Progress on Extraction of Medical Guidelines

Sean Gallagher, Hossein Hematialam, Wlodek Zadrozny

March 26, 2015

Abstract

Our objective in the project has thus far been to show that we can extract medical information from text, particularly medical guidelines. In doing this, we have taken three approaches, building links from analysis of constituent trees, building links from regular expression matches on plain text, and via answer type detection in an existing question answering system. So far, these approaches have shown recall on the order of 22% and precision between 57% and 100%. We intend to continue following multiple strategies, and to direct further development with more specific tests and testing data.

1 Preprocessing

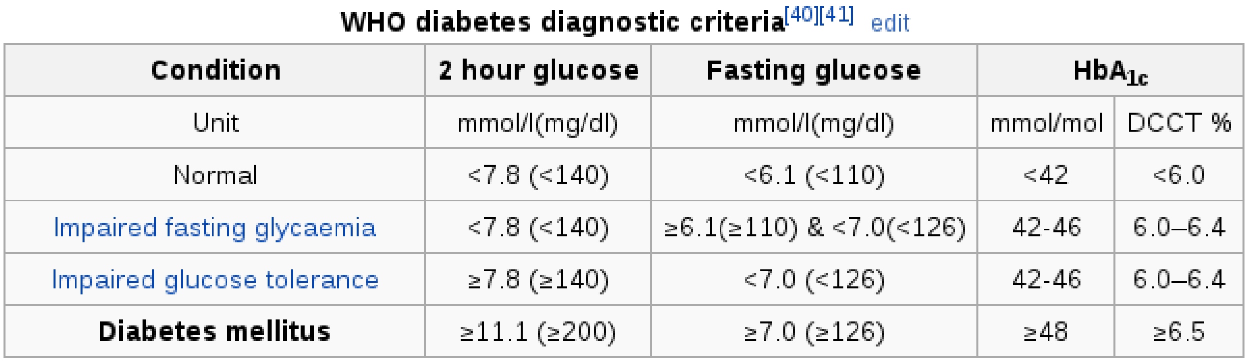
The American Diabetes Association(ADA) publishes standards of medical cares in diabetes annually in its ”Diabetes Care” journal. We consider ADA’s guideline as our primary text source. It provides positive statements, tables, and figures about diabetes. We started by converting the guidelines from PDF to text format, editing sentences only to manage encoding errors, the majority of which were bullet points. Tables and some figures pose a problem, becoming unstructured text with many ambiguities. For example, Table 9 in our source text is replaced with a sentence of more than 80 words.

In the next step, we tried to detect sentences of the text by LingPipe sentence detector which has reasonable accuracy and recall for our project. It detected 5,532 sentences from the text. 232 sentences contain ”A1C” term which is an important factor in diagnosing diabetes.

2 Constituent Trees

Our first attempt at extracting medical conditions and symptoms from unstructured text is by using hierarchical analysis of a constituent parse tree of the text. The system is relatively simple, following a general pattern:

1. Descend to the leaves of the tree, the original words.
2. Match classes of words for operators and numeric values.
3. Match the remaining words against known tests and diseases.



WHO diabetes diagnostic criteria[40][41] edit

Condition	2 hour glucose	Fasting glucose	HbA_{1c}	
Unit	mmol/l(mg/dl)	mmol/l(mg/dl)	mmol/mol	DCCT %
Normal	<7.8 (<140)	<6.1 (<110)	<42	<6.0
Impaired fasting glycaemia	<7.8 (<140)	≥6.1(≥110) & <7.0(<126)	42-46	6.0–6.4
Impaired glucose tolerance	≥7.8 (≥140)	<7.0 (<126)	42-46	6.0–6.4
Diabetes mellitus	≥11.1 (≥200)	≥7.0 (≥126)	≥48	≥6.5

Figure 1: Excerpt of the Wikipedia Article on Diabetes

4. Ascend the tree, combining analyses as follows:
   - A decimal value, an operator, and a test constitute a condition.
   - A condition and an ailment constitute a link, which is the program output.

There are several concerns about this approach, not the least of which are the limited scope of the links it can extract. It expects guidelines of the form ”Values of $A1C \geq 6.5$ may indicate diabetes,” but this causes serious brittleness. Consider the generally equivalent statement: ”A1C values predict diabetes when greater than approximately 6.5.”

There are also weaknesses with tables, which are relatively common in medical literature, and any document with more then a few such links. Take for example the Wikipedia article on diabetes. The condition information is stored in a structured way, as a table. The trouble is that it is not clear how to interpret the contents. ”$\geq$ 6.5” can be interpreted as a operator and value. But the rest of the content needs more context, and this is what we are aiming to solve in section four.

3 Regular Expressions

Modeling data is an essential part of information extraction from an unstructured data source like textual guidelines. Condition-action sentences provide information about the process flow in Clinical Guidelines. In this stage, we focus on modeling these sentences and extracting them automatically.

Condition-action sentences are not always in form of “`if` condition `then` action”. For example, in the sentence “Conditions that affect erythrocyte turnover and hemoglobin variants must be considered, particularly when the A1C result does not correlate with the patient’s clinical situation”, we have a condition-action sentence without an “if” term. On the other hand, conditions may refer to effects, intentions, or events rather than activities. For instance, we see an effect of a condition in “if the A1C is 7.0% and a repeat result is 6.8%, the diagnosis of diabetes is confirmed.” We see that these sentences provide valuable information for us. So, for this stage of project, we consider them as true condition-action sentences.

In order to identify condition-action sentences we used a simple pattern recognition by regular expressions. We consider four patterns for condition-sentences:

- ‘If condition, action.’
- ‘Action if condition’
- ‘When condition, action’
- ‘Action when condition’

We detected 81, 61, 62, and 60 sentences respectively, matching these patterns against the guidelines. You can see some result sentences below:

- Sentence: “If adults with diabetes choose to use alcohol, they should limit intake to a moderate amount (one drink per day or less for adult women and two drinks per day or less for adult men) and should take extra precautions to prevent hypoglycemia. (E)”

  Condition: “adults with diabetes choose to use alcohol”

  Action: “they should limit intake to a moderate amount (one drink per day or less for adult women and two drinks per day or less for adult men) and should take extra precautions to prevent hypoglycemia.”

- Sentence: “When people with type 1 diabetes are deprived of insulin for 12– 48 h and are ketotic, exercise can worsen hyperglycemia and ketosis (197); therefore, vigorous activity should be avoided in the presence of ketosis.”

  Condition when 1: “people with type 1 diabetes are deprived of insulin for 12 – 48 h and are ketotic”

  Action: “exercise can worsen hyperglycemia and ketosis -LRB- 197 -RRB- ; therefore , vigorous activity should be avoided in the presence of ketosis.”

- Sentence: “Metformin, if not contraindicated and if tolerated, is the preferred initial pharmacological agent for type 2 diabetes. (A)”

  Condition: “tolerated”

  Action: “is the preferred initial pharmacological agent for type 2 diabetes.”

Also we found 11, 7, 2, 5 sentences with these patterns in A1C sentences. We verified these sentences and their condition and action result manually. We observed 1 error in condition determining of A1C sentences because of existence of ”,” in the condition part. We also observed four faults in actions. Replacing a table with a sentence and existence of ”,” in condition and action segments are the reason of these faults. We achieved 80% accuracy in identifying condition-action A1C sentences. Our accuracy for the guideline condition-action sentences is around 57%. The decreased accuracy is the result of increasing ”when” sentences, sentences with 2 conditions, sentences constructed from entire tables, and increases in sentence complexity.

To achieve classification performance with regular expressions, we need to make some changes in the preprocessing stage (such as converting table sentences to a set of well structured sentences, changing passive verbs to active verbs, splitting a complex sentence to several smaller simple sentences, etc.)

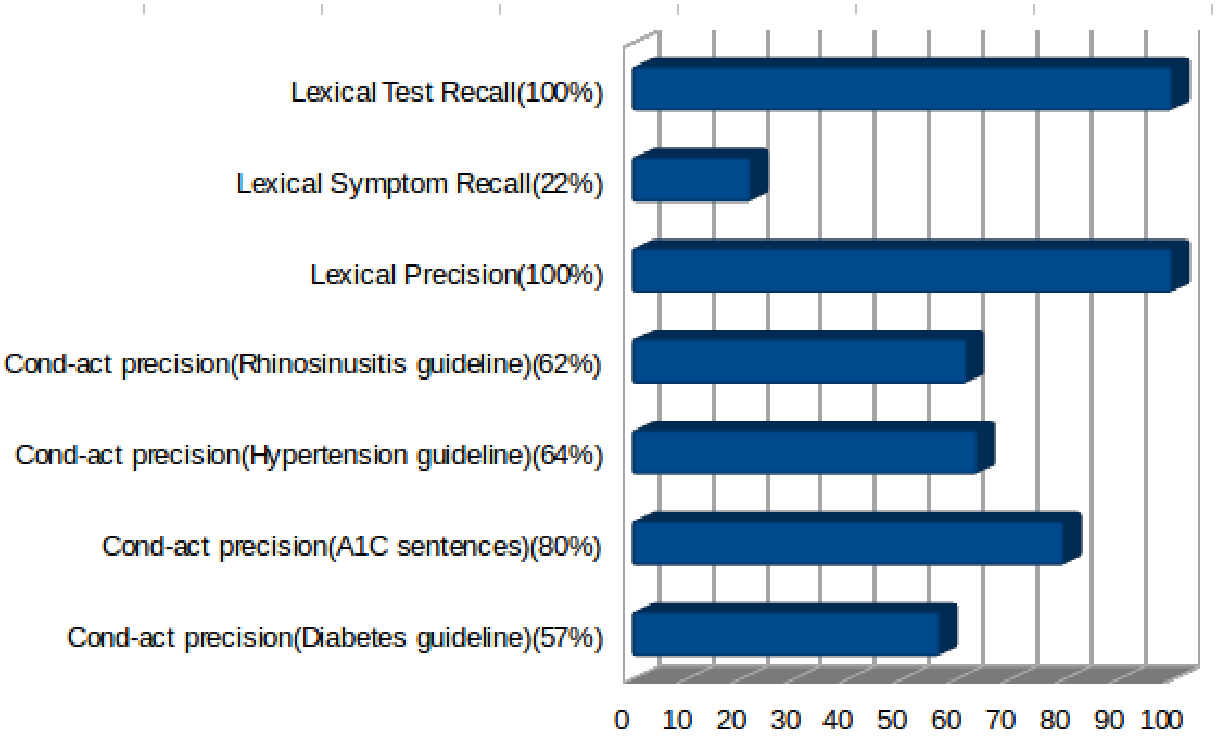


Figure 2: Accuracies of Extraction Modules (small samples)

4 Dependency Graphs

We aim to improve our recognition of diagnostic criteria using other knowledge also available on Wikipedia. We can do that with many of the tools we have already been developing for question answering with Watsonsim.

Firstly, rather than hard-coding the names of many diseases we want to detect, we can recognize the types of the words we encounter using lexical type detection. This way our approach will generalize better for new diseases. Secondly, we can use many Wikipedia sources to determine semantic relations between phrases. For example, we have determined that HbA_{1c}, as seen in the excerpt given above, is a synonym for the main article ”Glycated hemoglobin.” We do this by keeping a table of the redirects between Wikipedia articles. We also assume that phrases which label a link to an article are synonymous with the title of the target article, and we use this to measure synonymy.

4.1 Lexical Types

On the outset we intended to use DBPedia to detect lexical types, but these are not usually specific enough to be greatly helpful, and are synonymous with the target type only on very rare occasions. We attempted to supplement this using ConceptNet, but our initial tests, before parsing the entire knowledge base, still do not indicate very high recall rates. Instead, we have developed a simple rule-based method of extracting lexical types from supporting passages. This covers many cases and the results are reasonably specific but in several instances they make unnecessary distinctions. ("Sign" and "symptom" may often be used interchangably, even though Wikipedia makes a distinction; similarly for "writer" and "author.") At any rate, we are examining whether to include the NELL ontology, which does at least contain many of the phrases we are targeting.

Nonetheless, there are situations where knowing the exact type of a referent is not immediately helpful. Going back to the previous example, there is no direct statement that glycated hemoglobin is a medical test in the following passage, because it is actually the subject of an implied test of its quantity:

> Glycated hemoglobin (hemoglobin A1c, HbA1c, A1C, or Hb1c; sometimes also HbA1c or HGBA1C) is a form of hemoglobin that is measured primarily to identify the average plasma glucose concentration over prolonged periods of time.

This is not immediately helpful because it is not yet clear that glycated hemoglobin is the subject of any test. Unless NELL or a similar ontology has this information, some other general solution for reading such tables will need to be found, or else they may need to be handled more manually.

4.2 Link Expansion

When evaluating supporting passages to discover lexical types, we have found it useful to detect all possible names with the target lexical type, rather than simply to verify known names. Doing this was a departure from the existing QA framework but happened to be helpful even there.

The intuition for such a change is clear. The symptom of pain is relevant enough to diabetes for authors to choose to include it, but diabetes is not relevant enough to pain for the inverse to occur. As a result, any list of symptoms or conditions relating to diabetes will be incomplete unless the evidence for a lexical type comes from multiple passages.

The implications of the type search has the curious effect that the text analysis pipeline operates as a tree when a detected type suggests a new candidate of the same lexical type. We then further stimulate this effect by then sifting for additional names of that type among the newly generated evidence cyclically, making the pipeline into a graph, but the results tend to be very similar.

4.3 Examples and Progress

The dependency graph approach is not complete enough to extract full conditions, though it can find symptoms and signs. For example, a query for the symptoms of diabetes will find the source text that includes the following ten symptoms for diabetes:

frequent urination
increased thirst
increased hunger
weight loss
blurry vision
itchy skin
diabetic dermadromes
slow healing of cuts
fatigue
headache

The query, which is formed to imitate Jeopardy (although this soon will not be necessary), is the following: "This symptom indicates diabetes." The suggestion apparatus finds six symptoms, two of which are symptoms of diabetes: muscle pain and thirst.

For the closely related query “This sign indicates diabetes,” “Proteinuria” is the top result, although in this case it is read from the wiki page on the symptom rather than the disease.

The query ”This test indicates diabetes” discovers the type of the A1C test, and also suggests ”blood glucose fasting test.” As a module in the existing QA framework, the test is recognized. What remains to be seen is the extent to which the existing filtering and scoring play a role, and whether it can be separated cleanly.

The current recall of the lexical type apparatus on symptoms is about 22% for diabetes, zero for hypertension (which is fine since it does not have any to speak of), but also zero for sepsis. Sepsis has plenty of symptoms, unfortunately, they are couched in a syntax indicating that sepsis may not be the only cause.

On the other hand, the precision of recognized symptoms is very high because of the strict match rules. We integrated a more complete logic engine, and developed rules to match conjunctions, but we are uncertain how to recognize all such verbs as ”includes,” ”indicates,” ”is associated with,” and such syntax to improve recall without making the rules excessively verbose.

The situation with medical tests tend to be much clearer. The HbA_{1c}, as well as glucose tests, the Ham test and Fehling’s solution are all detected for the query “This test indicates diabetes,” along with numerous tests not mentioned on the Wikipedia diabetes article, such as a postprandial glucose test. For hypertension, it has strong support for “blood pressure” but also includes “pulmonary function test” and “renal glycosuria.”

5 Software

We used a number of software packages to develop the existing software. In particular, we used CoreNLP for constructing constituent and dependency parses, and LingPipe for additional constituent parses, which was the initial stage of our development. We used Apache Jena and Fuzeki with TDB for interfacing with the DBPedia RDF database as part of the lexical types setup, but DBPedia ontologies provided few sufficiently specific types. We later used logic-programming style rules supported by Jena’s Reasoner interface to detect types in passages, but after some limitations made processing prohibitively slow, we rewrote our lexical recognition rules for TuProlog.

The project source code was written in a mixture of Scala and Java, according to what was more appropriate at the time. Since several of our components run independently, the language difference rarely played a role in development.

6 Possible Next Steps

There are a number of avenues of improvement for our software. In particular, we believe we can improve the recall of the lexical type system if we use the relations in a transitive way. For example, as it now stands, finding HbA_{1c} in a list of guidelines is not sufficient to suggest it as a test. It also needs outside information, possibly found in the same search. But we believe it is tractable to detect such types from across many sources in a preprocessing step.

7 Conclusion

Our project consists of several generally independent modules, and represents a foray into possibilities of text mining for this particular domain. There remains some need for a more consistent and rigorous measure of its effectiveness, in part to direct development. It could also be incorporated into a single, possibly more stable system with more readily reproducible results. Even so, it has been shown to produce precise information from plain text using only machine reading and some automated filters.