} p(f^+ \mid \xi) &= \sum_d \left[\prod_{i \in a} P(f_i \mid d, \xi_i) \right] p(f_e^+ \mid d) p(d) \\ &= \sum_d \left[\prod_{i \in a} P(f_i \mid d, \xi_i) \right] P(d \mid f_e^+) p(f_e^+) \\ &\propto \mathbb{E} \left\{ \prod_{i \in a} P(f_i \mid d, \xi_i) \right\}, \end{aligned} \tag{3.25}$$

where bayes rule was used to rewrite $P(f_e^+|d)$, $P(f^+)$ is the proportionality constant and the expectation is wrt. the posterior of the diseases given those findings we treat exactly (i.e. do not transform). Reintroducing the explicit form for the transformed

conditional, we get:

$$\begin{aligned} P(f^+|\xi) &\propto \mathbb{E}\left\{\prod_{i\in a} e^{\xi_i \sum_j \theta_{ij} d_j - f^*(\xi_i)}\right\} \\ &= \prod_{i\in a} e^{-f^*(\xi_i)} \mathbb{E}\left\{\prod_{i\in a} e^{\xi_i \sum_j \theta_{ij} d_j}\right\}, \end{aligned} \tag{3.26}$$

where the factors independent of the summation has been moved outside the expectation. Taking the logarithm, this becomes:

$$\log P(f^+ \mid \xi) = C - \sum_{i\in a} f^*(\xi_i) + \log \mathbb{E}\Big\{ \prod_{i\in a} e^{\sum_j \xi_i \theta_{ij} d_j} \Big\}. \tag{3.27}$$

Here C is just the log likelihood of the positive findings. The above expression is the objective function that is to be minimised wrt. the variational parameters ξ when we seek upper bounds.

3.4 Optimisation of lower bound

The lower bound on the likelihood $P(f^+|q)$ for a set of positive findings has some value for some assignment of the variational parameters q. We seek the assignment of values to the variational parameters that maximise this lower bound. This can be considered as a maximum likelihood estimation problem, where d is a latent variable and f the observation, the probability of which is governed by q. In this context the EM algorithm can be used to maximise the likelihood $P(f^+|q)$. The EM algorithm provides sequential updates to the q parameters so that the log likelihood increases monotonically with each update.
The EM algorithm consists of performing an E step where the posterior $P(d|f^+, q^{old})$ for the old setting q^{old} of the variational parameters is evaluated. In the following M step the expectation with respect to $P(d|f^+, q^{old})$ of the complete data log likelihood $\log P(f, d|q)$ is maximised with respect to q. This will serve as the new assignment to the variational parameters:

$$q^{new} = \arg\max_q \sum_d P(d|f, q^{old}) \log P(f, d|q) \tag{3.28}$$

$$= \arg\max_q \mathbb{E}\{\log P(f, d|q)\} \tag{3.29}$$

These two steps are repeated until the q parameters converge. Considering the expectation in Equation 3.29, we have:

$$\begin{aligned}\mathbb{E}\{\log P(f,d|q)\} &= \mathbb{E}\{\log P(f|d,q)P(d)\} &(3.30)\\ &= \mathbb{E}\left\{\sum_i \log P(f_i^+|d,q) + \log P(d)\right\} &(3.31)\\ &= \sum_i \mathbb{E}\{\log P(f_i^+|d,q)\} + C, &(3.32)\end{aligned}$$

where C is a constant that is not dependent on q. There is variational parameters associated with each transformed finding and these are independent across findings. Therefore the maximisation of Equation 3.32 consist of separately maximising each term with respect to the corresponding $q_{\cdot|i}$ parameters. Using the explicit form for $P(f_i^+|d,q)$ we get:

$$\begin{aligned}\mathbb{E}\{\log P(f_i^+|d,q)\} &= f(\theta_{i0}) + \sum_j \mathbb{E}\{d_j q_{j|i} g_{ij}\}\\ &= f(\theta_{i0}) + \sum_j \sum_{d_j} P(d_j|f^+, q^{old}) d_j q_{j|i} g_{ij}.\end{aligned} \tag{3.33}$$

Recalling that $g_{ij} = f(\theta_{i0} + \frac{\theta_{ij}}{q_{j|i}}) - f(\theta_{i0})$, we have:

$$\begin{aligned}\mathbb{E}\{\log P(f_i^+|d,q)\} &= f(\theta_{i0}) + \sum_j P(d_j = 1|f^+, q^{old}) q_{j|i}[f(\theta_{i0} + \frac{\theta_{ij}}{q_{j|i}}) - f(\theta_{i0})]\\ &= f(\theta_{i0}) + \sum_j \mathbb{E}\{d_j\} q_{j|i}[f(\theta_{i0} + \frac{\theta_{ij}}{q_{j|i}}) - f(\theta_{i0})]\\ &= f(\theta_{i0}) + \sum_j y_{ij}(q_{j|i}),\end{aligned}$$

where we have introduced y_{ij} and used the fact that $d_j \in 0, 1$.

We seek to optimise the above expression with respect to $q_{\cdot|i}$ parameters. According to [5] this optimisation can be carried out by performing the following update while also normalising the q parameters:

$$\begin{aligned}q_{j|i} &\leftarrow q_{j|i} \frac{\partial y_{ij}(q_{j|i})}{\partial q_{j|i}}\\ &= q_{j|i}\mathbb{E}\{d_j\}\left[f(\theta_{i0} + \frac{\theta_{ij}}{q_{j|i}}) - f(\theta_{i0}) - f'(\theta_{i0} + \frac{\theta_{ij}}{q_{j|i}})\frac{\theta_{ij}}{q_{j|i}}\right].\end{aligned} \tag{3.34}$$

Expectation value

In each step of the iterative updates to the q parameters in Equation 3.34 we need the expectation $\mathbb{E}\{d_k\}$ of a single disease with respect to $P(d|f^+, q^{old})$. This section

derives the explicit result for this calculation that is needed in order to perform the optimisation of the lower bound.
Writing out the expectation, we have:

$$\begin{aligned}\mathbb{E}\{d_k\} &= \sum_d P(d \mid f^+, q^{old}) d_k \\ &= P(d_k = 1 \mid f^+, q^{old}) \\ &= \frac{P(d_k = 1, f^+ \mid q^{old})}{P(f^+ \mid q^{old})},\end{aligned} \tag{3.35}$$

where we again make use of the fact that disease variables are binary, i.e. $d_k \in \{0, 1\}$. We now focus on the numerator; the probability of f^+ and disease d_k:

$$\begin{aligned}P(f^+, d_k \mid q^{old}) &= \sum_{d \setminus d_k} P(f^+, d \mid q^{old}) \\ &= \sum_{d \setminus d_k} P(f^+ \mid d, q^{old}) P(d).\end{aligned} \tag{3.36}$$

In the conditional probability $P(f^+|d, q^{old})$ we can choose to treat some of the findings exactly and transform the rest, i.e. approximate these individual conditionals with a lower bound. Splitting the conditional probability $P(f^+|d, q^{old})$ into the findings f_e that are treated exactly and those that are approximated having indices $i \in a$, we have:

$$\begin{aligned}P(f^+, d_k \mid q^{old}) &= \sum_{d \setminus d_k} \left[\prod_{i \in a} P(f_i^+ \mid d, q^{old}_{\cdot|i}) \right] P(f_e^+ \mid d) P(d) \\ &= \sum_{d \setminus d_k} \left[\prod_{i \in a} P(f_i^+ \mid d, q^{old}_{\cdot|i}) \right] P(d \mid f_e^+) P(f_e^+) \\ &= P(f_e^+) \sum_{d \setminus d_k} \prod_{i \in a} e^{\sum_j q_{ij} g_{ij} d_j + f(\theta_{i0})} P(d \mid f_e^+) \\ &= P(f_e^+) \sum_{d \setminus d_k} \prod_{j} e^{\sum_{i \in a} q_{j|i} g_{ij} d_j + f(\theta_{i0})} P(d \mid f_e^+),\end{aligned} \tag{3.37}$$

where the explicit form of the transformed conditional from Equation 3.20 has been inserted. Moving factors $f(\theta_{\cdot 0})$ involving the background cause outside in a constant C_0 and then moving the factor that is not summed over outside the summation, we get:

$$
\begin{aligned}
P(f^+, d_k \mid q^{old}) &= P(f_e^+) C_0 \prod_j \sum_{d \setminus d_k} e^{\sum_{i \in a} q_{j|i} g_{ij} d_j} P(d \mid f_e^+) \\
&= P(f_e^+) C_0 e^{\sum_{i \in a} q_{k|i} g_{ik} d_k} P(d_k \mid f_e^+) \prod_{j \neq k} \sum_{d \setminus d_k} e^{\sum_{i \in a} q_{j|i} g_{ij} d_j} P(d \mid f_e^+) \\
&\propto e^{\sum_{i \in a} q_{k|i} g_{ik} d_k} P(d_k \mid f_e^+).
\end{aligned} \tag{3.38}
$$

Normalising the above expression we get the desired expectation value from Equation 3.35

$$
\mathbb{E}\{d_k\} = \frac{e^{\sum_{i \in a} q_{k|i} g_{ik} d_k} P(d_k = 1 \mid f_e^+)}{e^{\sum_{i \in a} q_{k|i} g_{ik} d_k} P(d_k = 1 \mid f_e^+) + P(d_k = 0 \mid f_e^+)}. \tag{3.39}
$$

3.4.1 Combining exact and transformed findings

Once a set of variational parameters for the findings we wish to transform have been obtained, the likelihood of the findings can be split into two factors; a factor that is the probability $P(f_e)$ of the positive findings that is treated exactly and another factor $F(f_t|q, f_e)$ (for the lower bound) that is dependent on the variational parameters (q for the lower bound and ξ for the upper bound):

$$
\begin{aligned}
P(f|q) &= \sum_d P(f|d) p(d) && (3.40) \\
&= P(f_e) \prod_j \sum_{d_j} e^{\sum_{i \in a} q_{j|i} g_{ij} + f(\theta_{i0})} P(d_j | f_e^+) && (3.41) \\
&= P(f_e) F(f_t|q, f_e) && (3.42)
\end{aligned}
$$

The above expression relates to the lower bound but a similar expression exists for the upper bound with the corresponding upper bound factor $F(f_t|\xi, f_e)$.

3.4.2 Calculating both bounds

Having described how a set of conditional probabilities can be transformed in order to achieve either an upper bound $P(f \mid \xi)$ or a lower bound $P(f \mid q)$ on the sought after probability $P(f)$, we will now describe the steps our algorithm goes through in order to calculate an interval wherein the true joint probability lies.

We describe the case where the set of transformed findings f_e are the same for both the upper and lower bound. This case is particularly efficient to calculate since many of the calculations can be reused in both bounds as we shall see.

Both the upper and lower bound can be written as the product of the probability of the exact findings and a factor that is dependent on the transformed findings f_t

for a given set of non-transformed findings and variational parameters:

$$P(f \mid \xi) = P(f_e)F(f_t \mid \xi, f_e) \tag{3.43}$$
$$P(f \mid q) = P(f_e)F(f_t \mid q, f_e). \tag{3.44}$$

This shows that the probability of the exactly treated findings can be calculated once at the outset and then multiplied with the upper bound factor and lower bound factor in order to get the respective bounds.

During the optimisation of both the ξ and q parameters, we will need to calculate posterior probabilities $P(d_j|f_e)$ given the non-transformed findings. In this step we also calculate the finding probability $P(f_e)$ and since we are transforming the same findings for the upper and lower bound, all these calculations can be reused in both contexts. Furthermore we do not calculate the posterior for each disease d_j, but only the relevant ones. The relevant diseases are those that are parent to at least one of the transformed findings. These are the steps of the algorithm:

1. Given a set of positive findings f and a number n_{exact} of these findings to treat exactly (i.e. not transform); sort the findings according to some given criteria. Take out the first n_{exact} findings (f_e) to be treated exactly and designate the remaining findings as transformed findings (f_a).

2. Calculate posterior $P(d_j|f_e)$ for $j \in \pi(f_t)$, where $\pi(f_t)$ is the set of diseases that are parent to at least one transformed finding.

3. **Minimise upper bound:** Calculate the best ξ parameter for each transformed finding by performing gradient descent to minimise the objective function from Equation 3.27.

4. **Maximise lower bound:** Calculate the best $q.$ distribution by iteratively applying the update rule from Equation 3.34 until convergence.

5. Given the variational parameters ξ and q, calculate the factors $F(f_t \mid \xi, f_e)$ and $F(f_t \mid q, f_e)$ and multiply these with the probability of the findings $P(f_e)$ in order to get the upper and lower bound respectively.

The criteria we use for selecting findings to transform follows the delta method described in [1, 5].

3.5 Interval Bounds on Posteriors

Using the equations shown so far we are able to provide both upper and lower bounds on joint probabilities. This is useful if we are looking for approximations to the likelihoods, but by themselves they cannot be used to provide bounds on posterior probabilities. Trying to do so would mean the numerator and denominator in the posterior calculation (3.9) would both be off in the same direction but by different amounts, giving no guarantees on the result.

However if we know both bounds, it is possible to provide interval bounds on the posterior probabilities. A way of doing this is presented in [5] and can be seen below:

$$\frac{P(f^+, d_j \mid q)}{P(f^+, \bar{d_j} \mid \xi) + P(f^+, d_j \mid q)} \leq P(d_j \mid f^+) \leq \frac{P(f^+, d_j \mid \xi)}{P(f^+, d_j \mid \xi) + P(f^+, \bar{d_j} \mid q)}. \quad (3.45)$$

Since we are calculating the positive posterior probability $P(d_j^+ \mid f^+)$ we can rewrite the formula as follows:

$$\frac{P(f^+, d_j^+ \mid q)}{P(f^+, d_j^- \mid \xi) + P(f^+, d_j^+ \mid q)} \leq P(d_j^+ \mid f^+) \leq \frac{P(f^+, d_j^+ \mid \xi)}{P(f^+, d_j^+ \mid \xi) + P(f^+, d_j^- \mid q)}. \quad (3.46)$$

Since $P(d_j^+ \mid f^+) = P(f^+, d_j^+) / \sum_{d_j} P(f^+, d_j)$ the above inequality is on the following form:

$$\frac{(x - \delta_l)}{(y + \delta_u) + (x - \delta_l)} \leq \frac{x}{x + y} \leq \frac{(x + \delta_u)}{(x + \delta_u) + (y - \delta_l)}, \quad (3.47)$$

where $x = P(f^+, d_j^+)$, $y = P(f^+, d_j^-)$ and δ_l and δ_u are both positive and represent the difference between the exact probability and the bounds. Note that the values are not necessarily the same across the inequalities.

To see the validity of this inequality we first consider the right side fraction, i.e. the upper bound on the posterior. Having a $\delta_l > 0$ will only decrease the denominator which in turn increases the fraction. Having a $\delta_u > 0$ will increase both the numerator and denominator but the numerator will increase by a greater factor, hence increasing the overall fraction. Similar arguments can be made to show that the lower poster bound is valid.

This provides us with a tighter bound, since we are able to normalize using the same bound for the same joint probability in both the numerator and denominator, compared to a naive approach of normalizing using the opposite bound only.

In order to calculate the joint probabilities used in Equation 3.46, we can rewrite them using the chain rule. Here it is shown for the upper bound probabilities:

$$P(f^+, d_j \mid \xi) = P(f^+, \mid d_j, \xi) P(d_j, \mid \xi) = P(f^+, \mid d_j, \xi) P(d_j) \quad (3.48)$$

where the ξ parameter disappears from the prior probability, since the prior does not depend on it.

An obvious use case for these interval posterior bounds, is to use them as a substitute for the exact probabilities to be presented to the doctor. Whenever the number of positive findings grows beyond what can be handled in real time, we can output these approximate bounds instead, which the doctor can use in the immediate assessments of the situation. If exact probabilities are still needed, then these could be provided as soon as they are available.

It could also be considered how this might be used in order to speed up the questioning method that is being used. In the simplest case we could just substitute

the posterior probabilities with either of the bounds. It might also be possible to use it in a way, such that we maximize or minimize the impact the bounds have on choosing the question. For example we might consider the entropy from either of these bounds, or some combination thereof.

Chapter 4

Experiments

In this part of the report we will be presenting results from running experiments using the various algorithms presented so far. We will present results for the caching algorithms for posteriors and sequential probabilities, approximations to likelihoods and posteriors and finally the question strategy.

For the caching algorithms we compare their processing time compared to the brute force alternatives. For the variational methods the accuracy of the approximations will be presented along with processing time for some of the runs. The question strategy will be evaluated by investigating how many questions needs to be asked before present diseases are ranked high on a list sorted by disease posteriors.

This chapter contains descriptions of the data we a working with that was provided by Ambolt. From this data we were able to generate artificial patient cases that we can run our experiments on.

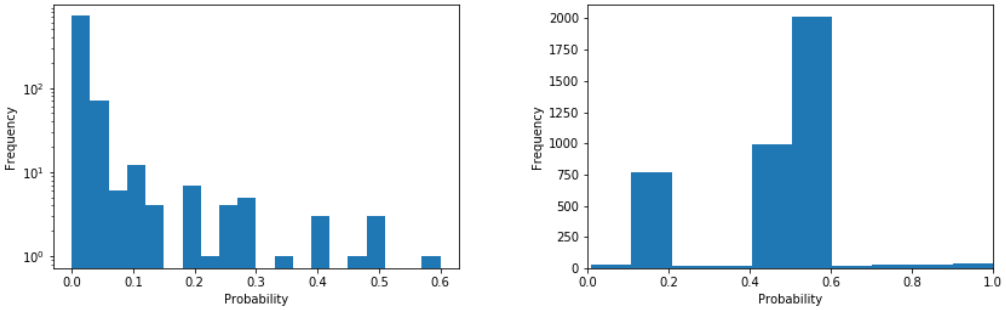
In experiments with variational methods we always have to decide which findings to transform and which findings to treat exactly. We use the delta method presented in [1] and [5] which is a heuristic for determining which findings are the best candidates for transformation. The findings that, when transformed, move the approximation result away from the true probability by the least amount, are the best ones to transform.

4.1 Data

The data used to build the $BN2O$ network, consist of 833 diseases along with their prior probabilities. About half of these have a prior probability of $< 0.01\%$ and the maximum prior has a value of 0.6 belonging to the disease 'Mavekatar' (see Figure 4.1a).

The noisy-OR parameters q_{ij} are also readily available to us from the data set. We have 3965 q parameters specifying the relationship between diseases and 440 symptoms. The q parameters are mostly centered around 50% (see Figure 4.1b).

The full data set contains detailed information such as which demographics a



(a) Histogram over the probabilities with a logscale on the y-axis.

(b) Histogram over the q_{ij} parameters needed to specify the noisy-OR gates.

Figure 4.1

certain disease prior pertains to, in which body part a symptom is experienced, which diseases can lead to other diseases and more. This type of information us not something we include for the purposes of our experiments. We extract and compile disease priors and q parameters according to a predefined demographic.

4.2 Synthetic Data

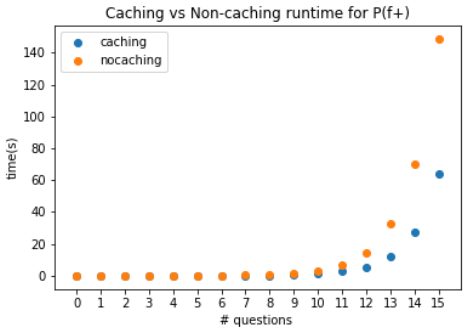
A patient case represents the diseases present in a patient along with the symptoms they are experiencing.

Specific cases can be generated by using the data encoded in the model. First we have to decide which diseases are present. This can be done by sampling each disease node according to its prior. Alternatively the presence of a set of diseases could be set manually if we wish to generate cases containing specific diseases.

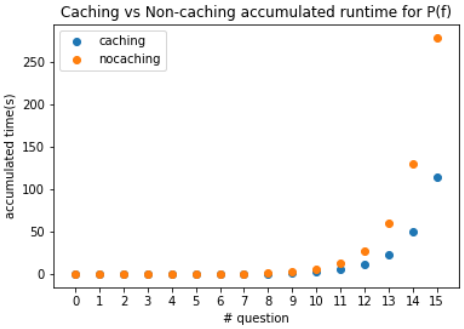
Once the disease layer has been instantiated each symptom can now similarly be assigned a 1/0 value based on the presence of the disease parents that can cause the symptom along with the noisy-OR gate assumption. The more present disease parents a symptom has, the more likely the symptom is to be present.

4.3 Caching for sequential joint finding probabilities

This section presents the results for using our caching strategy when determining probabilities of the form $P(f_m, ..., f_n)$ sequentially. We run two experiments for this setup with the purpose of examining the runtime for the caching method vs the brute force method; One where only positive findings will be added, and one where positive and negative findings are added alternatingly. The caching method uses algorithm 3 and algorithm 4 when a finding is added to the set of findings of interest, whereas the brute force method simply uses Quickscore in a straightforward manner and restarts calculations each time a finding is added.

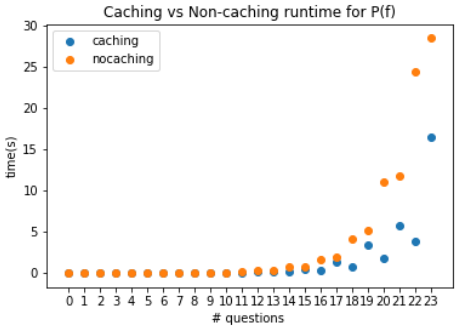


(a) Runtime for caching strategy when calculating joint positive finding probabilities sequentially.

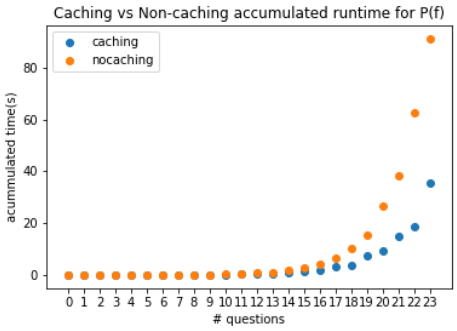


(b) The accumulated processing time from (a)

Figure 4.2



(a) Runtime for caching strategy when calculating joint finding probabilities sequentially and positive and negative findings are added alternately.



(b) The accumulated processing time from (a)

Figure 4.3

The results from the experiments focusing on the positive findings is seen in Figure 4.2a. Here a positive finding is added at each step along the x-axis. The y-axis gives the time required to add that finding. In Figure 4.2b the acummulated runtime is plotted. That is, the point at $x = 12$ gives the processing time required to calculate the probabilities up to and including question 12.

In Figure 4.3a we present the performance of the caching strategy in the setting where the questions asked are alternating between having positive and negative answers. Figure 4.3b shows the corresponding acummulated runtime.

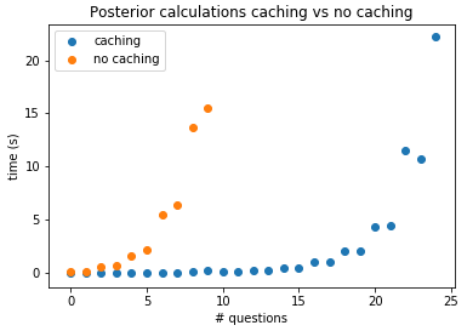
As expected the caching method shows some savings in processing time. Specifically the plots over the accumulated run times suggest that for a given number of questions, the time it takes to calculate the probability of the associated findings using caching, is less that half of that of the brute force alternative.

4.4 Caching for multiple posterior probabilities

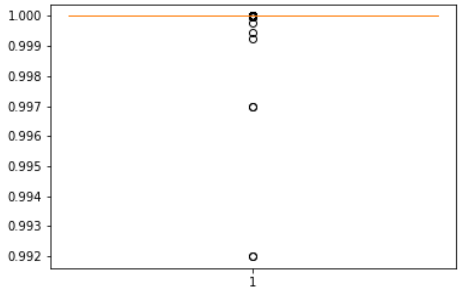
Experiments with the caching algorithm for calculating multiple posteriors $P(d_k|f)$ for $k = m, ..., n$ was evaluated on a case with 25 positive findings. We calculate the posterior for a set of diseases given a set of findings that is varied by alternately adding one of the positive findings and a random negative finding. The diseases over which we calculate the posteriors are chosen as the diseases that are a parent to at least one of the positive findings. This amounts to 422 posteriors for this specific case.
We compare two methods for calculating these posteriors. The first is through a more or less direct application of Quickscore. Since $P(d_k = 1|f) = P(d_k = 1, f)/P(f)$ two calls to Quickscore can be used to calculate quantity; first a call to determine $P(d_k = 1, f)$ and then another to determine $P(f)$. Since $P(f)$ doesn't change as d_k changes, saving this result once it has been retrieved the first time is a straightforward caching to utilise when calculating posteriors and will be used for the first base method, which we are comparing to. We could chose to compare our caching strategy to a completely brute force method, but since this first caching step is almost trivial we chose to include it, along with iteration over only relevant disease parents. We still refer to this base method as a non caching method despite of this minor level of caching. Our presented caching strategy (algorithm 1 and algorithm 2) still provide significant speedup compared to the non caching method as can be seen in Figure 4.4a. The fast caching method (blue) also absorbs negative findings since in this setting all findings are known at the outset, unlike the previous section where findings were added iteratively and therefore could not be absorbed.
In order to check that the two methods indeed produce the same results we compare the 422 posteriors generated by calculating the ratio of the caching result to the non cached result, which ideally would be one. We only compare the posteriors calculated after 10 findings has been added. In Figure 4.4b is a boxplot over these ratios showing that the vast majority is concentrated around 1 with a few outliers within one percent.

4.4.1 Joint finding probability approximations

In this section we present results for the variational methods. We show how the upper/lower bound algorithm provides a more narrow interval wherein the true likelihood $P(f)$ lies as the number of findings that are being transformed (i.e. approximated) decreases. This can be seen in Figure 4.5a where the log likelihood upper and lower bounds are shown in colours along with the log of the true probability, obtained via Quickscore, drawn as a line. This is an evaluation on a case with 15 positive findings. The lower bound proves to be much more crude compared to the upper bound as the number of transformed findings increases, whereas the upper bound gives much more useful results even when a large portion of the findings are being transformed. In Figure 4.5b we show the processing time for calculating the bounds in Figure 4.5a for each number of transformed findings. The processing time for the exact calculation was 48.9 seconds.



(a) Runtime for posterior caching algorithms(blue) presented in algorithm 1 and algorithm 2 vs non caching alternative (orange). The number of findings under consideration varies along the horizontal axis, and at each step the posteriors for 422 diseases are calculated.



(b) Boxplot of ratios between the posteriors $P(d_k|f)$ calculated with the caching method and the non caching method after 10 findings has been added. The vast majority of these ratios are concentrated around 1 indicating that the two methods produce the same probabilities with only a few outliers where the caching method produced a slightly smaller result.

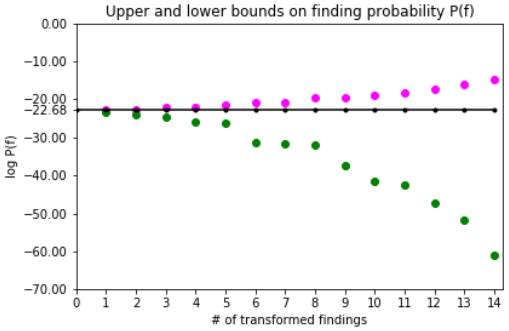
Figure 4.4

The overhead associated with transforming only a small number of the findings while treating the rest exactly (i.e. not approximating them) is too great to render the method useful. However this processing time decreases rapidly as the number of findings being transformed increases, bottoming out at 8 transformed findings. The fact that the processing time is not monotonic shows that we can sometimes achieve a better approximation (a more narrow bound) in less time. In this case, for instance, a better and faster bound is achieved by transforming 8 findings than by transforming 9 or more findings.

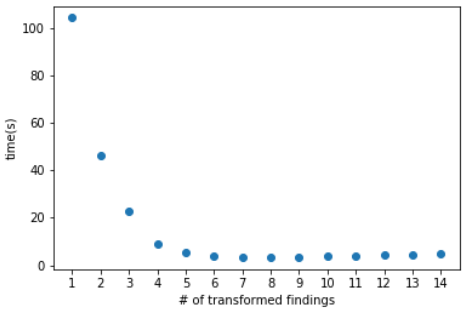
4.5 Bounds on posteriors

Having presented the bounds on the likelihood of findings, we now turn to task of calculating bounds on the posteriors $P(d_j|f)$. We present three such cases in Figure 4.6. Here we see how the bounds become more narrow as the number of transformed findings decreases. We evaluate the posterior for a selected disease in three different cases with 11, 12 and 17 positive findings respectively. All three plots show that the crudeness of the bound increases rapidly as more and more findings are approximated. It should be noted how large an effect changing the number of findings to transform can have on these bounds. All plots show jumps in the lower bound by factors on the scale of e^{10}; e.g. Figure 4.6b when going from 7 to 6 transformed findings.

We now try to fix the number of findings that we transform and give posteriors for multiple diseases for a specific case. In Figure 4.7a we present a histogram over the interval sizes, i.e. the upper bound minus the lower bound, for relevant posteriors. This case has 14 positive findings and 5 of these were transformed. The histogram

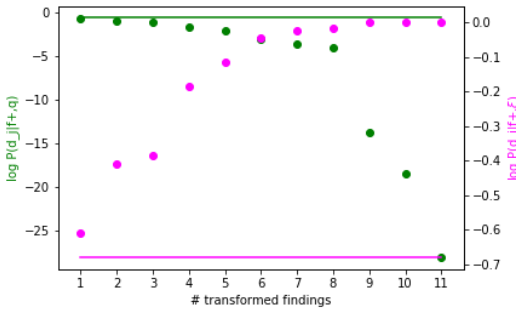


(a) Bounds on the joint probability $P(f)$ with the number of transformed findings varying along the x-axis. This is for a case with 15 positive findings.

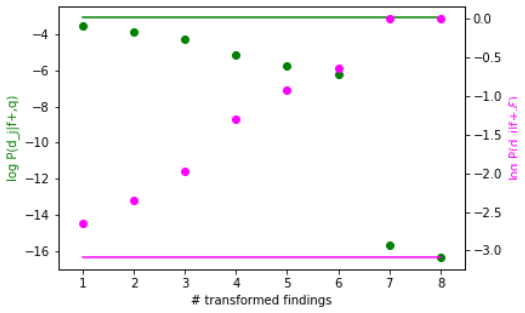


(b) The runtime for the corresponding number of transformed findings

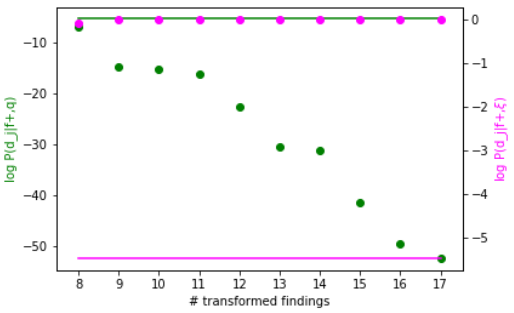
Figure 4.5



(a) Case with 11 positive findings

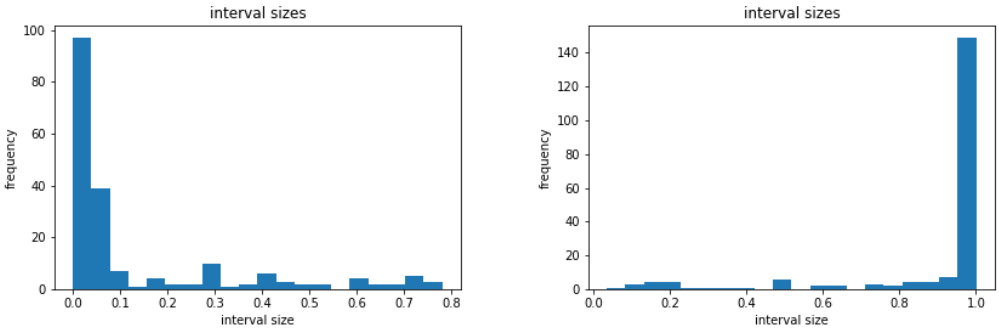


(b) Case with 12 positive findings



(c) Case with 17 positive findings

Figure 4.6: Bounds (magenta and green points) on the posterior $P(d_j \mid f^+)$ (solid) for different selected diseases with a varying number of findings being transformed. Figure 4.6a is a case with 11 positive findings, Figure 4.6b has 12 positive findings and Figure 4.6c 17 positive findings.



(a) The interval size on posterior bounds for 195 relevant diseases for a case with 14 positive findings, where 5 of these were transformed.

(b) The interval size on posterior bounds for 195 relevant diseases for a case with 14 positive findings, where 7 were transformed.

Figure 4.7: Caption

shows that the majority of the bounds sizes are concentrated at the lower end of the scale, meaning that many of the provided bounds are narrow. A few bounds prove to be quite useless with sizes of one half or more.

In Figure 4.7b we present the same plot only now 7 findings are transformed. This also shows how sensitive the posteriors bounds are to the number of findings being transformed. Transforming only 2 more findings renders almost all of the bounds useless.

4.6 Question strategies

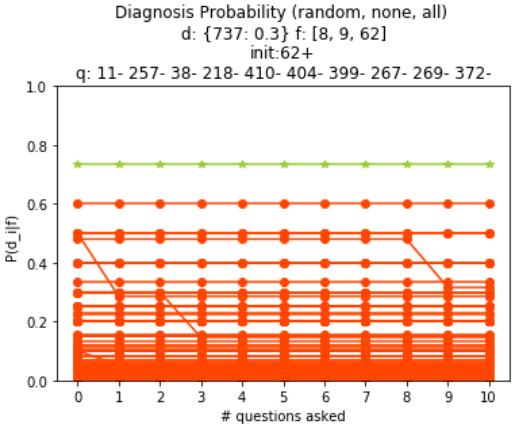
In this section we run experiments on different cases in order to see the effect of the different questioning methods. We also compare the performance over many different cases for the random and entropy based methods on relevant findings.

The text above each plot includes the settings and case it has been run with. The present diseases and their priors are denoted by 'd:', the present findings by 'f:', the initial findings by 'init:' and the questions being asked by 'q:'. The sign after initial findings and questions correspond to whether said finding was present and absent. Some of the plots will show the ranking of the diseases which is obtained by sorting the posteriors from high to low. This means that a good question strategy will ask questions that quickly lower the rank of the correct diseases.

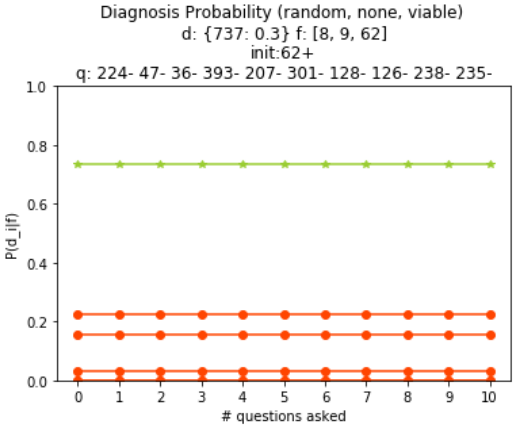
4.6.1 Case 1

The first case we consider has a disease with a prior probability of 0.3, and we initially know about 1 of the 3 findings. This in itself brings the posterior probability close to around 0.75, ranking it 1st out of all disease posteriors.

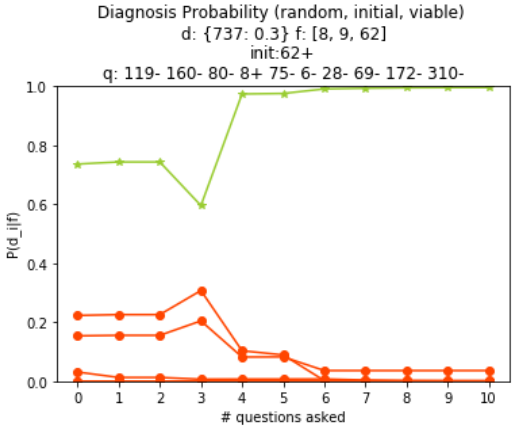
This can be seen in Figure 4.8a where the green line represents the posterior over the present disease. For this run we are asking completely random questions, which



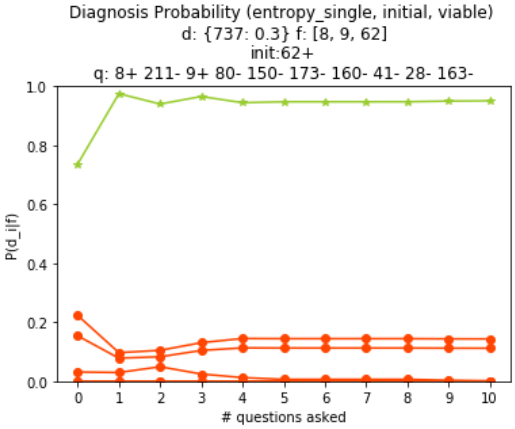
(a) Run with random question choices, without using relevancy to limit the question pool, and displaying all disease posteriors.



(b) Similar to (a), but only showing diseases related to the initial findings. It is a separate run on the same case.



(c) Run where questions were selected only among the finding that were relevant given the initial finding.



(d) Run using the entropy method on the findings that were relevant given the initial finding.

Figure 4.8: Case 1

are not improving the probability of the present disease noticeably.

By simply switching to displaying only diseases which are related to the initial findings as shown in Figure 4.8b, we are able to isolate the correct disease posterior, increasing our relative confidence in the diagnosis.

In Figure 4.8c we change which findings we consider for questioning, to only ask about those that are relevant given the initial findings. That is; pick out the disease parents of the known present symptoms and consider only the child findings of these diseases. This causes the diagnosis to improve significantly, with increased posterior probability of the correct disease, and reduced posterior probability of all other diseases. Although this is with the exception of the 3rd question, after which the diagnosis became less certain until the next question.

In Figure 4.8d we apply the entropy based method. This starts by choosing another positive finding, which is able to both increase the posterior probability of the present disease, as well as reduce the probability of the other relevant diseases that are not present, both of which are desirable outcomes of a questioning strategy. But after this there is no further significant improvement, although it does bring the posterior of the least likely disease close to 0.0.

4.6.2 Case 2

For this case we start with asking random questions among the relevant findings. This can be seen in Figure 4.9a. There is no guarantee that it will ask a good question within a reasonable number of questions, so it might be the case that the graph would have showed no improvement at all in the ranking of the present disease. It could also have happened that the first few question causes the present disease to become the most likely diagnosis.

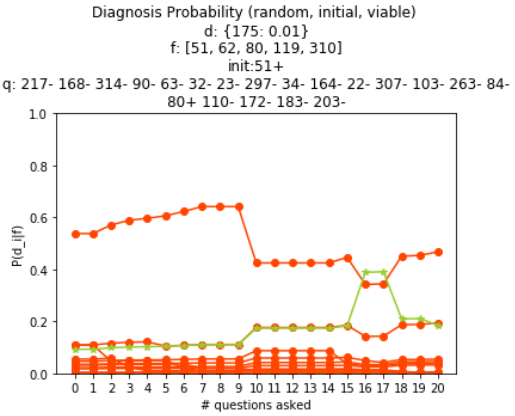
In Figure 4.9b we apply the entropy based method to this case. We see a slow improvement in the probability of the present disease, but it occasionally gets worse again. We also see a reduction in the posterior probability of the most likely, but incorrect, disease.

Looking at the plot for the disease ranking in Figure 4.9c We see that the present disease is occasionally ranked number 1, but does not stay that way.

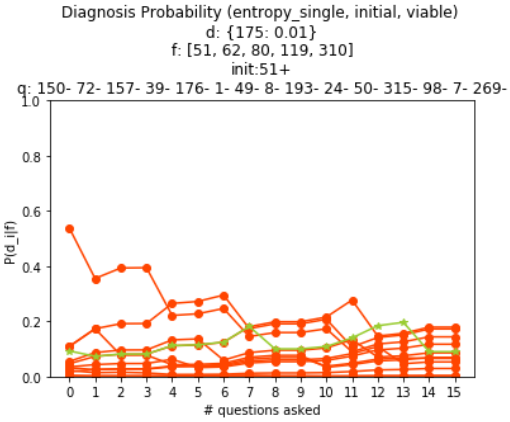
4.6.3 Case 3

The plots in Figure 4.10a and Figure 4.10b both chose random questions among the pool of relevant findings. Although the first one got lucky with the questions it asked, the second one does not manage to increase the posterior probability of the present disease much at all. In these cases the present disease achieves decent rankings (1 and 4), but in Figure 4.10b our posterior says the disease is probably not present, and in Figure 4.10a we still think another disease might also be present with a probability of about 0.6.

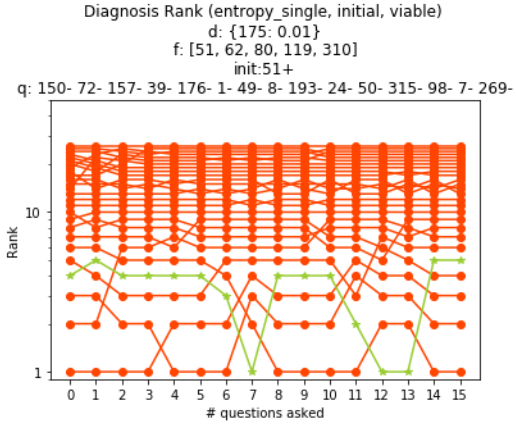
Figure 4.10c shows the entropy based method applied to the case. For the first 6 question there is an improvement in the probability of the correct disease, but after



(a) Random questioning with relevant findings.

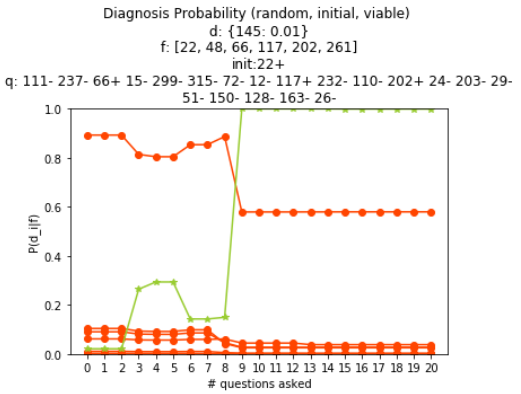


(b) Entropy method used for questioning, also among relevant findings.

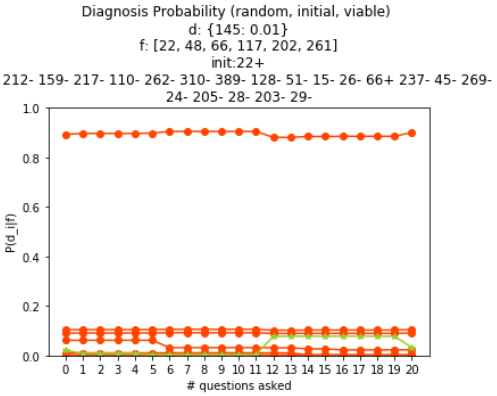


(c) Ranking of the diseases from the entropy based run in (b). Lower rank is better.

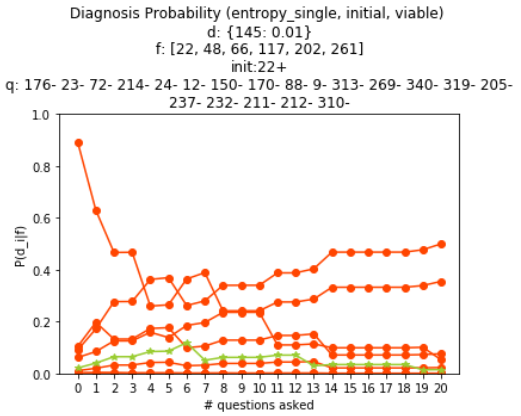
Figure 4.9: Case 2



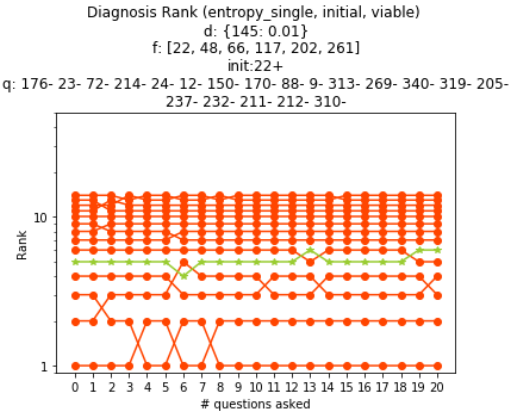
(a) Random questioning with relevant findings.



(b) A different random questioning run.

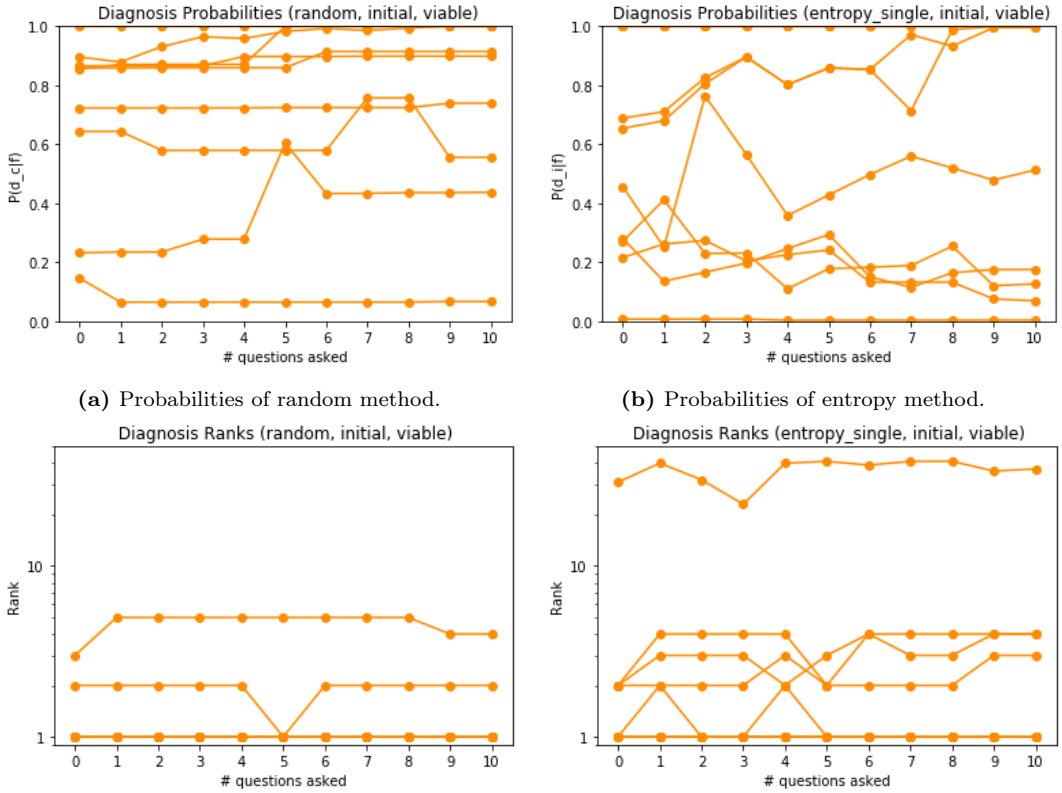


(c) Entropy based questioning with relevant findings.



(d) The ranking of diseases under entropy based questioning.

Figure 4.10: Case 3



(a) Probabilities of random method.

(b) Probabilities of entropy method.

(c) Rankings of random method.

(d) Rankings of entropy method.

Figure 4.11: Comparing the random and entropy based methods based on 10 cases each. The development in both the posterior probabilities and the corresponding rankings are shown.

that it drops off again. The posterior of the most likely disease does manage to get reduced despite causing an increase in entropy by becoming less certain. At the same time we see an increase in posterior probability of two other diseases that are not present.

Any increase in the posterior probability of the correct disease does not manifest as an improvement in the ranking as can be seen in Figure 4.10d.

4.6.4 Aggregate cases

Here we are comparing the performance of the random method in Figure 4.11a to the entropy based method in Figure 4.11b. We run 10 cases using each method, asking 10 questions using the specified questioning method.

The random method seems to have mostly straight lines, corresponding to unchanged probabilities although with some increasing probabilities near the top along with a few random spread out increases or decreases. The entropy based method on the other hand seems to have probabilities trending away from 0.5, which could be expected in general due to that decreasing the entropy of the posteriors. This is great

for posteriors that are already above, while it is not so good if they are below. In that case it might be suspected that it tries to avoid asking questions that are likely to cause those posteriors to rise, since that would mean an increase in entropy as they get near 0.5.

The ranks in these cases already seem pretty good, being mostly in top 5, with a single outlier in Figure 4.11d that could be attributed to randomness.

Chapter 5

Conclusion

The presented caching strategies demonstrated how major savings in processing time can be achieved when using Quickscore in certain settings. Regarding the sequential finding probabilities, the processing time for adding another finding was more or less halved. As for the multiple posterior calculations our experiment suggest that for some cases the caching method offers run time savings on the scale of at least one order of magnitude.

The effect of the entropy based questioning method is to reduce the entropy for the individual disease posterior probabilities being considered. This seems to work great if the posterior probability of the present disease is already close to or above 0.5, as any increase in this probability is then also a reduction in entropy. If the posterior is below 0.5 then this doesn't seem to be the optimal behaviour. For example if we look back at Figure 4.9b, the present disease starts at a low posterior probability of around 0.1 and as such has to get over the "hump" that is a probability around 0.5, corresponding to the highest entropy. If we look at the disease that starts around 0.6, we see that we do actually manage to move it away from 0.5, albeit having to move past said probability in a single step.

Based on the experiments we can see that a completely random questioning method does not work, as we do not learn anything new about the diseases that might potentially be present. But by simply limiting the questions to a pool of relevant questions we are able to improve the performance of random questioning on small cases. But as the number of relevant findings grows, the likelihood of asking a good question drops.

The entropy based method performs well in the first case, managing to increase our already confident belief in the correct diagnosis. However it runs into trouble with the other cases. As the number of relevant findings and associated diseases grows, the time it takes to calculate the entropy increases as well. It also experienced difficulties with increasing the probability of a disease, if this does not already start with a high posterior probability after the initial findings. When evaluating the entropy based questioning strategy on cases with multiple diseases results were not promising, as the belief in the present diseases remained unchanged and low ranked as the questioning

progressed.

We have demonstrated that it is possible to provide bounds on both joint and posterior probabilities using the variational algorithms. These bounds could prove useful both for determining which questions to ask, and for providing bounded posterior probabilities to a clinician doing diagnosis.

However these approximations are very sensitive to which findings are transformed. Figure 4.7a and Figure 4.7b showed that even small changes in the number of transformed findings has a major impact on the usefulness of the provided bounds. This also suggest that different findings contribute differently to the crudeness of the approximations - likewise this suggest that certain key findings are important to treat exactly in any given set of positive findings. If we allow the number of exactly treated findings to become too small, there is no longer room for all of these key findings to be treated exactly. This might be the explanation for the dramatic jumps away from the exact result, that could be seen in the evaluation of the approximate methods.

The variational methods are especially relevant when the number of positive findings that needs to be considered is high. The approximations provided by the variational methods could be used when calculating the expected value to determine the next question to ask, in order to reduce calculation time. Testing the sensitivity of the order of suggested questions to approximations in the underlying calculations is a candidate for future work.

Bibliography

[1] J. Aggerholm and C. Mathisen. "Inference Methods for Realtime Medical Diagnosis in BN2O Networks". In: (Jan. 2020), p. 37.

[2] Eta S Berner. *Clinical decision support systems*. Vol. 233. Springer, 2007.

[3] Feili Yu et al. "Multiple disease (fault) diagnosis with applications to the QMR-DT problem". In: *SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme - System Security and Assurance (Cat. No.03CH37483)*. Vol. 2. 2003, 1187–1192 vol.2.

[4] David Heckerman. "A Tractable Inference Algorithm for Diagnosing Multiple Diseases". In: *CoRR* abs/1304.1511 (2013). arXiv: 1304.1511. URL: http://arxiv.org/abs/1304.1511.

[5] T. S. Jaakkola and M. I. Jordan. "Variational Probabilistic Inference and the QMR-DT Network". In: *Journal of Artificial Intelligence Research* 10 (May 1999), pp. 291–322. ISSN: 1076-9757. DOI: 10.1613/jair.583. URL: http://dx.doi.org/10.1613/jair.583.

[6] Jianhui Luo et al. "Graphical models for diagnosis knowledge representation and inference". In: *IEEE Autotestcon, 2005.* IEEE. 2005, pp. 483–489.

[7] Blackford Middleton et al. "Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base. II. Evaluation of diagnostic performance". In: *Methods of information in medicine* 30 (Nov. 1991), pp. 256–67. DOI: 10.1055/s-0038-1634847.

[8] Mark A Musen, Blackford Middleton, and Robert A Greenes. "Clinical decision-support systems". In: *Biomedical informatics*. Springer, 2014, pp. 643–674.

[9] Thomas Dyhre Nielsen and Finn Verner Jensen. *Bayesian networks and decision graphs*. Springer, 2007.

[10] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1988. ISBN: 1558604790.

[11] Murali Sambasivan et al. "Intention to adopt clinical decision support systems in a developing country: effect of physician's perceived professional autonomy, involvement and belief: a cross-sectional study". In: *BMC medical informatics and decision making* 12.1 (2012), p. 142.

[12] Michael Shwe and Gregory Cooper. "An empirical analysis of likelihood-weighting simulation on a large, multiply connected medical belief network". In: *Computers and Biomedical Research* 24.5 (1991), pp. 453–475.

[13] Jiří Vomlel and Petr Tichavský. "Probabilistic inference with noisy-threshold models based on a CP tensor decomposition". In: *International Journal of Approximate Reasoning* 55.4 (2014). Special issue on the sixth European Workshop on Probabilistic Graphical Models, pp. 1072–1092. ISSN: 0888-613X. DOI: https://doi.org/10.1016/j.ijar.2013.12.002. URL: http://www.sciencedirect.com/science/article/pii/S0888613X13002910.

[14] F. Yu et al. "A Lagrangian Relaxation Algorithm for Finding the MAP Configuration in QMR-DT". In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 37.5 (2007), pp. 746–757.