The medTurk Book

Aug 7, 2014

Preface

ICBI at Georgetown University is proud to provide this simple, yet effective, tool for converting unstructured clinical notes into structured data for clinical research. This book contains instructions on what medTurk is, how to use it, and how to set it up.

- Robert M. Johnson (rmj49@georgetown.edu)

Contents

Chapter 1

Introduction

1.1 What is medTurk?

medTurk is web application that coordinates the ingenuity of humans to convert unstructured clinical notes into structured clinical data for research. medTurk assumes there are clinical research questions of interest that could be answered by reading patients' clinical notes. medTurk organizes these questions and their allowed answers, in what's called a Research Model (RM). RMs can can organize answers .

medTurk uses a Research Model (RM) to encapsulate the questions you want answered, allowable answers (i.e. control fields), and a 'trigger' that identfies phrases in clinical notes that probably contain the answer to the question sought.

To organize these questions and allowed answers, medTurk uses 'Research Models' or RMs for short. A RM is a simple JSON file that organizes questions, answers, and taxonomies.

medTurk allows multiple users at once to answer questions and the project status page reports the percantage of project complete. The following chapter guides the installation process. Chapter 3 gives instructions on how to setup your first project.

Chapter 2

Installation

In this chaper we guide you through the process of setting up medTurk. medTurk depends on Python and several Python packages. In addition, it requires cTAKES for processing clinical notes, a local running instance of UMLS, and a local running instance of MongoDB. This chapter instructs you on how to setup each of these components. We begin with Python's dependencies.

2.1 Python

medTurk requires a Python version of at least 2.7.6. You can use following line in a Mac Terminal or Window's Command Line to retrieve the current Python version:

```
python --version
```

Next, using pip, install the following Python packages:

```
pip install Flask
pip install mimerender
pip install pymongo
```

2.2 UMLS

medTurk uses UMLS to look up names associated with CUIs reported by cTAKES. The following webpage provides instructions on how to setup a local UMLS MySQL database:

`http://groups.csail.mit.edu/medg/projects/text/Load_UMLS_mysql.html`

Installation of this is necessary because medTurk always makes the assumption it is operated in a private, secure environment.

2.3 cTAKES

Install cTAKES by following the instructions provided here:

`https://cwiki.apache.org/confluence/display/CTAKES/cTAKES+3.1+User+Install+Guide`

2.4 mongoDB

Finally, install mongoDB by following the instructions provided here:

`http://www.mongodb.org/downloads`

Chapter 3

Research Models

This section gives a detailed explanation of the research models (RMs) used in medTurk. Research models form the foundation of a research project and encapsulate the logic requiredto present the right information to workers to answer questions.

3.1 How to Build a RM

Unfortunately, medTurk to date does not provide a graphical user interface (GUI) to create a RM. Because of this, RMs must currently be created by hand or by programming. A RM in the end, is nothing more than a JSON (javascript object notation) file. For clarity, here we present a simple type of RM. One that investigates information on smoking and lung cancer:

```
{
  "name": "Investigation of Smoking and Lung Cancer",
  "children": [
    {
      "name": "Smoking",
      "question": "How many times has this patient smoked?",
      "answertype": "text",
      "triggers": [
        {
          "type": "keyword",
          "name": "smoke"
        },
        {
          "type": "keyword",
          "name": "cigarette"
        }
      ]
    },
    {
      "name": "Lung Cancer",
      "children": [
        {
          "name": "SCLC",
          "question": "Was this patient diagnosed with small cell lung cancer?",
          "answertype": "radio",
          "answers": [
            "yes",
            "no"
          ],
          "triggers": [
            {
              "type": "cui",
              "name": "?"
            }
          ]
        },
        {
          "name": "NSCLC",
          "question": "Was this patient diagnosed with non-small cell lung cancer?",
          "answertype": "radio",
          "answers": [
            "yes",
            "no"
          ],
          "triggers": [
            {
              "type": "cui",
              "name": "?"
            }
          ]
        }
      ]
    }
  ]
}
```

At the root of this JSON, is the key **name**. This is where you name your research model. In this example, the name of the RM is **Investigation of Smoking and Lung Cancer**.

The next key is **children**. Underneath this, is a list of either active nodes (ANs) or passive nodes (PNs). A passive node does not contain a question/answer, but instead, contains two key/value pairs: ”name” and ”children”. Passive nodes exist in order to categorize answers underneath. In our current example, the node named **Cancer** is a PN. If a worker answers **yes** in the AN named **Lung Cancer**, then that answer is automatically tagged with **Lung Cancer**. This allows downstream analysis for example, to group all patients that were diagnosed with lung cancer in general.

ANs contain the questions the workers will be asked. For each question, an answer type is specified under the key **answertype**. medTurk currently only answer types: **radio** and **text**. Choosing **text** means that a text box will appear for the worker to input text in. If radio type is chosen, then the key/value pair of ”answers” must be present. Here, the value is a list of answers. In our current example, the AN of **Lung Cancer** has an answer radio type and has two possible answers: **yes** and **no**. Note that when choosing an **answertype** of **radio**, a worker may choose only one of the answers given in the list **answers**.

3.2 Available Models

Currently, medTurk contains only one research model that is used for extracting information on pediatric cancers, pediatric late effects, and pediatric cancer protocols. It is located in the following directory:

```
medturk/models/late_effects.json
```

Chapter 4

Creating Your Own Project

4.1 Data Preparation

medTurk requires patients' clinical notes to be in JSON format. Each JSON file should correspond to one patient. The name of the file is unimportant. Sample patient files can be viewed at:

`https://github.com/ICBI/medkitpy/sample/patients/`

For convenience, we provide a snippet of one file here:

```
{
  "id": "1",
  "records": [
    {
      "id": "1_1",
      "date": "2000-5-15",
      "note": "Patient with newly diagnosed T-cell acute
               lymphoblastic leukemia. Patient is being treated
               according to COG protocol AALL0434 (On Study)
               and is day X of Induction chemotherapy.
               Coming for follow-up and chemotherapy."
    },
    {
      "id": "1_2",
      "date": "2000-10-01",
      "note": "Patient with newly diagnosed T-cell acute
               lymphoblastic leukemia. Patient is being treated
               according to COG protocol AALL0434 Arm C (On Study)
               and is here today for follow-up and possible admission for
               Day x Consolidation chemotherapy (count dependent)."
    }
  ]
}
```

As can be seen, at the top level, medTurk assumes two key/value pairs. The first key of **id** is the patient's unique identifier. The key **records** maps to an array of all records associated with that patient..

Each record requires three key/value pairs: **id**, **date**, and **note**.

id must be a unique record id across all patient records.

date assumes a YYY-MM-DD format. These dates are used when answer questions in order to arrange clinical notes across a time axis.

note contains the clinical note.

cTAKES

Clinical notes must also be prepared for cTAKES processing. To do this, create a folder which contains two folders: 'input' and 'output'. In the input directory, each clinical note must be written as a .txt file. Using our previous the example, two text files would have to be generated:

```
1_1.txt
Patient with newly diagnosed T-cell acute lymphoblastic leukemia.
Patient is being treated according to COG protocol AALL0434
(On Study) and is day X of Induction chemotherapy. Coming for follow-up and chemotherapy.
```

```
1_2.txt
Patient with newly diagnosed T-cell acute lymphoblastic leukemia.
Patient is being treated according to COG protocol AALL0434 Arm C
(On Study) and is here today for follow-up and possible admission
for Day x Consolidation chemotherapy (count dependent).
```

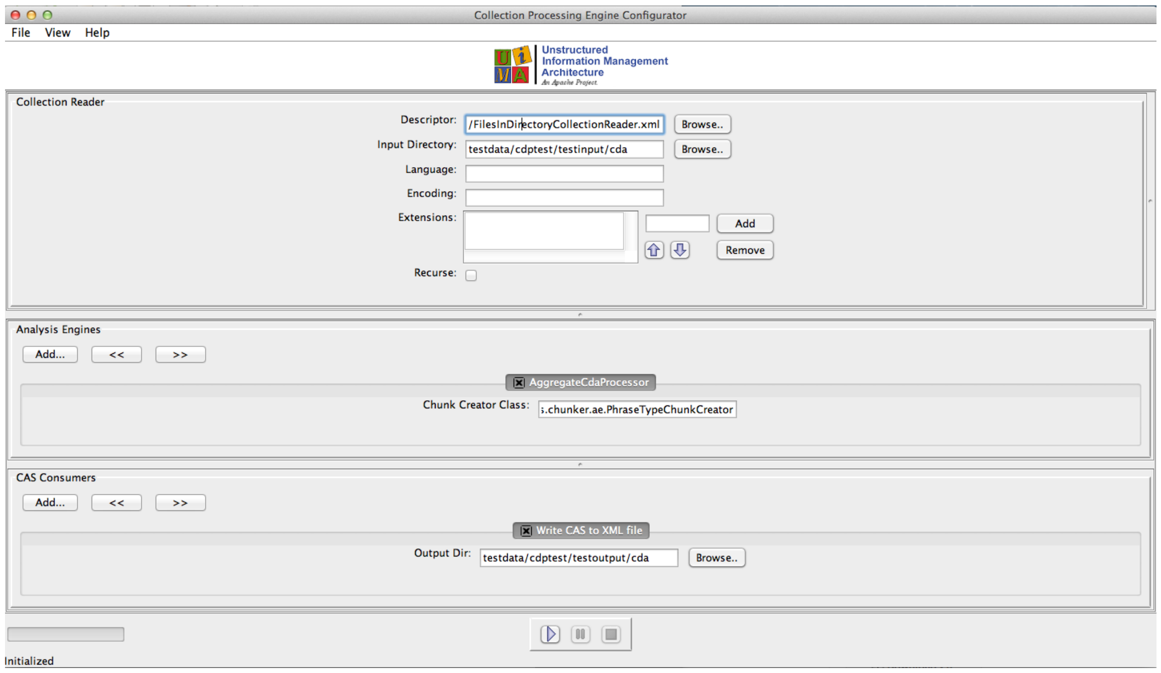
The name of the file must be in the format of *recordId*.txt where *recordId* is the recordId as found in the patients' JSON files. Thus, the file name is important here as medTurk uses this to map the processes clinical note back to its source.

4.2 Running cTAKES

In this section, we use cTAKES to process every clinical note contained the input folder described in the last section. To start up cTAKES, run the following command:

```
sh ./runctakesCPE.sh
```

You should see something like the following:



Collection Processing Engine Configurator

Collection Reader

Descriptor: /FilesInDirectoryCollectionReader.xml
Input Directory: testdata/cdptest/testinput/cda
Language:
Encoding:
Extensions:
Recurse:

Analysis Engines

AggregateCdaProcessor
Chunk Creator Class: ;.chunker.ae.PhraseTypeChunkCreator

CAS Consumers

Write CAS to XML file
Output Dir: testdata/cdptest/testoutput/cda

Initialized

cTAKES provides instructions on how to batch process the files contained in the 'input' directory . medTurk uses cTAKES to process each clinical note for concepts and stores concepts according to three criteria.

Negation

The first criteria is negation. The concept must have positive polarity, that is, it is not negated (e.g. the patient denies smoking).

Subject Analysis

The second criteria is the concept must have a subject of patient. As an example, there are times when the a physician might write that a patient's father had lung cancer. In this example, the concept of "lung cancer" would have a subject of "Patient's Father". medTurk only extracts concepts which act directly on the patient.

Confidence

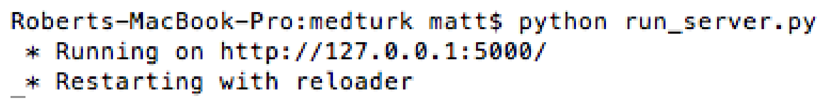
The last criteria is confidence. cTAKES reports a confidence number for each extraction and medTurk only keeps concepts which have a confidence value of "1.0".

4.3 Using medTurk

To start up medTurk, ensure UMLS and MongoDB is running. Next, run the following script:

```
python run_server.py
```

You should see something like the following:



```
Roberts-MacBook-Pro:medturk matt$ python run_server.py
 * Running on http://127.0.0.1:5000/
_* Restarting with reloader
```

Now, browse to the following URL in your browswer:

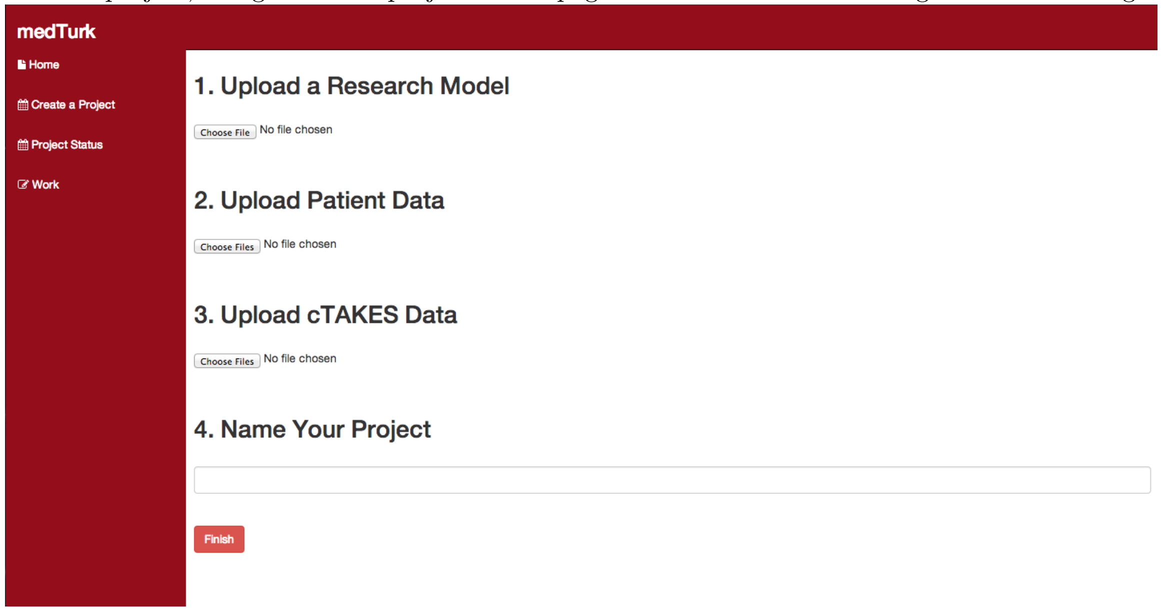
```
http://127.0.0.1:5000/medturk/index.html#/
```

Home Page

This is the landing page that contains a brief description of what medTurk is, and how to get started.

Create a Project Page

To create a project, navigate to the project status page. You should see something like the following:



Step 1

Step 1 let's you upload your RM by navigating to it on your local computer. As an example, you may upload the late effects research model provided in medTurk found in:

```
medturk/models/late_effects.json
```

Step 2

Here, you can upload your one or more patient JSON files in the format explained in Section 3.1. As an example, you may upload the sample patient files found in:

`medturk/example/patients/`

Step 3

The Step 2, allows you to upload the processed cTAKES files. An an example, you may upload the sample cTAKES files found in:

`medturk/example/ctakes/output/`

Step 4

Here you may pick a name for your project. Click Finish and the project will be created.

4.4 Navigating medTurk

In addition to the **Home** and **Create a Project Page** already mentioned, medTurk includes two additional pages.

Project Status Page

You may visualize the research model and view the percentage of project completion in the 'Project Status' page. This page also allows you to download answered questions in CSV format. This file can then be used for statistical analysis.

Work Page

You many answer questions by visiting this page. Each question provides clinical notes that are relevant to answering the question.