Natural-language understanding in robotics

(for domestic service robots, actually)

Mauricio Matamoros

2018.12.20

On Language

Definition

Saussure [1] formally defines the term language$_1$ (*fr. langue, de. „Spreache“*) as the “work of a collective intelligence”, in contrast with language$_2$ (*fr. parole, de. „Wort, Redeweise“*) which denotes the particular utterances of an individual.

On Language

Philology

Philology studies the historical growth, change, and adaptation of languages, specially focusing in *meaning*. It pays special attention to written literature.

Linguistics

Linguistics studies the phonetics and structure of languages, as well as their changes over time and the relationships among them.

On Language

Linguistics

Linguistics studies the phonetics and structure of languages, as well as their changes over time and the relationships among them.

- **Phonology and Phonetics** study the sounds of a language.
- **Morphology** studies the structure of words and its transforms.
- **Syntax** studies the grammatical aspects of a language, i.e. the structure of sentences.
- **Semantics** studies the meaning of words and sentences.
- **Pragmatics** studies the meaning of a sentence within a given context.

Formal and natural languages

According with Tarski [2] these are some of the differences between natural and formal languages:

Natural Languages

- Not finished
- Constantly changing
- Universal:
  anything that can be though, can be spoken of

Formal Languages

1. $\exists$, a structural definition $\forall$ symbol
2. $\exists$, a structural definition $\forall$ axioms or primitive statements
3. Inference rules allow transform sentences into axioms
4. All statements are unambiguous

Controlled Natural Languges

Natural languages are often untractable. Therefore al applications make use of a *Controlled Natural Language* subset that constrains the lexicon and syntax to whatever is relevant within the application domain.

Natural Language Processing

Natural Language Processing

Natural Language Processing (NLP) is the branch of Artificial Intelligence concerned about the interactions between computers and human (natural) languages, focusing on the creation of computer programs to process and analyze natural language.

Natural Language Understanding

Natural Language Understanding (NLU) is a sub-branch of NLP focused on enabling computers to process and understand human languages.

Natural Language Processing: Applications

- Spelling correction
- Grammar checking
- Question answering
- Dialog Systems
- Language detection
- Machine Translation
- Text classification
- Summarization
- Sentiment analysis
- Opinion mining
- Conversational Agents
  - Chatbots
  - Customer service
  - Client satisfaction survey

Natural Language Processing: Applications

- Spelling correction
- Grammar checking
- Question answering
- Dialog Systems
- Language detection
- Machine Translation
- Text classification
- Summarization
- Sentiment analysis
- Opinion mining
- Conversational Agents
  - Chatbots
  - Customer service
  - Client satisfaction survey

Robotics

Approaches

Classic

- Analytical
- Inspired in linguistics
- Requires programmer expertise

Stochastic

- Probabilistic
- Machine-learning inspired
- No expertise required

Approaches

Classic

Pros

- No corpora required
- Computationally cheap
- Easier to ground

Cons

- Hard to develop
- Memory expensive
- Human-error prone

Stochastic

Pros

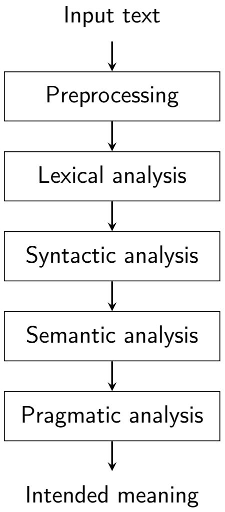
- Easy to develop
- Relatively fast
- Robust

Cons

- Requires [very] large corpora
- Computationally expensive
- Extending requires retraining

E L I Z A

Summary of analytic NLP methods



Summary of analytic NLP methods

Input text
↓
Preprocessing
↓
Lexical analysis
↓
Syntactic analysis
↓
Semantic analysis
↓
Pragmatic analysis
↓
Intended meaning

Preprocessing*

Cleans and split the whole text into paragraphs, paragraphs into sentences, and sentences into words (tokenization).

It is specially important in non-space delimited languages.

Summary of analytic NLP methods

Input text
↓
Preprocessing
↓
Lexical analysis
↓
Syntactic analysis
↓
Semantic analysis
↓
Pragmatic analysis
↓
Intended meaning

Lexical analysis (Lexer)*

Separates known and unknown words, and finds each word's canonical form (root).

- Stemmer
- Lemmatizer

Example

Hand me the apple juice

Gib mir den Apfelsaft

Hand · me (I) · the · apple-juice

gib (geben) · mir (ich) · den (der) · apfelsaft

Summary of analytic NLP methods

Input text → Preprocessing → Lexical analysis → **Syntactic analysis** → Semantic analysis → Pragmatic analysis → Intended meaning

Syntactic analysis (Parser)

Analyzes the relationship between the words in a sentence, and identifies their roles (POS-tagging and dependency graphs).
Also rejects unsupported inputs.

- Context free grammars (up to $O(n^3)$)
  - Generalized Phrase-Structure Grammar
  - Head-driven Phrase-Structure Grammar
  - Lexical-Functional Grammar
- Mildly Context-Sensitive Grammars (NP)
  - Tree-Adjoining Grammar
  - Combinatory Categorial Grammar
- Immediate Dominance, Linear Precedence, etc

Summary of analytic NLP methods

Input text
↓
Preprocessing
↓
Lexical analysis
↓
Syntactic analysis
↓
Semantic analysis
↓
Pragmatic analysis
↓
Intended meaning

Semantic analysis (Semantic parser)

Determines the meaning of a sentence and provides a structural [logic?] representation which can be grounded, i.e. mapped to representations of objects in the real world.

- First (and $n-th$) order logic
- Lambda calculus
- Semantic networks
- Semantic frames
- Description logics
- Semantic augmentation of CFG
- Natural Semantic Metalanguages
- Pustejovsky's Generative Lexicon
- Ontologies

Exercise: Lexical Analyzers

Lemmatizer

```
from nltk.stem import WordNetLemmatizer
nlp = WordNetLemmatizer()

print nlp.lemmatize('getting', 'v')
print nlp.lemmatize('feet', 'n')
print nlp.lemmatize('deadly', 'r')
print nlp.lemmatize('xyzing', '')
```

Stemmer

```
from nltk.stem import SnowballStemmer
nlp = SnowballStemmer('english')

print nlp.stem('getting')
print nlp.stem('feet')
print nlp.stem('deadly')
print nlp.stem('xyzing')
```

Summary of statistical NLP methods

	Preproc.	Lex	Syn	Sem
Entropy Maximization	☺	😐	☺	😐
Hidden Markov Models	😐	😐	☺	☹
Markov Decision Processes	—	—	😐	☺
Neural Networks	☹	☹	😐	☹
Probabilistic CFG	—	—	☺	☺
Statistical Decision Trees	—	—	☺	😐
Support Vector Machines	☺	—	☺	—

Other approaches (Not suitable for robotics)

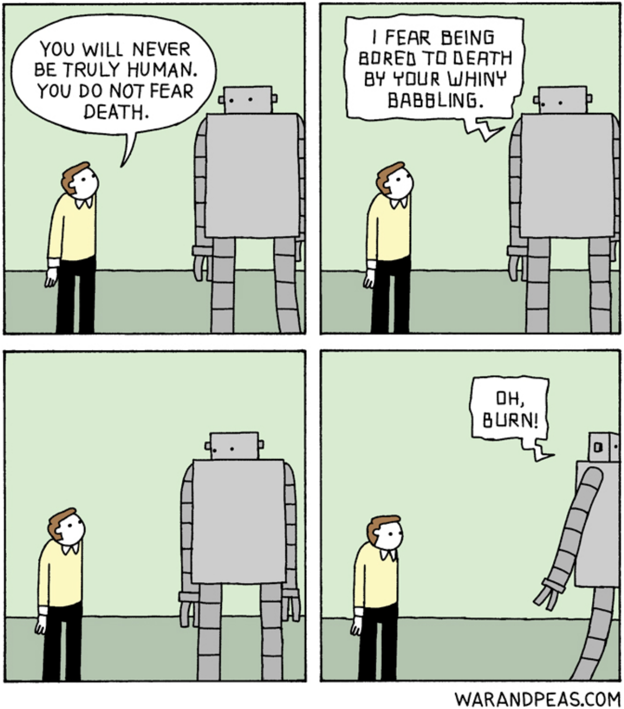
- Clustering
- K-means
- Linear Regression
- Markov Random Fields
- Mixture Models
- Perceptron and Kernel methods

Summary of statistical NLP methods

Take the best of both worlds to meet application requirements

In robotics

- No preprocessing (input clean from ASR)
- No lexical analysis (naive replacement)
- POS-Tagger (Stanford, if any)
- *Ad-hoc* semantics (mostly first order logic, if any)


YOU WILL NEVER BE TRULY HUMAN. YOU DO NOT FEAR DEATH.
I FEAR BEING BORED TO DEATH BY YOUR WHINY BABBLING.
OH, BURN!
WARANDPEAS.COM

How can we communicate with a computer?

- Coding

How can we communicate with a computer?

- ~~Coding~~
- Interface HW like smartphones (e.g. Telegram)

How can we communicate with a computer?

- ~~Coding~~
- Interface HW like smartphones (e.g. Telegram)
- Gestures and sign language

How can we communicate with a computer?

- ~~Coding~~
- Interface HW like smartphones (e.g. Telegram)
- Gestures and sign language
- Talk to it!

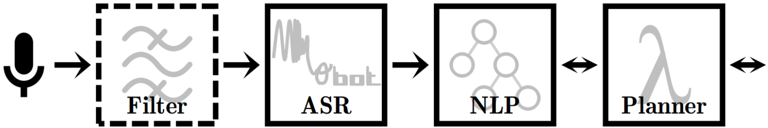
How can we communicate with a computer?

Observation

Always keep in mind

People do NOT want to read a manual

Golden pipeline



Golden pipeline

Cardioid for best results.

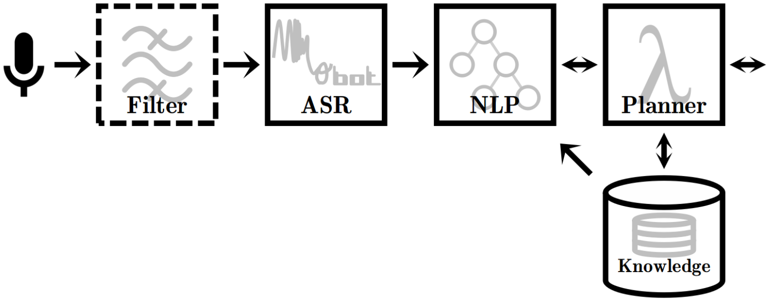
Normally absent (HARK is among the most used).

Mostly grammar-based solutions, but cloud services are used when possible.

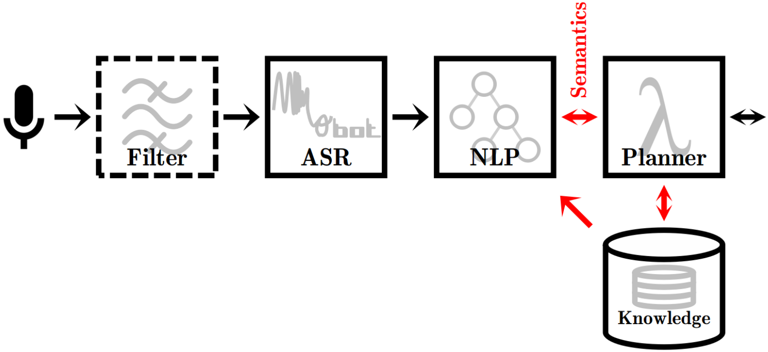
Varies widely. Stanford CoreNLP and LU4R are becoming popular. Mostly keyword spotting.

Varies widely. λ–calculus, KnowRob, and Watson are becoming popular. Mostly state machines.

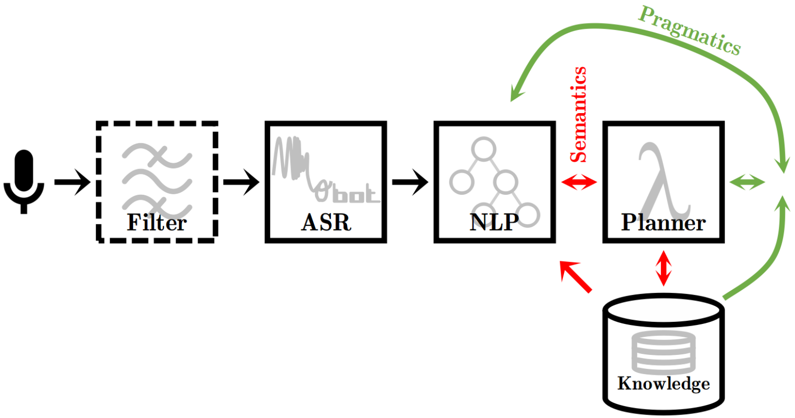
Golden pipeline



Golden pipeline



Golden pipeline



Challenges

Example: Ambiguity

John saw the girl with a telescope

Challenges

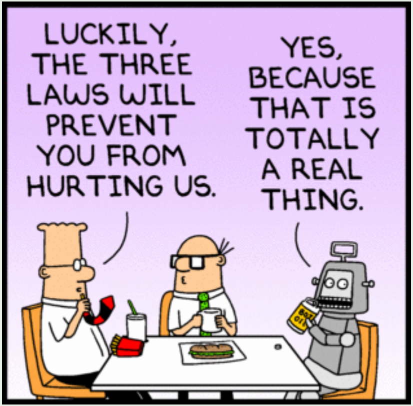
Example: Ambiguity

John saw the girl with a telescope

		
John	John used a telescope to cut the girl (in halves)	John used a telescope to see the girl
Girl	John used a saw to cut the girl that had a telescope	John had seen the girl who had a telescope

Challenges

Example: Sarcasm and Irony



Challenges

- Action Grounding
- Contextualized, time-sensitive dialogs
- Environmental awareness and reasoning

Robot actions alter the physical world

- Recognize what's doable and what isn't
- Consider world's physics
- Consider abstractions

Challenges

- Action Grounding
- Contextualized, time-sensitive dialogs
- Environmental awareness and reasoning

It shouldn't take longer to ask than to act

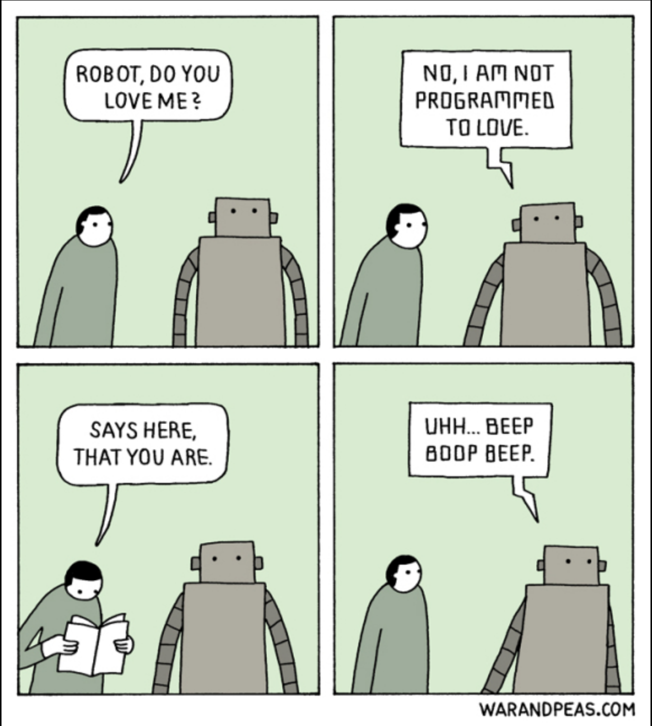
- People won't explain twice
- People won't explain the obvious

Challenges

- Action Grounding
- Contextualized, time-sensitive dialogs
- Environmental awareness and reasoning

Previous deeds and surroundings alter the interaction

- *. . . just like the other day*
- Most information is omitted when speaking


ROBOT, DO YOU LOVE ME?
NO, I AM NOT PROGRAMMED TO LOVE.
SAYS HERE, THAT YOU ARE.
UHH... BEEP BOOP BEEP.
WARANDPEAS.COM

Types of sentences

- Declarative
- Interrogative
- Imperative

Types of sentences

- **Declarative**
  Facts. Store new knowledge.
- Interrogative
- Imperative

Example

I lost my keys

Types of sentences

- Declarative
- **Interrogative**
  Questions. Retrieve knowledge.
- Imperative

Example

Have you seen my keys?

Types of sentences

- ~~Declarative~~
- ~~Interrogative~~
- **Imperative**
  Commands. Trigger an action.

Example

Find my keys, NOW!

Keyword spotting

Strategy

Find relevant words and discard the rest

Please, stop it!	*(stop)* $\rightarrow$ `stop()`
Could you help me to find my keys?	*(find, keys)* $\rightarrow$ `find(keys)`
I'd like to have cheese fingers and a coke	*(cheese, fingers, coke)* $\rightarrow$ `deliver(cheese-fingers, coke)`

Keyword spotting and Pattern matching

Strategy

Use keyword-based patterns to discern similar information

It is common to rely on the (augmented) CFG used by the ASR

Example

. . . and place the napkin over the cup

→ place/VB · (the/DT · napkin/NN)/NP · (over/IN)/PP · (the/DT · cup/NN)/NP
→ $S : VB_{\text{man}} NP_{\text{obj}} PP_{\text{stack}} NP_{\text{obj}}$
→ `place(napkin, glass)`

Frame-based dialog

Strategy

Fill the gaps (slots) in a dialog frame

Each slot in the frame has an associated question, and a set of keywords or patterns

1. Keyword-spot the main verb V (or find the most likely one)
2. Use V to select the adequate frame F (or the most likely one)
3. Use Keyword-spotting/pattern-matching to fill F
4. If there are gaps
    1. Select a gap and make the associated question
    2. Return to 3
5. Confirm information
6. Run

Moving the turtle using NL

```
import re
from robot import Robot

def parse(s):
  keywords = ['advance', 'step', 'turn', 'stop', 'spin']
  parts = re.split(r'\s+', s.lower())
  for word in parts:
    if word in keywords:
      return 'robot.{}()'.format(word)
  return None
#end def

def main():
  robot = Robot()
  while(True):
    s = raw_input('? ')
    action = parse(s)
    if not action is None:
      eval(action)
#end def
```

Moving the turtle using NL

```
import re
from robot import Robot

def parse(s):
  parts = re.split(r'\s+', s.lower())
  for word in parts:
    switcher = {
      'walk'      : "step",
      'run'       : "advance",
      ...
      'stop'      : "stop",
      'spin'      : "spin",
    }
    func = switcher.get(word, None)
    if not func is None:
      return 'robot.{}()'.format(func)
  return None
#end def
```

Moving the turtle using NL

Command

Please start spinning

Moving the turtle using NL

```python
import nltk
from robot import Robot

def parse(s):
  parts = nltk.word_tokenize(s.lower())
  stemmer = nltk.stem.SnowballStemmer('english')
  for word in parts:
    switcher = {
      'walk'      : "step",
      'run'       : "advance",
      ...
      'stop'      : "stop",
      'spin'      : "spin",
    }
    func = switcher.get(stemmer.stem(word), None)
    if not func is None:
      return 'robot.{}()'.format(func)
  return None
#end def
```

Moving the turtle using NL

Command

Turn left

Moving the turtle using NL

```
def parse(s):
  ...
    switcher = {
      'turn'     : "turn({})".format(parts),
      'walk'     : "robot.advance(0.5)",
      'run'      : "robot.advance(1.5)",
      'advance'  : "robot.advance()",
      'stop'     : "robot.stop()",
      'spin'     : "robot.spin()",
    }
    func = switcher.get(stemmer.stem(word), None)
  ...
#end def

def turn(parts):
  if 'left' in parts:     robot.rotate(90)
  elif 'right' in parts:  robot.rotate(-90)
  else:                   robot.turn()
#end def
```

Moving the turtle using NL

Now with a grammar (turning only)

```
% start S
S[F=?f, A=''] -> VB_STOP[F=?f]
S[F=?f, A=?a] -> VB_TURN[F=?f] DIR[A=?a]
S[F=?f, A=?a] -> VB_TURN[F=?f] PREP DET DIR[A=?a]
VB_TURN[F='rotate'] -> 'rotate' | 'turn' | 'go' | 'drive' | 'head'
VB_STOP[F='stop'] -> 'stop' | 'halt' | 'break' | 'enough'
PREP -> 'to' | 'towards'
DET -> 'the'
DIR[A='90'] -> 'left'
DIR[A='-90'] -> 'right'
DIR[A='180'] -> 'back' | 'around'
```

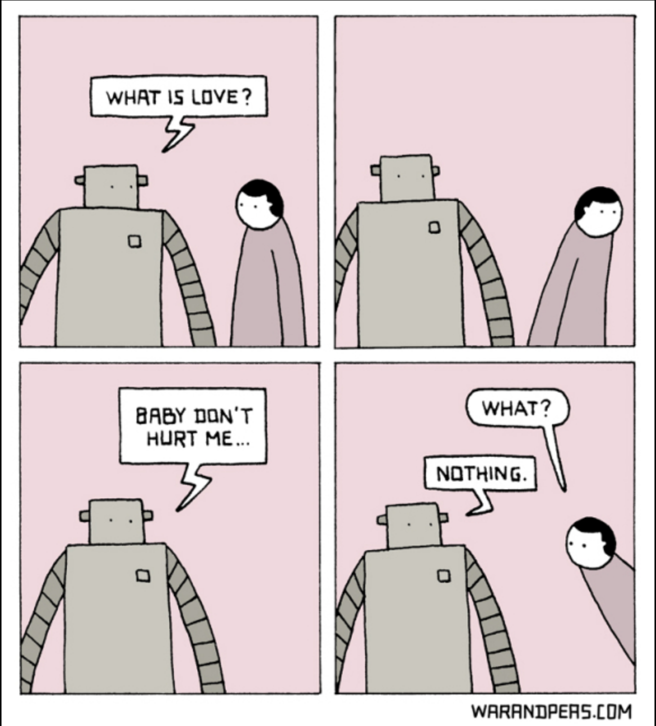
Moving the turtle using NL

```
def parse(s):
  try:
    fgrammar = nltk.grammar.FeatureGrammar.fromstring(
        grammar)
    fparser = nltk.parse.FeatureChartParser(fgrammar)
    trees = list(fparser.parse(nltk.word_tokenize(s.lower())
        ))
    if len(trees) < 1:
      return None
    tree = trees[0]
    return 'robot.{}({})'.format( trees[0].label()['F'],
        trees[0].label()['A'])
  except Exception as ex:
    print ex
    return None
#end def
```

Coupling an ASR engine

```
import speech_recognition as sr

reco = sr.Recognizer()
with sr.Microphone() as source:
reco.adjust_for_ambient_noise(source)
while True:
  try:
    audio = reco.listen(source)
    recognized = reco.recognize_google(audio, language='en-
        US')
   # Do NLP
  except:
    continue
#end while
```


WHAT IS LOVE?
BABY DON'T HURT ME...
WHAT?
NOTHING.
WARANDPEAS.COM

Suggested Bibliography

1. Nitin Indurkhya and Fred J. Damerau. *Handbook of natural language processing, volume 2*. CRC Press, 2010.
2. Dan Jurafsky and James H Martin. *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Prentice Hall, Pearson Education International, 2009.

Complementary Bibliography

1. James Allen. *Natural language understanding*. Pearson, 1995.
2. Eugene Charniak. *Statistical language learning*. MIT press, 1996.
3. Ralph Grishman. *Computational linguistics: an introduction*. Cambridge University Press, 1986.
4. Christopher D. Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. The MIT press, 1999.
5. George A. Miller. *Wordnet: a lexical database for english*. Communications of the ACM, 38(11):39–41, 1995.
6. Ruslan Mitkov. *The Oxford handbook of computational linguistics*. Oxford University Press, 2005.

References

[1] Ferdinand de Saussure. Cours de linguistique générale. Payot, Lausanne-Paris, 1916.

[2] Alfred Tarski. The concept of truth in formalized languages. Logic, semantics, metamathematics, 2:152–278, 1956.