Lecture 24:
Information Extraction
(Sequence Labeling, Named Entity Recognition, Relation Extraction)

Julia Hockenmaier
juliahmr@illinois.edu
3324 Siebel Center
Sequence Labeling
Pierre Vinken, 61 years old, will join IBM’s board as a nonexecutive director Nov. 29.

Pierre_NNP Vinken_NNP ,_, 61_CD years_NNS old_JJ ,_, will_MD join_VB IBM_NNP ‘s_POS board_NN as_IN a_DT nonexecutive_JJ director_NN Nov._NNP 29_CD _. 

**Task:** assign POS tags to words
Pierre Vinken, 61 years old, will join IBM’s board as a nonexecutive director Nov. 29.

Task: identify all non-recursive NP chunks
The BIO encoding

We define three new tags:

- **B-NP**: beginning of a noun phrase chunk
- **I-NP**: inside of a noun phrase chunk
- **O**: outside of a noun phrase chunk

Example:

```
Pierre_B-NP Vinken_I-NP ,_O 61_B-NP years_I-NP old_O ,_O will_O join_O IBM_B-NP ’s_O board_B-NP as_O a_B-NP nonexecutive_I-NP director_I-NP Nov._B-NP 29_I-NP ._O
```
Shallow parsing

Pierre Vinken, 61 years old, will join IBM’s board as a nonexecutive director Nov. 29.

[NP Pierre Vinken], [NP 61 years] old, [VP will join] [NP IBM] ‘s [NP board] [PP as] [NP a nonexecutive director] [NP Nov. 2].

**Task:** identify all non-recursive NP, verb (“VP”) and preposition (“PP”) chunks
The BIO encoding for shallow parsing

We define several new tags:

- **B-NP B-VP B-PP**: beginning of an NP, “VP”, “PP” chunk
- **I-NP I-VP I-PP**: inside of an NP, “VP”, “PP” chunk
- **O**: outside of any chunk

```
[NP Pierre Vinken], [NP 61 years] old, [VP will join] [NP IBM] ‘s [NP board] [PP as] [NP a nonexecutive director] [NP Nov. 2].
```

```
Pierre_B-NP Vinken_I-NP ,_O 61_B-NP years_I-NP old_O ,_O will_B-VP join_I-VP IBM_B-NP ‘s_O board_B-NP as_B-PP a_B-NP nonexecutive_I-NP director_I-NP Nov._B-NP 29_I-NP ._O
```
Pierre Vinken, 61 years old, will join IBM’s board as a nonexecutive director Nov. 29.

Task: identify all mentions of named entities (people, organizations, locations, dates)
The BIO encoding for NER

We define many new tags:

- **B-PERS, B-DATE, …:** beginning of a mention of a person/date...
- **I-PERS, I-DATE, …:** inside of a mention of a person/date...

Example:

```
[PERS Pierre Vinken] , 61 years old , will join [ORG IBM] ‘s board as a nonexecutive director [DATE Nov. 2] .
```

```
Pierre_B-PERS Vinken_I-PERS ,_O 61_0 years_0 old_0 ,_O will_0 join_0 IBM_B-ORG ‘s_0 board_0 as_0 a_0 nonexecutive_0 director_0 Nov._B-DATE 29_I-DATE ._O
```
Sequence Labeling

**Input:** a sequence of $n$ tokens/words:

Pierre Vinken, 61 years old, will join IBM’s board as a nonexecutive director Nov. 29

**Output:** a sequence of $n$ labels, such that each token/word is associated with a label:

**POS-tagging:** Pierre\_NNP Vinken\_NNP ,\_ , 61\_CD years\_NNS old\_JJ ,\_ , will\_MD join\_VB IBM\_NNP ‘s\_POS board\_NN as\_IN a\_DT nonexecutive\_JJ director\_NN Nov.\_NNP 29\_CD .\_.

**Named Entity Recognition:** Pierre\_B-PERS Vinken\_I-PERS ,\_O 61\_O years\_O old\_O ,\_O will\_O join\_O IBM\_B-ORG ‘s\_O board\_O as\_O a\_O nonexecutive\_O director\_O Nov.\_B-DATE 29\_I-DATE .\_O
BIO encodings in general

BIO encoding can be used to frame any task that requires the identification of non-overlapping and non-nested text spans as a sequence labeling problem, e.g.:

- NP chunking
- Shallow Parsing
- Named entity recognition
Sequence labeling algorithms

Statistical models:
  — Maximum Entropy Markov Models (MEMMs)
  — Conditional Random Fields (CRFs)

Neural models:
  — Recurrent networks (or transformers) that predict a label at each time step, possibly with a CRF output layer.
Maximum Entropy Markov Models

MEMMs use a **logistic regression** ("Maximum Entropy") classifier for each $P(t^{(i)} | w^{(i)}, t^{(i-1)})$

$$P(t^{(i)} = t_k | t^{(i-1)}, w^{(i)}) = \frac{\exp(\sum_j \lambda_{jk} f_j(t^{(i-1)}, w^{(i)}))}{\sum_l \exp(\sum_j \lambda_{jl} f_j(t^{(i-1)}, w^{(i)}))}$$

Here, $t^{(i)}$: label of the i-th word vs. $t_i = i$-th label in the inventory

This requires the definition of a **feature function** $f(t^{(i-1)}, w^{(i)})$ that returns an $n$-dimensional **feature vector** for predicting label $t^{(i)} = t_j$ given inputs $t^{(i-1)}$ and $w^{(i)}$

Training returns weights $\lambda_{jk}$ for each feature $j$ used to predict label $t_k$
Conditional Random Fields (CRFs)

Conditional Random Fields have the same mathematical definition as MEMMs, but:

— CRFS are trained globally to maximize the probability of the overall sequence,
— MEMMs are trained locally to maximize the probability of each individual label

This requires dynamic programming
— Training: akin to the Forward-Backward algorithm used to train HMMs from unlabeled sequences)
— Decoding: Viterbi
Named Entity Recognition (NER)
## Named Entity Types

These types were developed for the news domain as part of NIST’s Automatic Content Extraction (ACE) program.

Other domains (e.g. biomedical text) require different types (proteins, genes, diseases, etc.)

<table>
<thead>
<tr>
<th>Type</th>
<th>Tag</th>
<th>Sample Categories</th>
<th>Example sentences</th>
</tr>
</thead>
<tbody>
<tr>
<td>People</td>
<td>PER</td>
<td>people, characters</td>
<td>Turing is a giant of computer science.</td>
</tr>
<tr>
<td>Organization</td>
<td>ORG</td>
<td>companies, sports teams</td>
<td>The IPCC warned about the cyclone.</td>
</tr>
<tr>
<td>Location</td>
<td>LOC</td>
<td>regions, mountains, seas</td>
<td>The Mt. Sanitas loop is in Sunshine Canyon.</td>
</tr>
<tr>
<td>Geo-Political Entity</td>
<td>GPE</td>
<td>countries, states, provinces</td>
<td>Palo Alto is raising the fees for parking.</td>
</tr>
<tr>
<td>Facility</td>
<td>FAC</td>
<td>bridges, buildings, airports</td>
<td>Consider the Golden Gate Bridge.</td>
</tr>
<tr>
<td>Vehicles</td>
<td>VEH</td>
<td>planes, trains, automobiles</td>
<td>It was a classic Ford Falcon.</td>
</tr>
</tbody>
</table>

**Figure 18.1** A list of generic named entity types with the kinds of entities they refer to.

---

18.1.1 NER as Sequence Labeling

The standard algorithm for named entity recognition is as a word-by-word sequence labeling task, in which the assigned tags capture both the boundary and the type. A sequence classifier like an MEMM/CRF, a bi-LSTM, or a transformer is trained to label the tokens in a text with tags that indicate the presence of particular kinds of named entities. Consider the following simplified excerpt from our running example.

- **ORG** American Airlines, a unit of **ORG** AMR Corp., immediately matched the move, spokesman **PER** Tim Wagner said.
Features for NER

Lists of common names exist for many entities
  — Gazetteers (place names, www.geonames.org),
  — Census-derived lists of first names and surnames,
  — Genes, proteins, diseases, etc.
  — Company names

Such lists can be helpful, but:

... Zipf’s Law: these lists are typically not exhaustive,
  (and the distribution of names has a long tail)

... Ambiguity: many entity names either refer to different types of entities (Washington: person, places named after the person), or are used to refer to different types of entity (metonymy: Washington as reference to the US government)
Feature-based NER

identity of $w_i$, identity of neighboring words
embeddings for $w_i$, embeddings for neighboring words
part of speech of $w_i$, part of speech of neighboring words
base-phrase syntactic chunk label of $w_i$ and neighboring words
presence of $w_i$ in a gazetteer

$w_i$ contains a particular prefix (from all prefixes of length $\leq 4$)
$w_i$ contains a particular suffix (from all suffixes of length $\leq 4$)
$w_i$ is all upper case
word shape of $w_i$, word shape of neighboring words
short word shape of $w_i$, short word shape of neighboring words
presence of hyphen

Figure 18.5 Typical features for a feature-based NER system.

Train a sequence labeling model (MEMM or CRF), using features such as the ones listed above for English

— Word Shape: replace all upper-case letters with one symbol (e.g. “X”), all lower-case letters with another symbol (“x”), all digits with another symbol (“d”), and leave punctuation marks as is (“L’Occitane → “X’Xxxxxxxx”)

— Short Word Shape: remove adjacent letters that are identical in word shape “L’Occitane → “X’Xxxxxxxx” → “X’Xx””)
Neural NER

**Sequence RNN** (e.g. biLSTM or Transformer) with a CRF output layer.

**Input:** word embeddings, possibly concatenated with character embeddings and other features, e.g.:

![Diagram](blabla.png)

**Figure 18.8** Putting it all together: character embeddings and words together in a bi-LSTM sequence model. After Lample et al. (2016).

18.1.4 Rule-based NER

While machine learned (neural or MEMM/CRF) sequence models are the norm in academic research, commercial approaches to NER are often based on pragmatic combinations of lists and rules, with some smaller amount of supervised machine learning (Chiticariu et al., 2013). For example IBM System T is a text understanding architecture in which a user specifies complex declarative constraints for tagging tasks in a formal query language that includes regular expressions, dictionaries, semantic constraints, NLP operators, and table structures, all of which the system compiles into an efficient extractor (Chiticariu et al., 2018).

One common approach is to make repeated rule-based passes over a text, allowing the results of one pass to influence the next. The stages typically first involve the use of rules that have extremely high precision but low recall. Subsequent stages employ more error-prone statistical methods that take the output of the first pass into account.

1. First, use high-precision rules to tag unambiguous entity mentions.
2. Then, search for substring matches of the previously detected names.
3. Consult application-specific name lists to identify likely name entity mentions from the given domain.
4. Finally, apply probabilistic sequence labeling techniques that make use of the tags from previous stages as additional features.

The intuition behind this staged approach is twofold. First, some of the entity mentions in a text will be more clearly indicative of a given entity’s class than others. Second, once an unambiguous entity mention is introduced into a text, it is likely that subsequent shortened versions will refer to the same entity (and thus the same type of entity).

18.1.5 Evaluation of Named Entity Recognition

The familiar metrics of recall, precision, and $F_1$ measure are used to evaluate NER systems. Remember that recall is the ratio of the number of correctly labeled responses to the total that should have been labeled; precision is the ratio of the number of correctly labeled responses to the total number of responses made.

CS47 Natural Language Processing (J. Hockenmaier) https://courses.grainger.illinois.edu/cs447/ 19
Rule-based NER

The textbook gives an example of an iterative approach that makes multiple passes over the text:

— Pass 1: Use high-precision rules to label (a small number of) unambiguous mentions

— Pass 2: Propagate the labels of the previously detected named entities to any mentions that are substrings (or acronyms?) of these entities

— Pass 3: Use application-specific name lists to identify further likely names (as features?)

— Pass 4: Now use a sequence labeling approach for NER, keeping the already labeled entities as high-precision anchors.

The basic ideas behind this approach (label propagation, using high-precision items as anchors) can be useful for other tasks as well.
Relations and Relation Extraction
WordNet as a database of relations between *concepts*

**Hyponym relations** *(is-a relation)*
- cats are mammals

**Meryonym relations** *(part-of/has-a relations):*
  - **Part meronyms:** bumpers are parts of cars, cars have bumpers
  - **Member meronyms:** musicians belong to bands/orchestras,
  - **Substance meronyms:** dough contains flour

NB: some of these are inherited via hypernyms:
- ‘musician’ is a member meronym of ‘musical organization’,
- which has hyponyms such as ‘orchestra’, ‘band’, ‘choir’, etc.
Domain knowledge expressed as relations

Wikipedia’s **infoboxes** provide structured facts about **named entities**:  

These can be turned into **structured relations** between these entities, e.g.  

- location-of(UIUC, Illinois)  

or **RDF** (Resource Description Framework) **triples**  

\[(entity, relation, entity)\]:  

(UIUC, location, Illinois)

**Freebase** and **DBPedia** (2 billion RDF triples) are both very large knowledge bases of such relations, extracted from Wikipedia.
Relation Extraction from text

Citing high fuel prices, [ORG United Airlines] said [TIME Friday] it has increased fares by [MONEY $6] per round trip on flights to some cities also served by lower-cost carriers. [ORG American Airlines], a unit of [ORG AMR Corp.], immediately matched the move, spokesman [PER Tim Wagner] said. [ORG United], a unit of [ORG UAL Corp.], said the increase took effect [TIME Thursday] and applies to most routes where it competes against discount carriers, such as [LOC Chicago] to [LOC Dallas] and [LOC Denver] to [LOC San Francisco].

Can we identify that…
…American Airlines is part of (a unit of) AMR,
…United Airlines is part of (a unit of) UAL Corp,
…Tim Wagner is employed by (a spokesman of) AMR
Relation Extraction from text

Identify **relations between named entities**, typically from a small set of predefined relations.

<table>
<thead>
<tr>
<th>Relations</th>
<th>Types</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Physical-Located</td>
<td>PER-GPE</td>
<td>He was in <strong>Tennessee</strong></td>
</tr>
<tr>
<td>Part-Whole-Subsidiary</td>
<td>ORG-ORG</td>
<td>XYZ, the parent company of <strong>ABC</strong></td>
</tr>
<tr>
<td>Person-Social-Family</td>
<td>PER-PER</td>
<td>Yoko’s husband <strong>John</strong></td>
</tr>
<tr>
<td>Org-AFF-Founder</td>
<td>PER-ORG</td>
<td><strong>Steve Jobs</strong>, co-founder of <strong>Apple</strong>...</td>
</tr>
</tbody>
</table>

The 17 relations (orange) used in ACE:
A logical interpretation

We can construct a model for these relations:

- The **domain** (universe) is a set of named entities, partitioned into different types or classes of entities
- Each **relation** is a set of tuples of entities (restricted to relation-specific tuples of types)

<table>
<thead>
<tr>
<th>Domain</th>
<th>Classes</th>
<th>Relations</th>
</tr>
</thead>
<tbody>
<tr>
<td>United, UAL, American Airlines, AMR</td>
<td>United, UAL, American, and AMR are organizations</td>
<td>United is a unit of UAL</td>
</tr>
<tr>
<td>Tim Wagner</td>
<td>Tim Wagner is a person</td>
<td>PartOf = {\langle a, b \rangle, \langle c, d \rangle }</td>
</tr>
<tr>
<td>Chicago, Dallas, Denver, and San Francisco</td>
<td>Chicago, Dallas, Denver, and San Francisco are places</td>
<td>American is a unit of AMR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tim Wagner works for American Airlines</td>
</tr>
<tr>
<td></td>
<td></td>
<td>United serves Chicago, Dallas, Denver, and San Francisco</td>
</tr>
</tbody>
</table>

\[ \mathcal{D} = \{ a, b, c, d, e, f, g, h, i \} \]

\[ \mathcal{D} = \{ a, b, c, d \} \]

\[ \mathcal{D} = \{ e \} \]

\[ \mathcal{D} = \{ f, g, h, i \} \]
Rule-based relation extraction

Handwritten rules to identify **lexico-syntactic patterns** (Hearst, 1992) can be used for high-precision (and low-recall) relation extraction:

Agar is a substance prepared from a mixture of **red algae, such as Gelidium**, for laboratory or industrial use

The pattern “X, such as Y (and/or Z)” implies that X is a hypernym of Y and Z.

```
NP {, NP}* {,} (and|or) other NP_H  
NP_H such as {NP,*} {(or|and)} NP  
such NP_H as {NP,*} {(or|and)} NP  
NP_H {,} including {NP,*} {(or|and)} NP  
NP_H {,} especially {NP,*} {(or|and)} NP  
```

temples, treasuries, and other important **civic buildings**
red algae such as Gelidium
such **authors** as Herrick, Goldsmith, and Shakespeare
common-law **countries**, including Canada and England
European **countries**, especially France, England, and Spain

Figure 18.12  Hand-built lexico-syntactic patterns for finding hypernyms, using {} to mark optionality (Hearst 1992a, Hearst 1998).
Relation Extraction via supervised learning

Learn a classifier that identifies whether there is a relation between a pair of entities that appear in the same sentence (or nearby within a document).

Classifier output: $n+1$ classes for $n$ rels (incl. NONE)

Useful features:

— the words appearing in and next to the entities
— the words between the entities
— the NER types of both entities
— the distance between both entities (#words, #NERs, …)
— the syntactic path between the entities
Semi-supervised Relation Extraction

Use **high-precision seed patterns** (e.g. “X’s Y”) relations to identify **high-confidence seed tuples**.

Ryanair’s hub Charleroi -> (Ryanair, has-hub-in, Charleroi)

**Bootstrap a classifier** with increasing coverage:
- Find sentences containing entity pairs from seeds.
  - “Ryanair, which uses Charleroi as hub”
  - “Ryanair’s Belgian hub at Charleroi”
- These will contain new patterns (as well as some noise: “Sydney has a ferry hub at Circular Quay”)
- Noise needs to be controlled so as not to propagate (Confidence values, combined across patterns via noisy-or)
Distant Supervision for Relation Extraction

— Use a very large database of known relations (Freebase, DBPedia) to obtain a very large number of seed tuples.

  (John F. Kennedy, died-in, Dallas)
  (Princess Diana, died-in, Paris)
  (Elvis Presley, died-in, Memphis)

— Search large amounts of text for sentences containing pairs of entities in a known relation (plus entities in this list not in any known relation, to get no-relation examples)

— Process these sentences with NER, syntactic parsing, etc.

— Learn a classifier on these sentences to predict relations between entities that are not in the database

What is the intuition why this might work?
This returns a lot of noise: Elvis performed/lived/is buried in/sang about/… Memphis
But if trained on enough data, high-confidence predictions of this classifier are likely to be correct (since many true positive examples will be similar to each other)
Unsupervised Relation Extraction ("Open Information Extraction/IE")

Goal: Extract any relation (from large amounts of text, e.g. web) without being restricted to a predefined set of relations

Relations: Raw strings of words (often beginning with verbs, and possibly subject to some predefined syntactic constraints)

Example: The ReVerb algorithm:
— Run a POS tagger and entity chunker over each sentence
— Identify any potential relations (any string between entities that starts with a verb and obeys predefined constraints)
— Normalize relations (remove inflection, auxiliary verbs, adjectives, adverbs)
— Add relations that occur with at least N different arguments to database
— Train a classifier on small number (1000) hand-labeled sentences to obtain confidence scores for relations in the database.