OPINOSIS SUMMARIZER LIBRARY

Platform: platform independent
Required Software: JRE 1.6 and above

The Opinosis Summarizer generates very concise abstractive summaries of highly redundant text. It has primarily been tested on user reviews, but since the approach is fairly general, with a few minor tweaks it can be used on any redundant dataset. The Opinosis Summarizer library is a simple jar file. It requires that you have a work directory defined. This directory will hold all the input files, output files and any other resources. The following instructions will guide you on how to generate summaries using the Opinosis Summarizer.

DOWNLOAD LIBRARY & SET-UP DIRECTORY STRUCTURE

Once you have unpacked the zip file, you will see the following items in the directory:

opinosis_lib/ - Contains helper jar files
opinosis.jar – The library that performs the summarization task
documentation.pdf – Set-Up instructions
opinosis_sample – Sample directory structure of the work directory.

Now you need to define a new work directory similar to opinosis_sample. You must have the following directory structure.

```
<your_work_folder>/
      input/  - All the text to be summarized. One file per document.
      output/ - Summarization Results (opinosis summaries)
      etc/    - Other resources like opinosis.properties will be stored here.
```

Now copy the opinosis.properties file from opinosis_sample/etc/ into <your_work_folder>/etc/. This is the file that would contain all the application specific settings. See below on how to change these settings.

SET-UP INPUT FILES

Currently, Opinosis only accepts POS annotated sentences as the input. We expect that each input file to contain a set of related sentences that have POS annotations in the following format:

"that/DT has/VBZ never/RB happened/VBN before/RB ./."
" never/RB happened/VBN before/RB ./."
“…….”

Note that each sentence has to be on a separate line. To generate POS annotations in the required format, you could use the following POS Tagger. Create one input file per set of related sentences. For example one file for all sentences related to the "battery life of an ipod nano", and another for all sentences related to the "ease of use of the ipod nano".

RUN THE OPINOSIS SUMMARIZER

Assuming you have gone over the first two steps above, to start generating summaries type the following:

```
java -jar opinosis.jar -b <path_to_work_folder>
```

```
-b: base directory where input and output directories are found (work
directory).
```

All summary output will be found in `<path_to_work_folder>/output/`

If you want to run the examples, just to get an idea, execute the following:

```
java -jar opinosis.jar -b opinosis_sample/
```

OPINOSIS PARAMETER SETTINGS

To change the various properties for summary generation, just look into the opinosis.properties file found in the <your_work_folder>/etc/ directory. This file contains a list of configurable parameters. Here is an explanation of these parameters:

redundancy - Controls the minimum redundancy requirement. This enforces that a selected path contains at least the minimum specified redundancy. This has to be an absolute value. Setting this value to more than 2 is not recommended unless you have very high redundancies. This corresponds to σ_r in the paper.

gap - Controls the minimum gap allowed between 2 adjacent nodes. If you set this to a very large value, then your summaries may have grammatical issues. The setting recommended is between 2 and 5. The minimum acceptable setting is 2, and the default is 3. This corresponds to σ_{gap} in the paper and has to be an absolute value.

max_summary - The number of candidates to select as the summary. This corresponds to the summary size, σ_{ss} in the paper. This has to be an absolute value.

scoring_function - Which scoring functions to use?

1- only redundancy
2- 2- redundancy & path length
3- 3- redundancy & log(path length) -- default (and recommended)

collapse - Should we collapse structures? Recall may be low when structures are not collapsed. Possible values are true or false

run_id – This is just to give the current run a logical name. Any string describing the run would be ideal.

TECHNINAL QUESTIONS

For any questions or suggestions please send a note to [kganes2 at Illinois.edu].