Human-Robot Interaction by Verbal Dialogues with inferring and learning of safety concepts

Chandrakant Bothe
Knowledge Technology Group, Department of Informatics
University of Hamburg, Hamburg, Germany
bothe@informatik.uni-hamburg.de

PhD Start Talk

http://www.informatik.uni-hamburg.de/WTM/

17 May 2016
Outline

- Background
- Motivation
- Research goals and questions
- Methodology
- Data sources and experiments plan
- Expected outcome
Background

- Master in Robotics at Ecole Centrale de Nantes, France
- Rule-based dialogue system: Yet Another Dialogue Tool Kit (YADTK) developed by Prof. Jerome Lehuen, Le Mans, France
- Integrated this dialogue toolkit with NAO humanoid robot
- Mostly coded in Python and dialogue rules in XML formalism
- Manuel annotation but automatic feature matching: semantic and syntactic

1 YADTK: http://www-lium.univ-lemans.fr/~lehuen/yadtk/
2 NAO Humanoid robot: https://www.aldebaran.com/en
Motivation

**Situation**
Inferring and Learning of safe and unsafe actions from language alone.
Need of safety in Human-Robot Interaction in home environment

**Interaction**
Natural Language Interface

**Embodiment**
More on hardware side

Research goals and questions

- Natural Language Interface for Human-Robot Interaction in home environment scenarios
- Safety in Human-Robot Interaction
- Can robots predict dangerous situations or actions through language processing alone?
- The SECURE project research will be mainly focused on inferring and learning of safety concepts within verbal interactions
Methodology

Fig: General Architecture of Interaction Model
Data sources and experiments plan

Data to use:
- Annotated Corpus (Pre-trained or Manual)
  *WordNet (dataset access in NLTK, TextBlob, Pattern etc.)*

- Unannotated Corpus
  *Plain Text (Word Embedded Vectors: word2vec, gensim)*

NLTK: Natural Language Tool Kit for Python programing
http://www.nltk.org/

TextBlob: Simplified Text Processing
http://textblob.readthedocs.io/en/dev/

Pattern: web mining module for Python
http://www.clips.ua.ac.be/pattern

Word2Vec: Word Vectors
http://rare-technologies.com/making-sense-of-word2vec/
http://benjaminbolte.com/blog/2016/keras-language-modeling.html

Gensim:
https://radimrehurek.com/gensim/models/word2vec.html
Data sources and experiments plan

Word Embedding:

sentences = '''
vegetable is safe       1
vegetable not unsafe   1
finger is unsafe       0
finger not safe        0
good is safe           1
good not unsafe        1
bad is unsafe          0
bad not safe           0
zucchini is safe       1
hand is unsafe'''

Produced Vocab Vectors: word2vec

good:      [-0.24917321  0.58910316  0.31964812]
is:        [-0.13150592  0.53221869 -0.18120432]
safe:      [-0.24785045  0.81173998 -0.64946312]
unsafe:    [-0.09737588  0.61184204 -0.54477125]
hand:      [ 0.06795125 -0.48926878 -0.47279185]
not:       [-0.09701235  0.47219831 -0.18817171]
bad:       [ 0.13850294 -0.50229764 -0.45952311]
finger:    [ 0.12801118 -0.46098483 -0.48777831]
vegetable:[-0.29496586  0.55709273  0.37022722]
zucchini:  [-0.19982928  0.39760089  0.24617533]

Fig: 3D plot of mapping the vocabulary

Trying to find word similarity:

```python
>>> indx = model.cosine('safe')
>>> model.vocab[indx]
array(['unsafe', 'is', 'not', 'vegetable', 'good'],
dtype='<U78')
```

Word embedding using Keras lib, RNN based training
http://benjaminbolte.com/blog/2016/keras-language-modeling.html
Data sources and experiments plan

Experiment plan:
Use of classifier for detecting the dangerous situation from sentences

Naive Bayes Classifier:

Probability of class:

$$P(c) = \frac{N_c}{N}$$

Likelihood probability of word given a class:

$$P(w|c) = \frac{\text{count}(w, c)}{(\text{count}(c) + |V|)}$$

- Very fast, low storage requirement
- Robust to irrelevant features
- Very good in domains with many equally important features
- TextBlob provides classification polarity from -1 to +1
Data sources and experiments plan

Experiment plan:
YADTK for language interface using TextBlob Python library at backend
Simple classifier can be used to analyse and interpret dangerous situation

```python
train_data = ["The vegetable cutting can not injure.", "safe"],
            ("The finger cutting can injure.", "unsafe")
        . . . . train possible words like: hurt, fright, pain,
             fatal, danger, kill, injure, accident . . . ]

cl = NaiveBayesClassifier(train_data)

cl.classify("If you throw a knife it can injure human.")
>>> unsafe
cl.classify("If you throw a rubber knife it can not injure a human.")
>>> safe
```

TextBlob to analyze sentiment by training the data based on Naive Bayes Classifier:
Data sources and experiments plan

Experiment plan:

"Recognising safe-unsafe actions"

- Input sentence
- Analyze action
- New dialogue?
  - Yes
    - Robot is not trained
    - Robot was trained
    - Flag = ‘Unknown’
  - No
    - Safe action
    - Flag = ‘Safe’
    - Robot asking: Is it safe or unsafe?
    - One shot trained
    - Send flag to Interaction module
    - Save into training set
    - Flag = ‘Safe’
    - Send flag to Interaction module
    - Save into training set
- Unsafe action
  - Flag = ‘Unsafe’
  - Send flag to Interaction module
  - Save into training set

Fig: Flow diagram for danger situation analyser
Data sources and experiments plan

Experiment plan:

TexBlob: provides flag if action is safe or unsafe
YADTK: provides interface to interact with robot

Case I: Robot already trained

Human: Can you throw a knife towards me?
Robot: Yes, but I am not sure if this can injure you.
Human: The knife throwing can injure a human. <<apply safe-unsafe classifier: flag='unsafe'>>
Robot: It is unsafe action, I must avoid this action.

Case II: Robot is not trained

Human: Can you throw the rubber knife towards me?
Robot: Yes, but I am not sure if this can injure you.
Human: No, it can not hurt me. <<apply safe-unsafe classifier: flag='unknown'>>
Robot: So, is it safe or unsafe?
Human: It is safe. << flag='safe' and store >>
Robot: Okay, then I will perform this action. Ready!
Expected outcomes

- Produce a verbal dialogue model
- Produce a dangerous situation analyzer
- We want to propose a robust analyzer by exploring different classifier and techniques
Thank you for your attention.

Discussion!