Information

**Background**

Progress in medical natural language processing has stymied due to our inability to interpret medical abbreviations and acronyms (hereby just "abbreviations"). Approximately 30-50% of words in typical clinical text are medical abbreviations, meaning we can hardly process these texts at all in the current state of the science.

Interpreting medical abbreviations in text is a two-step process: (1) all possible meanings of the abbreviation must be identified, then (2) the correct meaning must be selected from the possible meanings (step 2 is also called "abbreviation sense disambiguation" which is a special case of word sense disambiguation).

The first step, and therefore the second step, rely on abbreviation "sense inventories" which are vocabularies of abbreviations (aka "short forms") and their meanings (aka "long forms" or "senses"). Until now, abbreviation inventories have been very incomplete, largely because abbreviations vary enormously between different geographic regions and medical specialties in the US. Obviously, this means abbreviation sense disambiguation hasn't been working very well.

Our team has managed to locate and combine abbreviation inventories from many different geographic regions and medical specialties into one massive inventory. We think this will really improve completeness and therefore disambiguation.

Unfortunately, the combined inventories have lots of redundancy, which makes disambiguation and subsequent downstream tasks much harder. The **objective** is to identify synonymous senses in the inventory and group them, to remove redundancy and simplify downstream tasks. The secondary objective is to demonstrate that this task, which in traditional "knowledge engineering" is often manual, can be automated.

**Sense Inventory**

(Current is under "modules" as "Step2Output")

|  |  |  |
| --- | --- | --- |
| Column | Description | Notes |
| GroupID | Unique identifier for sense group | <- this is what we want to generate |
| EntryID | Unique identifier for entry |  |
| SF | Short form |  |
| NormSF | Normalized short form | <- use this one, not the short form |
| LF | Long form |  |
| NormLF | Normalized long form | <- use this one unless not available, then use LF |
| Source | Source sense inventory |  |
| LFEUI | Long form entry unique identifier  *(only present in one source)* | <- can be used for training |

All other columns are either identifiers or exist for the purpose of *source transparency* (defined as ensuring that all information in the source also exists in the new version) and are not relevant because they're unique to only one source.

**Task**

Within each normalized short form (NormSF) compare all possible senses, and determine which ones are synonymous. Assign unique GroupIDs to each synonymous group.

**Train**

Important: A freely-available training dataset is implicit in the data. Senses with the same LFEUI are synonymous. This training set can be extracted and leveraged by yourself. We can assume throughout that senses with the same LFEUI belong to the same group.

In addition, we've provided 4 training sets which may be useful.

Train 1: The is the main dataset. Your test data should come entirely from this dataset. It was generated by comparing string similarity within normalized short forms and contains mostly positives.

Train 2: This dataset was generated by comparing string similarity more broadly for the purpose of generating pertinent negatives.

Train 3: Hand-generated by the annotators to contain pertinent positives and negatives.

Train 4: Medical synonyms provided by Jim Cimino ("Auditing the Unified Medical Language System with Semantic Methods", JAMIA, 1998, PMID: 9452984). This may or may not help but you can try.

If you need more negatives, they should be easy to generate using string similarity.

**Baseline**

Baseline performance is string similarity (normalized Levenshtein distance) with a threshold of 0.30. This should not be a hard baseline to improve on.

**Performance**

This task is a little different than other machine learning tasks in that it's better to miss positives (aka fail to group something) than to miss negatives (aka group something that shouldn't be grouped). So, we'll want to maximize the specificity here. Otherwise, use whatever performance metrics you prefer.

**Certainty**

Perhaps it would be useful to generate two columns for each entry, specifying the next closest GroupID (that the entry wasn't grouped with) and the certainty that the entry might belong to that GroupID.

**Mistakes**

Our annotators are very good, but of course we can't rule out the possibility of mistakes. If there's anything that is persistently misclassified, send it over and I can review it to ensure it is correct.

**Samples**

transforming growth factor β receptor 2|type II transforming growth factor-beta receptor|Y

2,3,7,8-tetrachlorodibenzo-para-dioxin|tetrachlorodibenzo-p-dioxin|Y

organophosphorus delayed neuropathy|organophosphorus ester-induced delayed neurotoxicity|Y

pea seed-borne mosaic potyvirus|pea seedborne mosaic virus|Y

polyadenylate binding protein 2|polyA binding protein 2|Y

vitamin D receptor|vitamin D receptor gene|N

serum- and glucocorticoid-regulated kinase|serum and glucocorticoid-inducible kinase 1|Y

serum- and glucocorticoid-regulated kinase 2|serum and glucocorticoid-inducible kinase 1|Y

PGI(2) synthase|PGI synthase|Y

apoprotein A-I|apolopoprotein A-1|Y