File: c:/ddc/Angel/Papers/Diagnosis.docx

Date: 2024 March

Medical Diagnosis Algorithm

Dennis de Champeaux

Unaffiliated/ Silicon Valley

ddc2@ontooo.com

Abstract

We describe a medical diagnosis algorithm that has been implemented and is available in a website. A patient simulator shows that all build-in diseases are correctly identified. The algorithm is generic in the sense it can be applied where the equivalent of diseases and symptoms can be identified. We discuss several settings in which such a system can be deployed: self-help, physician monitored, call center monitored, etc. The eRecord maintained by the algorithm can be used for additional symbolic reasoning.

Keywords: medical diagnosis algorithm.
Declarations of interest: none
Funding: This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

1 Introduction

Diagnosis is a form of abductive reasoning:

> It starts with an observation or set of observations then seeks to find the simplest and most likely explanation for the observations.

AI was early on interested in medical diagnosis as witnessed by MYCIN in the seventies [Mycin]. Medical practice has changed since then substantially. Early recognition of the onset of breast cancer, prostate cancer, lung cancer, colon cancer, diabetics, heart attacks, too high cholesterol etc. has become routine with periodic checkups using blood testing and numerous non-invasive imaging modalities. Symptom driven diagnosis is still being used, however, often complemented subsequently and validated with a test. Companies have seized opportunities. IBM [IBM] claims;

> AI can quickly and more accurately spot signs of disease in medical images such as MRIs, CT scans, ultrasounds and x-rays. Patients can be diagnosed faster and can begin treatment sooner.

Arterys [Arterys] claims:

> The first systematic review and meta-analysis of its kind finds that artificial intelligence (AI) is just as good at diagnosing a disease based on a medical image as healthcare professionals.

While there have been many articles that emphasize the role that AI can play in the medical arena, one can encounter a more careful position [Datarevenue]:

> AI won't replace doctors anytime soon. It's unlikely that AI will replace doctors outright. Instead, AI systems will be used to highlight potentially malignant lesions or dangerous cardiac patterns for the expert allowing the doctor to focus on the interpretation of those signals.

This paper describes an algorithm for symptom driven diagnosis in an interactive setting. After several rounds of confirming or denying conjectured symptoms the system converges on a differential diagnosis: an ordered sequence of relevant diseases.

Given the developments in the previous decades and the substantial variability of medical practices around the World we discuss also the different ways that an implementation of this algorithm can be used by different type of users in different type of settings.

MYCIN was implemented as an expert system, which had maintenance and scalability issues. These days a medical system should be at least:
-- Scalable: adding another disease, symptom, body location, etc. should just be an incremental effort
-- Maintainable: updating tools should be available instead of having to tinker with messy code
-- Transparency: the medical knowledge that is build-in should be inspectable
-- Testable: the algorithm that generates a differential diagnosis needs validation by which each build-in disease is correctly identified
The HealthCheck (HC) system that contains the algorithm described has these features, [HC].

To set the stage, we start with the ontology of the build-in medical knowledge. The actual knowledge base has a key feature regarding the frequency distribution of how often a symptom is used in the characterization of a disease. We proceed with the description of the iterative process of the diagnosis algorithm that starts with the user's initial input and generates in subsequent iterations pertinent symptoms to be decided. The system produces a differential diagnosis - a list of candidate diseases rank ordered by probability - when the probabilities pass a threshold. We discuss how the algorithm is validated.

2 Medical Ontology

The ontology of the HC system has, among others, the following core concepts: *Location*, *Body System*, *Abnormal Condition*, *Symptom* and *Disease*. We give details of only some of them (and their numerous subclasses) to focus on the role they play in diagnosis.

Characterizing the last three was done with ad hoc definitions:

-- Abnormal condition: an atypical state of the body that is usually not directly observable, which can be a (contributing) cause of a disease or a consequence of a disease

-- Symptom: an atypical state of the body that is usually directly observable, which is a defining/characterizing aspect of one or more diseases

-- Disease: an atypical, detrimental state of the body defined (among others) by a set of symptoms and optionally with associated abnormal conditions

These three concepts are represented by classes that inherit from the class *Condition* and thereby obtain, among others, the attributes *Name* and *Location*. The *Symptom* class adds the attributes *Minimum age*, *Maximum age*, the *Status condition* and the set of diseases in which the symptom

occurs. The *Status condition* captures for women the different state they can be in from pre-menstruation up to post-menopause.

The *Disease* class has many, many attributes among which: *Life threatening*, *Gender specific*, *Race specific*, *Minimum age*, *Maximum age*, the *Status condition*, and the set of symptoms that characterize a disease.

The HC system contains:

- 236 locations
- 102 body systems
- 498 diseases (and 94 'super-class' diseases)
- 83 abnormal conditions
- 928 symptoms
- 153 external causes
- 244 medication/drugs

3 Disease-symptom distribution

Symptoms differ how often they occur in the characterization of diseases. Virtually half (461) of all symptoms (928) are used only once in the description of a disease. At the other extreme, there is one symptom, nausea, which shows up in 85 diseases. The graph in figure 1 shows (*x*,*y*) pairs with the following interpretation: a symptom belongs to the set of size *x* if it used in the description of *y* diseases.

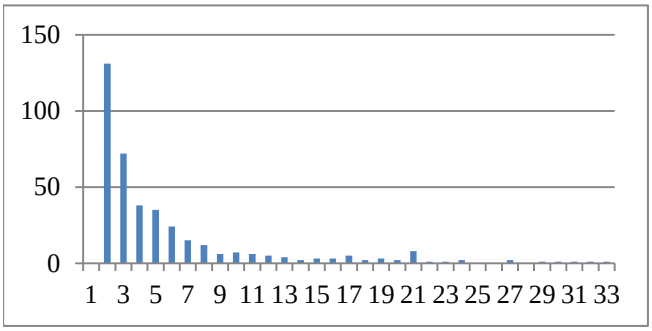

Figure 1. Showing the number of symptoms (*y*) that occur in x diseases. The value for x = 1 is omitted as well as the entries for 33 < *x*.

The graph omits at the left side that there are 461 symptoms that are used only in one disease. The first bar at the left side represent that there are 131 symptoms that are used twice. X-values larger than 33 are omitted because there are only 16 symptoms remaining and hence most of the buckets above 33 are empty. These 16 symptoms are quite important because they are used the most to characterize and identify diseases.

The disease-symptom frequency distribution illustrates that symptoms can be used for two exclusive tactics during the diagnosis process:

- Querying a symptom first that is used in many diseases separates maximally the diseases that are potentially (non) relevant
- Querying a symptom later that is used in a few diseases helps to narrow down the most probable disease.

Hence the key idea is NOT to hunt for the best explanation at first; instead the aim is to rule out diseases that do not satisfy the facts until a unique candidate has become most probable; most probable only because a disease may not always manifest itself as its descriptions prescribe and/or a symptom may not be recognized.

4 HC diagnosis algorithm

Symptoms can be caused by the side-effects of medication. Thus it is checked first whether the user's symptoms are just the consequence of medication. This is done when the user's eRecord has an entry for an earlier prescription.

The diagnosis algorithm proceeds by filtering out those disease that are gender specific and do not apply to the user; similarly diseases having an unmet age restriction can be filtered out. Candidate symptoms are screened also and filtered out similarly.

It would be nice if an instance of the class Disease could be class-like so that we could create an instance of it to represent a specific manifestation in a particular person on a specific moment. Instead, we use a class *DiseaseInst* with a particular disease as one as its attributes and other attributes regarding the actual manifestations of the disease. The instances of the class *DiseaseInst* are used usually below when we use 'disease'.

The user is asked – after filtering – whether a specific body location is problematic (using a male/female body image) and asked to provide one or more symptoms (using selections from pull down lists or by typing). After obtaining the primary concern, the user is queried iteratively several rounds by presenting symptoms to be confirmed or denied (by not confirming them). In case the user has specified a specific body location then candidate symptoms can be made available because locations and symptoms are cross referenced.

The following items, among others, are available after a next round of querying the user:
- selected body location, if any
- previous confirmed symptoms
- previous not confirmed symptoms
- considered diseases
- relevant test symptoms not yet presented
- new confirmed symptoms
- new not confirmed symptoms

Hence after the first round of querying the user we have only:

- selected body location: a specific location or no specific location
- previous confirmed symptoms: null
- previous not confirmed symptoms: null
- considered diseases: null
- relevant test symptoms not yet presented: null unless a specific body locations was selected
- new confirmed symptoms: one or more symptoms
- new not confirmed symptoms: null

Ignoring for now termination and other refinements, the symptoms to be presented to the user for a next round of querying are produced as follows:

-- the previous confirmed symptoms are updated with the new confirmed symptoms and similarly for the not confirmed symptoms
-- the collection of considered disease is extended by those diseases that are referred to by the new confirmed symptoms
-- each considered disease is re-ranked with respect to the new confirmed and new not confirmed symptoms; a boosted disease is one whose ranking improved due to the re-ranking
-- the collection of relevant test symptoms not yet presented is extended with those not yet encountered symptoms in boosted diseases and in the diseases that have been added to the collection of considered diseases
-- each symptom in the collection thus obtained is given a relevance number based on whether it has been obtained from a boosted disease and, among others, also from how often it occurs in the collection of considered diseases
-- at most 20 of the top ranking symptoms are taken from the collection of relevant test symptoms and shown to the user for a next round of querying

When there is a disease that has at least 50% of its required symptoms confirmed a differential diagnosis is presented to the user before more symptoms to be decided are presented. Such a page has a table with conjectured diseases, a probability estimate and an entry that can be selected to accept the conjectured disease. Accepting a disease causes the disease to be added to the user's eRecord and the diagnosis session will terminate. The conjectured diseases can be explored by the user to see their details, among which treatment options. The user can also elect to continue the diagnosis session.

The rank order of the diseases in the differential diagnosis is primarily determined by the ratio of confirmed symptoms. Life threatening diseases are displayed in red. [Disease diagnosis incidence frequency (gender specific, age specific, race specific, season specific) could be an additional way to rank them, but that data is typically not available.]

5 Validation through a patient simulator

The diagnosis algorithm has been validated with a simulator of a hypothetical user that selects iteratively the expected symptoms for each checked disease. All 498 diseases are correctly identified.

The simulator has trouble with a disease whose symptoms is a subset of another disease. This happens usually when an early phase of a disease has fewer symptoms than a version of an advanced phase; for example acute prostatitis and chronic prostatitis. Checking using a diagnosis setting in which only the shared symptoms are provided shows that the correct differential diagnosis is obtained also for this type of 'masked' diseases.

The simulator identifies two diseases with no symptoms: early aids and a dormant syphylis variant.

The average number of iterations to reach the correct differential diagnosis is 3.6. A user who reaches an iteration level over six receives the suggestion to try another search; backtracking is supported.

Analyzing the relationship between symptoms and diseases revealed 57 symptoms not occurring in any disease description; they are side-effects of medication. The analyzer found also 24 diseases having only one symptom while that symptom does not show up in any other disease.

6 Advanced diagnosis options

A user is offered initially the choice of an advanced diagnosis – likely taken by an expert user. Options are: selecting a particular body subsystem with its associated diseases, or selecting one or more disease families. These selections will prime the collection of considered diseases.

7 Usage modes of the HC diagnosis algorithm

The HealthCheck system that contains the described diagnosis algorithm has different modes of use:

- An anonymous web user without creating an account or alternatively by creating a private account for repeated use, which will initialize an eRecord
- A web user that interacts with the system using an account set up by a physician so that the physician gets notified of a new session
- A web user, using a physician created account, that is monitored by the system in real time so that experts in a call-center can be notified of worrisome conditions and intervene as necessary
- A user interacting with a prototype version using speech technology

A non-English, Dutch web version was easy to create because the presentation language is separated from the conceptual 'deep' level.

These modes have been of academic value thus far. Over 1800 anonymous users have accessed the HC system. Once they realize that they have to provide personal data – mainly age & gender, while there is the guarantee that ip-addresses are not collected – most people do not proceed using the system.

The use of a physician created account that would support remote diagnosis is a stretch. The ability to see a patient and the one-on-one interaction plays a major role in the traditional diagnosis process. The reimbursement for a non-face-to-face consult may not be established or has a reduced rate, which is a major inhibitor. The transparency provided by the eRecord of a HC type system is also something that physicians would have to become comfortable with.

The implemented ability to monitor a user session in a call-center is a small technical miracle but the liabilities in our litigious society makes it likely for now off-limits.

These considerations explain why companies avoid generic, symptoms based diagnosis and focus on niches like difficult cases, the interpretation of medical images, finding and using (other) markers in blood samples, etc.

Yet another candidate way of deployment could be a physician using the expert mode to validate fast an 'obvious' diagnosis. Also, this mode could provide access to best practice/alternative treatments assuming that the disease ICD taxonomies [ICD] would have these available and would always be upgraded – a major unrealized assumption. This workflow would update an HC eRecord (that support machine reasoning), which could in turn feed a meta medical database and provide upstream data to a billing system. Interacting with a patient at the same time with a HC-like system is, however, at this point not obvious. Overhearing a dialog and extracting the symptoms could be a way.

Still we conjecture that the HC-like system can be deployed as-is in settings where the physician/population ratio is low or zero. The user would in such a case be a trained expert acting as a mediator with a patient. An associated remote call center could than instruct the trained expert who would assist a patient in turn. This is actually proposed this year by [Olugboja].

The HC system has a feature that we consider noteworthy. Maintaining an eRecord for a patient has numerous benefits ranging from here-and-now patient care to longitudinal trends and potentially supporting medical research. An eRecord must certainly be human readable. The records in the HC system are also 'machine' readable: i.e. the representation uses the conceptual language of the ontology. (The two-way translators support this functionality.) This design supports an experimental AI agent 'to look over the shoulder' of a logged in user while having access to the user's eRecord. This permits to generate immediate warnings supported by what is available in the eRecord.

8 Reviewer and other questions

Reviewer questions:

- How does the algorithm handle complex cases or rare diseases?
    - The system assumes a single disease and provides an ordered list of candidate diseases. Diagnosis incidence frequencies (by age, gender, race, season, etc.) are not available – an omission in current practices.
    - A customized version for rare diseases would indeed be an opportunity.
- How about adaptability and robustness? How does the algorithm handle updates or changes to medical knowledge and practices over time?
    - Adding a disease, an attribute, an abnormal condition, body location, medication/ drug, etc. is 'trivial' from the perspective of the architecture. A recompilation integrates the new material throughout. This is a *key* advantage over earlier systems. See below how the maintenance should be done.
    - The system is already running for years without crashing ever.
- How about incorporating machine learning techniques for improved diagnosis?
    - This system has as 'backbone' an ontology developed using OO principles. Current chat-systems report that they *don't* have an ontology. It is unclear – to this author – how machine learning can provide a better system.
- Are there plans to expand the algorithm's capabilities beyond diagnosis, such as treatment recommendations or prognosis prediction?
    - Treatments are already incorporated. Upgrading/ maintaining them require indeed monitoring new developments.
    - Prognosis prediction is what patients want, but is often avoided by physicians – according to personal experiences.

Other question:

- What about deployment?
    - The system was developed during 2000/5 and has been available for anonymous users in the self-help mode. It provides more info than a google session but requires more input from the user. A user who is concerned, has pain, an infectious condition, etc. will go to urgent care or schedules an appointment instead.
    - A non-profit organization could use the system and extend it with more ICD10 encodings (also for obscure diseases) and maintain best treatments. This would be a go-to source for physicians also. A for-profit organization is conceivable as well with this value proposition.
    - Telemedicine options need to be explored further.

9 Conclusion and beyond

The diagnosis algorithm described is, in a sense, abstract. It does not depend on the 'real meaning' of what diseases and symptoms stand for; the feature that symptoms and diseases are cross referenced is the key. The skewed symptom-disease frequency distribution in the medical domain helps to narrow down reaching a differential diagnosis.

We outlined several different modes in which the HealthCheck system can be used: self-help, patient session results forwarded to the patient's physician and a patient's session monitored in real-time with a call-center.

The system generates formal eRecords that are also readable by machine. These can be used for reasoning to assist the treatments of a patient as well as for research and statistical purposes.

Regions where the physician/population ratio is low are candidates where the system backed up with a call-center – as described – appears feasible, see [Olugboja]. People in isolated settings are also candidate customers.

System source

The system can run on Windows and Unix machines using a Tomcat webserver. The code is available at: https://github.com/ddccc/HealthCheck

References

[Arterys] https://www.arterys.com/

[Datarevenue] Schmitt, M., Arificial Intelligence in Medicine.
https://www.datarevenue.com/en-blog/artificial-intelligence-in-medicine

[HC] http://www.healthcheck4me.info/

[IBM] https://www.ibm.com/watson-health/learn/artificial-intelligence-medicine

[ICD] WHO releases new International Classification of Diseases (ICD 10)
https://www.who.int/news-room/detail/18-06-2018-who-releases-new-international-classification-of-diseases-(icd-11)

[Mycin] Buchanan, B.G., Shortliffe, E.H. (1984). Rule Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project. Reading, MA: Addison-Wesley. ISBN 978-0-201-10172-0.
https://en.wikipedia.org/wiki/Mycin

[Olugboja] Olugboja, A. & E.A. Agbakwuru, 'Bridging Healthcare Disparities in Rural Areas of Developing Countries: Leveraging Artificial Intelligence for Equitable Access', in https://ieeexplore.ieee.org/xpl/conhome/10467177/proceeding