EC999: Named Entity Recognition

Thiemo Fetzer

University of Chicago & University of Warwick

January 24, 2017

Named Entity Recognition in Information Retrieval

- Information retrieval systems extract clear, factual information
- Can think of trying to answer questions such as: Who? What? Where? When?
- Named Entities are very important subtask to many information retrieval tasks.
- Named Entity recognition systems are build to *identify* and *classify* named entites of specific types.
- Typically they separate: Person, Organization, Date, Location,...
- NER is useful for:
  1. Named entities can be indexed, linked off, etc.
  2. Sentiment can be attributed to companies or products
  3. A lot of IE relations are associations between named entities
  4. For question answering, answers are often named entities.

Named Entity Identification

The decision by the independent MP **Andrew Wilkie** to withdraw his support for the minority **Labor** government sounded dramatic but it should not further threaten its stability. When, after the **2010** election, **Wilkie**, **Rob Oakeshott**, **Tony Windsor** and the **Greens** agreed to support **Labor**, they gave just two guarantees: confidence and supply.

Named Entity Categorization

The decision by the independent MP **Andrew Wilkie - PERSON** to withdraw his support for the minority **Labor - ORGANIZATION** government sounded dramatic but it should not further threaten its stability. When, after the **2010 - DATE** election, **Wilkie - PERSON**, **Rob Oakeshott - PERSON**, **Tony Windsor - PERSON** and the **Greens - ORGANIZATION** agreed to support **Labor - ORGANIZATION**, they gave just two guarantees: confidence and supply.

Named Entity Recognition

- Crucial for Information Extraction, Question Answering and Information Retrieval

  $\Rightarrow$ Up to 10% of a newswire text may consist of proper names, dates, times,...
- Relational information is built on top of Named Entities
- Many web pages tag various entities, with links to bio or topic pages, etc.

  $\Rightarrow$ knowledge graphs build on top of Wikipedia use NER parsing of articles.
- `https://www.wikidata.org/wiki/Q22686`

Three Approaches to Named Entity Recognition

1. Rule based retrieval: e.g. through regular expressions

   $\Rightarrow$ remember the email vs telefon number extraction regexes?

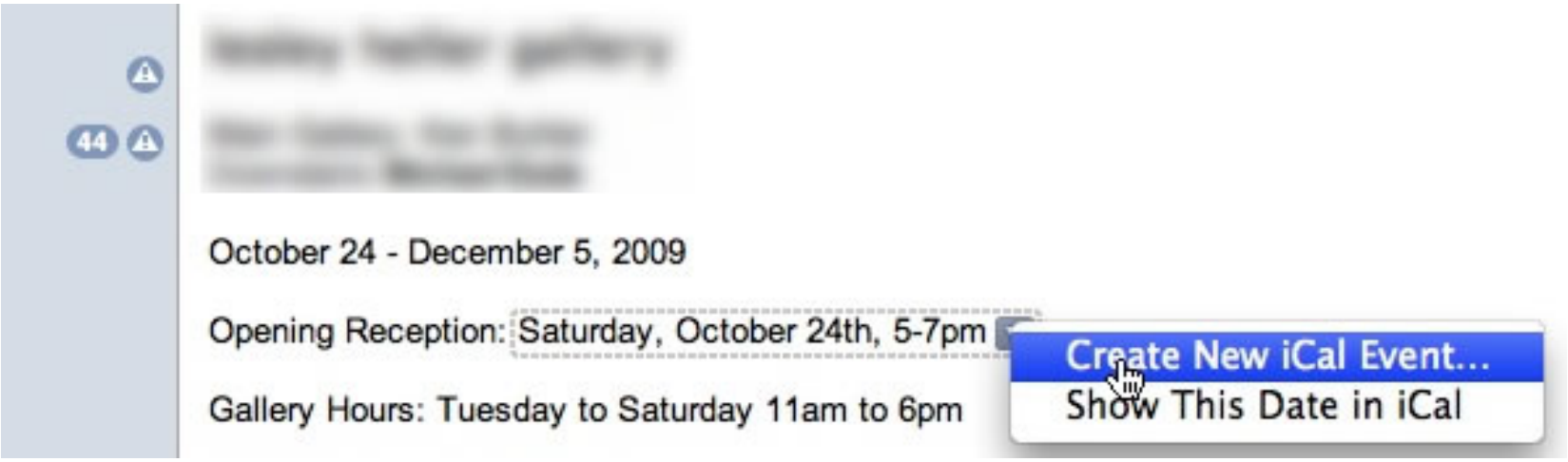
2. Classifiers

   In this course we will introduce a range, especially Naive Bayes, logistic regression, ...

3. Sequence models

   Hiden Markov Models, ...

Rule-based retrieval



Rule based extraction of dates using regular expressions

Rule-based retrieval

- If extracting from automatically generated web pages, simple regex patterns usually work as behind the scenes a common template forces a layout.

  HTML page

  ```
  <b>(.*?)</b>
  ```

- For certain restricted, common types of entities in unstructured text, simple regex patterns also usually work.

  Finding (US) phone numbers

  ```
  ([2-9][0-9]{2})[- .]([0-9]{3})[- .]([0-9]{4})
  ```

Rule-based retrieval

- Can leverage POS tagging to define a set of rules to extract training data for a classifier.
- Remember pairs of "NN - NN" extracted in collocation discovery exercise?
- Determining which person holds what office in what organization

  [person] , [office] of [org]

  This defines a sequence of nouns that could be extracted using a regex searching for [NN], [NN] of [NN].

Machine Learning Approach to Named Entity Recognition

Training

1. Collect a set of representative training documents
2. Label each token for its entity class or other (O)
3. Design feature extractors appropriate to the text and classes
4. . Train a sequence classifier to predict the labels from the data [more next week]

Testing

1. Receive a set of testing documents
2. Run sequence model inference to label each token
3. Appropriately output the recognized entities

Features for sequence labeling

- Words, current word, previous word,
- Parts of words such as indicators for suffixes common to names of places or locations.
- Part of speech tags
- Shapes of words

MEMM inference in systems

- Conditional Markov Model the classifiers makes a decision at a time conditional on the evidence in the neighborhood.
- Typically use a type of logistic regression classifier.
- Example for data used for classifier

w	POS p	NER
Three	CD	O
ISIL	NNP	ORG
terrorists	NNS	O
were	VBD	O
killed	VBN	O
in	IN	O
Rakka	NNP	**LOC**
.	.	O

Features at word $w_n =$ Rakka:
$\{w_{n-1} = "\text{in}", w_n = \text{Rakka}, w_{n+1} = ".", p_{n-1} = "IN", p_n = "NNP", ...\}$.

Named Entity Recognition in R

- There are a couple of built named entity recognition modules built into easy access NLP packages.
- Most notably are OpenNLP and the Stanford NLP, both require `rJava` package and a Java installation on the operating system.

  `install.packages('rJava')` and installation of `Java JDK 1.2 or higher`
- I will also show how to access purpose built classifiers built for very specific tasks can be made accessible to work in R.

OpenNLP

The Apache OpenNLP library is a machine learning based toolkit for the processing of natural language text.

It supports the most common NLP tasks, such as tokenization, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing, and coreference resolution. These tasks are usually required to build more advanced text processing services

`https://opennlp.apache.org`

Opening a pipeline between OpenNLP and R

```
# install.packages(c('NLP', 'openNLP')) install.packages('openNLPmodels.en', repos =
# 'http://datacube.wu.ac.at/', type = 'source')
library(NLP)
library(openNLP)
library(openNLPmodels.en)

TRUMP <- readLines("../../Data/Speeches/trump-foreignpolicy.txt")
TRUMP <- paste(TRUMP, collapse = " ")
TRUMP <- gsub("([A-Z]{4,})", "", TRUMP)
# create instance
word_ann <- Maxent_Word_Token_Annotator()
sent_ann <- Maxent_Sent_Token_Annotator()

TRUMP_annotate <- annotate(TRUMP, list(sent_ann, word_ann))

head(TRUMP_annotate)
```

```
##  id type     start end features
##   1 sentence     1  10 constituents=<<integer,3>>
##   2 sentence    12  34 constituents=<<integer,7>>
##   3 sentence    36  68 constituents=<<integer,9>>
##   4 sentence    70 168 constituents=<<integer,19>>
##   5 sentence   170 220 constituents=<<integer,10>>
##   6 sentence   222 327 constituents=<<integer,22>>
```

```
# combine Annotations with text
TRUMP_doc <- AnnotatedPlainTextDocument(TRUMP, TRUMP_annotate)
```

OpenNLP Named Entity Recognition types

OpenNLP can find dates, locations, money, organizations, percentages, people, and times.

```
# need to create instances of different types annotators supported values for kind are
# 'date', 'location', 'money', 'organization', 'percentage', 'person', 'misc'
person_ann <- Maxent_Entity_Annotator(kind = "person")
location_ann <- Maxent_Entity_Annotator(kind = "location")
organization_ann <- Maxent_Entity_Annotator(kind = "organization")

# pipeline for processing, NER annotators require sentence and word boundaries as inputs
pipeline <- list(sent_ann, word_ann, location_ann, organization_ann, person_ann)
TRUMP_annotate <- annotate(TRUMP, pipeline)
TRUMP_doc <- AnnotatedPlainTextDocument(TRUMP, TRUMP_annotate)

TRUMP_doc
```

```
## <<AnnotatedPlainTextDocument>>
## Metadata:  0
## Annotations:  1, length(s): 5475
## Content:  chars: 25389
```

OpenNLP Extract Named Entities from Annotated Document

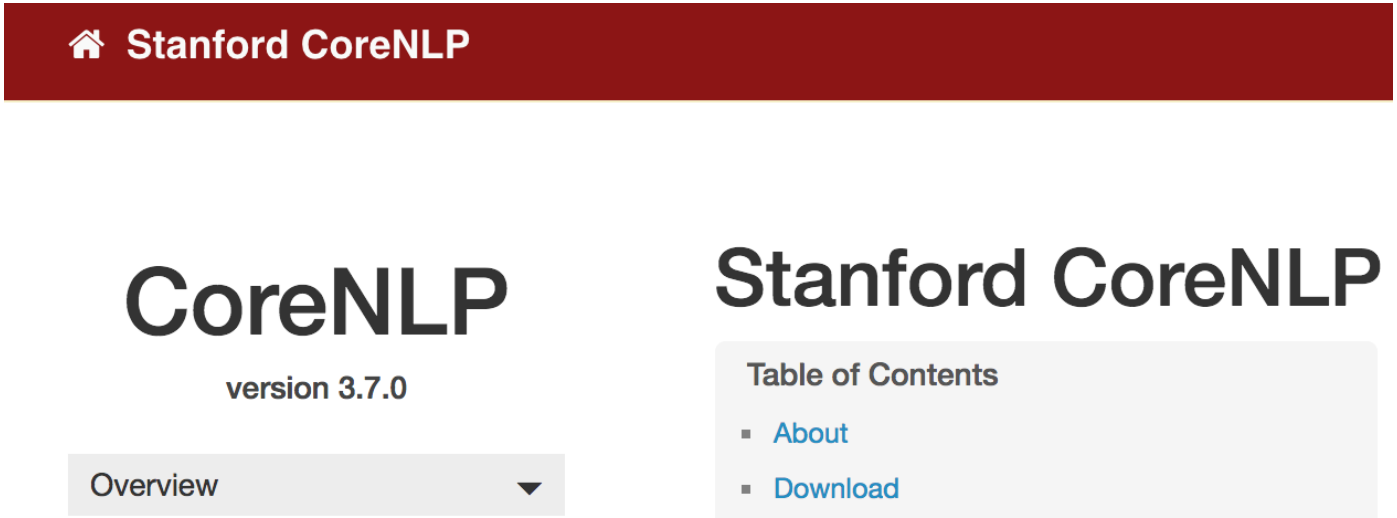
```
# Extract entities from an AnnotatedPlainTextDocument
entities <- function(doc, kind) {
    s <- doc$content
    a <- annotations(doc)[[1]]
    if (hasArg(kind)) {
        k <- sapply(a$features, `[[`, "kind")
        s[a[k == kind]]
    } else {
        s[a[a$type == "entity"]]
    }
}

sort(table(entities(TRUMP_doc, kind = "person")), decreasing = TRUE)[1:10]

##
##   Hillary Clinton           Clinton           Hillary Hillary Clintons              Bill
##                23                 9                 8                 6                 3
##    Bernie Sanders      Bill Clinton          Clintons   Abraham Lincoln    Bill Clintons
##                 2                 2                 2                 1                 1

sort(table(entities(TRUMP_doc, kind = "location")), decreasing = TRUE)[1:10]

##
## United States         China       America         Libya          Iran          Iraq
##            13             9             5             5             3             3
##   Middle East         Egypt        Israel      New York
##             3             2             2             2
```

OpenNLP Handling

- OpenNLP handling in R is a bit cumbersome, have to set up pipelines etc...
- Could write a custom function to perform a specific NLP task and return vector of , e.g. named entities.
- For SENNA, this type of processing pipeline set up is explained here: `http://www.trfetzer.com/a-simple-pipeline-to-access-senna-nlp-toolkit-through-r/`
- Next present StanfordNLP package, which is easier to work with in R.

Stanford CoreNLP

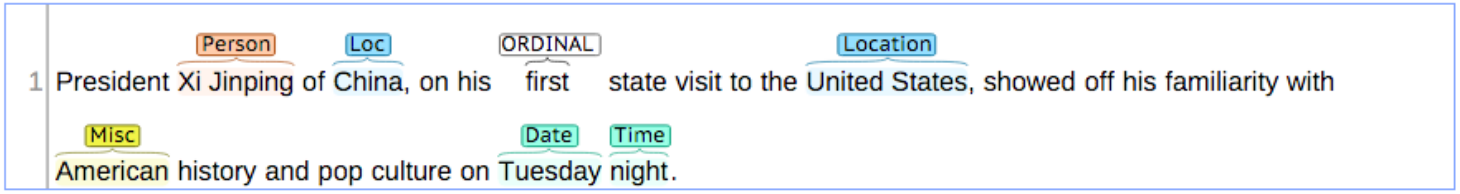
Stanford CoreNLP integrates many of Stanfords NLP tools, including the part-of-speech (POS) tagger, the named entity recognizer (NER), the parser, the coreference resolution system, sentiment analysis, bootstrapped pattern learning, and the open information extraction tools.



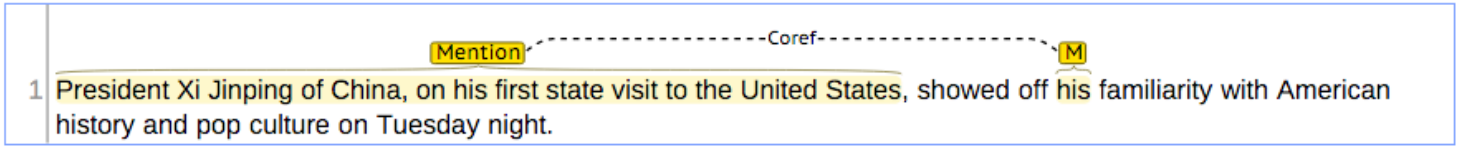
http://stanfordnlp.github.io/CoreNLP/

Stanford CoreNLP

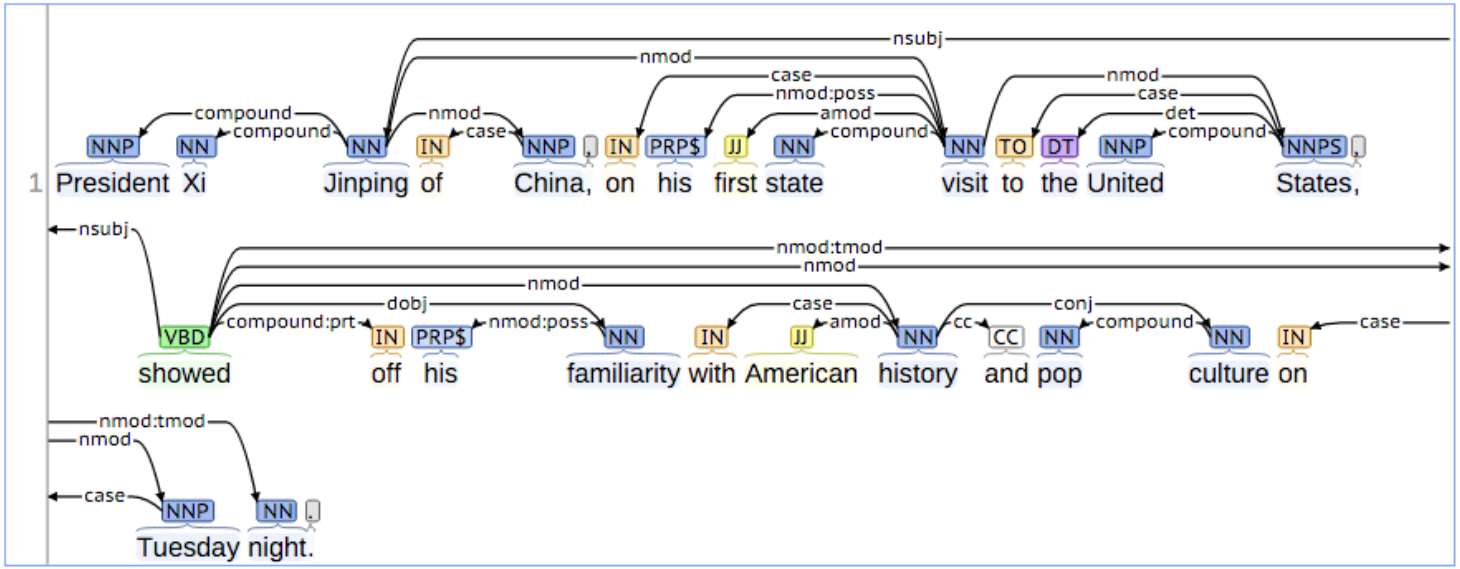
Named Entity Recognition:



Coreference:



Basic Dependencies:



http://stanfordnlp.github.io/CoreNLP/

CoreNLP R-package

```r
# install relevant packages devtools::install_github('statsmaths/coreNLP') this installs
# the full stanford corenlp coreNLP::downloadCoreNLP()
library(rJava)
library(coreNLP)
# creating an instance (opening the pipeline)
initCoreNLP()
str <- c("President George Bush spoke to the troops on board of an aircraft carrier in the Gulf of Mexico.")
output <- annotateString(str)
getToken(output)
```

```
##    sentence id     token     lemma CharacterOffsetBegin CharacterOffsetEnd POS      NER
## 1         1  1 President President                    0                  9 NNP        O
## 2         1  2    George    George                   10                 16 NNP   PERSON
## 3         1  3      Bush      Bush                   17                 21 NNP   PERSON
## 4         1  4     spoke     speak                   22                 27 VBD        O
## 5         1  5        to        to                   28                 30  TO        O
## 6         1  6       the       the                   31                 34  DT        O
## 7         1  7    troops    troops                   35                 41 NNS        O
## 8         1  8        on        on                   42                 44  IN        O
## 9         1  9     board     board                   45                 50  NN        O
## 10        1 10        of        of                   51                 53  IN        O
## 11        1 11        an         a                   54                 56  DT        O
## 12        1 12  aircraft  aircraft                   57                 65  NN        O
## 13        1 13   carrier   carrier                   66                 73  NN        O
## 14        1 14        in        in                   74                 76  IN        O
## 15        1 15       the       the                   77                 80  DT        O
## 16        1 16      Gulf      Gulf                   81                 85 NNP LOCATION
## 17        1 17        of        of                   86                 88  IN LOCATION
## 18        1 18    Mexico    Mexico                   89                 95 NNP LOCATION
## 19        1 19         .         .                   95                 96   .        O
##    Speaker
## 1     PER0
## 2     PER0
## 3     PER0
## 4     PER0
## 5     PER0
```

Other CoreNLP features

```
getDependency(output)
```

```
##    sentence governor dependent     type governorIdx dependentIdx govIndex depIndex
## 1         1     ROOT     spoke     root           0            4       NA        4
## 2         1     Bush President compound           3            1        3        1
## 3         1     Bush    George compound           3            2        3        2
## 4         1    spoke      Bush    nsubj           4            3        4        3
## 5         1   troops        to     case           7            5        7        5
## 6         1   troops       the      det           7            6        7        6
## 7         1    spoke    troops  nmod:to           4            7        4        7
## 8         1    board        on     case           9            8        9        8
## 9         1   troops     board  nmod:on           7            9        7        9
## 10        1  carrier        of     case          13           10       13       10
## 11        1  carrier        an      det          13           11       13       11
## 12        1  carrier  aircraft compound          13           12       13       12
## 13        1    board   carrier  nmod:of           9           13        9       13
## 14        1     Gulf        in     case          16           14       16       14
## 15        1     Gulf       the      det          16           15       16       15
## 16        1    spoke      Gulf  nmod:in           4           16        4       16
## 17        1   Mexico        of     case          18           17       18       17
## 18        1     Gulf    Mexico  nmod:of          16           18       16       18
## 19        1    spoke         .    punct           4           19        4       19
```

```
getSentiment(output)
```

```
##   id sentimentValue sentiment
## 1  1              1  Negative
```

NER for Lazy People: OpenNLP Access via Web API

```
url <- "http://www.huffingtonpost.com/entry/house-republicans-ethics_us_586bdb14e4b0de3a08f99e66?6ztihpvi"
api <- "http://juicer.herokuapp.com/api/article?url="
target <- paste(api, url, sep = "")

raw.data <- readLines(target, warn = "F")

rd <- fromJSON(raw.data)
dat <- rd$article

ENTITIES <- data.frame(do.call("rbind", dat$entities))

ENTITIES[1:10, ]
```

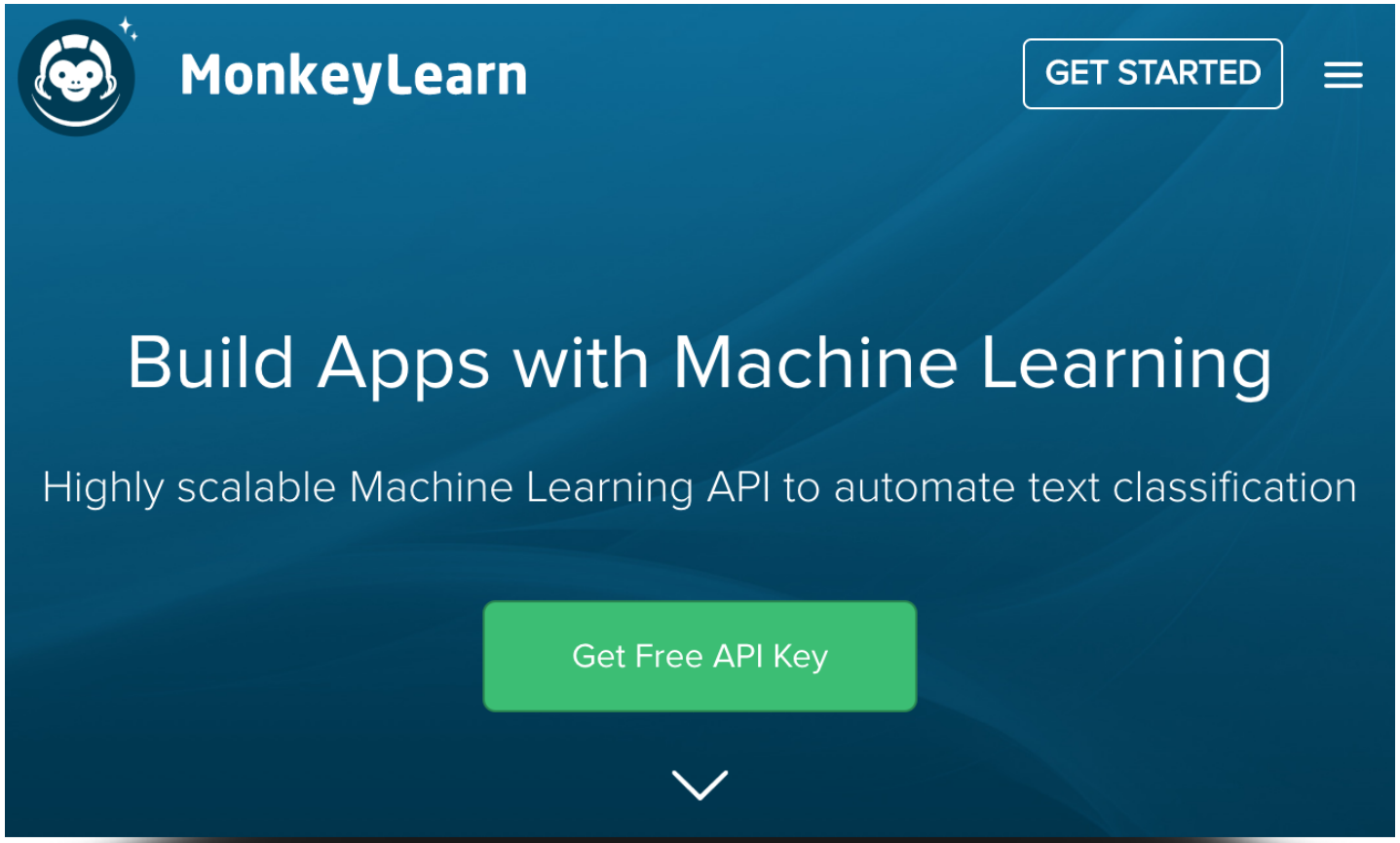
```
##            type                                                        text frequency
## 1        Person                                                       Trump         1
## 2  Organization Campaign for Accountability , Citizens for Responsibility         1
## 3        Person                                                 Nancy Pelosi         1
## 4  Organization                                                       House         3
## 5  Organization                                                People 's House         1
## 6        Person                                                   Goodlatte         1
## 7      Location                                                  Washington         2
## 8        Person                                                    Paul Ryan         1
## 9      Location                                                  WASHINGTON         1
## 10       Person                                                      Pelosi         1
```

$\Rightarrow$ this works off the Stanford NLP NER module, which we will also directly use in R.

NER for Lazy People (2): MonkeyLearn

- Can leverage existing extractors made available on MonkeyLearn platform.
- MonkeyLearn is a platform through which classifiers for very specific tasks are shared by other data scientists.
- They produce benchmark estimates on the performance of extractors and classifiers that allows you to compare different products.
- This is very useful resource for evaluation of your own proto-types against other classifiers.

NER for Lazy People (2): MonkeyLearn



Customers

http://monkeylearn.com/

NER for Lazy People (2): MonkeyLearn

MonkeyLearn | + Create Module | Explore | My Modules | Help | Thiemo Fetzer

Entity Extractor Public

★ Star 43

Extract entities from text.

Description | API

Description
Extract Entities from text using Named Entity Recognition (NER). NER labels sequences of words in a text which are the names of things, such as person and company names. This implementation labels 3 classes: PERSON, ORGANIZATION and LOCATION

Implementation
This NER tagger is implemented using Conditional Random Field (CRF) sequence models and is trained over a huge amount of data.

Example
Input:

```
In the 19th century, the major European powers had gone to great lengths to maintain a balance of powe
r throughout Europe, resulting in the existence of a complex network of political and military allianc
es throughout the continent by 1900.[7] These had started in 1815, with the Holy Alliance between Prus
sia, Russia, and Austria. Then, in October 1873, German Chancellor Otto von Bismarck negotiated the Le
ague of the Three Emperors (German: Dreikaiserbund) between the monarchs of Austria-Hungary, Russia an
d Germany.
```

Output:

```
HTTP/1.1 200 OK
Content-Type: application/json
X-Query-Limit-Limit: 1000
X-Query-Limit-Remaining: 998
X-Query-Limit-Request-Queries: 1
```


```
{
  "result": [
    {
      "count": 1,
      "tag": "LOCATION",
      "entity": "Europe"
    },
    {
      "count": 1,
      "tag": "LOCATION",
      "entity": "Prussia"
    },
```


http://monkeylearn.com/

NER for Lazy People (3): MonkeyLearn

```r
# set up an account on Monkeylearn and obtain an API reference 100k calls per month are for
# free devtools::install_github('ropensci/monkeylearn') set API key as environment variable
# Sys.setenv(MONKEYLEARN_KEY='')

library("monkeylearn")
text <- "In the 19th century, the major European powers had gone to great lengths to maintain a balance of pow
alliances throughout the continent by 1900. These had started in 1815, with the Holy Alliancebetween Prussia,
Then, in October 1873, German Chancellor Otto von Bismarck negotiated the League of the Three Emperors (Germa
output <- monkeylearn_extract(request = text, extractor_id = "ex_isnnZRbS")
output
```

```
##   count      tag            entity                          text_md5
## 1     1 LOCATION            Europe 97b50d3cf5012f5ba4aa2d40117da521
## 2     1 LOCATION           Prussia 97b50d3cf5012f5ba4aa2d40117da521
## 3     1 LOCATION   Austria-Hungary 97b50d3cf5012f5ba4aa2d40117da521
## 4     1 LOCATION           Austria 97b50d3cf5012f5ba4aa2d40117da521
## 5     1 LOCATION           Germany 97b50d3cf5012f5ba4aa2d40117da521
## 6     1   PERSON Otto von Bismarck 97b50d3cf5012f5ba4aa2d40117da521
## 7     2 LOCATION            Russia 97b50d3cf5012f5ba4aa2d40117da521
```