

# CMPE 561 NLP

## Relation Detection And Classification

Melce Hüsünbeyi & Berna Erden

# Information Extraction

- The process of identifying the structured information from unstructured or semi-structured text.
- To be more exact, to detect the documents in a large database which satisfy the conditions of a specific information need.
- Wide range of applications in domains
  - biomedical literature mining,
  - business intelligence,
  - unstructured electronic text on Web, and namely question answering.

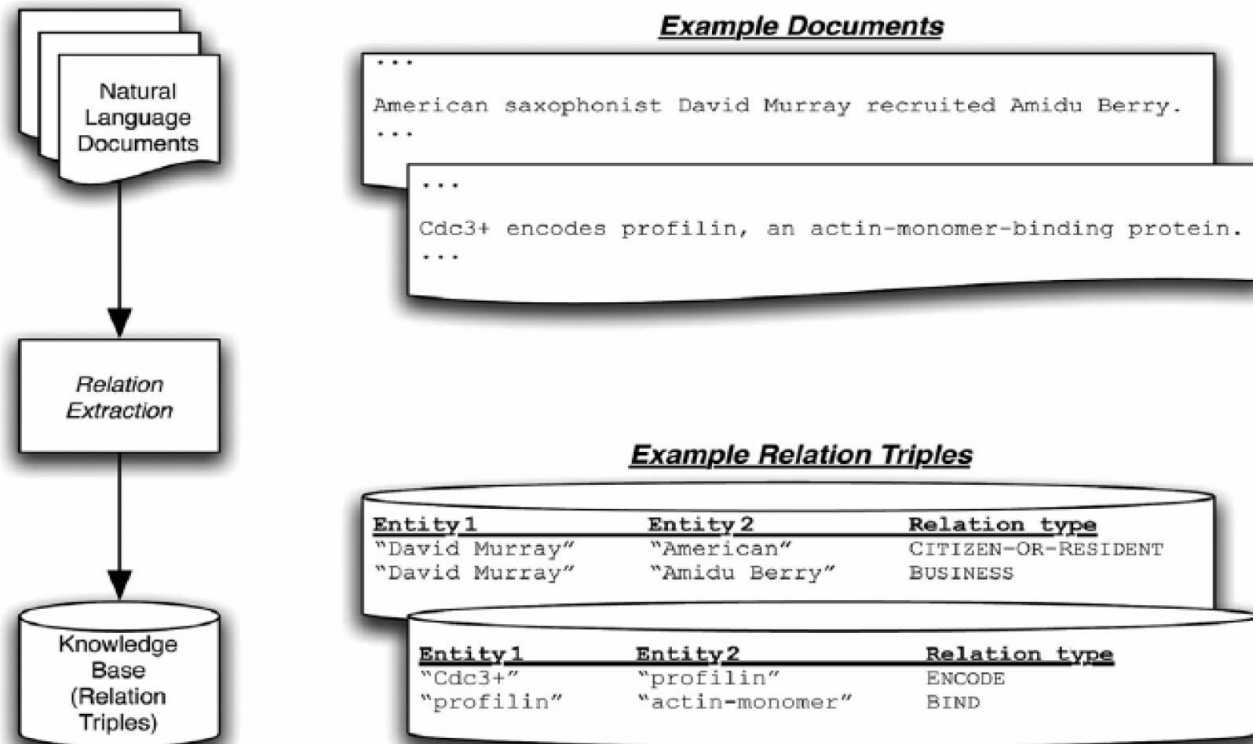
# Information Extraction

- Two principal tasks of information extraction:
  - Named Entity Recognition
  - Relation Extraction

# Relation Detection

- To detect and characterize the semantic relations between entities
- The input: natural language documents containing unstructured text.
- The RE system takes these documents for identify relations described in the text data and annotates them with a label describing the type of relation.
- The output: relation mention tuples, which consist of the entity mentions that take part in the relation and the relation type.

# Relation Extraction Task Pipeline



# Approach

- The most standard information extraction approach is for a pair of entities in the same sentence, whether we can classify the relation between the two entities as the predefined relation type, or not.
- A classification problem
- Techniques that have been proposed for the classification such as supervised and semi-supervised learning approaches.

# Supervised Learning Approaches

- Formulate the relation detection task as a binary classification
- For a sentence  $S = w_1, w_2, \dots, e_1, \dots, w_j, \dots, e_2, \dots, w_n$  where  $e_1$  and  $e_2$  are entities.  $f(\cdot)$  which is a mapping function:  
$$f_R(T(S)) = \begin{cases} +1 & \text{if } e_1 \text{ and } e_2 \text{ are related according to relation } R \\ -1 & \text{otherwise} \end{cases}$$
- Features extracted from sentence  $S$  are represented with  $T(S)$

# Supervised Learning Approaches

- According to existence of positive and negative relation examples,  $f$  (.) function composes discriminative classifiers like Support Vector Machines (SVMs) which classify any detected relation.
- The property of input to the classifier training distributes supervised approaches as feature based methods and kernel methods.



# Feature based methods - Lexical features

- the easiest way to cover evidence from contexts.
- Specific words that are between and enclosing the two entities
- The part-of-speech tags of these words such as nouns, verbs, adverbs, adjectives, numbers, foreign words
- Fixed window of  $k$  words before the first entity and their part of speech tags
- Fixed window of  $k$  words after the second entity and their part of speech tags
- Combination with the lexical features constitutes conjunctive feature.

# Feature based methods- Syntactic features

- Chunk base phrase paths
- Bags of chunk heads
- Dependency tree paths
- Constituent tree paths
- Tree distance between the arguments
- Existence of precise construction in a constituent structure
- Obtained from the parse trees of the sentence
- Another alternative method is deriving syntactic paths through trees.

# Feature based methods-Name entity tag features

- For the two entities
- The two candidate arguments
- Concatenation of the two entity types
- Head words of the arguments
- Bag of words from each of the arguments
- For instance, Stanford four class named entity tagger provides label for each word 'person, location, organization, miscellaneous, none'.

Some of the features which derived for classifying the relationship between 'American Airlines' and 'Tim Wagner'.

Entity-based features	
Entity <sub>1</sub> type	ORG
Entity <sub>1</sub> head	<i>airlines</i>
Entity <sub>2</sub> type	PERS
Entity <sub>2</sub> head	<i>Wagner</i>
Concatenated types	ORGPERS
Word-based features	
Between-entity bag of words	{ <i>a, unit, of, AMR, Inc., immediately, matched, the, move, spokesman</i> }
Word(s) before Entity <sub>1</sub>	NONE
Word(s) after Entity <sub>2</sub>	<i>said</i>
Syntactic features	
Constituent path	<i>NP ↑ NP ↑ S ↑ S ↓ NP</i>
Base syntactic chunk path	<i>NP → NP → PP → NP → VP → NP → NP</i>
Typed-dependency path	<i>Airlines ←<sub>subj</sub> matched ←<sub>comp</sub> said →<sub>subj</sub> Wagner</i>
<p><b>Figure 22.15</b> Sample of features extracted while classifying the &lt;American Airlines, Tim Wagner&gt; tuple.</p>	

# Kernel Methods

- When generating features involving long-range dependencies is computationally expensive and infeasible
- Kernel methods is alternative to feature-based methods
- A kernel function represents a similarity between the observations by computing the inner products of them.
- The observations can be defined as string, sequence of words, parse trees etc.

# Sequence-based Kernel Methods

- Defined over the shortest dependency paths between two arguments
- The similarity of dependency paths is measured by having nodes in common
- Nodes: words , part-of-speech tags, and entity types.
- Restricted to have the same length.
- For example, no common words, but in the same form and the same length; so a non-zero similarity value is obtained.
  - e.g. protestors -> seized <- stations
  - troops -> raided <- churches
  - Person -> VBD <- Facility

# Sequence-based Kernel Methods

- Considering context around the entities, possible to learn the relationship between two entities
- In a subsequence kernel function, the similarity between two sequences depends on their subsequences
- A sequence is divided into nodes, and each node is represented a feature vector.
- The inner product of the feature vectors gives the similarity between two nodes. Then, the similarity between two subsequences is calculated by the product the similarities of each pair of the nodes in the same position. In last step, a weighted sum of all the subsequence similarities shows the similarity between two sequences.

# Tree-based Kernel Methods

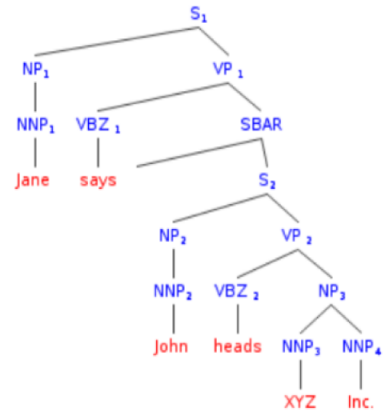
- The similarities between parse trees
- Parse tree is extracted for a given sentence, and subtrees contribute to nodes.
- If two subtrees have the matching attributes, the comparison of child-sequences is recursively processed.
- A weighted sum of the number of common subtrees gives the similarity value.



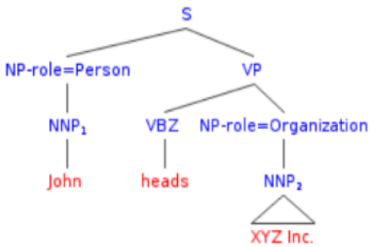
# Tree-based Kernel Methods

- Extending the idea to dependency parse trees that provide rich structural information like word identity, POS tags, phrase tags, entity type, entity level, relation delivers good performance .
- In the convolution tree kernels, the kernel function is computed by the production of vector space representations of parse trees .
- All kernel function defined can be trained using a classifier like SVM, Voted perceptron.

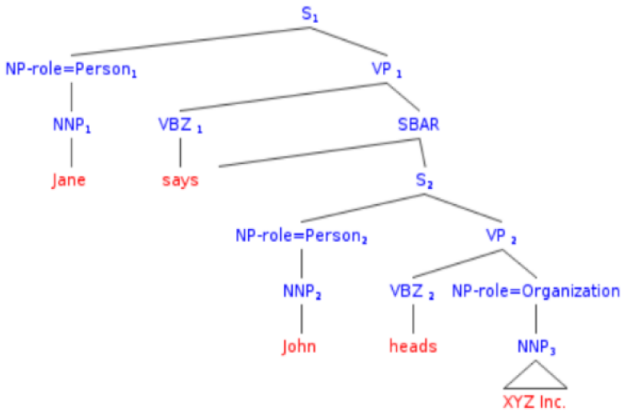
# Tree-based Kernel Methods



(a) Parse Tree



(b) Positive Relation Example



(c) Negative Relation Example

# Tree-based Kernel Methods

- Jane says John heads XYZ Inc.
- S2 includes both the related entities (John and XYZ Inc, labeled as positive)
- S1 covering the related entities and the entity Jane is not in a relation with the other entities, labeled as negative
- Each node has three attributes:
  - Entity role (T.role = {person, organization, none}),
  - chunk type (T.chunk = {NP, VP, etc.})
  - the text it covers (T.text).

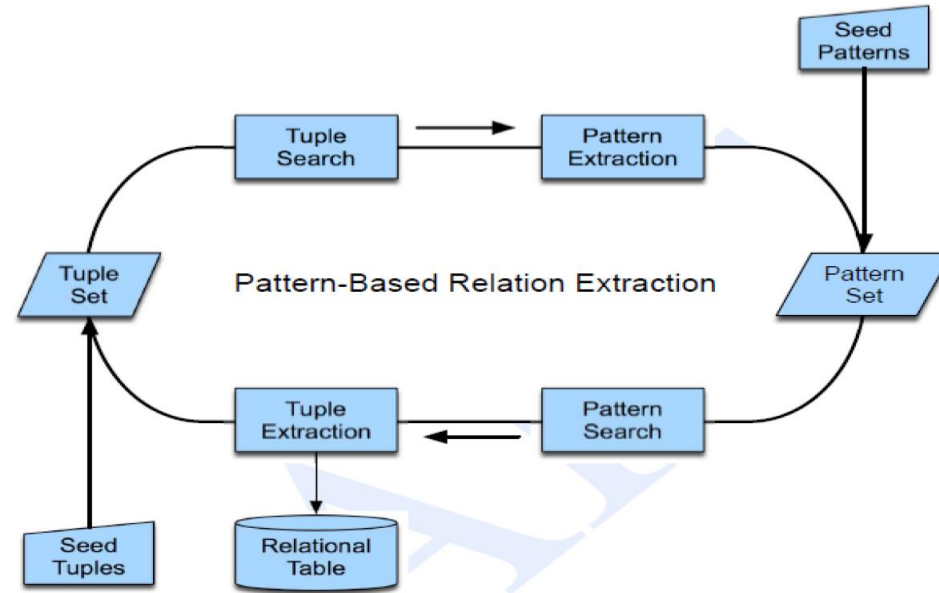
# Semi-supervised Learning Approaches

- Learn the relation instances and extraction patterns from a small set of labeled data or with the guide of known relation instances previously obtained.

# Bootstrapping

- New relations and patterns are derived via a small set of seed tuples or seed patterns
- The relationship Headquartered
- The seed pairs such as <Microsoft, Redmond>, < Google, Mountain View> and <Facebook, Palo Alto>
- “Google’s headquarters in Mountain View” and “Redmond-based Microsoft” produce the patterns like ‘ORG’s headquarters in LOC’ and ‘LOC-based ORG’.
- An unreliable sentence “Google, Mountain View” that contributes the pattern <ORG, LOC>, this pattern should not be used.

# Bootstrapping



# Distant Supervision

- Computed over the freely available knowledge bases such as Wikipedia, Freebase
- For example, Freebase is a knowledge base that stores several thousand relations in structured form
- Use various features like lexical, syntactic and named entity tags obtained over the sentences containing the entity pairs, the problem can be solved as a classification problem.

# Evaluation of Relation Analysis Systems

- In the evaluation of Supervised models, the precision, recall and f-measure express the performance of the system.
- For evaluation semi-supervised methods same metrics (precision, recall, f-measure) are used with different procedure. Due to production of a large number of new patterns and relations, approximate estimation is used.



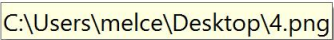
# Datasets for Relation Extraction Systems

- For the ACE data, automatically converts many relations which are nested, nominal entity mentions and non-nested, named or pronominal entity mentions. For instance, the first entity is mapped from 'one' to 'Amidu Berry' in the membership relation defined in 'Amidu Berry, one half of PBS'.

# Datasets for Relation Extraction Systems

---

```
<doc id='15'>
  <text>
    <p>
      <s id='s17'>
        <w l='american' p='NNP' phr='B-NP'>American</w>
        <w l='saxophonist' p='NN' phr='I-NP'>saxophonist</w>
        <w l='david' p='NNP' phr='B-NP'>David</w>
        <w l='murray' p='NNP' phr='I-NP'>Murray</w>
        <w l='recruit' p='VBD' phr='B-VP' voice='act'>recruited</w>
        <w l='amidu' p='NNP' phr='B-NP'>Amidu</w>
        <w l='berry' p='NNP' phr='I-NP'>Berry</w>
        <w p='CC' phr='I-NP'>and</w>
        <w l='dj' p='NNP' phr='I-NP'>DJ</w>
        <w l='awadus' p='NNP' phr='I-NP'>Awadi</w>
        <w p='NN'>.</w>
      </s>
    </p>
  </text>
  ...
</doc>
```



---

# Datasets for Relation Extraction Systems

- The BioInfer data which is encoded in XML, contains of sentence and word token markup in the domain of bioinformatics. This corpus uses token standoff annotation for entities and nested relations.

# Datasets for Relation Extraction Systems

---

```
<doc id='15'>
  <text>
    <p>
      <s id='s11'>
        <w l='beta-catenin' p='NN' phr='B-NP'>Beta-catenin</w>
        <w l='be' p='VBZ' phr='B-VP'>is</w>
        <w p='RB' phr='I-VP'>also</w>
        <w l='find' p='VBN' phr='I-VP' headv='yes' voice='pass'>found</w>
        <w p='IN'>in</w>
        <w p='DT' phr='B-NP'>these</w>
        <w l='structure' p='NNS' phr='I-NP'>structures</w>
        <w p='.'>.</w>
      </s>
    </p>
  </text>
  ...
</doc>
```

---