User Overview of SynthaCorpus

Microsoft

July 26, 2017

In its simplest form a text corpus consists of a set of documents, each containing text, comprising a sequence of words[1]. A synthetic corpus has the same form, but the sequence of words in a document may be randomly chosen, and the representation of the words may be made up.

Like real corpora, synthetic corpora have observable properties such as total number of word occurrences, vocabulary size, term frequency distribution, document length distribution, letter frequency distribution, word length distribution and so on.

Real corpora have other properties such as internal structure and external linking which are not currently modelled by SynthaCorpus.

There are a number of scenarios in which SynthaCorpus may be useful:

1. when you need to work with a private corpus you're not entitled to see,
2. when someone wants to replicate results you obtained on a corpus you can't share for reasons of confidentiality, or size,
3. when you want to realistically scale up a corpus
4. when you want to engineer an artificial corpus with specific properties, e.g. to stress test an IR system,

1 Getting going with SynthaCorpus

1.1 Obtaining SynthaCorpus

Assuming you are working with a Unix style shell, e.g. on Linux or MacOSx, under Cygwin, or under Ubuntu Bash for Windows 10 ....

```
git clone "https://github.com/Microsoft/Synthacorpus"
```

1.2 Prerequisites

To use the system you need to build executables from their sources. For this you need to have installed either gcc and make or Microsoft Visual Studio 2015.

To use the system you will also need to have perl 5 installed. Finally, to generate PDF plots (optional) and view them you will need to have installed the gnuplot package and a PDF viewer such as acrobat reader, okular, preview, or a browser such as edge.

1.3 Building with gcc / make

If you have gcc, make, perl and gnuplot installed and in your path you can just type:

```
cd SynthaCorpus/src    # Change to the src directory of the cloned repository
perl exerciser.pl
```

The Exerciser script builds the executables from scratch and then runs most of the tools, starting with a corpus derived from some Project Gutenberg books.

[1] In CJKT languages, such as Chinese, there is no explicit segmentation into words. SynthaCorpus doesn't yet attempt to deal with CJKT text.

1.4 Building with Visual Studio 2015

To build the executable using Visual Studio 2015 (VS 2015), please navigate to `SynthaCorpus/src/Buildexes` using the file explorer, then double click on `BuildAllExes.sln`. That should open the VS 2015 application. In the lower part of the toolbar at the top of the VS window, select `Release` and `x64` from the first and second drop-downs. Then click on `Rebuild solution` from the `Build` menu.

If the build succeeds you should see

```
========== Rebuild All: 8 succeeded, 0 failed, 0 skipped ==========
```

in the output pane at the bottom, and all eight exes will be in:

```
SynthaCorpus/src/BuildAllExes/x64/Release/
```

You can then run the Exerciser script:

```
cd SynthaCorpus/src    # Change to the src directory of the cloned repository
perl exerciser.pl useVS   # An option containing VS removes any GCC
                          # exes and uses VS ones
```

2 Other documentation

A companion document `developer_overview.pdf` gives more detail on uilding and using the system. It also provides information for developers who wish to understand how things work, or to contribute enhancements or extensions to the project. That document lists some useful extensions which contributors may wish to work on.

A set of HOWTOs in this directory is planned to provide detailed information on each of the main components of the system. Currently only `how_to_synthesize_a_corpus.pdf` which describes `corpusGenerator.exe` has been written.

The mathematics and science behind synthesizing a corpus with specified properties is *described elsewhere*.

3 Tools provided

At the most fundamental level, the `SynthaCorpus` project provides executables for:

1. Extracting the properties of a text corpus. (`corpusPropertyExtractor.exe`)
2. Generating a synthetic corpus with specified properties. (`corpusGenerator.exe`)
3. Generating a set of known item queries from a text corpus. (`queryGenerator.exe`)

It also provides perl scripts to achieve:

Emulation of a corpus. i.e. Extract the properties of a base corpus, then use the extracted properties to generate a mimic corpus. (`emulateARealCorpus.pl`)

Corpus sampling . i.e. taking samples of a base corpus and extracting their properties. Sample sizes range from 1% to 100%. Plotfiles are generated showing how the properties change with increasing sample size. A growth model is also generated, allowing for potential scaling up. The script also allows for temporal growth scenarios. (`samplingExperiments.pl`)

Scaling up a corpus . i.e. generating a corpus many times larger than a real one, which has properties which might be expected of a larger version of the real corpus. (`scaleUpASample.pl`)

4 Corpus formats

Currently the executables and scripts described above expect text to be encoded in UTF-8 and stored in a single file recorded in one or other of only two formats:

TSV Each document (with TABs and newlines removed) is represented by a single record. The text of the document is in column 1, and a (numeric) static score is expected to be in column 2. Other columns of metadata may optionally be present.

STARC Simple Text Archive format allows documents to contain control characters, by prepending each record with an ASCII-encoded length. The STARC format is specified in the Developer Overview.

5 Limitations on size

The size of corpus which can be generated, or whose properties can be extracted, is limited by the amount of RAM you have. Hash tables to record all the n-grams in a corpus tend to be very large, and the programs in `SynthaCorpus` are written to favour simplicity of coding at the expense of memory demands. Start with small corpora as in the example in the next section, and gradually work up. If your RAM is insufficient your CPU utilisation percentage will eventually drop, and progress may slow dramatically.

6 Example usage

Below are sequences of commands (similar to the content of the `exerciser.pl` script) which create a STARC file from the Project Gutenberg documents, emulates it, generates sets of known item queries for the base corpus and also the emulated one, runs sampling experiments on the base corpus, scales up a 1% sample and compares the scaled-up sample with the original. The first version uses the executables built by `gcc` and the second those built by Visual Studio 2015.

6.1 GCC example

```
cd SynthaCorpus/src
mkdir ../Experiments
./convertPGtoSTARC.exe ../ProjectGutenberg/*.txt > ../Experiments/PG.STARC
./emulateARealCorpus.pl PG Piecewise markov-5e dlhisto ngrams3
./queryGenerator.exe corpusFileName=../Experiments/PG.STARC
    propertiesStem=../Experiments/Base/PG  -numQueries=1000
./queryGenerator.exe corpusFileName=../Experiments/Emulation/Piecewise/markov-5e_dlhisto_ngrams3/PG.starc
    propertiesStem=../Experiments/Emulation/Piecewise/markov-5e_dlhisto_ngrams3/PG  -numQueries=1000
./samplingExperiments.pl PG
./scaleUpASample.pl PG 100 Linear markov-5e dlnormal ind
./queryGenerator.exe corpusFileName=../Experiments/Scalingup/Working/PG.tsv
    propertiesStem=../Experiments/Scalingup/Working/PG  -numQueries=1000
```

6.2 VS 2015 example

```
cd SynthaCorpus/src
mkdir ../Experiments
BuildAllExes/x64/Release/convertPGtoSTARC.exe ../ProjectGutenberg/*.txt > ../Experiments/PG.STARC
./emulateARealCorpus.pl PG Piecewise markov-5e dlhisto ngrams3
BuildAllExes/x64/Release/queryGenerator.exe corpusFileName=../Experiments/PG.STARC
    propertiesStem=../Experiments/Base/PG  -numQueries=1000
BuildAllExes/x64/Release/queryGenerator.exe
    corpusFileName=../Experiments/Emulation/Piecewise/markov-5e_dlhisto_ngrams3/PG.starc
    propertiesStem=../Experiments/Emulation/Piecewise/markov-5e_dlhisto_ngrams3/PG  -numQueries=1000
./samplingExperiments.pl PG
./scaleUpASample.pl PG 100 Linear markov-5e dlnormal ind
BuildAllExes/x64/Release/queryGenerator.exe corpusFileName=../Experiments/Scalingup/Working/PG.tsv
    propertiesStem=../Experiments/Scalingup/Working/PG  -numQueries=1000
```