Language & Technology

Wrapping Up

Alëna Aksënova & Aniello De Santo

Stony Brook University
alena.aksenova@stonybrook.edu
aniello.desanto@stonybrook.edu

TL;DR — Be Skeptical

- Technological advances can bring much good but...
- Humans overestimate technology.
  Loebner Prize/Turing test
- Don't buy the hype.
  deep learning, artificial intelligence, culturomics, neuroscientism, ...
- Technological advances also come with downsides.
  social biases, rising job requirements, mass surveillance, ...

The State-of-the-Art in Language Technology

- Current technologies often involve **n-grams models**.
  - word completion/prediction
  - stylistic analysis
  - OCR
  - spell checking
  - web search
  - word meaning
  - machine translation
- Frequency information from corpora/tree banks is used to add **probabilities** to the models.
- Caveat: in this class we have (mostly) ignored most modern trends
  $\Rightarrow$ Neural Networks and Deep Learning

The State-of-the-Art in Language Technology

- Current technologies often involve **n-grams models**.
  - word completion/prediction
  - stylistic analysis
  - OCR
  - spell checking
  - web search
  - word meaning
  - machine translation
- Frequency information from corpora/tree banks is used to add **probabilities** to the models.
- Caveat: in this class we have (mostly) ignored most modern trends
  $\Rightarrow$ Neural Networks and Deep Learning

Language is Complex

- Language comes easy to humans,
  but it is **incredibly complex**.
  - culture-specific rules of turn taking
  - building sentence meaning from word meaning
  - tons of variation across languages and dialects
  - hidden structure of sentences
  - how that hidden structure is inferred
- We have to deal with this complexity.
- Probabilities won't cut it because of **Zipf's law**.
- (Psycho)Linguistic insights will help us evaluate/improve existing technologies.

Why There is no Hope for bare N-Gram Models

n-grams can only consider local information, but language often uses **non-local information**.

Example

- Suppose the user is typing *May I of* on their phone.
- Do we suggest *off* or *offer* as the best word completion?
- **Bigram Frequencies**
  I offer **0.00014%**
  I off 0.00001%
- But what if the user had typed *May I quickly of*?
- **Bigram Frequencies**
  quickly offer 0.00000025%
  quickly off **0.000002%**

Scaling Up Doesn't Help

- Okay, so bigrams don't work, but trigrams would:
  I quickly offer is more frequent than *I quickly off*
- But what if the user had typed

May I really quickly of	4-grams!
May I really really quickly of	5-grams!
⋮	

- This isn't feasible, we quickly run into **data sparsity issues**.

Non-Local Dependencies

- Dependencies in language aren't limited to a fixed number n of words, they can span arbitrary distances:
  - *The man that I think Bill thinks I think . . . Bill punched seems angry.*
  - *The men that I think Bill thinks I think . . . Bill punched seem angry.*

The Linguistic Moral of the Story

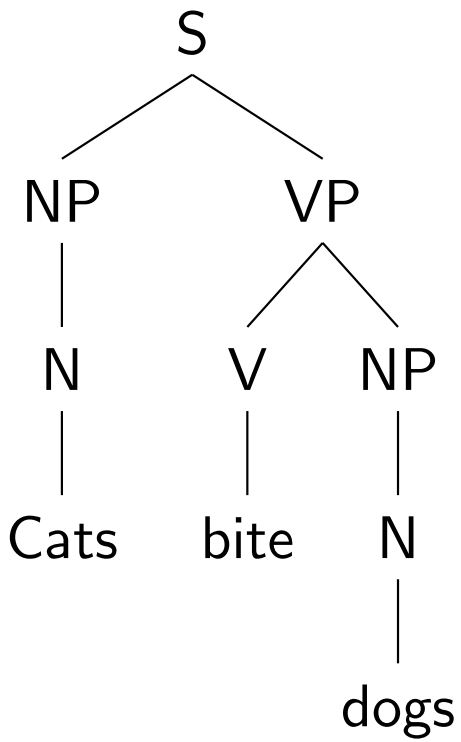
- n-gram models will never work perfectly, not even if we had unlimited resources.
- **Reason:** Linguistic dependencies can be unbounded and thus span over more than n words.

How Complex are Sentences?

- We have seen that n-**gram models are insufficient**.
- But what should we use instead?

Sentences Have Hidden Structure

- Linguists have known for a long time that sentences are not just sequences of words.
- They involve a lot of hidden structure $\Rightarrow$ **trees!**



Some Evidence for Hidden Structure

- Some but not all strings of words can be moved around.

(1) a. It is old ugly dogs that cats bite __.
b. * It is dogs that cats bite old ugly __.

- Some but not all strings can be coordinated.

(2) a. Cats bite old ugly dogs and scratch young cute dogs.
b. * Cats bite old ugly and scratch young cute dogs.

- Verbal agreement is not determined by closest noun.

(3) a. The woman is tired.
b. The women are tired.
c. * The women that buried the woman is tired.
d. The women that buried the woman are tired.

Even More Evidence for Hidden Structure

- Questions front the **structurally highest auxiliary**,
  not the first one in the string.

(4) a. The man who is looking for a job is exasperated.

b. * Is the man who __ looking for a job is exasperated?

c. Is the man who is looking for a job __ exasperated?

- lots of evidence from experiments
  self-paced reading, eye tracking, ERP, fMRI

Related Linguistics Courses

- LIN 101 *Human Language*
- LIN 110 *Anatomy of English Words*
- LIN 200 *Language in the USA (online!)*
- as background for sentence models:
  LIN 311 *Syntax*
  LIN 346 *Language & Meaning*
  LIN 347 *Pragmatics*
- as background for speech recognition:
  LIN 201 *Phonetics*
  LIN 301 *Phonology*

Possible Courses Spring 2020 (taught by yours truly)

- LIN 425 *Computational Psycholinguistics*
- LIN 260 *Language & Mind*

Related Linguistics Courses

- LIN 101 *Human Language*
- LIN 110 *Anatomy of English Words*
- LIN 200 *Language in the USA (online!)*
- as background for sentence models:
  LIN 311 *Syntax*
  LIN 346 *Language & Meaning*
  LIN 347 *Pragmatics*
- as background for speech recognition:
  LIN 201 *Phonetics*
  LIN 301 *Phonology*

Possible Courses Spring 2020 (taught by yours truly)

- LIN 425 *Computational Psycholinguistics*
- LIN 260 *Language & Mind*

Computational-oriented Faculty in Linguistics

Jeff Heinz

Thomas Graf

Jiwon Yun

Check out our lab and brand new M.A. Program.

Computational-oriented Faculty in Linguistics

Jeff Heinz

Thomas Graf

Jiwon Yun

Check out our lab and brand new M.A. Program.

Computational Opportunities in Linguistics

1. **LIN 488 UG Teaching Practicum (Fall 2019)**
   Be a UG TA for LIN 120! If interested, shoot me an email by the end of May. *Prerequisite*: A decent grade in this class.
2. **LIN 488 Internship (Fall 2019)**
   I might need research assistants for my dissertation project. Get in touch if interested.
   Prerequisite: LIN 311 Syntax, or talk to me.
3. **LIN 628 - Computational Syntax (Fall 2019)**
   Mathematical models of sentence structure (it's a grad class).
   Prerequisite: LIN 311 Syntax, or talk to Thomas Graf.
4. **LIN 220 Computational Linguistics (Spring 2020)**
   A follow up to this class. Taught by me.

Two Course Recommendations Outside Linguistics

1. SOC 330 *Media and Society*
   (Jason J. Jones; no programming, but a bit of computational sociology and big data)
2. AMS 103 *Applied Math in Technology*
   (Matthew Reuter; includes a live cookie baking session)

Programming Concepts

- **Universal**
  - strings, integers, lists
  - variables
  - print, input
  - conditionals
  - while loops
  - for loops
  - custom functions
  - regular expressions
- **Python-Specific**
  - counters
  - positions and slices
  - built-in functions like `len`, `str.lower`, `...`

Sticking with Python

- You invested lots of time and energy on picking up some programming skills. They **should not go to waste**!
- The most important thing: find yourself a **project**!

Project suggestions

- Computational analysis in a book report
- Automatic menu generator for New York hipster restaurants
- Facebook bot to like all your friends' posts and leave a short reply
- Script to delete all unread emails that are older than 30 days
- Reminder to take a break after 2h in front of the screen
- Automating something you frequently do manually

Some Project Ideas from Reddit

Check out the Reddit thread:

> *Our department keeps a calendar of activities in excel. I use python to import that spreadsheet, break the activities into itemized tasks, put due dates on each task, and spit out those tasks for each teammate.*
>
> *I stitched together a video encoding script, a mp3 tag reading script and a YouTube uploading script to automatically upload all of the old music I'd recorded to YouTube. Saved me 100s of hours of manual work.*

Some Project Ideas from Reddit

Check out the Reddit thread:

> *Our department keeps a calendar of activities in excel. I use python to import that spreadsheet, break the activities into itemized tasks, put due dates on each task, and spit out those tasks for each teammate.*

> *I stitched together a video encoding script, a mp3 tag reading script and a YouTube uploading script to automatically upload all of the old music I'd recorded to YouTube. Saved me 100s of hours of manual work.*

Some Project Ideas from Reddit

Check out the Reddit thread:

> *Our department keeps a calendar of activities in excel. I use python to import that spreadsheet, break the activities into itemized tasks, put due dates on each task, and spit out those tasks for each teammate.*
>
> *I stitched together a video encoding script, a mp3 tag reading script and a YouTube uploading script to automatically upload all of the old music I'd recorded to YouTube. Saved me 100s of hours of manual work.*

Some Project Ideas from Reddit [cont.]

This is a pretty silly usage, but the autopy package is really great for making auto-clickers for various web-games.

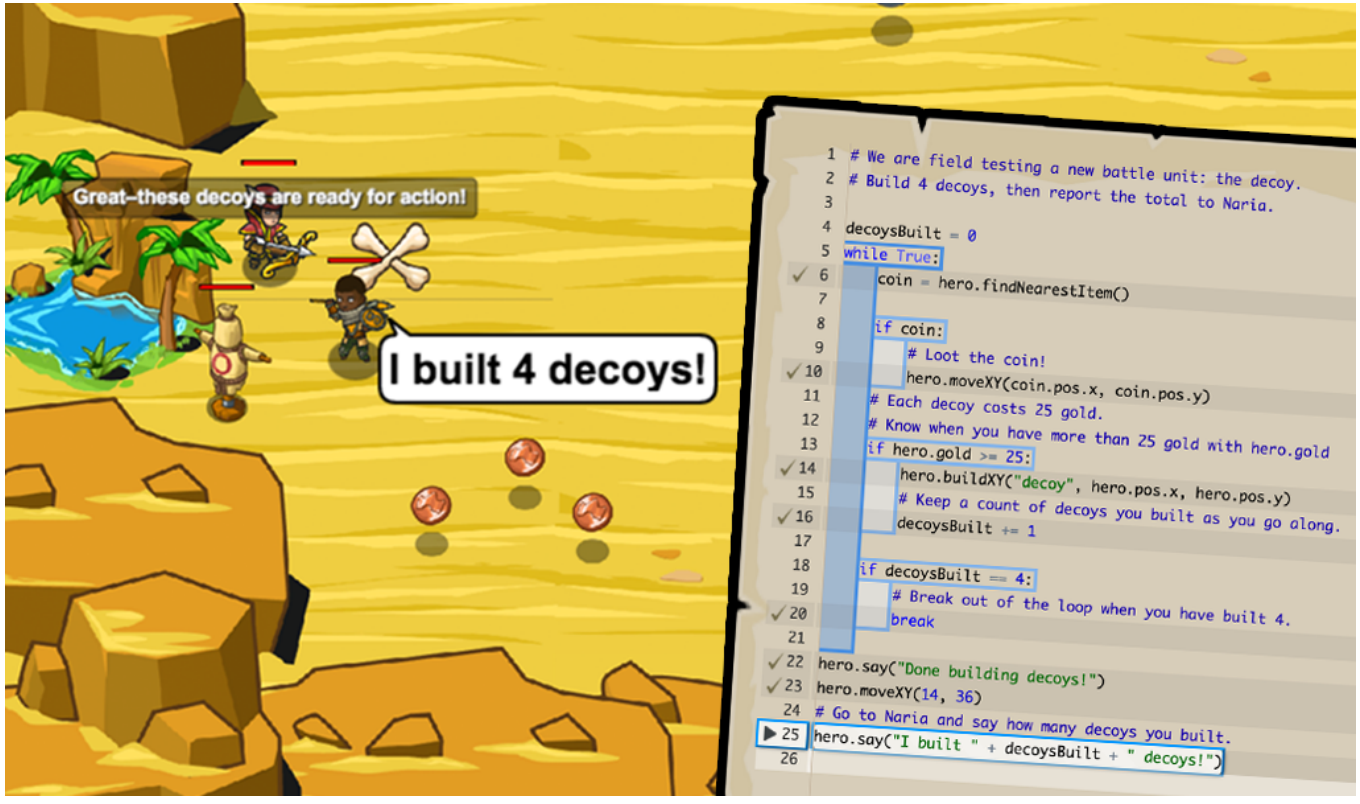
I have a script that checks the Powerball and Megamillions jackpot size and calculates the expectation value of a play. If either is positive, it sends me a text to buy a ticket.

Not complete yet, but I am using python on a RasPi to automate the temperature, humidity, and lighting for my two pythons' cages

Other Fun Stuff with Python

- Python coding for *Minecraft*
  http://www.instructables.com/id/Python-coding-for-Minecraft/
- Modding *The Sims 4*
  http://simswiki.info/wiki.php?title=Tutorials:TS4_General_Modding
- Creating hi-res texture mods (e.g. for Resident Evil 4 HD)
  https://youtu.be/u_8I3vaVrfE
- *Gray Hat Python* book on hacking with Python

Code Combat & Code Wars



If you liked Code Combat, check out Code Wars!

How much Python Do I Know?

- We covered enough Python for 90% of all problems.
- But Python can do a lot more:
  - strides
  - dictionaries (of which Counters are a special case)
  - unpacking
  - zipping
  - recursive functions
  - object oriented programming with classes
  - generators
  - decorators
- And there's many powerful libraries:
  - pyplot
  - pandas
  - ntlk
  - scipy
  - scikit-learn
- Don't worry about any of this for now.

Additional Teaching Materials

If at some point you need to know more Python for your project, check out some of these:

- The remainder of our textbook
  Automate the Boring Stuff With Python
- The programming historian
  http://programminghistorian.org/
- Python programming for the humanities
  http://www.karsdorp.io/python-course/
- Natural language processing with Python
  http://www.nltk.org/book/
- Mark Summerfield's book *Programming in Python 3*
- For more theory, check out the book *Language and Computers*

Don't do it Alone!

- Form learning/reading groups with your classmates.
- Check out SBU undergraduate clubs and the events they host.
- Join online communities.
- Need advice? Send me an email or come to my office hours!

The End (?)