

ANLP Assignment 1 2016

due: Tuesday 18 October, 3pm, electronic submission (see section 1.4)

1 Overview

1.1 Tasks

In this assignment, you will use language modeling to detect what language a document is written in. Specifically, you will write Python code to do the following:

- build a trigram language model over characters (not words!): read in a text file, collect counts for all character 3-grams, estimate probabilities, and write out the model to a file.
- generate random output according to a similar model provided by us (which you'll need to read in from a file).
- read in a test document and compute its perplexity according to one of these models.

You will also need to write a short description of what you did and why, and answer some questions.

1.2 Motivation and goals

By completing this assignment, you will practice skills that are important for doing NLP, and for other research in Informatics. This assignment will allow you to demonstrate that you:

- understand some of the concepts and issues involved in language modeling and working with real language data;
- can implement a basic Python program; and
- have some of the skills needed to write a good scientific report. These include being able to clearly explain/justify decisions you have made, describe results you have obtained, and draw appropriate conclusions from them. We don't just care that you got the right answer, we also care that you are able to explain how and provide evidence justifying your answer.

1.3 Working with others (and not)

The assignments for this class are intended to be done in *pairs*: by working with another student, you can discuss ideas and work things out together. Ideally, try to find a partner with a different skill set to your own, although this may not be possible in all cases. You may choose your own partner, but you must work with different partners for each assignment.

You may discuss any aspects of the assignment with your partner and divide up the tasks however you wish; but we encourage you to collaborate on each part rather than doing a strict division of tasks, as this will enable better learning for both of you.

You may also discuss high-level concepts and general programming questions with others in the class; however you may NOT share code or answers directly with other groups. For example, it is ok to suggest using a particular library function to help solve a task; it is not ok to send (or describe) a code snippet showing exactly how you used that function on what arguments.

Your code and report must be your own group's work. Any key ideas obtained from outside sources should be appropriately cited, and direct quotes must also be included in quotation marks. If you re-use code snippets from outside sources, these should also be clearly marked and the source cited.

See the School's guidance on plagiarism and academic misconduct for more information:

<http://web.inf.ed.ac.uk/infweb/admin/policies/guidelines-plagiarism>

Note that you are required to take reasonable measures to protect your assessed work from unauthorised access. For example, if you put any such work on a public repository then you must set access permissions appropriately (permitting access only to yourself, or your group in the case of group practicals like this one).

1.4 Submitting your assignment

Only one student in your pair needs to submit, but make sure both student's ID numbers (s...) are included in the report!

Submit your assignment using the following command on a DICE machine:

```
submit anlp 1 <file1.py> <file2.pdf>
```

where you replace **<file1.py>** with the name of the file (or files, you can add more filenames if needed) with your python code and **<file2.pdf>** with the name of the file with your report.

Notes:

- Your report *must* be either a .pdf document or .txt (plain text) file, please convert other files types (eg Word documents) into .pdf format before submitting.
- You can submit more than once. Identically named files will overwrite earlier submitted versions. All submissions are timestamped, so do not submit anything after the deadline—it will be considered late, and receive a mark of 0. [Even if you submitted an earlier version, we won't be able to give you credit for that one if it's overwritten!]
- Make sure you **save** your code and report **before** submitting, that the filenames are the ones you intend to submit, and that you go through the whole submission process (you may need to answer some y/n questions) and get a message in the terminal that submission succeeded.

2 Assignment specification

2.1 Data

The data we use for this project is part of the Europarl corpus, which requires a license to use. The University has a license, so to download the data you must be inside the University intranet. You may use the data for your University work but may not distribute it to others.

From a University computer, the data is available at:

<http://www.inf.ed.ac.uk/teaching/courses/anlp/data/assignment1-data.tgz>

On a DICE machine you can download and unzip it with Archive Manager. You should find that the data consists of:

- `training.en` - English training data
- `training.es` - Spanish training data
- `training.de` - German training data
- `model-br.en` - a pre-trained language model file
- `test` - test document

2.2 Code

To help students who are new to programming, we've provided a very small amount of code that may help you get started with the tasks specified below. The code is available here:

<http://www.inf.ed.ac.uk/teaching/courses/anlp/hw/code/asgn1-helper.py>

You are not required to use the provided code, and if you are an experienced programmer you may prefer not to.

2.3 Your tasks

1. (10 marks) Before starting to code, you should always have an idea of what your program's expected output should be, so you can check that your implementation is correct. In the last part of this assignment, you'll be asked to compute the perplexity of some real language data under a language model that's also computed from real data. But to test your code, you'll want a simple test case that you've done by hand.

Consider the following trigram character language model, where `#` is a symbol used to mark both the start and end of a sequence, and probabilities are written as $P(c_i | c_{i-2}, c_{i-1})$. So, $P(a | \#b)$ means the probability of the character `a` given that the immediately preceding character is `b` and the one before that is `#`.

$P(a \# \#) = 0.2$	$P(b \# \#) = 0.8$	$P(\# \# \#) = 0.0$
$P(a \#a) = 0.2$	$P(b \#a) = 0.7$	$P(\# \#a) = 0.1$
$P(a \#b) = 0.15$	$P(b \#b) = 0.75$	$P(\# \#b) = 0.1$
$P(a aa) = 0.4$	$P(b aa) = 0.5$	$P(\# aa) = 0.1$
$P(a ab) = 0.6$	$P(b ab) = 0.3$	$P(\# ab) = 0.1$
$P(a ba) = 0.25$	$P(b ba) = 0.65$	$P(\# ba) = 0.1$
$P(a bb) = 0.5$	$P(b bb) = 0.4$	$P(\# bb) = 0.1$

What is the perplexity of the sequence `##abaab#` under this model? Show your work.

Hint: "Show your work" means, roughly speaking, that you should write down the relevant equation, what you plugged into it, and the result, with at most a few sentences of explanation. You do not need to explain all the assumptions behind the equation or what it is used for in general; please keep your answer brief.

2. (10 marks) For the three languages we gave you, the language identification task is too easy if we include the entire character set, because there are some characters that only

occur in one of the languages. So, we will ask you to preprocess the data to make things a bit more interesting.

Write a function called `preprocess_line` that takes a line of text (string) as an argument. It should return a new string which removes all characters from the line that are not in the following set: characters in the English alphabet, space, period (full stop), or digits. (That is, remove characters with accents and umlauts and the other punctuation marks). Your function should also lowercase all remaining characters and convert all digits to '0'.

Depending on how you write the rest of your program, you may do slightly more in this function than just what we have specified. If so, add a comment to your code explaining what else you did and why. You do *not* need to justify the steps we explicitly asked for.

You should use this function in building your language model and scoring the test document; i.e., your language model should be over n-grams of lowercase letters plus a few other characters as noted.

To get credit for this task, include the code for this function in your report.

3. (10 marks) We have provided a pre-trained language model in the file `model-br.en`. This model was trained on a corpus of English, but not necessarily the same one we gave you. Each line of this file lists three characters followed by a tab and a number. The number represents the model's estimated probability of the third character given the previous two. For example, one of the lines looks like this (where `<tab>` is a tab character):

`and<tab>2.507e-01`

which means that for this model, $\hat{P}(d|a,n) = 0.2507$. The file is alphabetical, so at the beginning you will see a lot of trigrams that include spaces and punctuation characters.

By looking at the language model probabilities in this file, can you say anything about the kind of estimation method that was used? You can't be completely certain without seeing the corpus, but based on your knowledge of English and of different estimation methods, you should be able to narrow down the possibilities.

Write a paragraph or so explaining which method(s) might have been used, and say what evidence your guess is based on.

Note: We designed our LM file for simplicity, not space efficiency. Real LM files are far more efficient but far harder to read!

4. (30 marks) Now write code to build your own trigram character language model. You will need to read in a training file, collect counts, estimate probabilities, and write the model probabilities into a file.

In your report, describe the method you used to estimate the probabilities, and briefly justify why you used that method. Your description should include, but not be limited to, an equation. You should also explain any simplifying assumptions you have made. Please see the Marking Guide below for hints about what is considered a good explanation; most pairs should only need 1-2 paragraphs here. Your report need **not** describe how you

implemented the method in your code, but please do include comments in the code itself indicating what you did.

Also include in your report an excerpt of the language model for English, displaying all n -grams and their probability with the two-character history ng . Explain what you would expect to find and why. Do the results match your expectations?

Hint 1: In deciding on an estimation method, consider what you will need to do in the rest of the assignment, and make sure your method will work for what you need to do. You can do well on this question by implementing a simple method, as long as it works for the other tasks you will need to do (see also Section 3).

Hint 2: Consider what kinds of data structures will be useful for storing your counts and probabilities. For example, you might want to use a dictionary with an entry for each possible history, where the value stored is the distribution over next characters given that history. (You'll then need to decide what kind of structure that distribution is!)

5. (10 marks) Write a function or method called **generate_from_LM** that uses a language model to generate random output sequences. That is, the sequences should be generated according to the probabilities in the language model.

Using this code, generate 300 characters of random output for each of two different models: (a) the model you estimated from the English training data, and (b) the model in **model-br.en** (you will need to write code to read the model in from the file).

Include the output from each model in your report. Comment *briefly* (max 1 paragraph) on any differences you see, and what seems likely to have caused them.

Hint: You can do most of this question before you have implemented your own model.

6. (15 marks) Extend your program to read in a test document and compute (and output) its perplexity under the estimated language models.

What is the perplexity of the test document we provided under each of the three language models you estimated from the training documents? How can you use these numbers to guess the language of the test document without looking at it?

Suppose we ran your program on a new test document and told you the perplexity under your English LM. Would this be enough for you to determine if the document is written in English? Why or why not?

Hint: How will you know if you've implemented perplexity correctly? First, consider what is a reasonable value to expect (look back over the Entropy slides as a starting point). Second, consider how to test your code. One way would be to create a test file with the LM from Question 1, read it in, and use it to compare your program's perplexity result to your hand computation from Question 1. Try coming up with other tests as well.

7. (15 marks) Extend your work in some other way. Explain briefly what additional question(s) you decided to address, and what results you got.

As noted in the marking guidelines (below), if you do a good job on questions 1-6, with only minor errors, you will still be getting an A. This question should be considered **extra**,

beyond what is expected of most students. For example, implementing more complex methods, comparing different methods or models, or making a more in-depth exploration of whichever method you choose. The time it takes to answer this question is not necessarily proportional to how many extra marks you'll get, though it will help you learn more, **if** you already fully understand the previous parts.

WARNING: you will only get credit for this part if your basic method is correct. This question is intended to challenge students who already feel solid on the basic material in the course, and want to go further. **Do not** attempt this question unless you have completed all the previous questions, and are sure that you've done a good job on those. Also, **do not** attempt this question if you have personally spent 16 or more hours on the assignment already. The maximum number of marks awarded for this question last year was 10, or 1% of your overall course mark. Unless you really whizzed through the earlier questions, please stop now and spend your time and energy revising other course materials or getting more sleep. Both are likely to have a much bigger impact on your final mark.

3 Marking

3.1 The British marking scale

Assignments will be marked on the usual British scale:

Numeric mark	Equivalent letter grade	Approximate meaning
< 40	F	fail
40-49	D	poor
50-59	C	acceptable
60-69	B	good
70-79	A3	very good/distinction
80-100	A1, A2	excellent/outstanding/high distinction

Please note the University specifications for marks above 70:

A1 90-100 Often faultless. The work is well beyond what is expected for the level of study.

A2 80-89 A truly professional piece of scholarship, often with an absence of errors.

As 'A3' but shows (depending upon the item of assessment): significant personal insight / creativity / originality and / or extra depth and academic maturity in the elements of assessment.

A3 70-79

Knowledge: Comprehensive range of up-to-date material handled in a professional way.

Understanding/handling of key concepts: Shows a command of the subject and current theory.

Focus on the subject: Clear and analytical; fully explores the subject.

Critical analysis and discussion: Shows evidence of serious thought in critically evaluating and integrating the evidenced and ideas. Deals confidently with the complexities and subtleties of the arguments. Shows elements of personal insight / creativity / originality.

Structure: Clear and coherent showing logical, ordered thought.

Presentation: Clear and professional with few, relatively minor flaws. Accurate referencing.

Figures and tables well constructed and accurate. Good standard of spelling and grammar.

3.2 What are we looking for?

Write your answers to each question separately (with question number) in your report. It is possible that different people may mark different questions, so please try to make each answer stand alone. If need be you can refer back or forward to your answers to other questions, but try to keep answers as self-contained as possible.

To do well on this assignment (70-79), focus on doing the following:

- Complete at least tasks 1-6 and provide working solutions, even if they are simple. You can mention in one or two sentences if you think there is a better solution and why, but you will not get extra marks for a detailed explanation of a complex method you did not implement. You must implement the solutions yourself, and should not use existing language modelling toolkits. You may use other support code provided you cite the source.
- Answer the questions correctly, clearly and concisely.
- Include brief but clear comments in your code indicating what each function does, including what its arguments and return values are. See the comments inside triple quotes in the lab code for examples—you do not need to comment nearly every line as we did, but a few well-placed comments explaining what each block is doing can be useful.
- Describe the results of your work, not a history of it. So, avoid descriptions like this one:

“We implemented [Method 1], which is formulated as [equation], because [reason]. [Considerably more explanation of Method 1 here]. However, we then realized that [Method 1] wouldn’t work [reasons here], so instead we ended up using [Method 2]. [Explanation of Method 2].”

This is not appropriate style for a scientific report, and it is confusing. The reader initially thinks you used one method, but then later discovers you actually did something else. The reader (whether a marker reading your assignment or a scientist reading a research paper) does not want to know the history of your ideas. The reader wants to know what you actually ended up doing, and why. The above explanation should be rewritten as follows:

“We used [Method 2] to implement probability estimation in this assignment, because [Method 1] does not work [reasons here], and [other reasons for using Method 2]. [Explanation of Method 2].”

- Explain the terms in your equations, and say what they mean within the context of this report (if appropriate, give actual values). When you are doing an assignment, this shows the marker that you actually understand the equation yourself. When you are writing a longer paper or thesis, in many cases the reader will not actually know ahead of time what the terms in your equations mean so explaining them also helps the reader understand.

We will **not** be assigning any marks based on the quality or style of your code (provided it works), since many students are still just learning. However we do insist that you turn in your code (a) to discourage plagiarism, and (b) since it may help us provide useful feedback.