Patient symptoms elicitation process for breast cancer medical expert systems: A semantic web and natural language parsing approach

O.N. Oyelade  
Department of Computer Science, Faculty of Physical Science, Ahmadu Bello University, Zaria, Nigeria, olaide_oyelade@yahoo.com

A.A. Obiniyi  
Department of Computer Science, Faculty of Physical Science, Ahmadu Bello University, Zaria, Nigeria, aaobiniiyi@yahoo.com

S.B. Junaidu  
Department of Computer Science, Faculty of Physical Science, Ahmadu Bello University, Zaria, Nigeria, abuyusra@gmail.com

S.A. Adewuyi  
Department of Radiotherapy & Oncology, Ahmadu Bello University, Zaria, Nigeria, sadewuyi2003@yahoo.com

Follow this and additional works at: https://digitalcommons.aaru.edu.jo/fcij

Part of the Computer Engineering Commons

Recommended Citation
Patient symptoms elicitation process for breast cancer medical expert systems: A semantic web and natural language parsing approach

O.N. Oyelade, A.A. Obiniyi, S.B. Junaidu, S.A. Adewuyi

Department of Computer Science, Faculty of Physical Science, Ahmadu Bello University, Zaria, Nigeria

Department of Radiotherapy & Oncology, Ahmadu Bello University, Zaria, Nigeria

Received 22 June 2017; revised 16 October 2017; accepted 26 November 2017

Available online 21 December 2017

Abstract

Information gathering from patient by clinicians during diagnostic procedures may sometimes require some skills to adequately collect required information that will be sufficient for the procedure. A situation where this information gathering may prove difficult in when a diagnostic decision making support system (DDSS) will have to gather such information from patient before carrying out the diagnostic procedure. Research has proven that it is more challenging to ensure user or patient inputs, in their raw form, maps into the list of acceptable medical terms for diagnostic tasks. This paper therefore proposes a formalized input generating model that addresses this shortcoming through the creation of an inference process, breast cancer lexicon, rule set and natural language processing (NLP). We developed an input generation algorithm which uses the python natural language processing capability in first filtering and generation the first pre-input collection. Furthermore, this algorithm then feeds in the pre-input word collection as input into the inference engine which has in its memory the rule set and ontology-based lexicon developed. Finally, this generates a list of acceptable tokens that will be sent into the medical expert system or DDSS for the diagnosing breast cancer. This proposed model was tested on a breast cancer based DDSS earlier designed by this authors, and result shows that the inference support of this model generates additional input of about 64% compared to when the patient's input where sent in as input in is state.

Keywords: Medical terms; Semantic web; Natural language processing (NLP); Rule sets; Inference; Heuristics

1. Introduction

There has being a long term effort to close the gap between the understanding of computer and that of human. The fields of artificial intelligence, semantic web and natural language processing are major influencing technologies providing techniques for bridging this gap [1,11]. This becomes necessary because of advances in the development of semantic web, intelligent or expert system that are able to interfaces that allows human's view to be understood by machine [2,12]. However, it has been observed that an impaired input into such expert systems will negatively affect the reasoning and as well result of the system. Hence, an efficient model will close this gap when its uses NLP, seeing that NLP has the potential of generating meaning and adding missing words to raw text [3]. Meanwhile, natural language processing will effectively provide computer—human interaction [4,5], greatly minimizing the challenges of computer understanding human inputs.

Though, different DDSS might have been developed. However, the issue of adequately parsing users' input of natural language is hardly stressed, as it increases the cost of mapping those user's input into a set of terms medically acceptable [6,7], by the DDSS. We note that when necessary inputs — symptoms and observed signs in patients — are not...
sent into a DDSS, it will definitely affect the accuracy of diagnoses — most likely making false positive diagnoses to be assumed as true positives.

In this paper, we proposed a formalized input model for generating a list of medically acceptable terms or tokens that will be passed into a DDSS as a set of needed input that will support the diagnoses process. Our approach is to use a lexicon patterned after the clinical guideline or protocol of the domain of consideration. This lexicon is by an inference engine that uses a rule set. Meanwhile, a natural language parser is first used, also two heuristics, in filtering the raw text sent in by the user/patient.

2. Related work

Natural Language Web Interface for Database (NLWIDB) was developed. By Ref. [4], and NLWIDB allows a user to query database in a language more like English, through a friendly interface over the Internet [5]. Used cTAKES to parse several patient notes to identify concepts relating with diseases and symptoms for patients. By so doing, they showed that modifier combinations need to be used in concert with note sections to capture what is true for the patient at the time of the note. Also, a model, using XML, that provides access to clinical information in patient reports for a broad range of clinical applications, and to implement an automated method using natural language processing that maps textual reports to a form consistent with the model was designed by Ref. [8]. Furthermore, the use of the approaches of getting disease names with the help of classifiers and another way is using the patterns with the help of NLP for getting the information related to diseases was proposed by Refs. [2,9] compared some forms of usage of NLP in their work. These are using MetaMap Transfer (MMTxE) with a negation, detection algorithm (NegEx), another is using an alpha version of a locally developed NLP application called MPLUS2, and the last one uses keyword searching.

Furthermore, HITEEx (Health Information Text Extraction) is built on top of Gate framework and uses Gate as a platform, is a natural language processing (NLP) software application that works by assembling plug-ins into pipeline applications, along with other standard NLP plug-ins. Solves problems in medical domain, such as principal diagnoses extraction, discharge medications extraction and others [10]. MIDAS, an NLP based approach, is an expert system that is able to suggest medical diagnosis from the radiological/clinical patient records, based on information extraction and machine learning from clinical histories of previously diagnosed patients [13]. The LifeCode NLP engine uses a large number of specialist readers whose particular outputs are combined at various levels to form an integrated picture of the patient’s medical condition(s), course of treatment and disposition [14]. This [15] study seeks to find out if patients with multiple sclerosis MS could be identified from their clinical notes prior to the initial recognition by their healthcare providers. Patients were classified as MS or not using Naive Bayes classification.

3. The formalized input model: mapping patient input into acceptable medical terms

Recall that the main aim of this paper is to ensure that patients are allowed to enter their input into the system in an open ended pattern, and then the proposed formalized model maps the input into acceptable list of symptoms. Fig. 1 is the mechanism for achieving this mapping technique. Then following is a list out of the components of the model, briefly describing what each of the component does.

3.1. Components of the model for mapping user input into acceptable terms

The components of Fig. 1 are here listed as follows:

a Patient input box: This is a text-based input graphical user interface (GUI). It enables user to type in their symptoms in sentential form. This input is passed onward into the next phase of the model, the natural language parser.

b Natural language (NL) Parser: In this research, the natural language toolkit (NLTK), is an open source Python library for natural language processing (Bird et al., 2009).
This research employs its ability to effectively parse natural language combined with its capacity to use WordNet. This tool kit is to be used in word tokenization and other necessary natural language operations that will be useful for this model.

c **Heuristics:** this is used as a means of solving the problem of disambiguating users input, and also to prepare input for the lexicon database. The first heuristic is the heuristic 1 (synonym mapper). Once the NLTK has done the necessary parsing by generating useful words, this heuristic 1, works with the lexicon database in gathering all synonyms of the word passed to it. These words are then carefully chosen as they can be interchanged in the context of usage. The heuristic 2, also known as the homonym mapper, also gathers all the relating words to the current word which are spelled and pronounced the same way but might have meant different things to the user and implemented framework. This heuristic also works with the lexicon database. Once the outputs of the two heuristics are gathered, and then are sent into the inference model which straightly maps those words into their intended terms used in the domain of operation of the framework. Note that for accuracy, user may be prompted to ratify to some conclusions mapped this model.

d **Inference Model:** This consist of the inference engine and semantic lexicon (ontology taxonomy/thesauri), and mapping rules. When feed with the semantic lexicon and the mapping rules, the inference engine generate the acceptable terms for usage in the framework. Section 3.2 details and explain these semantic lexicon and the mapping rules.

e **Natural language Lexicon Database (WordNet):** Just as mentioned in (c), an English language lexicon database will be used alongside the process of mapping.

Fig. 2 is a show of how information or data flows from the beginning of the model until the end of it when its output (list of symptoms entered by patient or user) is fed into the framework as input for the clinical reasoning process.

To give a skeletal implementation of this model, an algorithm is written to demonstrate how all these components interact while working on a user's implemented instance. Algorithm 1 outlines the steps for achieving out the user input mapping into acceptable medical terms for the domain of cancer — with breast cancer as the focus of this research.
Algorithm 1. Mapping user input into acceptable terms.

Lines 1–5 outline the variables that will be used in the body of the pseudo code. While lines 6–8 does some basic initializations obvious from the algorithm, lines 10–18 loops through the tokenized words so as to send them through the heuristics, the English lexicon database, and the inference model. Observe that the variable *psynonyms and phomonyms* temporarily store the results of the two heuristics before finally sending them out into semantic input token/acceptable terms storage. Having shown this, the next section now considers the full algorithm and where this model fits into it.

Lines 11–12 of Algorithm 1 was definitely implemented by the python code snippet in Algorithm 2. The diagnostic application logs in the users input in a file for read up by this python snippet in Algorithm 2. User inputs are read in and stop-words are filtered out leaving other words to be lemmatized. Thereafter, synonyms and hyponyms are generated for the lemmatized words. These collections are structurally stored in lines 12–13 of Algorithm 1. Furthermore, lines 15–16 of Algorithm 1 uses the breast cancer lexicon shown in Fig. 3, modeled as an expert knowledgebase (assumed to be the knowledge and experiences of oncologists) to reconcile the collections of Algorithm 2.

Fig. 3. Breast cancer lexicon.
Algorithm 2. Python implementation of the user input filtering.

```python
from mypython import *
from itertools import chain, product

f = open(file_name, 'r')
raw_text = f.read()

text_in_sentence = get_sentence_form(raw_text)
all_tokens = []
for sento in text_in_sentence:
    for filter_words in filter_stopwords(sento):
        all_tokens.append(filter_words

def get_word_form(text):
    words = word_tokenize(text)
    return words
```

Algorithm 3. Python Implementation graphs shown in Fig. 5.

```python
def lemmatization_of_word(word):
    wnl = nltk.WordNetLemmatizer()
    w = wnl.lemmatize(word)
    return w

def get_word_form(text):
    porter = nltk.PorterStemmer()
    w = porter.stem(word)
    return lemmatization_of_word(w)
```

Observe that some user defined functions were invoked from python snippets of Algorithm 2; Algorithm 3 therefore lists the implementations of the user defined functions.

3.2. Input model lexicon, inference process and inference rule

The national cancer institute (NCI) has for some time maintained thesaurus for cancer. While carrying out this research, this thesaurus was downloaded and studied. This document, modeled with OWL DL sized up to about 320 MB. Findings on this thesaurus revealed that it combines all forms of cancer, and all possible concepts in cancer. However, this research finds it too generic and large for consideration during implementation. Hence, this research embarks on following the clinical protocol of breast cancer combined with the support of some oncologist at the teaching hospital in modeling a sizable thesaurus for this work. Fig. 3 shows the thesaurus/lexicon developed during this research. The key words in this domain are listed in a hierarchical pattern, showing their relationship in their domain of application (breast cancer). This thesaurus or lexicon was molded using Protégé 3.8. The entries that resulted in the buildup in Fig. 3 were collated from the sources mentioned earlier on through a collection of standard procedures of diagnosing, treating and handling breast cancer. This knowledge engineering phase of the research adheres to the method of creation of ontology by Ref. [16].

Majorly, the reasoning process adopted in this research is the rule-based approach which is slightly contrary to the decision tree approach which is characterized by drawing up classification and regression trees. Recall that the input model described in section 3.1 was designed to map user/patient input with those which are acceptable in the medical domain.
that this research concerns itself with. As a result, another subtle inference process enabled for information generation is the inference power inherent in coherently structured ontologies. Reasoning in ontologies are achieved through the relationships (is-a or subsumption and property relationship for example) and through the use of reasoners such as Pellet and Hermit. Also, the principle of classification, in addition to the principle of subsumption, is a means for generalization and as well inference of new facts. This paper therefore takes advantage of the following principles of classification and subsumption in building its ontologies for inference support.

a. Principles of subsumption or Inheritance principle: if $S$ is a child of $P$ then all properties of $P$ is also properties of $S$.

b. Principles of classification: Each hierarchy must have a single root; each class (except for the root) must have at least one parent; each class must differ from each other class in its definition.

For example, the class ClinicalFeatures in Fig. 3 subsumes the classes Signs, Symptoms and ClinicalInvestigation. So, by the rule of subsumption, when ClinicalFeatures is selected as user input, the logical process infers that the user will likely want to supply as input the items that ClinicalFeatures subsumes. This inference making therefore increases the size of input. Now, under the Symptoms class, let’s assume the user inputs BreastLumps as seen in Fig. 3, how does our input generating system reasons that if there is BreastLumps as a symptom, then by medical reasoning the symptom Pain might have also manifested? This question is answered by the rule-based reasoning discussed subsequently in this section. Similarly, the three classes Signs, Symptoms and ClinicalInvestigation share the same root ClinicalFeatures, in as much that they are subsumed in it.

On the other hand, we present the inference making process of our rule-based approach. Based on interaction with oncologists (one of which is the fourth author), this research crafted out a rule set base on an approach that oncologists employ in gathering information their patients. The inference process defined by Rule 1 assumes that user/patient can directly or indirectly provides input on symptoms felt and risk factors exposed to. The WHY (for example, why does some symptoms manifests or what could have being wrongly placed to make some particular symptoms to manifest?) questions answered helped in building this rule set.

1. \text{Symptoms(\textit{s})} \land \text{integer[\geq 30,\leq 65](\textit{age})} \land \text{risks(\textit{age},\textit{s})} \Rightarrow \text{InferredSymptom(\textit{s})}
2. \text{FamilyHistory(\textit{fh})} \land \text{Symptoms(\textit{s})} \land \text{causes(\textit{fh},\textit{s})} \Rightarrow \text{InferredSymptom(\textit{s})}
3. \text{Gender(\textit{g})} \land \text{swrlb:equal(\textit{g,Female})} \land \text{Symptoms(\textit{s})} \land \text{risks(\textit{g},\textit{s})} \Rightarrow \text{InferredSymptom(\textit{s})}
4. \text{Environment(\textit{e})} \land \text{Symptoms(\textit{s})} \land \text{causes(\textit{e},\textit{s})} \Rightarrow \text{InferredSymptom(\textit{s})}
5. \text{HistoryOfContraceptive(\textit{hr})} \land \text{Symptoms(\textit{s})} \land \text{risks(\textit{hr},\textit{s})} \Rightarrow \text{InferredSymptom(\textit{s})}
6. \text{HormonalReplacement(\textit{hr})} \land \text{Symptoms(\textit{s})} \land \text{casuses(\textit{hr},\textit{s})} \Rightarrow \text{InferredSymptom(\textit{s})}

Rule 1 consists of five (6) rules within its set. The aim of this rule set is to enrich and support the input or values generated by the preceding inference making process of ontologies described earlier. The first rule on line 1 shows that given a patient whose age gap lies within 30–65 years, then all the breast cancer symptoms associated with those patients in that age group. Also, on line 2, to ensure that all the symptoms that points to patient's history are also considered as inferred symptoms. Furthermore, 99% breast cancer patients are female, hence the symptoms associated with female breast cancer patients are then added up to the inferred symptom list on line 3. Line 4 states that the symptoms that affects patient who have exposed themselves to a particular environmental influences capable of cell mutation must now be added as instances of InferredSymptom class. Wrong use of contraceptives was also a risk factor of breast cancer we gleaned, hence line 5 simply checks if that condition holds in the patient, and then lists all symptoms in the lexicon that relates with this risk factor. Finally, line 6 is rule which checks a patient's exposure to hormonal replacement as a risk factor which if found, positions the patients to having symptoms listed in the lexicon which are related to this risk factor. Looking at the right hand side of the rule (RHS), we will observe that the resulting lists of symptoms in rules 1–6 are classified as instances of another class referred to as InferredSymptom. In each of the rule in Rule set 1, the left hand side (LHS) is the condition that must first be met before the RHS is executed, and the syntax of rule language used is that of semantic web rule language (SWRL).

Using pellet API, the rule set in Rule 1 was implemented using Java programming language. The application reads in the lexicon in Fig. 3, commands pellet to execute the rules in turn and then stores the retrieved results as instances or members of the class InferredSymptom which was already modeled in the underlying ontology. Therefore, the rule set combined with the lexicon in Fig. 3 forms the knowledge representation of the model presented in Fig. 1. In the next section, we present the implementation of the model and the result of the model in diagnosing breast cancer patients.

4. Implementation and result discussion

Again, the aim of this paper is to proffer solution to the limitation of insufficient, inaccurate and mostly open-ended style of input generation for medical expert system's diagnosis processes. This research particularly proposes to add this facility to a medical expert system built on Select and Test (ST) Algorithm for diagnosing breast cancer. Though [6] had already implemented ST for the purpose of diagnosing some general ailment, the authors of this paper however in Ref. [17] had modified the ST algorithm in enhancing is approximation or accuracy. This work of [17–19] is what is referenced as the enhanced ST algorithm for diagnosing breast cancer while that of [6] is referred to the existing ST algorithm. In Ref. [6], there was no automated facility for collecting input into the algorithm for reasoning purposes, hence the solution in this paper. Now, the approach taken by this paper in building such facility was to implement Algorithm 1 with Java. Recall that Algorithm itself has some function call to some Python files as earlier explained. Fig. 4 therefore is a Java snippet of the connection to the lexicon in Fig. 3, the execution of python files, the initialization of a class to launch pellet for execution Rule 1, and finally the generation and display of medically acceptable input into the algorithm described in Refs. [17–19].

Since the result of this paper serves as input into the system in Ref. [17], we then present the input generated for [17] in this paper. So far, an algorithm (Algorithm 1) for collecting user/patient input in an open-ended pattern was designed and presented; this algorithm first filters and generates a collection of words or terms through a python application; these terms are then matched against a database (breast cancer lexicon) of medically acceptable terms in breast cancer. Therefore, so far we have created an input model which intelligently maps users/patients loose or unprofessional inputs from into a list of medical or oncological terms that the enhanced ST application described in Ref. [17] will collects as its input in diagnosing breast cancer patients. Three major text containers are visible in Fig. 5: the prompting window, the user/patient response window, and the result of the generated input window. The system described in Ref. [17] prompts the user of the application a series of questions that helps in gathering input from the user/patient. The responses of the user to those prompts are filed in for the python application in Algorithm 1. The result of Algorithm 1, described in section 3 as input for the breast cancer diagnosing system in Ref. [17], is displayed as a list item of words in the rightmost box of Fig. 5. We note here that this list generated input or medically acceptable words/terms were collated from the lexicon described in Fig. 3, a breast cancer lexicon developed with the fourth author.
Meanwhile, the word collection of the Python snippet shown in Algorithm 2 is expected to have some words repeated. Recall that these collated words of Algorithm 2 and the final word list generated by the implementation of Rule 1 will have some words repeated because of potential emphasis (through repetition of statements or words) made by the user who keyed in the input. Fig. 6 therefore, is a graphical representation of words frequency as supposedly entered by a particular user (of application described in Ref. [17]) as input were made. The repetition of words is graphed in the form of frequency of occurrence of words (on y axis of the rightmost graph of Fig. 6) against the occurring words (on x axis of the rightmost graph of Fig. 6).

Looking at the graph tilted Word Frequency Chart; it will be observed that some words occurred at the frequency of 60, 55 and 49 times while some words had a frequency less than 10. On the other hand, the graph to the leftmost of Fig. 6 only gives a cumulating effect of words/terms generated by the algorithm. This implies that about 600 words (as input into the application described in Ref. [17]) were generated for the responses of this particular user/patient. While the frequency ($f$) denotes the weight attach to symptoms entered, the cumulating words show the density ($d$) of symptoms felt. Hence the formula:

$$\text{Intensity of disease as described by patient } (i) = \frac{f}{d}$$

Algorithm 4 shows the python implementation of the result shown in Fig. 6.

Algorithm 4. Python Implementation graphs shown in Fig. 5.

```python
    def draw_word_frequency_graph(textlist):
        mergedlist = []
        for mlist in textlist:
            mergedlist.extend(mlist)
        wordfreq = {}
        for w in mergedlist:
            wordfreq.append(mergedlist.count(w))
        counts2 = zip(mergedlist, wordfreq)
        counts = Counter(counts2)
        labels, values = zip(*counts.items())
        mybar = plt.bar(range(len(labels)), values, color='green', alpha=0.4)
        ax = plt.axes()
        ax.set_xticks(range(len(labels)), labels)
        ax.set_xticklabels(labels)
        plt.xlabel('Words')
        plt.ylabel('Frequency')
        plt.title('Word Frequency Chart')
        plt.legend()
        plt.show()

    def draw_word_cumulative_frequency_graph(textlist):
        mergedlist = []
        for mlist in textlist:
            mergedlist.extend(mlist)
        fdist1 = FreqDist(mergedlist)
        fdist1.plot(50, cumulative=True)
```

Fig. 6. Graphical representation frequency of acceptable terms from user.
On the other hand, to show the different impact made by the proposed formalized input model, Table 1 displays the impact of the proposed formalized input model.

We note that the system in Ref. [17] was an implementation and comparison of the improved ST algorithm against that of [6]. So then, when this proposed formalized input model was disabled in Ref. [6], the result shows a false positive. However, when the formalized input model proposed in this paper was used in diagnosing breast cancer in Ref. [17], the outcome was a true positive diagnosis. This is due to the approximation of the input into the systems. Table 2 shows two sets of user inputs under different conditions.

### 5. Conclusion

In summary, it has been shown that the proposed formalized input model in this paper will provide support for generating acceptable medical terms or tokens as input for medical diagnostic systems. This formalized input model for providing regularized input into decision support system (a medical expert system) was tested on a breast cancer diagnostic system in Ref. [17]. Our approach employs the use of natural language parsers, crafted rules sets, a domain based lexicon, and an inference engine. Result shows that the proposed model produced an improvement of 64% as against a model that accepts unprocessed input from patients/users. An improvement on this model may consider the use of machine learning algorithms in augmenting the functionality of this formalized input model.

### References


