Homework 2 - Language Modeling of Web Data

NLP/ML/Web - Fall 2015 - Andrew Rosenberg

Due Monday, April 6 at 11:59pm

Your task is to run a set of language modeling experiments based on web data.

Problem 1) Language Model Training and Evaluation

1a. Collect at least 100,000 words of data. Identify between 10 and 33% of the data as test data. Identify between 10 and 33% of the data as development data.

Use your discretion about the distribution of train, development and test data, but report your decision in the report.

1b. Train an N-gram language model (without backoff or smoothing). Identify an optimal value of N, based on evaluation of the development data. Report the test set perplexity of this model.

1c. Smoothing. Repeat the previous evaluation using a smoothing technique (Laplace, Good-Turing, Knesser-Ney, etc.).

1d. Backoff. Repeat the previous evaluation using Interpolation Backoff. Start with N+1 grams, where N is the optimal value previously identified. Identify interpolation coefficients from development data.

Problem 2) Language Model Adaptation.

2a. Collect another set of at least 50,000 words of data from a distinct domain. Identify 10 and 33% of the data as test data. Identify between 10 and 33% of the data as development data. We'll call this Domain B and the domain of the previous question Domain A.

2b. Use the code from the previous section to identify the best language modeling value for N, and smoothing/backoff configuration) for this new domain. Minimally report the best configuration and perplexity of the evaluation data.

2c. Evaluate the best performing Domain A model on the Domain B evaluation data.

2d. Linear Interpolation. Use linear interpolation to adapt the best performing language model from the previous section to this new domain. Identify the optimal value of lambda (λ) on Domain B development data. Report the identified λ.

2e. Evaluate the perplexity of this interpolated model on the evaluation data.

2f. **Linguistics Oriented Requirement:** Read Fernando Pereira's 2000 paper, "Formal Grammar and Information Theory: Together Again?". Write a 2-4 page response which minimally 1. identifies the major claims of the paper, 2. assess the approaches used to address these claims and 3. assesses the conclusions drawn. Any additional discussion on the relationship between statistical estimation and formal grammar is very appropriate in this response.

Deliverables

- All source code and libraries (or pointers to download) required for your project.
- A README/Report file whose contents are described below.
- A response to the Linguistics Oriented Requirement.
- 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 – including which matrix library (if any) you use.
- 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 data and the types of document representations 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 impact of N-gram size and smoothing on perplexity within a single domain? A graph/chart of the train and development perplexity at different values of N. How much worse is cross-domain performance than within domain performance? Is interpolation an effective domain adaptation strategy for the domain(s) that you're working with?

- Any other points of interest – running time, complexity, unique qualities of your implementation, etc.

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.)
- **10 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.)
- **20 points - Linguistics Oriented Requirement** Have you given a thoughtful response to the paper? Is the response coherent and well-written? Does the response represent creative thinking about the tension between Shannon- and Chomksy-derived theory?
- **7 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/`.
- **8 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.
- **20 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 representations been examined?