Using a Body Function NER Use Case to Drive Improvements to Context Scoping Tasks: Preliminary Report

By Guy Divita, Jessica Lo, Chunxiao Zhou, Julia Porcino, Elizabeth Rasch

# Motivation for the Work

This work was motivated by the desire for the Social Security Administration to retrieve Body Function mentions within their documents to help them adjudicate disability claims. While there is a question around the utility of knowing about body function mentions within documents, particularly as they relate to disability adjudications, this group was motivated to work on this task as a mechanism to improve the algorithms that support body function extraction, namely sectionizing, sentence chunking, and context scoping annotators, using body function mentions as the use case. Body Function mentions are often embedded in complex formatted text in the form of lists, slot: values, and oddly punctuated sentences in PT/OT and H&P clinical notes. The motivation for improving these mechanisms that decompose the formatting and scoping components to work well with Body Function mentions will also improve all those other tasks that also require sectionizing, section chunking, and context scoping.

# Methods

## Data

Data Characteristics

Training and Testing Set Characteristics

This is a hook to talk about whether or not the training set statistically has the same characteristics as the testing set.

## Guidelines

## Annotations

## Algorithms: Two Approaches

The quest for machine extraction of body function from clinical texts is taking two directions. A rule based, explainable approach and a machine learning approach. The rule-based approach us based upon the Apache project UIMA (Unstructured Information Management applications). Stanford’s Named Entity Recognizer, which relies upon an underlying Conditional Random Field (CRF) statistical machine learning modeling algorithm has been chosen as the machine learning approach to start with.

## A Rule based NLP Pipeline

Alistair is built upon a UIMA framework. There is a legacy of clinical NLP tools including cTAKES, CLAMP, and Sophia built upon this framework. Such a framework is based upon the notion of passing each document being analyzed and their associated annotations from function to function. Each function adds to, changes, or can delete the annotations associated with the document. A tenant of this framework is the notion of piecing together a recipe of these functions, often referred to as annotators, into a pipeline. Each annotator takes on an atomic level piece of work to build up a set of useful annotations referring to the original text.

Alistair includes a traditional set of atomic annotators within its pipeline to decompose each text into syntactic and semantic pieces of evidence to enable downstream annotators to use business rules to accurately identify body function mentions in the text.

The recipe is illustrative to the kinds of evidence is being accumulated for each document, which enable the downstream annotators to base their decisions upon.

### The Syntactic Pipeline

For the most part, the annotators listed here do obvious tasks that need no further explanation. There are exceptions and white lies of course, which will be noted for the seemingly mundane tasks for Tokenization, Sentence Chunking, and Date and Time identification. As it turns out, within the richly heterogenous data we are processing, those tasks are not as straightforward and error-free as is ultimately needed.

|  |  |  |
| --- | --- | --- |
| **Annotator** | **Explanation** | **Notes** |
| Line Annotator with Blank Lines | Creates line annotations. | This enables an algorithm to walk through lines of text. Special consideration around multiple blank lines. Multiple blank lines indicate a topic shift. Thus, one needs to keep track of those kinds of lines, rather than filter them out, when looking for paragraph breaks.  This annotator does not work well when there are no newlines in text, as is the case for the BTRIS data we are using. Special ameliorations are needed for such data. |
| Regex Shape Annotator | Creates annotations for emails, phone, URLs, zip codes and common redaction artifacts found in clinical text. | Identifying the easy things you don’t want, makes the task of identifying things you do want easier. Identifying these entities makes sure that downstream annotators erroneously pick up entities that are these. |
| Date And Time Annotator | Identifies dates and times via regular expressions |  |
| Token Annotator | Chunks the text into space delimited units. It creates WordTokens and WhiteSpaceTokens. | The tokenizer used here also creates attributes describing if the token has punctuation, is only punctuation, has numbers, is only numbers, starts with upper case, is camel case, ends with sentence ending punctuation. |
| BTRIS Redaction Annotator | Removes the text that is the redacted text. | A particularity of this data set, redacted names are in Section Names, such as *History of [First Name id = XXXXX ] Illness:* Taking out the redaction enables a bunch of sections to be correctly identified.  Note: This annotator causes trouble when converting the to BIO format because it removes tokens. But, when turned off, performance is actually helped? In the test set by reducing the number of false positives by 3. Gotta look to see if those false positives are true positives. |
| Date By Lookup Annotator | Identifies parts of temporal expressions by items listed in a date lexicon as being a date. |  |
| Date And Time By Token Annotator | Corrective Annotator | There are oddball dates that get missed by the regular expression annotator before tokenization. This annotator identifies dates that bounded by each token. |
| Checkbox Annotator | Identifies and analyzes entities like  smoking: yes [ ] no [x ] | The BTRIS data does not have this kind of data in it. |
| Slot Value Annotator | Identifies and analyzes slot and value entities into a content heading entity and an answer entity.  Example:  *Denies Alcohol*: yes  Slot Value entities are telegraphic sentences which lack an explicit verb. They are quick ways of data capture and easy human comprehension but do not syntactically parse in the same way sentences in prose do. | There are a lot of variations to slot: value formats within clinical text in general, and within the BTRIS dataset. Getting this structure correct is paramount. However, there are many ambiguous examples which flummox the current iteration of this annotator. |
| Sentence Chunker | Chunks the text into sentences and Lists. | Like the slot value annotator, correctly identifying the bounds of when a sentence begins and ends is paramount. The variation of text found in clinical text have flummoxed all the sentence chunkers tried thus far. None have worked 100% of the time. Many of the downstream errors are attributed to sentence chunking failures. |
| Term Annotator | Chunks together tokens into terms based on dictionary lookup. Categorization and syntactic information from the dictionary are tagged onto the terms created. | The UMLS SPECIALIST Lexicon is employed to chunk general English into terms. There are annotator specific lexica also employed, including a date lexicon, a lexicon of section names, a lexicon of assertion terms. The current pipelines employ 20 lexia of one kind or another. |
| Assertion Evidence Annotator | Identifies evidence for negation, conditional statements, hypothetical statements, whether the mention is about the patient (subject), whether the mention is historical, and who is saying the mention. | The algorithm employed is a re-write of Wendy Chapman’s ConTEXT algorithm in java. The Lexica came from her rules, and greatly augmented from work done by three groups at the University of Utah combining each group’s rules. Who is saying the mention (Attribution) is the newest extension to this algorithm and was done for this project. The annotation guidelines stipulated to ignore patient authored statements. Thus, the need to identify who is saying what. This is a nice mechanism to identify symptoms going forward. |
| Unit of Measure Annotator | Identifies things that are measured, are like terms, but not something to be looked up. Things like test results, pulse rate, Ejection Fractions. Degrees of range of motion. | This employs, for the most part, a combination of dictionary lookup for the units part, and regular expression for the numeric parts. The dictionary used for this is a snapshot of NLM’s UCUM resource. Not perfect, but useful. |
| Terms Shape Annotator | This annotator identifies number and units of measure ranges. |  |
| Punctuation Terms Annotator | Corrective Annotator: creates terms that are only punctuation like *+++* | The current lexical lookup ignores runs of only punctuation. Thus making it impossible to create terms that are only punctuation. There are many test results that are only punctuation. This annotator was created specifically for this task to pick up such entities. |
| Person Tokens Annotator | Identifies persons in the text. | The BTRIS data has persons already redacted, so this annotator is not useful currently. This annotator will play more of a role when used with non-redacted text. |
| Slot Value Repairs | Corrective Annotators: There are various failings of the current slot value annotator that these corrective annotators fix, using downstream annotations not available to the slot value annotator |  |
| CCDA Section Header Annotator | Creates Section Headings | Based, for the most part, on Dictionary lookup, using an augmented version of HL7’s list of approved section headings.  The list was augmented a lot for this task because OT/PT specific sections don’t appear within the CCDA domain (yet) |
| CCDA Panel Section Header Annotator | Creates Panel Headings | Panels are sections within clinical documents that list test results for blood tests, primarily. Panels are ignored for this task. |
| CCDA Section Annotator | Creates Section Zones | Think of a section zone as the paragraph(s) related to the section name. |
| Sentence Section Repair | Corrective Annotator: Once section headings are determined, there is need to adjust (erroneous) sentence boundaries to exclude section names. |  |
| Quoted Utterance Annotator | Identifies quoted text | Symptoms are typically found in “quoted text”, so it’s useful to find them. Quoted text does not play a role in the Body Function task. |
| Sentence Repairs | Corrective Annotators: | Removes lists that only have one element to them and turns those back into sentences. Sentences that end with a number also caused issues because the numbers look like list delimiters. So lists that have list delimiters like “1. 2.” that have the list delimiter ordering out of order are likely not lists, but sentences that end with numbers.  Sentences that have tabs in them are likely to be from multi-column formats, where, within the process of OCRing them, the OCR software injected tabs to indicate a new column. |
| Assertion Annotator | Creates assertion attributes to all annotations based on the assertion evidence noted before | This is the crux of the ConTEXT algorithm as noted before. |
| SectionName in Terms Attribute Annotator | Adds the section name to each term in the document. | It is useful to know what section a term is mentioned in. This is useful to filter out mentions found that come from sections you do not care about. |

### Dealing with Data without Newlines

The dataset we have the honor and privilege to work with has a particular idiosyncrasy: it contains no newlines. While not usually worth noting the underbelly of data quality control and data normalization, it is worth noting here because our other datasets, ones coming from a variety of providers from around the world, also occasionally include documents that have no newlines. I am noting the two ameliorations that are necessary to normalize the data to be used with our tools.

Our diligent annotators bless their hearts, annotated these documents without the benefit of newlines. They were able to infer where the section names were from experience in looking at and writing pt/ot notes. However, our sectionizer utterly failed without having newlines. Amelerations were necessary to get all the software downstream from the sectionizer to work. We came up with two amelerations which have proved to help.

It was noted that in 3 of the 5 document types we processed had a pattern of three spaces followed by a section name followed by a colon. That pattern was used to inject a newline before the section name via a simple regular expression and had the advantage of doing no harm to the documents that were not formatted that way.

In documents where there no obvious clues, section names were looked up. For those section names found that had sentence ending clues to the left, and where the section names were in all upper case, or starting with upper case, in camel case form, followed by a colon, a newline was injected in whitespace before the section name. This fix, while not perfect, worked enough to make the documents much easier to visibly read by a human, and made most of the downstream processing work as designed.

These fixes were added to the UIMA Readers employed. A UIMA reader is part of the UIMA framework workflow that involves combining a Reader with a pipeline, then adding one or more writers to spew out processed documents in various formats.

A UIMA reader reads in and converts documents into a container object that is passed from annotator to annotator.

The fixes were injected into the readers to minimize having to have what is a localized issue put into a generalized pipeline. While currently an option to be turned on, even when turned on, each document is analyzed to see if there are any newlines in the document before fixes are attempted.

### The Body Function Pipeline

|  |  |  |
| --- | --- | --- |
| **Annotator** | **Explanation** | **Notes** |
| Body Location Annotator | Creates body location annotations |  |
| Body Strength Annotator | Creates Body Strength annotations |  |
| Body Range Of Motion Annotator | Creates Range of Motion annotations |  |
| Body Reflex Annotator | Creates Reflex annotations |  |
| Body Function Qualifiers Annotator | Creates Qualifier annotations |  |
| Body Function Annotator | Creates Body Function Mentions |  |

The body function pipeline’s purpose is to identify Body function mentions. That is, an utterance that includes a body location, a body function type such as strength, range of motion or reflex along with some kind of qualifier related to the body function type.

The body function pipeline has been appended to the syntactic pipeline. The body function pipeline relies upon having terms in the document already looked up and classified prior to the next set of annotators and knowing what sections those terms occurred in.

A detail. The guidelines and subsequent human annotations created a Body Function Type label, with a type attribute which has an enumerated value as one of “Body Strength”, “Range of Motion” or “Reflex”. For the convenience of building the tool from existing components, those attributes were turned into labels.

A detail: The guidelines indicate some sections to ignore. These include Goals, Plan, Education, Family History, Medications, Referrals, Interventions, Gait, Balance, Coordination, Mobility, Motor learning, Motor Function, Follow-up and Recommendation sections. One oddity, there were several mentions in the training set that came from a common section labeled *Impressions and Plan*. Impressions and plan sections were not filtered out.

The Body Function Location, Body Function Strength, Body Function Range of Motion and Body Function Reflex Annotators each create their respective annotations from terms noted to have those categories as attributes from the upstream term lookup step. Annotations were not made from sections that were specifically noted to be ignored and annotations were not made from mentions that were not about the patient. As noted above, all terms have as an attribute, the section they are within, and if the term is about the patient or not.

#### Secret Sauce within the Body Function Qualifiers

The Body Function Qualifiers annotator is a little more complicated. It starts with iterating through the body function type annotations, finding the sentence they are in. If there is one or more terms that could have a Qualifier category in the sentence or there is a unit of measure or number in the sentence, a Qualifier is made. But there are exceptions.

Sometimes there is both a strength and pain mention in the same sentence and the qualifier found is really about the pain, not the strength. Although less frequent, mentions about coordination and balance were found with strength and range of motion mentions in the same statement and the qualifiers were not about the body function type we are looking for.

To thwart these confusions, mentions that were categorized as pain, coordination, or sensation when found within a window of six tokens of the other body function type kind of mentions, would inhibit the creation of a qualifier. To this end, a small lexicon of pain and coordination terms was created to support this. While this works well, it is noted that a term like pinch ad were both found to be about pain or strength depending upon context beyond the scope of this task.

This is too restrictive when there are both mentions about body function types and pain. Loosening the restriction if the body function type and pain mentions were conjoined by “and” helped performance.

There were a number of qualifier candidates occurring within statements that had mentions a body function type and a body location but the qualifiers were not about the body function type. Common among these were mentions peoples ages and time mentions. The same scoping rules get applied as above. There were a number of confounding terms also found to be in the vicinity of body function type mentions, that when seen, would indicate the qualifier would not be attributed to the body function type. A lexicon of such confounding terms was created and used with the above scoping rules as well. Such terms include “fine motor activities”, “NIHFA score”, and “MRI” to mention a few.

It is noted that the guidelines indicate to ignore a body function mention if the mention is in a conditional or hypothetical statement. Such statements are typically found within sections that are explicitly ignored, so are filtered out by default without need for further mechanisms to handle them. However, they do occur in un-named sections and in valid sections. While the AssertionEvidenceAnnotator and AssertionAnnotator mark every term with a conditional attribute based on the context algorithm, it was found that using this attribute filtered out too many qualifier candidates, hurting performance. There were a number of terms that are clear triggers for indicating a conditional statement that commonly were found in this data set. Among them were terms like “at risk for” and ironically “candidate”. These were made into confounding terms which trigger the same scoping mechanism.

An oddity of software created for one purpose, confounding another, a number of terms categorized as evidence found in page headers and page footers (there are no page headers or footers in the BTRIS dataset, but do occur in our other datasets) were interacting with terms that were also tagged as pain. It was found to help performance not to filter out those terms that were both tagged as pain and evidence to be found within a page header or footer. [When I have time I’ll track that down to remember why]

Scoping rules are a common theme to NLP, making it important to accurately attribute the scope of what section a mention is in as well as what sentence or slot value a mention is. There are a number cases where the text is not pristine sentences, or lists, or slot values. As mentioned above, the qualifier annotator iterated through each body function type, looking for a qualifier within the scope of the sentence or slot value to conjoin the two.

Occasionally there were texts where there were no sentence breaks or multiple colons causing the sentence scoping to go awry. There were a number of these cases where no qualifier was found for a body function type within scope. I added a modification to the scope of where to look for a qualifier candidate when no qualifier could be found within a sentence. Looking to the right by 266 characters (empirically set) to find a qualifier for a body function type improved performance and that rule is now enshrined in a new annotation type called “semantic window” which shows up in the analyzed documents under these circumstances.

#### Body Function Annotator

The Body Function annotator iterates through each utterance (sentence, slot:value ). For each of these, it notes all the Body Function evidences (body location, Strength, range of motion, reflex, qualifiers) and creates an annotation of the maximal spans amongst them within the utterance. Unless, the evidence was attributed to the patient.

Small correction, a body function mention is created when there is both a body function type found and a qualifier. [There were a few cases from the data where this didn’t occur, which will be noted below and hopefully discussed in future work.]

That leaves a number of body location annotations not within the scope of a body function mention. These unattributed body location annotations get subsequently removed within this annotator.

## Lexica

The seed candidate terms for each of the label types came from UMLS sources. Multiple mechanisms were used to start collecting candidate terms. Seed terms were gathered by finding the root term in the UMLS and extract all descendent terms.

### Body Location

For Body Location in particular, SNOMED has the modifier (Body Structure) attached to a number of terms. All of these were extracted, then manually culled to remove those terms which would not be relevant. These culled terms involved terms with “cell” and “cell structure”, “cardiac”, “vein”. This yielded 53,641 terms with UMLS concept identifiers to work with. 36 body laterality terms including left-sided, proximal and distal”, were manually added to cover parts of body location expressions in the text. An additional 24 terms were added to cover body location expressions found in the training set. These were mostly abbreviations like “r le” and a few more colloquial terms like “core” and “quad”. There are 53,704 terms total in the body location lexicon.

### Body Strength

The bulk of Body Strength terms gathered from the UMLS came from terms with the token “strength” in them. There are 4739 such terms. Some of these, admittedly are overly broad, for example having to do with the strength of contractions, and the strength of medication. A number of these were manually filtered out when seen. Sixty two terms were added from expressions seen in the text, not otherwise found. These were mostly in the *form [body location] [extensor|extensors|extension|extensions|ext].* Note that 34 terms having to do with *muscle weakness* were included as part of the body strength lexicon. There are a total of 4802 body strength terms.

### Range of Motion

Descendent terms from Range of Motion were gathered from SNOMED (screen scraping from the SNOMED CT Browser). These were augmented from terms in the UMLS with range of motion, extension, flexion as part of the term. It should be noted that a number of these terms came from MEDCIN in particular. While most of the terms came from SNOMED, LOINC, the National Cancer Institute Thesaurus, Ontology of Consumer Health Vocabulary (ochv), Rheumatoid Arthritis Ontology, along with MeSH, ICD-10CM had some coverage. 16 additional terms were added to cover range of motion expressions found in the training text. There are 793 range of motion terms in the range of motion lexicon.

### Body Qualifiers

A fair amount of body qualifiers are numeric, and are covered by regular expression mechanisms to identify units of measure. To this end, a lexicon of units of measure is being used to identify the units part. That lexicon is derived from the Unified Code for Units of Measure (UCUM) provided by the National Library of Medicine. This resource was altered for the body function task. In particular, all single letter units were commented out, because they were causing too many false positives. In addition, the terms “feet” and “foot” and “field” were likewise commented out. The UCUM lexicon includes 946 entries.

A lexicon needed to be gathered to cover the non-numeric qualifiers. 9755 Terms which had a semantic type of *Qualifier* were taken from the UMLS. This was augmented by terms that descended from *finding* concepts “weakness”, “observation of reflex”, and “hyperflexia”. In all, 2807 concepts from the UMLS were gathered. 104 terms were added to this resource to cover terms in the training text that were not already known as a body qualifier. The Body Qualifier lexicon has 2910 terms.

### Not Body Location

It was useful to gather terms, that when identified, would rule out a body function expression. These were labeled as *Confounding Terms*. Top among these terms was *NIHFA score* and *MRI*. There were 17 such terms added. An additional 34 terms were added, without any semantic label, to chunk up the text to prevent fallacious qualifiers, mostly around temporal entities. Among these were terms like *alert and oriented x 3* and *age-matched norms*. There were 32 such terms added.

### Pain

A small lexicon of 62 terms that indicate pain of some sort was gathered. One term in this lexicon turns out to be ambiguous with a body strength term – pinching.

### Balance and Coordination

A number of qualifiers were erroneously identified because they were within expressions that also included body location and sometimes strength but were referring to balance and coordination. A small lexicon of 13 balance and coordination terms were gathered to identify balance and coordination terms rather than strength or range of motion terms to combat these errant qualifiers.

# Evaluation

### Fuzzy/Partial Matching Criteria

The evaluation is being done at the entity level, with success being counted if there is overlap between the span of the gold annotation and the candidate annotation. For all of its flaws, this mechanism is being used because it was baked into the framework tools long ago.

### Token based Matching Criteria

# Results

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Testing** | **F-1 Score** | **Recall** | **Precision** | **Accuracy** | **PPV** | **TP** | **FN** | **FP** |
| Mentions | .6125 | .9452 | .4532 | .441 | .4532 | 276 | 16 | 333 |
| Mentions(1) | .6149 | .9452 | .4554 | .443 | .4554 | 276 | 16 | 330 |
| Qualifiers | .5699 | .8593 | .4263 | .3985 | .4263 | 446 | 73 | 600 |
| Type | .6378 | .8888 | .4974 | .4682 | .4974 | 480 | 60 | 485 |
| Location | .4696 | .8287 | .3276 | .3069 | .3276 | 271 | 56 | 556 |

Mentions(1) – This was a run with the de-identification annotator turned off.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Training** | **F-1 Score** | **Recall** | **Precision** | **Accuracy** | **PPV** | **TP** | **FN** | **FP** |
| Mentions | .6861 | .9675 | .5315 | .5223 | .5315 | 775 | 26 | 683 |
| Qualifiers | .6571 | .9419 | .5045 | .4893 | .5029 | 1264 | 78 | 1249 |
| Type | .7333 | .9711 | .5890 | .5789 | .5890 | 1313 | 39 | 916 |
| Location | .6041 | .9192 | .4499 | .4328 | .4499 | 899 | 79 | 1099 |

# CRF Modeling

The CRF model transformed the GATE annotations into BIO format[site] which then was trained using the Stanford NLP NER tool, which uses an underlying Conditional Random Field (CRF) machine learning algorithm to do the identification and classification of the labels. The evaluation was done at the token level, giving credit to tokens in the candidate text if those tokens appear in the gold standard.

## Results

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Testing** | **F-1 Score** | **Recall** | **Precision** | **Accuracy** | **TP** | **FN** | **FP** |
| Mentions | .6578 | .8376 | .5415 | .9845 |  |  |  |
| Qualifiers | .6586 | .8442 | .5399 | .9972 |  |  |  |
| Type | .7287 | .8657 | .6291 | .9976 |  |  |  |
| Location | .5776 | .7548 | .4677 | .9931 |  |  |  |

# Failure Analysis

#### On the Training Set

Particular attention is being paid to the qualifiers, since the creation of a qualifier relies upon correctly identifying the Body Function Type span as well as the qualifier, in the context of a valid section within a sentence which does not have confounding evidence within it.

##### False Negatives

By the numbers: The most prevalent term missed was *weakness*. That being said, there were 34 cases where *weakness* was correctly identified, but 93 cases where identifying *weakness* was a false positive. In the cases here weakness was missed,

5 of the 7 cases involving weakness also involved confounding mentions related to balance, coordination and pain. 1 Case was a scoping case, where there was a list of test results followed

no weakness identified on R side of body. As it turns out “no weakness” is in the Body Qualifier lexicon as a qualifier, but not also tagged as “strength” as it should have been.

One case of weakness incorrectly attributed to a patient mention, triggered by the word “note” in the sentence.

##### False Positives

(Every Developer’s complaint: False Positives that are True Positives)

Weakness, impaired.

Scoping

Section Names and Scoping

Sentence Scoping. 32 cases were caused by scoping, where a series of values either delimited by colons, semi-colons or periods limited the scope of what those numbers were referring to. For example, for *… 3 trials : Right : 60 , 60 , 62 Left: 60, 43, 50 Gauge was measured in ….* Missed the scope that these were grip strength measurements.

On the Testing Set

# Discussion

# Future Work

# 