Information Retrieval

Assignment 4:
Synonym Expansion with Lucene and WordNet

Patrick Schäfer (patrick.schaefer@hu-berlin.de)
Marc Bux (buxmarcn@informatik.hu-berlin.de)
Synonym Expansion

• Idea: When a user searches a term $K$, implicitly search for all synonyms of $K$
  – $S \text{ AND } T \rightarrow (S \text{ OR } S' \text{ OR } ..) \text{ AND } (T \text{ OR } T' ..)$

• Popular method

• Usually increases recall and decreases precision

• Requires a high quality synonym lexicon

• Can be extended to also include hyponyms (‘banana’ is a hyponym to ‘fruits’).
WordNet

- Lexical database with semantic relationships
- Maintained since 1985
- Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets).
- ~66,000 words, ~180,000 Synsets

- Contains different relationship types: hypernymy, hyponymy, causation, antonymy, holonym, meronym ...
Some Relationship Types

- Antonyms are words with opposite meanings:
  \[
  \text{bad} \text{ is an antonym of } \text{good}
  \]

- Hyponyms are specific instances of a category:
  \[
  \text{red} \text{ is a hyponym of } \text{color}
  \]

- Hypernyms describe categories of instances:
  \[
  \text{color} \text{ is a hypernym of } \text{red}
  \]

- Holonyms define a relationship between terms (one is part of the other):
  \[
  \text{tree} \text{ is a holonym of } \text{trunk}
  \]

- Meronyms are the opposite of holonyms:
  \[
  \text{trunk} \text{ is a meronym of } \text{tree}
  \]
Task

- Implement synonym expansion within **Lucene (v6.3)** for the IMDB movie plots.
- You can reuse your existing code from assignment 3 (**using word tokenization and stop word removal, but no stemming**).
- Use WordNet as lexicon
  - current release, **WordNet 3.1**
- For simplicity, we will only consider **Boolean (AND, OR, NOT) term search**.
- No phrase or proximity search any more
Example Synsets from WordNet

**[well]:** [considerably] [intimately] [easily] [comfortably] [wellspring] [substantially] [advantageously] [good] [swell] [fountainhead]

**[good]:** [commodity] [expert] [sound] [respectable] [secure] [estimable] [effective] [honest] [serious] [ripe] [near] [unspoiled] [dear] [just] [salutary] [goodness] [proficient] [skilful] [adept] [thoroughly] [soundly] [unspoilt] [dependable] [right] [upright] [beneficial] [safe] [well] [honorable] [full] [practiced] [skillful]

**[better]:** [expert] [meliorate] [sound] [respectable] [best] [secure] [good] [estimable] [wagerer] [effective] [honest] [serious] [ripe] [easily] [near] [unspoiled] [dear] [just] [salutary] [proficient] [skilful] [adept] [break] [bettor] [amend] [considerably] [intimately] [unspoilt] [dependable] [comfortably] [right] [upright] [ameliorate] [improve] [beneficial] [safe] [well] [punter] [substantially] [advantageously] [honorable] [full] [practiced] [skillful]
• You can search synsets directly at WordNet:
  http://wordnetweb.princeton.edu/perl/webwn
Query Expansion in Lucene

- There are two options:

  - At **indexing time**: Add all expansions to all terms of a document when indexing it.
  
  - At **search time**: When searching a keyword K, rewrite query in disjunction of all expansions of K.

  - Query: plot:Berlin AND plot:wall AND type:television


- Note: If K is part of more than one synset, use all

  - No disambiguation
Getting Started

• Download WordNet 3.1 files at
  – http://wordnetcode.princeton.edu/wn3.1.dict.tar.gz

• Extract noun, verb, adj, adv files:
  – data.[noun, verb, adj, adv] (synsets)
  – [noun, verb, adj, adv].exc (base forms)

• Parse synsets from these plain files using syntax:
  – http://wordnet.princeton.edu/man/wndb.5WN.html
Data File Format

- Each data file begins with a copyright notice. Skip this.
- Each synset is encoded in one line.
- Each line has the format:
  \[
  \text{synset\_offset \ lex\_filenum \ ss\_type \ w\_cnt \ word \ lex\_id \ [word \ lex\_id...]} \\
  p\_cnt \ [ptr... \ [frames...] \ | \ \text{gloss}
  \]

- \textit{w\_cnt}: Two digit hexadecimal integer indicating the number of words.

- Example line (synset):
  
  00007846 03 \text{n} 06 \text{person} 0 \text{individual} 0 \text{someone} 0 \text{somebody} 0 \text{mortal} 0 \text{soul}
  0 421 \text{a} 00004475 n 0000 \text{@} 00007347 n 0000 \text{#m} 07958392 n 0000 +
  01562007 \text{a} 0501 + 00388736 \text{v} 0203 + 04626138 n 0101 + 00729535 \text{v} 0101
  \text{%p} 04624919 n 0000 \text{%p ...}
Exception List File Format

• The first field of each line is an inflected form, followed by a space separated list of one or more base forms of the word.

• Examples:
  
  better good well
  bigger big

• Meaning: all synsets of good and well apply to better (but not the reverse).
Complications I

• Use only single-token synonyms
  – Ignore all synonyms with more than one token
  – These are formatted by a “_” in the name (e.g., house_of_cards)

• Special adjective syntax
  – Remove (p), (a) and (ip) from adjectives (e.g. galore(ip) ).
  – https://wordnet.princeton.edu/man/wninput.5WN.html
Complications II

• Merge synsets of words appearing in the verb, nouns, adj, adv files, such as reason (noun) and reason (verb).

• Consider a synset as set
  – Example: Synset of cause = {reason,grounds}
  – Create the following synonym relations: cause-reason, cause-grounds, reason-grounds and all reverse relations reason-cause, grounds-cause, grounds-reason.

• BUT do not apply this rule transitively
  – Example: cause = {grounds} and grounds={earth} should not create cause-earth!
  – Syn-relationships in WordNet do not form an equivalence class
Complications III

- The **exception lists** are not symmetric. The inflected form is merged with all synsets of its base forms but not the reverse.
- An exception given in adj.exc only adds the synsets defined in the data.adj file. An exception in noun.exc only adds the synsets defined in the data.noun file.
- So you have to keep the synsets in noun, adj, adv, verb separated for the exception lists.
- I.e., given an exception in adj.exc: better good well
  
  \[ \text{syns (better)} := \text{syns}_{\text{adj}}(\text{better}) \cup \text{syns}_{\text{adj}}(\text{good}) \cup \text{syns}_{\text{adj}}(\text{well}) \cup \text{good} \cup \text{well} \]

  **But not** \[ \text{syns(well)} := \text{syns}_{\text{adj}}(\text{better}) \cup \ldots \]

  **And not** \[ \text{syns(better)} := \text{syns}_{\text{noun}}(\text{better}) \cup \ldots \text{syns}_{\text{noun}}(\text{well}) \]
The exception files define base and inflected forms for irregular words. WordNet applies lemmatization for regular words based on rules like big, bigger, biggest. But you can skip this.

[https://wordnet.princeton.edu/man/morphy.7WN.html](https://wordnet.princeton.edu/man/morphy.7WN.html)

Some true results for reference

Only sysnets: 60993 words with 153394 synonyms.

Synsets & exception lists: 66126 words with 176476 synonyms.
Getting started

• in BooleanSearchWordnet.java, implement the functions:
  – public void buildSynsets(String wordnetDir)
    (used to parse the wordnet files and build the synonym index)
  – public void buildIndices(String plotFile)
    (used to parse the file and build the lucene index)
  – public Set<String> booleanQuery(String queryString)
    (parses the query string and returns the title lines of any entries in the
     plotFile matching the query)
  – public void close()
    (can be used to close Lucene index, Threadpool, etc.)
Test your Program

• we provide you with a modified:
  – `queries_wordnet.txt` file containing exemplary queries
  – `results_wordnet.txt` file containing the expected results of running these queries
  – `main` method for testing your code (which expects as parameters the corpus file, the queries file and the results file)

• you can check your synonym expansion for plausibility on the WordNet website:
Deliverables

- by Thursday, 26.01.17, 23:59 (midnight)
- submission: archive (zip, tar.gz)
  - contains Java source files, any used libraries, and your compiled jar named `BooleanQueryWordnet.jar`
  - file name (of submitted archive): your group name
- upload to https://hu.berlin/24377
  - if this doesn’t work, send via mail to buxmarcn@informatik.hu-berlin.de
- test your jar before submitting by running our queries on gruenau2
  - `java -jar BooleanQueryWordnet.jar <plot list file> <wordnetDir> <queries file> <results file>
  - you might have to increase the JVM’s heap size (e.g., -Xmx8g)
  - your jar must run and answer all test queries in ‘queries_wordnet.txt’ correctly
Presentation of Solutions

• you are be able to pick when and what you’d like to present (first-come-first-served):
  – monday: https://dudle.inf.tu-dresden.de/inforet_ue4_mo/
  – tuesday: https://dudle.inf.tu-dresden.de/inforet_ue4_tu/

• presentation will be given on 30./31.01.17

• One team can present their Lucene WordNet Indexer.
• Two teams can present their Lucene Query Expansion.
Competition

- Search as fast as possible.
- Stay under 40 GB memory usage.
- We will call the program using our eval tool:
  - We will use different queries and -Xmx40g parameter

- We will evaluate twofold:
  a) The total query time.
  b) The total time for building the index.
1. **did not change or remove any code** from BooleanQueryWordnet.java
2. **did not alter the functions’ signatures** (types of params, return values)
3. only use the **default constructor** and don’t change its parameters
4. **did not change the class or package name**
5. named your jar **BooleanQueryWordnet.jar**
6. **tested your jar on gruenau2** by running
   
   java -jar BooleanQueryWordnet.jar plot.list wordNetDir queries_wordnet.txt results_wordnet_wordnet.txt
   
   (you might have to increase Java heap space, e.g. -Xmx6g)
7. ascertained that the **queries in queries.txt were answered correctly**
8. **Make sure to upload a zip file named by your group name.**
Next Steps

• this week: evaluation of assignment 3

• next weeks: Q/A sessions for assignment 4.

• Upload your solution by Thursday, 26.01.17, 23:59 (midnight)