ayat-rashad / **ayat-rashad.github.io**

Branch: master ▾ ayat-rashad.github.io / **triples.ipynb** Find file Copy path

ayat-rashad Update triples.ipynb `7135cdf` on Jun 19, 2016

1 contributor

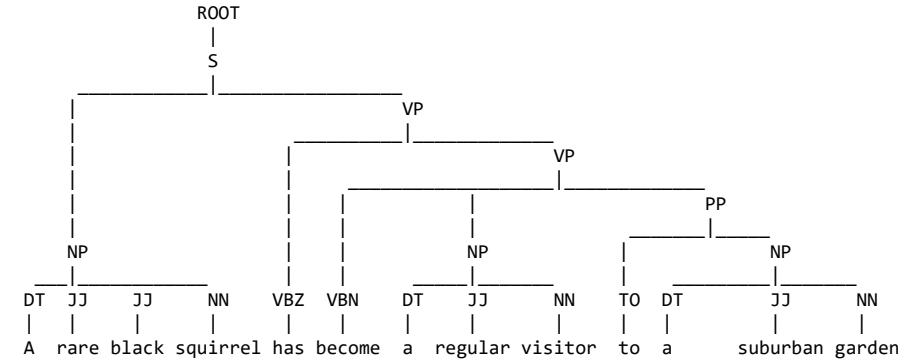
218 lines (217 sloc) 6.63 KB

Extracting relations, or (subject, predicate, object) triples:

https://playwithml.wordpress.com/2016/06/15/extracting-relations-or-subject-predicate-object-triples/
(https://playwithml.wordpress.com/2016/06/15/extracting-relations-or-subject-predicate-object-triples/)

In [1]:
```
import os

os.environ['STANFORD_PARSER'] = 'stanford-parser'
os.environ['STANFORD_MODELS'] = 'stanford-parser'
```

In [12]:
```
from nltk.parse.stanford import StanfordParser
from nltk.tree import ParentedTree, Tree

parser = StanfordParser()

# Parse the example sentence
sent = 'A rare black squirrel has become a regular visitor to a suburban garden'
t = list(parser.raw_parse(sent))[0]
t = ParentedTree.convert(t)

t.pretty_print()
```



In [10]:
```
def find_subject(t):
    for s in t.subtrees(lambda t: t.label() == 'NP'):
        for n in s.subtrees(lambda n: n.label().startswith('NN')):
            return (n[0], find_attrs(n))

def find_predicate(t):
    v = None

    for s in t.subtrees(lambda t: t.label() == 'VP'):
        for n in s.subtrees(lambda n: n.label().startswith('VB')):
            v = n
        return (v[0], find_attrs(v))

def find_object(t):
    for s in t.subtrees(lambda t: t.label() == 'VP'):
        for n in s.subtrees(lambda n: n.label() in ['NP', 'PP', 'ADJP']):
            if n.label() in ['NP', 'PP']:
                for c in n.subtrees(lambda c: c.label().startswith('NN')):
                    return (c[0], find_attrs(c))
            else:
                for c in n.subtrees(lambda c: c.label().startswith('JJ')):
                    return (c[0], find_attrs(c))

def find_attrs(node):
    attrs = []
    p = node.parent()

    # Search siblings
    if node.label().startswith('JJ'):
        for s in p:
            if s.label() == 'RB':
                attrs.append(s[0])

    elif node.label().startswith('NN'):
        for s in p:
            if s.label() in ['DT','PRP$','POS','JJ','CD','ADJP','QP','NP']:
                attrs.append(' '.join(s.flatten()))

    elif node.label().startswith('VB'):
        for s in p:
            if s.label() == 'ADVP':
                attrs.append(' '.join(s.flatten()))

    # Search uncles
    if node.label().startswith('JJ') or node.label().startswith('NN'):
        for s in p.parent():
```

```
            for s in p.parent():
                if s != p and s.label() == 'PP':
                    attrs.append(' '.join(s.flatten()))

        elif node.label().startswith('VB'):
            for s in p.parent():
                if s != p and s.label().startswith('VB'):
                    attrs.append(s[0])

        return attrs
```

In [13]:
```
print find_subject(t)
print find_predicate(t)
print find_object(t)
```

```
(u'squirrel', [u'A', u'rare', u'black'])
(u'become', [u'has'])
(u'visitor', [u'a', u'regular', u'to a suburban garden'])
```

In [9]:
```
sent = 'The hotel is located in USA'
t = list(parser.raw_parse(sent))[0]
t = ParentedTree.convert(t)

print find_subject(t)
print find_predicate(t)
print find_object(t)
```

```
(u'hotel', [u'The'])
(u'located', [u'is'])
(u'USA', [])
```

In [ ]: