Named Entity Recognition

Named Entity Recognition (NER) aims to classify proper names, dates/times, measures, and stylized terms.

**Proper Names:** Common named entities are people, companies, and organizations. Ex: *Elvis Presley, IBM, Department of the Interior.*

**Dates/Times:** Dates and times appear in a wide variety of forms. Ex: *November 9, 1997, 11/9/97, 10:29 pm.*

**Measures:** Measures can usually be identified by looking for units. Ex: *45%, 5.3 lbs, 65 mph, $1.4 billion*

**Other:** Application-specific needs may look for other stylized terms. Ex: *URLs, email addresses, phone numbers, social security numbers.*

Difficulties

- No dictionary will contain all existing proper names.
- New proper names are constantly being created.
- The same real-world entity can use many variants of the same proper name.
- Proper names are often abbreviated or turned into acronyms.
- Not all acronyms are proper names.

Proper Nouns Identification

Just recognizing proper nouns can be tricky because:

- In mixed case text, the first word of each sentence is capitalized.
- Some text is not written in mixed case (e.g., headlines, teletype reports).
- Spoken language does not indicate case.
- In some languages, case does not signal proper nouns (e.g., Chinese, Japanese, German)

Proper Name Ambiguity

There is a surprising amount of ambiguity even among proper names. For example:

- **People vs. Companies:** Ford, John Hancock, Philip Morris
- **People vs. Locations:** Jordan, Washington, Georgia, JFK (person vs. airport)
- **People vs. Months:** April, June
- **People vs. Organizations:** Brigham Young (person vs. university)
- **Acronyms vs. Organizations:** MRI (Magnetic Resonance Imaging vs. Mental Research Institute)
- **Other:** May, Mark, Turkey, Calvin and Hobbes
(Minimum) Preprocessing required for NER

- Tokenization
- Sentence segmentation
- POS tagging
- NP chunking (for proper names)

Three Common Approaches

- Hand-coded rules
  - Good: Usually performs the best, especially for specialized applications.
  - Bad: Expensive to build. Domain-specific.

- Machine learning
  - Good: Can be easily adapted for new domains.
  - Bad: Still need domain-specific annotated training corpus.

- Multilingual
  - Good: May work for a variety of languages.
  - Bad: May not work as well as a language-specific system.

Hand-crafted NER Systems

Rule-based NER can be very effective, but requires a lot of manual effort.

- Acquiring or building lists of people, companies, locations, organizations, etc.

- Rules to classify proper names often consider capitalization, lists, keywords, and local context.

  Examples of Keywords:
  - person titles (e.g., Mr., Jr., Ph.D.)
  - company designators (e.g., Corp., Inc., Co.)

- Special rules to classify other entities often look for punctuation marks (e.g., 10:06), special symbols (e.g., % or $), and surface structure (e.g., 10/08/97).

Examples of Contextual Patterns

- \{TITLE\}\{PERSON\}
  - Ex: “U.S. President George Bush”, “Mr. Frank Leonard”

- \{PERSON\}, the \{TITLE\} of \{ORGANIZATION\}
  - Ex: “Fred Martin, the CEO of XYZ Corp.”

- \{PERSON\} joined \{COMPANY\}
  - Ex: “Mary Smith joined Microsoft.”

- headquarters in \{LOCATION\}
  - Ex: “headquarters in London”

- \{LOCATION\}, \{LOCATION\}
  - Ex: “Salt Lake City, Utah”
Surface Structure

Some entities can be identified solely through surface structure patterns. For example:

- Social Security Number: ###-##-####
- Phone Number: (###) ###-####
- Date: ##/##/####
- URL: www.xxxxxx.xxx/xxx/xxxx.html
- Email Address: xxxxx@xxx.xxxxx.xxx

Machine Learning Approaches

- requires large annotated training corpus
- uses statistics or other learning algorithms
- doesn’t require human intuition, but may produce odd classifications
- can re-train for each domain (if you have or create annotated corpus for that domain)

A Bootstrapping Model for Named Entity Recognition (Collins & Singer)

- Bootstrapping methods learn information from unannotated texts using a few seed examples or rules to kickstart the process.

- Collins & Singer use a seed decision list of rules that are matched against the training text to label some examples.

- Collins & Singer developed a technique that exploits redundancy and makes a co-training assumption that there are two independent ways to recognize named entities: by looking for words in the NPs themselves (Spelling Rules) and by looking for phrases around them (Context Rules).

  An example spelling rule: CONTAINS("Mr.") → PERSON
  An example context rule: FULLSTRING("city of") → LOCATION.

Data

- Collins & Singer used training instances derived from nearly a million sentences from the New York Times.

- Each sentence was parsed and an instance was extracted if a word sequence satisfied the following criteria:
  1. The words are consecutive proper nouns inside an NP, and the last word is the head of the NP.
  2. The NP either (1) appears with an appositive modifier, whose head is a singular noun, or (2) is inside a PP that attaches to an NP whose head is a singular noun.

    The context then becomes either (1) the appositive NP or (2) the NP to which the PP attached, coupled with the preposition.
NP/Context Pairs

But Robert Jordan, a partner at Steptoe & Johnson who took ...
NP = Robert Jordan  Context = a partner

By hiring a company like AT&T ...
NP = AT&T  Context = company like

Hanson acquired Kidde Incorporated, parent of Kidde Credit, ...
NP = Kidde Incorporated  Context = parent
NP = Kidde Credit  Context = parent of

Decision Lists

• A decision list is an ordered list of rules that assign classes to instances.

• Given an instance to classify, the rules are applied to the instance, in order, until one rule matches. Then that rule's label is assigned to the instance.

• If none of the rules match the instance, then the instance is left unlabeled.

An Example Decision list

If FullString = “New York” → LOCATION
If FullString = “California” → LOCATION
If Contains(“Mr.”) → PERSON
If Contains(“Corp.”) → COMPANY
If Contains(“Inc.”) → COMPANY
If Contains(“Co.”) → COMPANY
If Contains(“Michael”) → PERSON
If FullString = “Jordan” → LOCATION

Rule Predicates

These are some of the predicates that Collins & Singer used, which were matched against a text fragment:

FullString=<string>  Exact string match
Contains(<word>)  Matches if the word is contained in the text
AllCap1  Matches if the text is a single word that is all capitals (e.g., “IBM”)
AllCap2  Matches if the text is a single word that consists only of capitals and periods, with at least one period (e.g., “N.Y.”)
NonAlpha=<string>  Matches if the text contains the given non-alphabetic characters, in order, when the alphabetic characters are removed. (e.g., “..&.” matches “A.T.&T.”)
(Approximation of) the Collins & Singer Learning Model

1. \(N = 5\)
2. Spelling\_Dlist = Seed\_Dlist
3. Labeled\_Instances = Apply(Spelling\_Dlist, Instances)
4. Temp\_Context\_Dlist = Induce\_Dlist(\textbf{CONTEXT}, Labeled\_Instances)
5. Context\_Dlist = Get\_Best\_Rules(Temp\_Context\_Dlist, \(N\))
6. Labeled\_Instances = Apply(Context\_Dlist, Instances)
7. Temp\_Spelling\_Dlist = Induce\_Dlist(\textbf{SPELLING}, Labeled\_Instances)
8. Spelling\_Dlist = Get\_Best\_Rules(Temp\_Spelling\_Dlist, \(N\))
9. \(N = N + 5\)
10. IF ((\(N > \text{MAX\_RULES}\)) or (\(\text{NUM\_ITERATIONS} = \text{MAX\_ITERATIONS}\)))
    THEN Create\_Final\_Dlist(Spelling\_Dlist, Context\_Dlist, Instances)
    ELSE Go to Step #3

Inducing a Decision List

1. Generate all possible rules that could apply to the training data.
2. Compute the frequency of each rule by counting the number of times that it applies to the training instances (regardless of whether the rule’s class matches).
3. Compute the probability of each rule as:
   \[P(\text{rule}_i) = P(\text{rule}_i's\ class | \text{rule}_i\ applies)\]
4. Discard all rules with frequency < \text{MIN\_FREQ} or probability < \text{MIN\_PROB}.
5. Sort the remaining rules by their frequency.

Accuracy

- Collins & Singer evaluated their model on about 88,000 NP/Context pairs for training and 1,000 for testing.
- They trained on 3 named entity classes: LOCATION, PERSON, ORGANIZATION.
- Their system got 83.3% accuracy.
- Their system got 91.3% accuracy when the OTHER gold standard instances were removed from the test set.

Conclusions

- Named Entity Recognition can perform quite well! NER is one of the most successful areas of NLP.
- But, named entities are so common that even 90% accuracy means that a lot of mistakes still exist.
- For a specific application, NER systems often need to be tweaked to perform satisfactorily.
  - Mistagging one common domain entity can be a disaster.
    Ex: imagine tagging “Jordan” as a country in sports articles!
- Beware of downloading large lists without checking the entries. Ambiguous terms must be filtered or they will cause you grief!
  Ex: “Mark” as a name, “Jordan” as a location, “Apple” as a company.