Homework 3 - Information Extraction

NLP/ML/Web - Spring 2015 - Andrew Rosenberg

Due Monday April 27 at 11:59pm

Your task is to use both structured and unstructured data to find a person's **date of birth**.

Problem 1) Identify 50 famous people, where "famous" is defined as having a wikipedia page dedicated to them. Of these 50 identify 40 as "training" and 10 as "testing". For this set construct two new-line separated files containing list of names – one for training and one for testing. Also construct two new-line separated list of birthdates, for each of the people. The date of birth should be structured as DD-MM-YYYY. If a date component (day, month or year) cannot be discovered replace DD, MM or YYYY with the string "xx". For example, if I can only find that Barack Obama was born in 1961, the string should read "xx-xx-1961", but if I can find his full birthday it should read "04-08-1961"

Problem 2) Identify 50 non-famous people, where "non-famous" is defined as **not** having a wikipedia page dedicated to them. These can be anyone in the world, though you should know their date and place of birth. They may be family members, lesser known people, classmates, etc. Of these 50 identify 40 as "training" and 10 as "testing". Construct similar files of names and dates of birth for the non-famous people as has been done above.

Problem 3) Exploiting Structured Data.

3a. Write a program that takes as input a new-line separated list of names and produces a new line separated list of dates of birth, one line per name, using the same format as above.

This program should construct a Freebase, Wikipedia or DBpedia query and parse the response to generate a date of birth.

You may use some or all of your training data to tune how your program constructs queries. Do not tune the query construction on any test data.

Note: your code will be evaluated on unseen queries.

3b. Write a program that takes two date of birth files – the true DOBs and the hypothesized file produced by your program and evaluates the performance. Your program should return 3 accuracy values: the percentage of correct 1) full birth dates, 2) birth year

and month, and 3) birth year only.

Problem 4) Exploiting Unstructured Data.

For the non-famous people, you will need to get creative about how to find place and date of birth information. Here are some sources to consider: 1) Web Search APIs. Many people without wikipedia pages have information about them on the web. Can you find reference to their age on a dated page? 2) Social Media. Has the person's birthday been mentioned on Facebook or Twitter? Do they have a public Google+ or other profile that might contain this?

Note: This is a very open ended question and a difficult problem. This is meant to be a short homework. Unless you expect to make use of this work on your term project, do not spend an inordinate amount of time. Full "correctness" marks will be given to assignments that have a single functional strategy for identifying the birthday of non-famous people from an unstructured source. This strategy need not work for all people.

4a. Write a program that takes as input a new-line separated list of names and produces a new line separated list of dates of birth, one line per name, using the same format as above. Different from the previous program, this should search unstructured data rather than structured data like DBpedia or Freebase.

You may use your training data to tune how you search for this information. Do not tune the process on the test data. Note: your code will be evaluated on unseen queries.

4b. Evaluate the performance on non-famous people using the evaluation code you wrote for the previous question.

Deliverables

- All source code and libraries (or pointers to download) required for your project.
- A README/Report file whose contents are described below.
- All data.
- All required configuration files

Note: You may use external tools and packages to perform this assignment. Your README must describe how to install these, and you must be able to manipulate all of the appropriate parameters.

The README can be in any electronic format (txt, pdf, doc, google docs, open office) and should minimally include the following

- A list and description of every file included in the submission.
- A description of how your code is compiled (if it is compiled)
- A description of how to run your code.

- A high level description of the task – here: A description of the structured and unstructured resources you are exploring. This should be just a few paragraphs long, and doesn't necessarily need a lot of mathematical notation. It should be understandable to a reasonably informed reader.
- A report of the experiments. What is the performance on your test queries? Do you think your approach is robust to new queries?
- Any other points of interest – running time, complexity, unique qualities of your implementation, etc.

Note: There is no "Linguistics Oriented Component" to this assignment.
Grading:

- **15 points - Compilation** Each file must compile without error or warning into an executable as described above. (Note: for python, no points are awarded for compilation, but execution is worth 30 points.)
- **15 points - Execution** Each executable must run without error or warning on valid input using the command line parameters described above. (Note: for python, no points are awarded for compilation, but execution is worth 30 points.)
- **15 points - README** Does your README documentation completely and accurately describe the task and approach taken? Does it satisfy the content requirements (i.e. how to compile and run your project, file listing, etc.)
- **12 points - Within Code Documentation** Every function should include a comment minimally describing 1) what it does, 2) what its inputs are and 3) what its output is. Are there effective other comments throughout the code? You may use a javadocs, or pydoc, or other standard. For a good read check out the google style guides: `https://code.google.com/p/google-styleguide/`.
- **13 points - Style** Is the structure of your program clear and coherent? Are functions and variables given self-explanatory names? Are functions used to aid intelligibility of the code? Are functions used to reduce repeated blocks of code? Is indentation, spacing, use of parentheses, use of braces consistent, and sensible? For example, if you use brackets on the same line at the start of a block, always do so. If you place a brace on the line following the start of the brace, always do so. If you put a space between variables and operators, e.g. `if (i == j)`, always do so. So, `if (i == j) i = j+k;` is bad. It should be `if (i == j) i = j + k;` or if you prefer `if (i==j) i=j+k;`. You will be graded on consistency in these decisions, not on any particular style.
- **25 points - Correctness** Is/Are the algorithm(s) implemented correctly? Have an appropriate number of word/document representations been used and used correctly?

- **5 points - Instructor's Discretion** Has this assignment gone beyond the minimal requirements in a substantive way? Is it especially clear? Is the code especially well written? Is the response particularly thoughtful or insightful? Have non-trivial approaches been developed and exploited?