ΔΙΑΧΕΙΡΙΣΗ ΠΕΡΙΕΧΟΜΕΝΟΥ ΠΑΓΚΟΣΜΙΟΥ ΙΣΤΟΥ ΚΑΙ ΓΛΩΣΣΙΚΑ ΕΡΓΑΛΕΙΑ

Τεχνικές NLP – Σχεδιαστικά Θέματα

Natural Language Processing

- Επεξεργασία δεδομένων σε φυσική γλώσσα
  - → Κατανόηση φυσικής γλώσσας από τη μηχανή
  - → Παραγωγή φυσικής γλώσσας από τη μηχανή
- Υποπεδίο της Τεχνητής Νοημοσύνης
  - ευφυείς μηχανές => κατανοούν την ανθρώπινη γλώσσα
  - AI-complete ή AI-hard πρόβλημα
- Οι περισσότερες NLP τεχνικές βασίζονται σε machine learning.

Μοντελοποίηση της γνώσης

- Machine Learning
  - Στηρίζεται σε μοντελοποιημένη εμπειρική γνώση για
    - → Αναγνώριση προτύπων
    - → Εκμάθηση κανόνων και δυνατότητα λήψης αποφάσεων από τη μηχανή
- Πηγές εμπειρικής γνώσης για NLP:
  - Corpora
  - Λεξικά
  - Σημασιολογικά Δίκτυα
  - κλπ.

Web-focused NLP

- Κίνητρο:
  - Διευκόλυνση στην πρόσβαση και τον εντοπισμό πληροφορίας στο δίκτυο.
- Προκλήσεις:
  - Τεράστιος όγκος πληροφορίας
  - Ποκιλομορφία των δεδομένων
  - Πολλές γλώσσες, τρόποι γραφής, είδη κειμένων κλπ.
- Μηχανές Αναζήτησης:
  - Τι εμπειρία χρειάζεται για να τις χρησιμοποιήσει κανείς?
    - Διατύπωση του ερωτήματος με τρόπο φιλικό στη μηχανή και όχι φιλικό ως προς τον άνθρωπο

Τι χρειάζεται για μια απλή ερώτηση?

- «Ποιό κινητό με συμφέρει να αγοράσω αν έχω διαθέσιμα 100 ευρώ;»
- Για να απαντηθεί αυτόματα από το web θα χρειαζόταν πχ:
  - Ανάκτηση των σχετικών με μοντέλα κινητών σελίδων
    → Information Retrieval techniques
  - Επεξεργασία των σελίδων για εξαγωγή του μοντέλου και της τιμής
    → Html Parsing, Information Extraction
  - Εξαγωγή των σχολίων και των προτάσεων που αναφέρονται σε κάθε μοντέλο
    → Text Mining, Syntactic Parsing, Anaphora Resolution
  - Υπολογισμός μετρικής ποιότητας ανάλογα με τα σχόλια
    → Sentiment Analysis
  - Ταξινόμηση και επιστροφή αποτελεσμάτων
- Χρειάζονται NLP μέθοδοι

Language Processing Technologies

- Τεχνολογίες βασισμένες στο NLP:
  - Word Sense Disambiguation
  - Syntactic Parsing & Reference Resolution
  - Machine Translation
  - Spoken Language Systems
  - Opinion Mining
  - Summarization
- Υπαρκτές τεχνολογίες και αντίστοιχα εργαλεία
  - (με περιθώρια βελτίωσης φυσικά…)

Word Sense Disambiguation

- Αμφισημία
  - Μέρους του λόγου
  - Σημασιολογική
  - Πραγματολογική
- Disambiguation (αποσαφήνιση)
  - Σύνολο τεχνικών επίλυσης αμφισημίας
    - Τι θα προτείνατε εσείς σε κάθε περίπτωση;

Επίλυση Αμφισημίας

- Part-Of-Speech tagging
- Σε τι βασίζεται ο tagger;

```
>>> from nltk.corpus import brown
>>> ambiguous=[(w,t) for w,t in brown.tagged_words() if w=='answer']
>>> len(ambiguous)
149
>>> unique=set(ambiguous)
>>> len(unique)
3
>>> unique
set([('answer', 'NN'), ('answer', 'NN-HL'), ('answer', 'VB')])
>>> noun_sents=[s for s in brown.tagged_sents() if ('answer','NN')in s]
>>> len(noun_sents)
105
>>> noun_sents[0]
[('The', 'AT'), ('answer', 'NN'), ('is', 'BEZ'), ('that', 'CS'), ('it', 'PPS'), (
'was', 'BEDZ'), ('a', 'AT'), ('combination', 'NN'), ('of', 'IN'), ('circumstances
', 'NNS'), ('.', '.')]
>>> verb_sents=[s for s in brown.tagged_sents() if ('answer','VB') in s]
>>> len(verb_sents)
42
>>> verb_sents[0]
[('It', 'PPS'), ('is', 'BEZ'), ('a', 'AT'), ('desperate', 'JJ'), ('effort', 'NN')
, ('to', 'TO'), ('prop', 'VB'), ('up', 'RP'), ('a', 'AT'), ('sagging', 'VBG'), ('
candidate', 'NN'), ('who', 'WPS'), ('has', 'HVZ'), ('proven', 'VBN'), ('he', 'PPS
'), ('cannot', 'MD*'), ('answer', 'VB'), ('any', 'DTI'), ('questions', 'NNS'), ('
about', 'IN'), ('New', 'JJ-TL'), ("Jersey's", 'NP$'), ('problems', 'NNS'), ('.',
'.')]
```

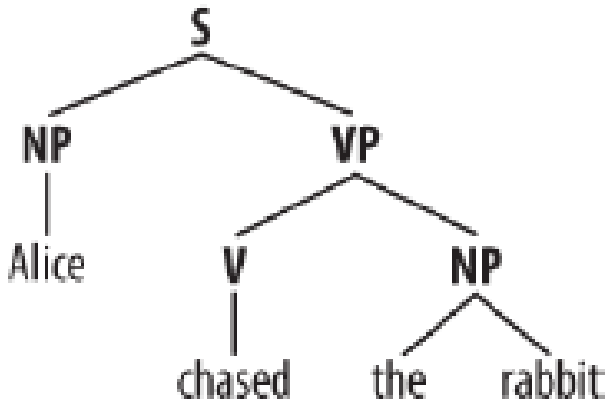
Επίλυση Αμφισημίας

- Word Sense Disambiguation
- Τι είδους πληροφορία χρειάζεται για να πραγματοποιηθεί;

```
>>> from nltk.corpus import wordnet
>>> syns=wordnet.synsets('human')
>>> for syn in syns:
...     print '----'+str(syn.name)+'----'
...     print wordnet.synset(syn.name).lemma_names
...     print wordnet.synset(syn.name).definition
...     print wordnet.synset(syn.name).examples
...
----homo.n.02----
['homo', 'man', 'human_being', 'human']
any living or extinct member of the family Hominidae characterized by superior in
telligence, articulate speech, and erect carriage
[]
----human.a.01----
['human']
characteristic of humanity
['human nature']
----human.a.02----
['human']
relating to a person
['the experiment was conducted on 6 monkeys and 2 human subjects']
----human.a.03----
['human']
having human form or attributes as opposed to those of animals or divine beings
['human beings', 'the human body', 'human kindness', 'human frailty']
```

Syntactic Parsing

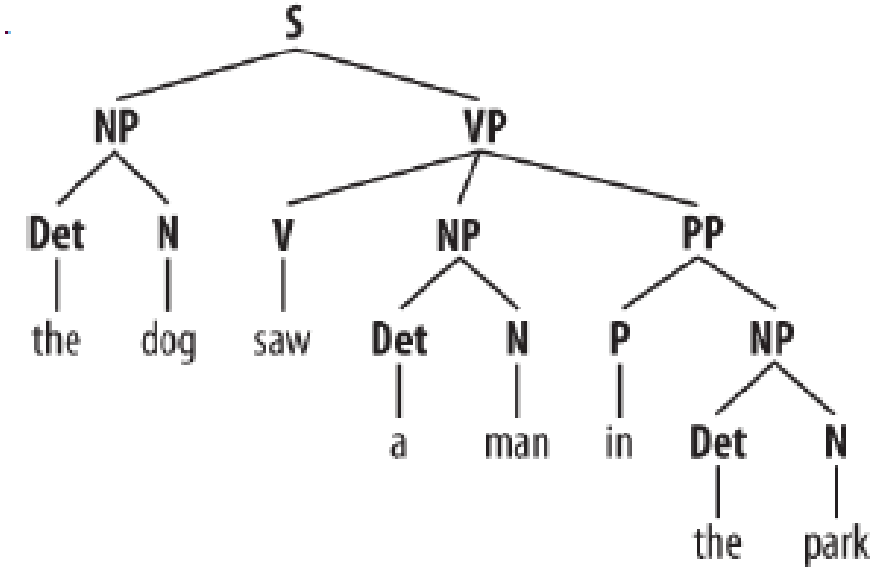
- Αναπαράσταση αποτελέσματος ανάλυσης

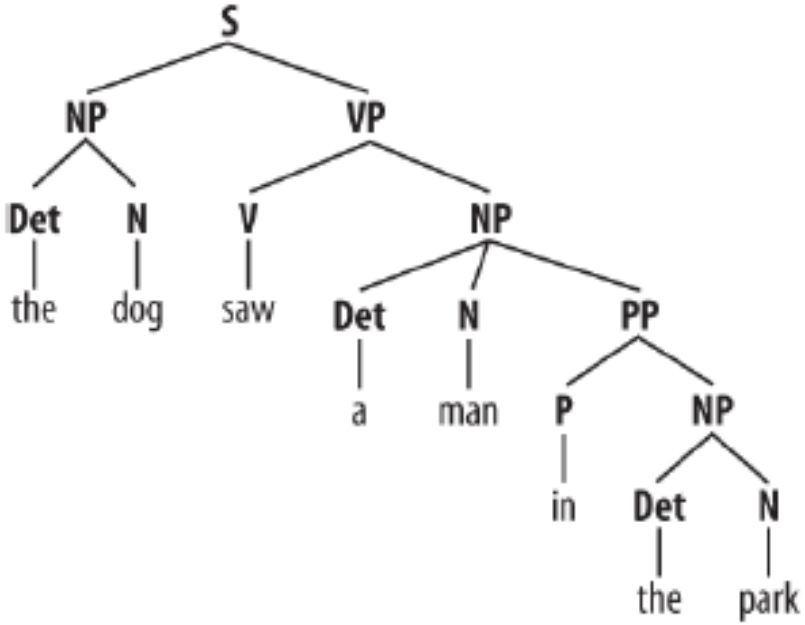



```
(S
   (NP Alice)
   (VP
      (V chased)
      (NP
         (Det the)
         (N rabbit))))
```

Διφορούμενη Σύνταξη

- Μια πρόταση μπορεί να έχει παραπάνω από μια δυνατές συντακτικές αναλύσεις
- Πως επιλέγουμε;





Reference Resolution

- “Owen confessed he is now living in Thierry Henry's shadow.”

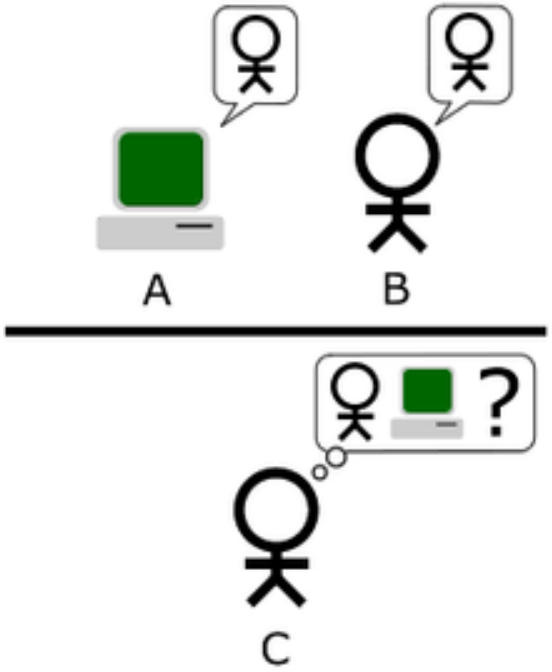
```
<s id="s1">
  <ne id="ne1" gId="nv1" AAcat="pn" AAper="per3" AAnum="sing" AAgen="any">
    <nphead id="AAh1">
      <W Lpos="NNP">Owen</W>
    </nphead>
  </ne>
  <ve id="ve1" gId="nv2">
    <W Lpos="VBD">confessed</W>
  </ve>
  <ne id="ne2" gId="nv3" AAcat="pers-pro" AAper="per3" AAnum="sing" AAgen="masc">
    <nphead id="AAh2">
      <W Lpos="PRP">he</W>
    </nphead>
  </ne>
  <ve id="ve2" gId="nv4">
    <W Lpos="VBZ">is</W>
    <W Lpos="RB">now</W>
    <W Lpos="VBG">living</W>
  </ve>
  <W Lpos="IN">in</W>
  <ne id="ne3" gId="nv5" AAcat="pn" AAper="per3" AAnum="sing" AAgen="neut">
    <mod id="AAm1" AAcat="AApre">
      <W Lpos="NNP">Thierry</W>
      <W Lpos="NNP">Henry</W>
      <W Lpos="POS">'s</W>
    </mod>
    <nphead id="AAh3">
      <W Lpos="NN">shadow</W>
    </nphead>
  </ne>
  <W Lpos=".">.</W>
  <AAante current="ne2" rel="ident">
    <anchor antecedent="ne1" />
  </AAante>
</s>
```

Machine Translation

```
>>> from nltk.book import *
>>> babelize_shell()
NLTK Babelizer: type 'help' for a list of commands.
Babel> help
NLTK Babelizer Commands:
All single-word inputs are commands:
help: this help message
languages: print the list of languages
language: the name of a language to use
Babel> languages
chinese english french german greek italian japanese korean portuguese russian sp
anish
Babel> How are you today?
Babel> german
Babel> run
0> How are you today?
1> Wie gehen Sie heute?
2> How do you go today?
Babel>
```

Spoken Language Systems

- Turing Test: βασική μέθοδος αξιολόγησης συστημάτων τεχνητής νοημοσύνης.
  - Ο C επικοινωνεί γραπτώς με τους A και B.
  - Του ζητάται να αποφασίσει ποιός είναι άνθρωπος και ποιος μηχανή.



Spoken Language Systems

- Alice Bot
- NLTK chatbots

```
>>> nltk.chat.chatbots()
Which chatbot would you like to talk to?
  1: Eliza (psycho-babble)
  2: Iesha (teen anime junky)
  3: Rude (abusive bot)
  4: Suntsu (Chinese sayings)
  5: Zen (gems of wisdom)

Enter a number in the range 1-5: 1
 Therapist
---------
Talk to the program by typing in plain English, using normal upper-
and lower-case letters and punctuation.  Enter "quit" when done.
========================================================================
Hello.  How are you feeling today?
>A little bored.
I see.
>You see what?
We should be discussing you, not me.
>ok
I see.  And what does that tell you?
>What does what tell me?
Why do you ask?
>I didn't understand.
Let's change focus a bit... Tell me about your family.
>I have a brother.
Now that you have  a brother, what will you do next?
>quit
Thank you, that will be $150.  Have a good day!
>>>
```

Σχεδιαστικά Θέματα

Ιδιαιτερότητες της Python &
Απαιτήσεις του πεδίου

Web εφαρμογές

- Ιδιαιτερότητα των εφαρμογών NLP στο web:
  - Μεγάλος όγκος δεδομένων
  - Απρόβλεπτη είσοδος:
    - Δεν ισχύει κανένας σχεδόν κανόνας
    - Τα πάντα μπορούν να πάνε στραβά με την είσοδο (και θα πάνε)
- Κρίσιμα ζητήματα:
  - Διαχείριση μνήμης
  - Επιλογή κατάλληλων δομών δεδομένων για τη φόρτωση στη μνήμη
  - Υλοποίηση αλγορίθμων με ικανοποιητικούς χρόνους για συστήματα πραγματικού χρόνου
  - Error handling

Python Specifics - Reference Types

- Ανάθεση

```
>>> a=[2,5,1,6]
>>> b=a
>>> a.append(4)
```

  - Τι περιέχουν τα a και b?

- Ισότητα

```
>>> a=[3,5,1,2]
>>> b=[3,5,1,2]
>>> a==b
True
>>> a is b
False
>>> id(a)
3077817420L
>>> id(b)
3077902924L
```

  - Γιατί?

Python Specifics – Call by value

- Call by value
  - Αντιγραφή της τιμής των παραμέτρων στη νέα θέση μνήμης
  - Για reference types ουσιαστικά γίνεται αντιγραφή της διεύθυνσης

```
>>> def test(x,l):
...     x+='hi!'
...     l.append(6)
...     print x
...     print l
...
>>> a='world '
>>> lst=[0,2,1,4]
>>> test(a,lst)
world hi!
[0, 2, 1, 4, 6]
>>> a
'world '
>>> lst
[0, 2, 1, 4, 6]
```

Python Specifics -Memory Management

- Garbage collection
- Δεν χρειάζεται καταστροφή των αντικειμένων
- Διατηρείται μετρητής αναφορών στα αντικείμενα
- Το αντικείμενο καταστρέφεται όταν ο μετρητής γίνει 0

```
>>> a=[1,2,3]
>>> b=a
>>> b.append(4)
```

- Η λίστα που περιέχει τα [1,2,3,4] καταστρέφεται όταν βγουν εκτός εμβέλειας οι αναφορές a και b.
- "Things disappear when no one is looking at them" ☺

Space and Time Tradeoff

- Στο σχεδιασμό χρειάζεται οικονομία σε χώρο και χρόνο.
  - Χώρος: πόση από την κύρια μνήμη καταλαμβάνει ο αλγόριθμος στο runtime.
    - Δεν φορτώνονται πολλαπλά αντίγραφα της ίδιας πληροφορίας.
    - Φορτώνουμε μόνο ότι θα χρειαστούμε.
  - Χρόνος: πόσο χρόνο σε σχέση με το μέγεθος του προβλήματος θα κάνει ο αλγόριθμος.
    - Επιλέγουμε τον αλγόριθμο με τη μικρότερη πολυπλοκότητα
    - Επιλέγουμε τις κατάλληλες δομές δεδομένων για τη φόρτωση στη μνήμη.
    - Δεν επαναλαμβάνουμε εργασίες που μπορούν να γίνουν μία φορά!
- Τα δύο μεγέθη λειτουργούν με tradeoff.
- Χαρακτηριστικό παράδειγμα: η δημιουργία ευρετηρίων για μια συλλογή κειμένων.
  - Σπαταλάται χώρος
  - Εξοικονομείται χρόνος αναζήτησης

Επιλογή δομής δεδομένων

- Κριτήρια:
  - Μέγεθος του προβλήματος
  - Πως θα χρησιμοποιηθεί
- Ενέργειες που καθορίζουν το κόστος:
  - Lookup: αναζήτηση/έλεγχος ύπαρξης δεδομένων στη δομή.
  - Insert: εισαγωγή δεδομένων στη δομή.

Συνήθεις κλάσεις

- Οι νεότερες γλώσσες προγραμματισμού προσφέρουν συνήθως δύο βασικές κλάσεις:
  - Dictionary
    - Δυναμικού μεγέθους.
    - Κάθε εγγραφή είναι του τύπου <key, value>.
    - Το κλειδί εισάγεται σε hashtable.
    - Οι εγγραφές δεν ταξινομούνται.
  - List:
    - Δυναμικού μεγέθους.
    - Κάθε εγγραφή είναι μόνο value.
    - Δεν υπάρχει οργάνωση.
    - Υποστηρίζει κλήσεις ταξινόμησης.

Υπέρ-Κατά

- Dictionaries:
  - Πολύ γρήγορα lookups.
  - Πιο αργή προσθήκη εγγραφών (hashing process).
  - Για να ταξινομηθεί πρέπει να εξαχθεί η λίστα των κλειδιών.
- Lists:
  - Αργά lookups. Συνήθως γίνεται σειριακή αναζήτηση.
  - Γρήγορη προσθήκη εγγραφών.
  - Μπορεί να ταξινομηθεί.

Τι χρησιμοποιούμε?

- Κριτήρια:
  - Πόσες θα είναι οι εγγραφές? Πχ για < 10 δεν αξίζει το κόστος του hashing.
  - Πόσο συχνά θα ψάχνουμε στη δομή? Πχ αν σε κάθε insert αντιστοιχεί και ένα lookup ή περισσότερο το hashing μπορεί να συμφέρει.
  - Θα ταξινομήσουμε? Αν και οι περισσότερες γλώσσες υποστηρίζουν απευθείας μετατροπή των κλειδιών του Dictionary σε λίστα για να γίνει ταξινόμηση.
- Πως επιλέγουμε?
  - Μαντεύοντας. Ο γενικός κανόνας είναι “rule of thumb”. Κάθε πρόβλημα είναι διαφορετικό και μέχρι να το δεις να δουλεύει δεν ξέρεις με σιγουριά.

Python Specifics

- Κακές συνήθειες:
  - Αφήνω στην τύχη τους τύπους δεδομένων των μεταβλητών - δεν τις αρχικοποιώ
  - Ορίζω της μεταβλητές σε τυχαία σημεία μέσα στον κώδικα
  - Τα ονόματα των συναρτήσεων/μεταβλητών δεν ακολουθούν καμία σύμβαση
  - Ουδέν σχόλιον...
  - Γράφω συναρτήσεις-σεντόνια
  - Επαναλαμβάνω κώδικα που κάνει την ίδια δουλειά
  - Άλλες?

Recommended Reading

- "Natural Language Processing with Python"
  - Chapter 4: Writing Structured Programs