SCC.413 Applied Data Mining - Natural Language Processing Coursework on Language Identification

Paul Rayson, Ignatius Ezeani

Context

Language identification is a necessary prerequisite for a large number of NLP tasks such as spell checking, speech recognition, optical character recognition, machine translation, text-to-speech synthesis, authorship attribution, forensic analysis and is used for search engines and summarisation. This task also focuses on n-grams since we have covered this topic (but at the word level) in the lectures e.g. unigrams and bigrams are used for POS tagging and classification.

Task

Your task is to use *character bigram* information to correctly classify language samples from three different languages: English, Czech and Igbo. By character bigrams we mean pairs of alphanumeric characters, punctuation marks and even spaces between words. For example, for the text "We love SCC413.", the character bigrams are "we", "lo", "ov", "ve", "sc", "cc" (ignoring spaces, punctuation and numbers).

You have been provided with large text samples (around a million words) of each language and you should build a character bigram language model of each language from a training subset. The language model can take the form of a list of normalised frequencies (e.g. frequency per million) of all the character bigrams in the training data. You will need to devise a metric by which you compare the language models against one derived from a test sample in order to determine the language of the test sample. You should then use test subsets of the language files to evaluate the accuracy of your language identification system. You should make sure that you separate training data from test data. The English and Czech language samples are available on the SCC.413 Moodle page and the Igbo samples from our GitHub repository[1]. Your system only needs to distinguish between the three given languages. The language samples may contain XML tags, so your implementation should ignore any text within <angled brackets like this>, and the Igbo data is provided in JSON format as well as plain text.

Relevant papers

If you wish, you can carry out some background research using, for example, a Google Scholar search for "n-gram language identification". However, we've provided a list of good papers below. They describe a variety of approaches to n-gram language identification. You can ask us about these academic papers (or others you found) and the algorithms they employ. You should choose a simple algorithm to implement like these or you may wish to implement something more complex using a machine learning approach, like we cover in the module.

Adams and Resnik (1997). A language identification application built on the client/server platform. ACL workshop From Research to Commercial Applications: Making NLP Work in Practice. http://www.aclweb.org/anthology/W97-0907

[1] https://github.com/IgnatiusEzeani/IGBONLP/tree/master/ig_monoling

Bashir Ahmed, Sung-Hyuk Cha, and Charles Tappert (2004). Language Identification from Text. Using N-gram Based Cumulative Frequency Addition. Proceedings of Student/Faculty Research Day, CSIS, Pace University, May 7th, 2004. http://www.csis.pace.edu/~ctappert/srd2004/paper12.pdf

William B. Cavnar and John M. Trenkle (1994). N-Gram-Based Text Categorization. In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.53.9367

Penelope Sibun and Jeffrey C. Reynar (1996). 'Language Identification: Examining the Issues.' Proceedings of the Fifth Conference on Document Analysis and Information Retrieval, pp 125—135. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.52.4524

Deliverables

You should deliver the code (in a programming language of your choice e.g. Java, Perl, Python, C etc) and a short report about the experiment. The tool that you develop does not need to have a complex user interface. Command line solutions implemented, for example in Perl, are perfectly fine. Marks awarded are not just based on successful solutions, but your reflection on the algorithm, the implementation and your evaluation of it. An initial implementation itself should be feasible in two hours but we are issuing the coursework several weeks ahead of the deadline so that you can discuss it with us during the lab sessions, and to enable you to tackle it earlier and plan your time relative to other deadlines. Please get in touch with us if you have questions about the algorithm, training, testing or the write up of the report. The awarded marks will be holistically based on the code and report submission. Both the code (as a zip file) and report (in PDF format) should be submitted on Moodle.

Final report

The final report that you deliver along with the code should contain the following sections. Reports should be a maximum of three A4 pages long, font size 11 as a minimum, with minimum 2cm margins.

Introduction: an overview of your submission including a very brief justification of your choice of implementation language(s).

Algorithm: describe the language identification procedure and metric that you have implemented. If you derived it from one or more of the academic papers above (or others) state this and say how you changed the algorithm(s), if at all. Describe enough detail so that someone could replicate your solution if it differs from a published paper.

Results and evaluation: Quantify your training and test data and show accuracy results for the test samples in table and graph format. You should show results for at least three different-sized test samples. Be clear about any use you made of cross-validation methods[2].

Conclusion: What conclusions can you draw based on your evaluation? How much training data is really needed for a reliable system? What is the minimum size test sample that you can reliably identify the language of? Speculate about whether your solution is suitable for the identification of other languages, and whether any features beyond character n-grams might be needed (you do not need to implement these).

References: any papers or websites that you've cited in the text.

[2] https://en.wikipedia.org/wiki/Cross-validation_(statistics)

Format, plagiarism and referencing
You must observe the University's regulations on plagiarism when producing your essay. https://www.lancaster.ac.uk/student-and-education-services/exams-and-assessment/regulations/plagiarism/
There are many referencing styles available, such as Harvard, APA, Chicago and Vancouver. You are encouraged to use Harvard referencing for SCC coursework, and to support you with this the Library have created a Harvard (Lancaster University Library) style, which has an online guide and a printable guide which is available at https://lancaster.libguides.com/harvard. This gives examples of how to cite in the body of your report, and how to create a reference list. The Library's guide for SCC has links to key databases such as ACM Digital Library and IEEE Xplore (see the Using Library Databases section):
http://lancaster.libguides.com/computing

Mandatory cover page and declaration
There should be two pages at the very front of the report, as follows:

1. A title page with the following information:
<Your Own Name> <The Title of the Project> <Degree Scheme Title> <Date of Submission>

2. A single page containing a statement of originality (declaration), with text exactly as follows:
I certify that the material contained in this coursework is my own work and does not contain unreferenced or unacknowledged material. Regarding the electronically submitted work, I consent to this being stored electronically and copied for assessment purposes, including the School's use of plagiarism detection systems in order to check the integrity of assessed work. I agree to my coursework being placed in the public domain, with my name explicitly included as the author of the work.
Name:
Date:

Indicative marking guidelines and expectations

	Program	**Report**
A+	Working program that meets the requirements for an A, but with an extension to more than three languages and use of more than one machine learning approach for comparison purposes. Cross-validation employed well.	Report as for an A, but showing evidence of working solution for more than three languages, and write up of machine learning comparison of approaches.
A	Working program that meets the requirements for a B, showing a method to home in on a minimum size test sample that can be predicted, or other advanced features. Good evaluation approach used e.g. cross-validation.	Report as for a B, which describes a comprehensive solution to the coursework with some speculation about the suitability for other languages, or machine learning approaches.
B	Working program that meets the requirements for a C, with some repetition of training for three different size training sets or framework or script to run the code multiple times to achieve multiple test runs.	Report as for a C, but with more extensive results and attempts to answer 1-2 questions posed in the conclusion description above.
C	Working program that meets the requirements for a D, with a prediction of which language the test sample is from either using a simple algorithm from one of the papers cited or a machine learning approach.	A more complete report that fully describes the implemented solution with justification of some of the design or implementation language choices. Lacking clarity on some issues.
D	Working program that counts bigrams, and performs a rudimentary comparison of trained model to a test file.	A basic report that factually or partially describes the solution and method but with no reflection on what worked and what didn't or what could be improved.
F	No working program submitted.	Report does not meet any of the requirements listed above.