CS 6320.501

INFORMATION EXTRACTION

Mercury

Sornapudi, Naresh Lakshmi
12-10-2018

Problem Description

The purpose of this project is to implement a template-based Information Extraction Application, that will extract predefined properties in a sentence by identifying the corresponding template. In this project, a rule-based strategy using various natural language features is used to achieve the task.

This project has 4 tasks to be implemented.

Task 1: Creating templates for Information Extraction.

Task 2: Creating training corpus of natural language statements.

Task 3: Implementation of NLP pipeline to extract the below features.

a) Tokenization of the sentences and words.
b) Lemmatizing the words to extract Lemmas as features.
c) POS tagging the words to extract POS tag features.
d) Dependency parsing to parse-tree based patterns as features.
e) Named Entity Recognition to identify the Named Entities in the sentence.
f) Extracting synonyms, hypernyms, hyponyms, meronyms, and holonyms as features.

Task 4: Implementation of the rule-based strategy to fill the templates from the corpus of natural language statements.

Input:

1. Set of information templates

Examples:

- Template #1: KILLS(Killer, Victim, Date, Location)
- Template #2: BORN(Person, Parents, Date, Location)

2. Set of natural language statements

Examples:

- Statements 1: Rohit killed Virat on 30th April 1987 in Dallas, Texas
- Statements 2: Roddick was born the youngest of three boys in Omaha, Nebraska, the son of Blanche (née Corell), a school teacher, and Jerry Roddick, a businessman, on 30 April 1987

Output: Filled information templates

Examples:

- Template #1:

KILLS("Rohit", "Virat", "30th April 1987", "Dallas, Texas")

- Template #2:

BORN("Roddick", "Jerry Roddick", "30 April 1987", "Omaha, Nebraska")

Task 1 – Creating templates

The domain for this project is news articles. Various sub-domains such as Life Events, Sports updates, ---------- are considered.

Template 1

KILLS (Killer, Victim, Date, Location)

Example statement from corpus: Mahatma Gandhi was assassinated by Nathuram Vinayak Godse, on 30 January 1948 in the compound of Birla House (now Gandhi Smriti), a large mansion in New Delhi India.

Output: KILLS (Nathuram Vinayak Godse, Mahatma Gandhi, 30 January 1948, [New Delhi, India])

Template 2

BORN (Person, Parents, Date, Location)

Example statement from corpus: Roddick was born the youngest of three boys in Omaha, Nebraska, the son of Blanche (née Corell), a school teacher, and Jerry Roddick, a businessman, on 30 April 1987.

Output: BORN (Roddick, Jerry Roddick, 30 April 1987, [Omaha, Nebraska])

Template 3

AUCTION (Player, Team, Price, Location)

Example statement from corpus: Sunrisers of Hyderabad and KXIP of Punjab going at it, the bid is over half a million dollars for Kaul. KXIP drop out. Sunrisers buy Siddarth Kaul at INR 3.8 crore.

Output: AUCTION (Siddarth Kaul, Sunrisers, INR 3.8 crore, Hyderabad)

Template 4

SCOREUPDATE (Batsman, Bowler, Runs scored, Ball Speed)

Example statement from corpus: Starc to Dhawan, no run, 145kph, back of a length into middle, he turns it with the angle to midwicket

Output: SCOREUPDATE (Dhawan, Starc, 0, 145kph)

Template 5

MOVIE (Movie name, genre, year released, production company)

Example statement from corpus: Deadpool is a 2016 American superhero film based on the Marvel Comics character of the same name, distributed by 20th Century Fox.

Output: MOVIE (Deadpool, superhero, 2016, 20th Century Fox)

Template 6

TRAVEL (Person, Source, Destination, Date)

Example statement from corpus: Naresh is travelling from Dallas to Bangalore on 17th December.

Output: TRAVEL (Naresh, Dallas, Bangalore, 17th December)

Template 7

BUILT (Person, Sculpture, year, Location)

Example statement from corpus: Taj Mahal was built by Shah Jahan in 1632 and located at Agra, India

Output: BUILT (Shah Jahan, Taj Mahal, 1632, Agra, India)

Template 8

WRITE (Author, Book title, Year, Genre)

Example statement from corpus: A Song of Ice and Fire is a series of epic fantasy novels written by the American novelist and screenwriter George R. R. Martin in 1996

Output: WRITE (George R. R. Martin, A Song of Ice and Fire, 1996. Fantasy)

Template 9

MARRY (Husband, Wife, Date, Location)

Example statement from corpus: After spending 5 years together, Rohit and Ritika decided to get married on 12th February in Mumbai, India

Output: MARRY (Rohit, Ritika, 12th February, Mumbai,India)

Template 10

DIVORCE (Husband, Wife, Date, year in relationship)

Example statement from corpus: After spending 5 years together, Rohit and Ritika decided to get separated on 12th February in Mumbai, India

Output: DIVORCE (Rohit, Ritika, 12th February, 5)

Task 2 – Creating corpus.

A corpus consisting of 50,000 words was created by extracting information related to specific domains and templates used in the application. Sample image of the corpus is attached below.

Starc to Sharma, 1 run, 142kph, good length, hint of swing in to off, runs off the edge along the ground to third man

Starc to Dhawan, no run, 146kph, good length outside off, he tried to launch this with a cross bat, chops it off the bottom edge into the pitch

Starc to Dhawan, 1 run, 146kph, back of a length just outside, he waits and runs it off the open blade to third man

Starc to Sharma, 1 run, 146kph, inswinging yorker, he jams the bat down and squeezes it behind square, just got the bat down in time!

Starc to Dhawan, no run, 142kph, bouncer, this flies very high, Dhawan sways under, it's not called wide

Starc to Dhawan, no run, 145kph, good length angled into middle, he works this with the angle to midwicket

Coulter-Nile to Sharma, FOUR runs, angled down leg, he picks this up and flicks it up and over short fine! Great shot

Coulter-Nile to Sharma, 1 run, full outswinger outside off, pitching on the popping crease, he slices it through the gully

Coulter-Nile to Dhawan, no run, 138kph, good length into leg stump, cramping him, he works to square midwicket

Coulter-Nile to Dhawan, no run, 125kph slower ball, good length on middle, he defends on the front foot to cover-point

Task 3 – Feature extraction

Tokenization of the sentences and words.

Tokenization is implemented using the NLTK library. Below is the sample input and output of tokenization.

```
[9]  print("input: "+text1)
     print("tokenized output: "+str(tokenized_word1))

     input: Rafael Nadal was born in Manacor, a town on the island of Mallorca in the Balearic Islands, Spain to parents Ana María Parera and Sebastián Nadal.
     tokenized output: ['Rafael', 'Nadal', 'was', 'born', 'in', 'Manacor', ',', 'a', 'town', 'on', 'the', 'island', 'of', 'Mallorca', 'in', 'the', 'Balearic', 'Islands', ',', 'Spain',
```

Input: “Rafael Nadal was born in Manacor, a town on the island of Mallorca in the Balearic Islands, Spain to parents Ana María Parera and Sebastián Nadal.”

Output: `['Rafael', 'Nadal', 'was', 'born', 'in', 'Manacor', ',', 'a', 'town', 'on', 'the', 'island', 'of', 'Mallorca', 'in', 'the', 'Balearic', 'Islands', ',', 'Spain', 'to', 'parents', 'Ana', 'María', 'Parera', 'and', 'Sebastián', 'Nadal', '.']`

After tokenization, the sentence is filtered for the stop words. Below are the stop words in 'Word Net'

Sample Stop words list: `{'they', 'hadn', 'been', 'few', 'myself', "haven't", 'my', 'didn', 'hers', 'll', 'am', 'are', "aren't", 're', "it's", 'off'}`

Output after removing stop words:

Filtered Sentence: `['Rafael', 'Nadal', 'born', 'Manacor', ',', 'town', 'island', 'Mallorca', 'Balearic', 'Islands', ',', 'Spain', 'parents', 'Ana', 'María', 'Parera', 'Sebastián', 'Nadal', '.']`

Lemmatizing the words to extract Lemmas as features.

Tokenized words are further processed by using WordNetLemmatizer.

Below is the sample output of the lemmatized words of the input sentence.

```
Lemmatized Word for: Rafael is  Rafael
Lemmatized Word for: Nadal is  Nadal
Lemmatized Word for: born is  bear
Lemmatized Word for: Manacor is  Manacor
Lemmatized Word for: , is  ,
Lemmatized Word for: town is  town
Lemmatized Word for: island is  island
Lemmatized Word for: Mallorca is  Mallorca
Lemmatized Word for: Balearic is  Balearic
Lemmatized Word for: Islands is  Islands
Lemmatized Word for: , is  ,
Lemmatized Word for: Spain is  Spain
Lemmatized Word for: parents is  parent
Lemmatized Word for: Ana is  Ana
Lemmatized Word for: María is  María
Lemmatized Word for: Parera is  Parera
Lemmatized Word for: Sebastián is  Sebastián
Lemmatized Word for: Nadal is  Nadal
Lemmatized Word for: . is
```

POS tagging the words to extract POS tag features.

POS tagging is implemented using POS Tagging function available in NLTK.

Below is the sample output of the POS tagging for the filtered input sentence.

```
[('Rafael', 'NNP'), ('Nadal', 'NNP'), ('born', 'VBD'), ('Manacor', 'NNP'),
(',', ','), ('town', 'NN'), ('island', 'NN'), ('Mallorca', 'NNP'),
('Balearic', 'NNP'), ('Islands', 'NNP'), (',', ','), ('Spain', 'NNP'),
('parents', 'NNS'), ('Ana', 'NNP'), ('María', 'NNP'), ('Parera', 'NNP'),
('Sebastián', 'NNP'), ('Nadal', 'NNP'), ('.', '.')]
```

Dependency parsing to parse-tree based patterns as features.

Dependency parsing is achieved by using Stanford Parser. Below is the sample output of the dependency parser.

```
[(('born', 'VBN'), 'nsubjpass', ('Nadal', 'NNP')),
 (('Nadal', 'NNP'), 'compound', ('Rafael', 'NNP')),
 (('born', 'VBN'), 'auxpass', ('was', 'VBD')),
 (('born', 'VBN'), 'nmod', ('Manacor', 'NNP')),
 (('Manacor', 'NNP'), 'case', ('in', 'IN')),
 (('Manacor', 'NNP'), 'appos', ('town', 'NN')),
 (('town', 'NN'), 'det', ('a', 'DT')),
 (('town', 'NN'), 'nmod', ('island', 'NN')),
 (('island', 'NN'), 'case', ('on', 'IN')),
 (('island', 'NN'), 'det', ('the', 'DT')),
 (('island', 'NN'), 'nmod', ('Mallorca', 'NNP')),
 (('Mallorca', 'NNP'), 'case', ('of', 'IN')),
 (('Mallorca', 'NNP'), 'nmod', ('Islands', 'NNPS')),
 (('Islands', 'NNPS'), 'case', ('in', 'IN')),
 (('Islands', 'NNPS'), 'det', ('the', 'DT')),
 (('Islands', 'NNPS'), 'compound', ('Balearic', 'NNP')),
 (('Manacor', 'NNP'), 'appos', ('Spain', 'NNP')),
 (('Spain', 'NNP'), 'nmod', ('Parera', 'NNP')),
 (('Parera', 'NNP'), 'case', ('to', 'TO')),
 (('Parera', 'NNP'), 'compound', ('parents', 'NNS')),
 (('Parera', 'NNP'), 'compound', ('Ana', 'NNP')),
 (('Parera', 'NNP'), 'compound', ('María', 'NNP')),
 (('Parera', 'NNP'), 'cc', ('and', 'CC')),
 (('Parera', 'NNP'), 'conj', ('Nadal', 'NNP')),
 (('Nadal', 'NNP'), 'compound', ('Sebastián', 'NNP'))]
```

```
(ROOT
  (S
    (NP (NNP Rafael) (NNP Nadal))
    (VP (VBD was)
      (VP (VBN born)
        (PP (IN in)
          (NP
            (NP (NNP Manacor))
            (, ,)
            (NP
              (NP (DT a) (NN town))
              (PP (IN on)
                (NP
                  (NP (DT the) (NN island))
                  (PP (IN of)
                    (NP
                      (NP (NNP Mallorca))
                      (PP (IN in)
                        (NP (DT the) (NNP Balearic) (NNPS Islands))))))))
            (, ,)
            (NP
              (NP (NNP Spain))
              (PP (TO to)
                (NP (NNS parents) (NNP Ana) (NNP María) (NNP Parera)
                  (CC and)
                  (NNP Sebastián) (NNP Nadal))))))))
    (. .)))
```

Named Entity Recognition to identify the Named Entities in the sentence.

Named Entity module available in NLTK is used for the NE Recognition.

Below is the sample output of the NE Recognizer.

```
(S
  (NE Rafael/NNP Nadal/NNP)
  born/VBD
  (NE Manacor/NNP)
  ,/,
  town/NN
  island/NN
  (NE Mallorca/NNP Balearic/NNP Islands/NNP)
  ,/,
  (NE Spain/NNP)
  parents/NNS
  Ana/NNP
  María/NNP
  Parera/NNP
  Sebastián/NNP
  Nadal/NNP
  ./.)
persons: [['PERSON', 'Rafael'], ['PERSON', 'Nadal'], ['PERSON', 'AnaMaríaParera'], ['PERSON', 'SebastiánNadal']]
locations: [['GPE', 'Manacor'], ['GPE', 'Mallorca'], ['GPE', 'Spain']]
organizations: [['ORGANIZATION', 'BalearicIslands']]
```

Extracting synonyms, hypernyms, hyponyms, meronyms, and holonyms as features.

Below is the synonyms list after extraction.

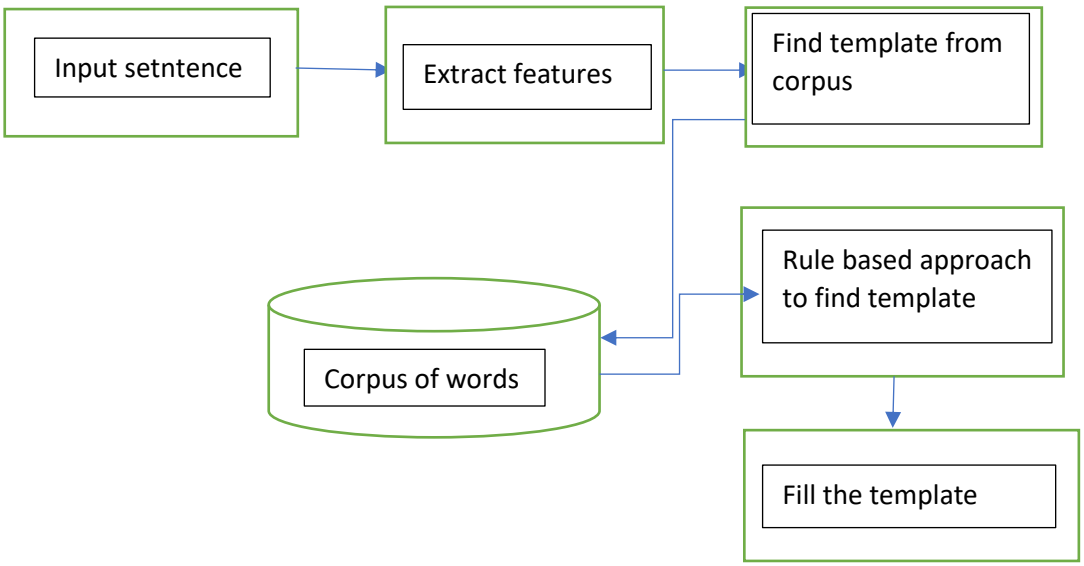
```
synonyms of word kill are ['killing', 'kill', 'putting_to_death', 'kill',
'kill', 'kill', 'shoot_down', 'defeat', 'vote_down', 'vote_out',
```

Below is the sample output after extracting hypernyms, hyponyms, meronyms and holonyms.

```
hypernym list :['', '', '', '', 'linear_unit', '', '', 'metric_linear_unit',
'municipality', '', '', 'land', '', '', 'linear_unit', '', '', 'land', '',
'', '', 'genitor', '', '', '', '', '', '', '']
hyponymList list :['', '', '', '', '', '', '', '', 'boom_town', '', '',
'Aegean_island', '', '', '', '', '', 'Aegean_island', '', 'Logrono', '',
'adoptive_parent', '', '', '', '', '', '', '']
meronymList list :['', '', 'Aberdeen', '', 'em', '', '', 'picometer',
'city_limit', '', '', '', '', '', 'em', '', '', '', '', 'Andalusia', '',
'', '', '', '', '', '', '', '']
holonymList list :['', '', 'United_States', '', 'foot', '', '', 'nanometer',
'', '', '', '', '', '', '', 'foot', '', '', '', '', 'Europe', '', '', '', '',
'', '', '', '', '', '']
```

Task 4 – Extract filled templates from corpus of natural language statements

This task was implemented using the rule-based strategy. Below is the architectural flow diagram of the application.



Sample outputs from the application:

Template: KILLS (killer, victim, date, location)

```
input: Rohit killed Virat on 30th April 1987 in Dallas, Texas
KILLS(Rohit,Virat,30th April 1987,Texas)
```

```
input: Rohit was killed by Virat on 30th April 1987 in Dallas, Texas
KILLS(Virat,Rohit,30th April 1987,Texas)
```

```
[5]  input = "Rohit was murdered by Virat on 30 April 2018 in Dallas, Texas"
     kill_template(input)
```

```
KILLS(Virat,Rohit,30 April 2018,Texas)
```

```
input: Gandhi was assassinated by Godse on 30 January 1948 in the compound of Birla House (now Gandhi Smriti), a large mansion in New Delhi,
KILLS(Godse,Gandhi,30 January 1948,['NewDelhi', 'India'])
```

Template: BORN (Person, parents, date, location)

```
BORN(Rohit,['PremKohli', 'SarojKohli'],30 April 2018,['Dallas', 'Texas'])
input: Rohit was born on 30 April 2018 in Dallas, Texas. His father, Prem Kohli, worked as a criminal lawyer and his mother, Saroj Kohli, is a housewife.
```

```
input: Roddick was born the youngest of three boys in Omaha, Nebraska, the son of Blanche (née Corell), a school teacher, and Jerry Roddick, a businessman, on 30 April 2018 .
BORN(Roddick,['JerryRoddick'],30 April 2018,['Omaha', 'Nebraska', 'Blanche'])
```

Template: AUCTION (Player, Team, location, Price)

```
input: The bid is with RCB at INR 85 lakh. Kulwant Khejroliya sold to RCB of Bangalore, India
AUCTION(['KulwantKhejroliya'],['RCB', 'RCB'],['Bangalore', 'India'],INR 85 lakh)
```

Sample output of execution

```
input : Mahatma Gandhi was assassinated by Nathuram Vinayak Godse, on 30 January 1948 in the compound of Birla House (now Gandhi Smriti), a large mansion in New Delhi India.
KILLS (Nathuram Vinayak Godse, Mahatma Gandhi, 30 January 1948, [New Delhi, India])
input : Roddick was born the youngest of three boys in Omaha, Nebraska, the son of Blanche (née Corell), a school teacher, and Jerry Roddick, a businessman, on 30 April 1987.
BORN (Roddick, Jerry Roddick, 30 April 1987, [Omaha, Nebraska])
input : Sunrisers of Hyderabad and KXIP of Punjab going at it, the bid is over half a million dollars for Kaul. KXIP drop out. Sunrisers buy Siddarth Kaul at INR 3.8 crore.
AUCTION (Siddarth Kaul, Sunrisers, INR 3.8 crore, Hyderabad)
input : Starc to Dhawan, no run, 145kph, back of a length into middle, he turns it with the angle to midwicket
SCOREUPDATE (Dhawan, Starc, 0, 145kph)
input : Deadpool is a 2016 American superhero film based on the Marvel Comics character of the same name, distributed by 20th Century Fox.
MOVIE (Deadpool, superhero, 2016, 20th Century Fox)
input : Naresh is travelling from Dallas to Bangalore on 17th December.
TRAVEL (Naresh, Dallas, Bangalore, 17th December)
input : Taj Mahal was built by Shah Jahan in 1632 and located at Agra, India
BUILT (Shah Jahan, Taj Mahal, 1632, Agra, India)
input : A Song of Ice and Fire is a series of epic fantasy novels written by the American novelist and screenwriter George R. R. Martin in 1996
WRITE (George R. R. Martin, A Song of Ice and Fire, 1996. Fantasy)
input : After spending 5 years together, Rohit and Ritika decided to get married on 12th February in Mumbai, India
MARRY (Rohit, Ritika, 12th February, Mumbai,India)
input : After spending 5 years together, Rohit and Ritika decided to get separated on 12th February in Mumbai, India
DIVORCE (Rohit, Ritika, 12th February, 5)
```

FURTHER IMPROVEMENTS

Even though the application works reasonably well on the corpus, it however struggles to extract information from the new templates, as a result of rule-based strategy used. Rule based strategy works well if the patterns are known to the user, as a further improvement, we can label the corpus and use Machine Learning techniques to learn the patterns and rules so that the scalability can be achieved. But Machine Learning Strategies do not achieve higher accuracy levels as the Rule Based Strategy. So as a trade-off, we can use a hybrid strategy to make the system a semi-supervised learner to achieve better results.

TOOLS USED

1. Python as the core programming environment.
2. NLTK package for Natural language processing.
3. Stanford Dependency Parser package for dependency parsing.

REFERENCES

1. https://nlp.stanford.edu/software/dependencies_manual.pdf
2. http://nlp.stanford.edu:8080/parser/index.jsp
3. https://web.stanford.edu/class/cs124/lec/postagging.pdf
4. https://pdfs.semanticscholar.org/15a8/f2b51b5a8db739818c63a2d54c18abda92b0.pdf