Cross-lingual Link Discovery System: Tutorial

Lukáš Žilka

December 7, 2013

Importing Wikipedia Dump to Database

Input: XML Wikipedia Dump of the desired language version of Wikipedia from `dumps.wikimedia.org` (e.g. enwiki-20110803-pages-articles.xml.bz2)

Input: Lang-links Dump of the Wikipedia whose page_id's will be used as concepts (usually the English langlinks file) from `dumps.wikimedia.org` (e.g. enwiki-20110803-langlinks.sql.gz)

Input: Redirect Dump of the desired language version of Wikipedia from `dumps.wikimedia.org` (e.g. enwiki-20110803-redirect.sql.gz)

Output: Wikipedia in DB

1. Run the following command to create 3 .sql files out of the Wikipedia dump:
   `# bzcat enwiki-latest-pages-articles.xml.bz2 | python scripts/wikixray.py -f`
2. Import the `data/mediawiki.sql` into the desired destination database:
   `# cat data/mediawiki.sql | mysql -u<db user> -p<db password> <wikidb>`
3. Import the created .sql files into the database:
   `# cat {page,revision,text}.sql | mysql -u<db user> -p<db password> <wikidb>`
4. Import the .sql file with langlinks. E.g:
   `# zcat enwiki-langlinks.sql.gz | grep "es" | mysql -u<db user> -p<db password> <wikidb>`
5. Import the .sql file with redirects. E.g:
   `# zcat enwiki-redirects.sql.gz | mysql -u<db user> -p<db password> <wikidb>`
6. Configure the PrepareWikiDb section in `config.xml`:

   db URL string for connection to the database where we've just imported the aforementioned .sql files

   disambigStr string which identifies that the given Wikipedia page serves as a disambiguation page

   hnDisambigStr string which identifies that the given Wikipedia page serves as a human being disambiguation string

   lang language code of the Wikipedia's language

   ```
   <PrepareWikiDb:Cs>
          <db>mysql://root:root@localhost/wikidb_cs</db>
          <disambigStr>disambig</disambigStr>
          <hnDisambigStr>hndis</hnDisambigStr>
   </PrepareWikiDb:Cs>
   ```

7. Run the PrepareWikiDb.

Building ESA Background

Input: Wikipedia in DB

Output: ESA Background in DB

1. Configure ArticleIndexer section in `config.xml`:

   db URL string for connecting to the database

   indexPath file path to the folder where the article index will be stored

   stopwordsFile file path to the file with stop-words

   stemmerClass full Java path to the Snowball class that does the stemming

   query.articleIter SQL query in @db to retrieve the list of articles for indexing

```
<ArticleIndexer>
        <db>mysql://root:rootf@localhost/wikidb_cs</db>
        <indexPath>/xdisk/ndx_cs</indexPath>
        <stopwordsFile>res/stopwords.cs.txt</stopwordsFile>
        <stemmerClassx>org.tartarus.snowball.ext.EnglishStemmer</stemmerClassx>
        <stemmerClass>common.lang.CzechStemmer</stemmerClass>
        <query>
                <articleIter>
                        SELECT page_id AS id\, page_id AS page_id\,
                               'cs' AS lang\, page.page_title\,
                               text.old_text AS content FROM page_concepts AS page
                        LEFT JOIN revision ON page_id = rev_page
                        LEFT JOIN text ON rev_text_id = old_id
                        WHERE page_id > ? LIMIT 100
                </articleIter>
</query>
</ArticleIndexer>
```

2. Run `clldsystem.esa.ArticleIndexer`.

3. Configure ESAIndexBuilder in `config.xml`:

   lang language code (for naming the database tables)

   db destination database (tables called `@lang_ndx` and `@lang_terms` will be created)

   esaIndexPath path with the article index (from the previous step)

```
<ESAIndexBuilder:en>
       <lang>en</lang>
       <db>mysql://root:root@localhost/clw</db>
       <esaIndexPath>/home/zilka/devel/kmi/tmp/esaindex</esaIndexPath>
</ESAIndexBuilder:en>
```

4. Run `clldsystem.esa.ESAIndexBuilder`. Note: this will work only on systems where the `sort` utility is on `PATH` (UNIX-like systems should not have problems)

5. Done. The ESA Background for @lang, resides in @db from the ESAIndexBuilder configuration.

Preparing ESA Vector Index

Input: Database with Wikipedia in some language

Input: ESA Background for the corresponding language

Output: ESA Vector Index Database

Output: ESA Vector Index Files

1. Configure ESAIndexer in `config.xml`:

   articleDb URL string for connecting to the database with the articles
   esaDb URL string for connecting to the database with the ESA Background
   destDb URL string for connecting to the ESA Vector Index destination database
   destTableName database table name for the ESA Vector Index in the destination database
   esaVectorSize number of dimensions that will be stored for each article in the ESA Vector Index
   lang language code for the Wikipedia and ESA Background
   indexPath file path to the folder where the article index will be stored
   stopwordsFile file path to the file with stop-words
   stemmerClass full Java path to the Snowball class that does the stemming
   query.articleIter SQL query in @articleDb to retrieve the list of articles for

```
<ESAIndexer>
        <articleDb>mysql://root:root@lwkm012/wikidb_zh</articleDb>
        <esaDb>mysql://root:root@localhost/clld</esaDb>
        <destDb>mysql://root:root@localhost/clw</destDb>
        <destTableName>esa_vector_index_test</destTableName>
        <esaVectorSize>100</esaVectorSize>
        <lang>zh</lang>
        <stopWordsFile>res/stopwords.zh.txt</stopWordsFile>
        <stemmerClass></stemmerClass>
        <query>
        <articleIter>
                SELECT page_id AS id\, page_id\, page.page_title\, text.old_text AS content
                FROM page_concepts AS page LEFT JOIN revision ON page_id = rev_page
                LEFT JOIN text ON rev_text_id = old_id WHERE page_id > ? LIMIT 100
        </articleIter>
        </query>
</ESAIndexer>
```

2. Run ESAIndexer. Note: this takes a considerable amount of time and can be parallelized; the script that runs this in parallel is in `scripts/run_esaindexer`

3. Edit `evi_builder/memndx/prepare.py` in the bottom, and fill in the database connection details to the @destDb from the previous step, and the @destTableName.

4. Make directory for the Index. E.g.:
   `# mkdir /data/ndx_en`

5. Change to that directory and run `evi_builder/memndx/prepare.py`.

6. Now, there should be the built index files in the current directory (`/data/ndx_en`). Those can be used with ESAIndexSearcher, through `esa_search` MySQL function, specifying this path as an argument.