

LAC: Language and the Computer

Computational Linguistics with Python

Introduction, Organization

Overview of NLP, Main Issues

Francis Bond


Division of Asian Studies

`bond@ieee.org`

Introduction

- Self Introduction
- Administrivia
- Why use computers in linguistics
- What this course is (and isn't)
- Texts and Words; Natural Language Understanding (HTML)
- Tutorial
 - Getting to know each other (what do you want)
 - Getting to know the NLTK

Self Introduction

- BA in Japanese and Mathematics
- BEng in Power and Control 
- PhD on “Determiners and Number in English contrasted with Japanese, as exemplified in Machine Translation”
- 1991-2006 NTT (Nippon Telegraph and Telephone)
 - Japanese - English/Malay Machine Translation
 - Japanese corpus, grammar and ontology (Hinoki)
- 2006-2009 NICT (National Inst. for Info. and Comm. Technology)
 - Japanese - English/Chinese Machine Translation
 - Japanese WordNet

-
- 2009-2022 NTU (Nanyang Technological University, Singapore)
 - Wordnets: Chinese, Malay, Indonesian, ...
 - Multilingual Wordnet
 - NTU Multilingual Corpus
 - 2022- UPOL (Palacký University Olomouc)
 - Wordnets
 - LLMs
 - ...

Assessment

- *Continuous Assessment* (100%)
 - Assignment One (30%)
 - Assignment Two (30%) Group work
 - Final In-Class On-Line Open-Book Programming Challenge (30%)
 - * Individual work (one program each)
 - * In class 4-6 hour exam
 - Participation (10%)
 - * Every week there will be short problems to do in class

Extra Credit

- If you submit a patch¹ that gets accepted to the NLTK or another tool we use
 - you can get 1-5% extra credit (depending on the size/difficulty)
Mark $n \propto 10^{n-1}$ lines of code/documentation
 - You can't go over 100%
- A patch can involve
 - fixing a bug in code
 - extending the code with new capabilities
 - fixing a bug in or extending documentation
 - * spelling error
 - * rewording
 - * translating

¹a short set of commands to correct a bug in a computer program

Why use Computers in Linguistics?

- Linguistics without computers is like taking a walk (or a long hard hike)
 - It can be very pleasant
 - You can see a lot of details
 - There is only so much ground you can cover
- Using a software tool is like catching the MRT
 - Very efficient for set routes
 - You have to adapt to it
 - Hard to customize
- Programming is like driving a car
 - It is expensive to start off (you have to learn!)
 - You are free to go where you want to

The goal of this course

To learn enough programming
to flexibly analyze data
and then do something with it

- The language will be Python
- We will use the NLTK toolkit
- You will be able to write your own programs by the end

LAC Prerequisites

➤ A little linguistic knowledge

- You know what a word is
- You know what a part of speech is
- You know what a parse tree is

If you don't know these, you will have to do a little background reading

➤ No computational knowledge

- You have to be ready to learn
- ⊗ If you are a very experienced Python programmer, then you will not learn so much
- If you can program, but in a different framework, then you will learn something new, and I will expect more from your code

What this course isn't

- We won't be learning how to build cars
 - this is the prerequisite for further NLP courses
 - ... but we won't be writing taggers and parsers yet
- Just an introduction to Python
 - We will be motivated by NLP
- Very easy (this is a feature, not a bug)
 - but it is very fun
 - I want you to learn enough to be useful to you

the Three Virtues of a Programmer

Laziness The quality that makes you go to great effort to reduce overall energy expenditure. It makes you write labor-saving programs that other people will find useful, and document what you wrote so you don't have to answer so many questions about it.

Impatience The anger you feel when the computer is being lazy. This makes you write programs that don't just react to your needs, but actually anticipate them. Or at least pretend to.

Hubris The quality that makes you write (and maintain) programs that other people won't want to say bad things about.

Larry Wall, Tom Christiansen, Randal L. Schwartz and Stephen Potter (1996) *Programming Perl* 2nd Ed, O'Reilly

But enough about me

➤ Language Poll

➤ Natural

- * Mandarin
- * Bahasa Malay
- * Tamil

...

➤ Artificial

- * PERL
- * C/C++
- * Java

...

Readings

- Core readings are all from the NLTK book.
- Supplementary readings may be assigned, but all the sources will be on-line.
- All Wikipedia articles have been checked by me, and I will watch them for changes. (extend the web of trust)
- **You must read the material before class**
I will assume that you have done so
 - You get good at programming by programming
 - ⇒ that is how we should spend our time

Acknowledgements

- Thanks to Graham Wilcock for the inspiration for this course, and permission to adapt his course notes.
- Thanks to Steven Bird, Ewan Klein and Edward Loper for releasing the NLTK
- Thanks to Guido van Rossum, Python Benevolent Dictator for Life (BDFL).
- Definitions from WordNet 3.0