108-NLP @ NCTU

e-Lab 2

Boaz

Good morning!

Please make sure your microphone/video is OFF

We'll take a break or two 🕘🕚🕛🕐

And now… let's try something new!

Online Polling

Go to PollEV.com, enter NLPFUN (or PollEV.com/NLPFUN)

Outline

Lab 1 re-cap

NLP libraries

Demo

Assignment

Python is not JavaScript :)

https://docs.python-guide.org/

Writing “Pythonic” code:

“Exploiting the features of the Python language to produce code that is clear, concise and maintainable.”

https://docs.python-guide.org/writing/style/

Use StackOverflow

Google “**<question>** in Python”

Even better:

Google: “site:stackoverflow.com **<question>** in Python”

Lab 1 Tips

```
1 def get_stopwords(document):
2   df = pd.read_csv('https://bit.ly/nlp-stopwo
3   stopwords = df.i.to_list()
4   stopwords.append('i')
5   return stopwords
```

```
i
me
my
myself
we
our
ours
ourselves
you
you're
you've
you'll
you'd
your
yours
yourself
yourselves
he
him
his
himself
she
she's
her
hers
herself
it
it's
its
itself
they
them
```

Lab 1 Tips

```
15 def tokenize(document):
16     words = document.split(' ')
17     words = ConTo_lowerCase(words)
18     words = RemoveStopWord(words)
19     words = RemovePunctuation(words)
20     return words
21
```

Document: He said “I don’t want it”
After split: He, said, “I, don’t, want, it”
After stopwork removal: said, “I, want, it”
After punctuation removal: said, I, want, it

Lab 1 Tips

```
def tokenize(document):
  stopwords = pd.read_csv('https://bit.ly/nlp-stopwords', header=None)[0].to_list()
  tokens = re.split('\W+', document.lower())
  tokens = [token for token in tokens if (token not in stopwords and token != '')]
  return tokens
```

```
def tokenize(document):
  try:
    tokenize.stopwords
  except AttributeError:
    tokenize.stopwords = pd.read_csv('https://bit.ly/nlp-stopwords', header=None)[0].to_list()
  tokens = re.split('\W+', document.lower())
  tokens = [token for token in tokens if (token not in tokenize.stopwords and token != '')]
  return tokens
```

Does it make sense?

```
> "GM’s OnStar, IBM’s Watson combine to market brands to drivers"

* "GM’s OnStar, IBM’s Watson combine to market brands to drivers" (1.0)
* "Uber partners with GM’s Maven car-sharing program" (0.524959268716631)
* "Cyber Monday sales biggest online shopping day in U.S. history" (0.5043329526689814)
* "Snap’s Spectacles make their debut in wacky vending machine" (0.497007091795403)
* "Cook ups Apple support for fight against AIDS" (0.4963156604307634)
```

```
* GM’s OnStar, IBM’s Watson combine to market brands to drivers
  > GM’s OnStar, IBM’s Watson combine to market brands to drivers (1.0000)
  > U.S. proposes requiring vehicles to ’talk’ to each other to avoid crashes (0.5472)
  > HERE, automakers team up to share data on traffic conditions (0.5416)
  > FCC chief unveils scaled-back business data reforms (0.4595)
  > Fiat Chrysler recalls 1.1 million cars, SUVs for rollaway issue (0.4240)
```

- lab2-<studentID>
- lab2_<studentID>

- hyphen: join words (or parts of words): He is a well-known professor

– en dash: “Our class is from 9:00–10:00”, “Non–English newspaper”

— em dash: “The new student—who came from Europe—entered the room.”

https://en.wikipedia.org/wiki/Dash

Lab 1 Solution

Code From Scratch: A TF-IDF-based News Recommendation System in Python

Boaz Shmueli
Apr 8 · 2 min read

Warning: for educational purposes only :)

https://medium.com/@shmueli/a-tf-idf-based-news-recommendation-system-from-scratch-75e73c2acc63?source=friends_link&sk=6c276a4c5e687aabc7870a4ba4fca1e5

Libraries

NLTK (Natural Language ToolKit)

A Python library for Natural Language Processing

- ✓ **Tokenization**
- ✓ Classification
- ✓ Stemming
- ✓ Tagging
- ✓ Parsing
- ✓ Semantic reasoning

https://www.nltk.org/

Source: https://github.com/nltk/nltk/

Demo

NLTK parts-of-speech

```
POS tag list:

CC      coordinating conjunction
CD      cardinal digit
DT      determiner
EX      existential there (like: "there is" ... think of it like "there exists")
FW      foreign word
IN      preposition/subordinating conjunction
JJ      adjective       'big'
JJR     adjective, comparative  'bigger'
JJS     adjective, superlative  'biggest'
LS      list marker     1)
MD      modal   could, will
NN      noun, singular 'desk'
NNS     noun plural     'desks'
NNP     proper noun, singular   'Harrison'
NNPS    proper noun, plural     'Americans'
PDT     predeterminer   'all the kids'
POS     possessive ending       parent\'s
PRP     personal pronoun        I, he, she
PRP$    possessive pronoun      my, his, hers
RB      adverb  very, silently,
RBR     adverb, comparative     better
RBS     adverb, superlative     best
RP      particle        give up
TO      to      go 'to' the store.
UH      interjection    errrrrrrrm
VB      verb, base form take
VBD     verb, past tense        took
VBG     verb, gerund/present participle taking
VBN     verb, past participle   taken
VBP     verb, sing. present, non-3d     take
VBZ     verb, 3rd person sing. present  takes
WDT     wh-determiner   which
WP      wh-pronoun      who, what
WP$     possessive wh-pronoun   whose
WRB     wh-abverb       where, when
```

```
LS      list marker     1)
MD      modal   could, will
NN      noun, singular 'desk'
NNS     noun plural     'desks'
NNP     proper noun, singular   'Harrison'
NNPS    proper noun, plural     'Americans'
PDT     predeterminer   'all the kids'
```

SpaCy

- Tokenization
- Named entity recognition
- Pre-trained word vectors
- Part-of-speech tagging
- Labelled dependency parsing
- Syntax-driven sentence segmentation
- Text classification

https://spacy.io

Demo

Assignment

Use the same Colab notebook for both part 1 and part 2

1. Calculate and print the
   - ✓ **5 most frequent 2-grams**
   - ✓ from the **Reuters news dataset** (content) available at bit.ly/nlp-reuters
   - ✓ where both tokens are PROPER NOUNS
   - ✓ using NLTK word_tokenize, POS tagger
   - ✓ no need to remove punctuation, no need to remove stop words
2. Calculate and print the
   - ✓ **5 most similar articles to seed_id = <student_ID> % 1000**
   - ✓ from the **Buzzfeed News dataset** (content), available at bit.ly/nlp-buzzfeed
   - ✓ use the **SpaCy POS tagger** and **tokenizer** (**en_core_web_sm** model)
   - ✓ use each token's combined lemma + POS (e.g., "give_VERB" or ("give, "VERB)) as the *term* for input to TF-IDF
   - ✓ remove stopwords (can use token.is_stop)
   - ✓ use the 512 most frequent **terms as features**
   - ✓ if desired, you can use TF-IDF code from Medium article, or can use your own TF-IDF code
   - ✓ you can use a library function to compute **cosine_similarity**

SOP for assignment submission

- [ ] Open your Python notebook
- [ ] Rename your notebook to lab2-<studentID>.ipynb
- [ ] “Restart and run all”
- [ ] Make sure the output is correct
- [ ] Download the Python file (File, Download .py)
- [ ] Share the Google colab using “Get shareable link”; copy link
- [ ] Submit both link and file in e3:
  - [ ] Paste link into e3 (link starts with https://colab.research.google.com/…)
  - [ ] Upload file into e3 (file name is lab2-<studentID>.py)