Lecture 3.1:
Machine Learning for Named Entity Recognition

Günter Neumann & Feiyu Xu
LT-lab, DFKI
The who, where, when & how much in a sentence

- The task: identify lexical and phrasal information in text which express references to named entities NE, e.g.,
  - person names
  - company/organization names
  - locations
  - dates&times
  - percentages
  - monetary amounts

- Determination of an NE
  - Specific type according to some taxonomy
  - Canonical representation (template structure)
Example of NE-annotated text

Delimit the named entities in a text and tag them with NE types:

<ENAMEX TYPE="LOCATION">Italy</ENAMEX>‘s business world was rocked by the announcement <TIMEX TYPE="DATE">last Thursday</TIMEX> that Mr. <ENAMEX TYPE="PERSON">Verdi</ENAMEX> would leave his job as vice-president of <ENAMEX TYPE="ORGANIZATION">Music Masters of Milan, Inc</ENAMEX> to become operations director of <ENAMEX TYPE="ORGANIZATION">Arthur Andersen</ENAMEX>.

- “Milan” is part of organization name
- “Arthur Andersen” is a company
- “Italy” is sentence-initial ⇒ capitalization useless
NE and Question-Answering

• Often, the expected answer type of a question is a **NE**
  - *What was the name of the first Russian astronaut to do a spacewalk?*
    - Expected answer type is **PERSON**
    - *Name the five most important software companies!*
      - Expected answer type is a list of **COMPANY**
    - *Where is does the ESSLLI 2004 take place?*
      - Expected answer type is **LOCATION** (subtype **COUNTRY** or **TOWN**)
  - *When will be the next talk?*
    - Expected answer type is **DATE**
German Named Entity

System Demo
NE and Web Mining: DFKI System WAG
The Result is an Indexed List of Named Entities
WAG's Control Flow

Query formation → Keywords

Google

N relevant documents (N=100) → Shallow NL processing

NE_q & EAT

Paragraph filtering

Inverted NE index

Stream of sentences (tokens, lexems, NE)

Sentence Equi-class ranking

{ <s_i, s_j, s_{NE-ea-t} > | query-compatible }

Exact answer

NE relevant documents (N=100)
### Average Results (40 Questions)

<table>
<thead>
<tr>
<th>Metric</th>
<th>Exact Answer</th>
<th>Sentence Answer</th>
<th>Snippet</th>
</tr>
</thead>
<tbody>
<tr>
<td>MRR (N=5)</td>
<td>0.174</td>
<td>0.171</td>
<td>0.0975</td>
</tr>
<tr>
<td>Recall (Top 1)</td>
<td>0.38</td>
<td>0.38</td>
<td>0.255</td>
</tr>
<tr>
<td>Recall (Top 3)</td>
<td>0.48</td>
<td>0.46</td>
<td>0.435</td>
</tr>
</tbody>
</table>
Difficulties of Automatic NER

• Potential set of NE is too numerous to include in dictionaries/Gazetteers
• Names changing constantly
• Names appear in many variant forms
• Subsequent occurrences of names might be abbreviated

⇒ list search/matching does not perform well
⇒ context based pattern matching needed
Difficulties for Pattern Matching Approach

Whether a phrase is a named entity, and what name class it has, depends on

• Internal structure:
  „Mr. Brandon“

• Context:
  „The new company, SafeTek, will make air bags.“

• Feiyu Xu, researcher at DFKI, Saarbrücken
NE and chunk parsing

- POS tagging plus generic chunk parsing alone does not solve the NE problem (ignoring type assignment for the moment)
  - Complex modification; target structure
    - [[1 Komma 2] Mio Euro]
    - CARD NN CARD NN NN

- POS tagging and chunk parsing would construct following syntactical possible but wrong structure
  - [1 Komma] [2 Mio] [Euro]
NE and chunk parsing

- Postmodification
  - Date expression with target structure
      CARD NN CARD
  - Wrong structure when generic chunk parsing
    - Am [3. Januar] [1967]
      CARD NN CARD
NE and chunk parsing

- Coordination of unit measures
  - target structure
    - [6 Euro und 50 Cents]
      CARD NN KON CARD NN
  - Generic chunk analysis
    - [6 Euro] und [50 Cents]
      CARD NN KON CARD NN
NE and chunk parsing

- **Person names**
  - **Target structure**
    - [John F. Kennedy]
      NE   NE   NE
  - **Generic chunk parsing**
    - [John F.] [Kennedy]
      NE   NE   NE
NE Co-reference


- Martin Marietta can be a person name or a reference to a company
- If MM is not part of an abbreviation lexicon, how do we recognize it?
  - Also by taking into account NE reference resolution.
NE is an interesting problem

- Productivity of name creation requires lexicon
  free pattern recognition
- NE ambiguity requires resolution methods
- Fine-grained NE classification needs fine-grained decision making methods
  - Taxonomy learning
- Multi-linguality
  - A text might contain NE expressions from different languages, e.g., output of IdentiFinder™
Why Machine Learning NE?

• System-based adaptation two new domains
  • Fast development cycle
  • Manual specification too expensive
  • Language-independence of learning algorithms
  • NL-tools for feature extraction available, often as open-source
• Current approaches already show near-human-like performance
  • Can easily be integrated with externally available Gazetteers
• High innovation potential
  • Core learning algorithms are language independent, which supports multi-linguality
  • Novel combinations with relational learning approaches
  • Close relationship to currently developed ML-approaches of reference resolution
## Different approaches

### Different degree of NL-preprocessing
- Character-level features (Whitelaw & Patrick, CoNLL, 2003)
- Tokenization (Bikel et al., ANLP 1997)
- POS + lemmatization (Yangarber et al., Coling 2002)
- Morphology (Cucerzan & Yarowsky, EMNLP 1999)
- Full parsing (Collins & Singer, EMNLP 1999)

### Supervised learning
- Training is based on available very large annotated corpus
- Mainly statistical-based methods used

### Unsupervised learning
- Training only needs very few seeds and very large un-annotated corpus: Topic of this lecture
## Current performance of supervised methods (CoNLL, 2003)*

<table>
<thead>
<tr>
<th>English</th>
<th>precision</th>
<th>recall</th>
<th>F</th>
<th>German</th>
<th>precision</th>
<th>recall</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIJZ03</td>
<td>88.99%</td>
<td>88.54%</td>
<td>88.76±0.7</td>
<td>FIJZ03</td>
<td>83.87%</td>
<td>63.71%</td>
<td>72.41±1.3</td>
</tr>
<tr>
<td>CN03</td>
<td>88.12%</td>
<td>88.51%</td>
<td>88.31±0.7</td>
<td>[KSNM03]</td>
<td>80.38%</td>
<td>65.04%</td>
<td>71.90±1.2</td>
</tr>
<tr>
<td>KSNM03</td>
<td>85.93%</td>
<td>86.21%</td>
<td>86.07±0.8</td>
<td>ZJ03</td>
<td>82.00%</td>
<td>63.03%</td>
<td>71.27±1.5</td>
</tr>
<tr>
<td>ZJ03</td>
<td>86.13%</td>
<td>84.88%</td>
<td>85.50±0.9</td>
<td>MMP03</td>
<td>75.97%</td>
<td>64.82%</td>
<td>69.96±1.4</td>
</tr>
<tr>
<td>CMP03b</td>
<td>84.05%</td>
<td>85.96%</td>
<td>85.00±0.8</td>
<td>[CMP03b]</td>
<td>75.47%</td>
<td>63.82%</td>
<td>69.15±1.3</td>
</tr>
<tr>
<td>CG03</td>
<td>84.29%</td>
<td>85.50%</td>
<td>84.89±0.9</td>
<td>BON03</td>
<td>74.82%</td>
<td>63.62%</td>
<td>68.88±1.3</td>
</tr>
<tr>
<td>MMP03</td>
<td>84.45%</td>
<td>84.90%</td>
<td>84.67±1.0</td>
<td>CC03</td>
<td>75.61%</td>
<td>62.46%</td>
<td>68.41±1.4</td>
</tr>
<tr>
<td>CMP03a</td>
<td>85.81%</td>
<td>82.84%</td>
<td>84.30±0.9</td>
<td>ML03</td>
<td>75.97%</td>
<td>61.72%</td>
<td>68.11±1.4</td>
</tr>
<tr>
<td>ML03</td>
<td>84.52%</td>
<td>83.55%</td>
<td>84.04±0.9</td>
<td>[MLP03]</td>
<td>69.37%</td>
<td>66.21%</td>
<td>67.75±1.4</td>
</tr>
<tr>
<td>BON03</td>
<td>84.68%</td>
<td>83.18%</td>
<td>83.92±1.0</td>
<td>[CMP03a]</td>
<td>77.83%</td>
<td>58.02%</td>
<td>66.48±1.5</td>
</tr>
<tr>
<td>MLP03</td>
<td>80.87%</td>
<td>84.21%</td>
<td>82.50±1.0</td>
<td>WNC03</td>
<td>75.20%</td>
<td>59.35%</td>
<td>66.34±1.3</td>
</tr>
<tr>
<td>WNC03*</td>
<td>82.02%</td>
<td>81.39%</td>
<td>81.70±0.9</td>
<td>CN03</td>
<td>76.83%</td>
<td>57.34%</td>
<td>65.67±1.4</td>
</tr>
<tr>
<td>WP03</td>
<td>81.60%</td>
<td>78.05%</td>
<td>79.78±1.0</td>
<td>HV03</td>
<td>71.15%</td>
<td>56.55%</td>
<td>63.02±1.4</td>
</tr>
<tr>
<td>HV03</td>
<td>76.33%</td>
<td>80.17%</td>
<td>78.20±1.0</td>
<td>DD03</td>
<td>63.93%</td>
<td>51.86%</td>
<td>57.27±1.6</td>
</tr>
<tr>
<td>DD03</td>
<td>75.84%</td>
<td>78.13%</td>
<td>76.97±1.2</td>
<td>WP03</td>
<td>71.05%</td>
<td>44.11%</td>
<td>54.43±1.4</td>
</tr>
<tr>
<td>Ham03</td>
<td>69.09%</td>
<td>53.26%</td>
<td>60.15±1.3</td>
<td>[Ham03]</td>
<td>63.49%</td>
<td>38.25%</td>
<td>47.74±1.5</td>
</tr>
<tr>
<td>baseline</td>
<td>71.91%</td>
<td>50.90%</td>
<td>59.61±1.2</td>
<td>baseline</td>
<td>31.86%</td>
<td>28.89%</td>
<td>30.30±1.3</td>
</tr>
</tbody>
</table>

Main features used by CoNLL 2003 systems

<table>
<thead>
<tr>
<th>System</th>
<th>lex</th>
<th>pos</th>
<th>aff</th>
<th>pre</th>
<th>ort</th>
<th>gaz</th>
<th>chu</th>
<th>pat</th>
<th>cas</th>
<th>tri</th>
<th>bag</th>
<th>quo</th>
<th>doc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Florian</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Chieu</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Klein</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Zhang</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Carreras (a)</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Curran</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Mayfield</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Carreras (b)</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>McCallum</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Bender</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Munro</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Wu</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Whitelaw</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Hendrickx</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>De Meulder</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Hammerton</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 3: Main features used by the sixteen systems that participated in the CoNLL-2003 shared task sorted by performance on the English test data. Aff: affix information (n-grams); bag: bag of words; cas: global case information; chu: chunk tags; doc: global document information; gaz: gazetteers; lex: lexical features; ort: orthographic information; pat: orthographic patterns (like Aa0); pos: part-of-speech tags; pre: previously predicted NE tags; quo: flag signing that the word is between quotes; tri: trigger words.
Learning Approaches in CoNLL

- Most systems used
  - Maximum entropy modeling (5)
  - Hidden-Markov models (4)
  - Connectionists methods (4)
- Near all systems used external resources, e.g., gazetteers
- Best systems performed hybrid learning approach
Details of Two Unsupervised NE Learning Methods

• Unsupervised NE Classification
  • Michael Collins and Yoran Singer, 1999

• Unsupervised Learning of Generalized Names
  • Yangarber, Lin, Grishman, 2002
  • Lin, Yangarber, Grishman, 2003
Unsupervised NE: idea

- Define manually only a small set of trusted seeds
- Training then only uses un-labeled data
- Initialize system by labeling the corpus with the seeds
- Extract and generalize patterns from the context of the seeds
- Use the patterns to further label the corpus and to extend the seed set (bootstrapping)
- Repeat the process unless no new terms can be identified
Unsupervised NE-learning: idea

- Trusted seeds
- NE Database
- annotator
- Unlabeled corpus

- NE Candidate selection
- Labeled corpus
- pattern learner

- Patterns

- NE Data base
- Labeled corpus

Candidate selection
Unsupervised NE classification
based on Michael Collins and Yoran Singer, EMNLP 1999

- The task: to learn a decision list to classify strings as person, location or organization

The learned decision list is an ordered sequence of if-then rules

... says Mr. Gates, founder of Microsoft ...

R_1: if features then person
R_2: if features then location
R_3: if features then organization
...
R_n: if features then person
Outline of Unsupervised Co-Training

- Parse an unlabeled document set
- Extract each NP, whose head is tagged as proper noun
- Define a set of relevant features, which can be applied on extracted NPs
- Define two separate types of rules on basis of feature space
- Determine small initial set of seed rules
- Iteratively extend the rules through co-training
Two Categories of Rules

- The key to the method is redundancy in the two kind of rules.

...says Mr. Cooper, a vice president of...

Paradigmatic or spelling

Syntagmatic or contextual

Huge amount of unlabeled data gives us these hints!
The Data

- 971,746 New York Times sentences were parsed using full sentence parser.
- Extract consecutive sequences of proper nouns (tagged as NNP and NNPS) as named entity examples if they met one of the following two criteria.
- Note: thus seen, NNP(S) functions as a generic NE-type, and the main task is now to sub-type it.
Kinds of Noun Phrases

1. There was an appositive modifier to the NP, whose head is a singular noun (tagged NN).
   • …says [Maury Cooper], [a vice president]…

2. The NP is a complement to a preposition which is the head of a PP. This PP modifies another NP whose head is a singular noun.
   • …fraud related to work on [a federally funded sewage plant] [in [Georgia]].
(spelling, context) pairs created

• … says Maury Cooper, a vice president…
  • (Maury Cooper, president)

• … fraud related to work on a federally funded sewage plant in Georgia.
  • (Georgia, plant_in)
## Features

for representing examples for the learning algorithm

<table>
<thead>
<tr>
<th>Set of spelling features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full-string=x (full-string=Maury Cooper)</td>
</tr>
<tr>
<td>Contains(x) (contains(Maury))</td>
</tr>
<tr>
<td>Allcap1 IBM</td>
</tr>
<tr>
<td>Allcap2 N.Y.</td>
</tr>
<tr>
<td>Nonalpha=x A.T.&amp;T. (nonalpha=..&amp;.)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Set of context features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Context = x (context = president)</td>
</tr>
<tr>
<td>Context-type = x appos or prep</td>
</tr>
</tbody>
</table>

It is strongly assumed that the features can be partitioned into two types such that each type alone is sufficient for classification.
# Examples of named entities and their features

<table>
<thead>
<tr>
<th>Sentence</th>
<th>Entities(Spelling/Context)</th>
<th>(Active) Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>But Robert Jordan, a partner at Steptoe &amp; Johnson who took ...</td>
<td>Robert Jordon/partner</td>
<td>Full-string=Robert_Jordon, contains(Robert), contains(Jordan), context=partner, context-type=appos</td>
</tr>
<tr>
<td></td>
<td>Steptoe &amp; Johnson/partner_at</td>
<td>Full-string=Steptoe_&amp;_Johnson, contains(Steptoe), contains(&amp;), contains(Johnson), nonalpha=&amp; , context=partner_at, context-type=prep</td>
</tr>
<tr>
<td>By hiring a company like A.T.&amp;T. ...</td>
<td>A.T.&amp;T./company_like</td>
<td>Full-string= A.T.&amp;T., allcap2, nonalpha=..&amp;. , context=company_like, context-type=prep</td>
</tr>
<tr>
<td>Hanson acquired Kidde Incorporated, parent of Kidde Credit, for ...</td>
<td>Kidde Incorporated/parent</td>
<td>Full-string=Kidde_Incorporated, contains(Kidde), contains(Incorporated), context=parent, context-type=appos</td>
</tr>
<tr>
<td></td>
<td>Kidde Credit/parent_of</td>
<td>Full-string=Kidde_Credit, contains(Kidde), contains(Credit), context=parent_of, context-type=prep</td>
</tr>
</tbody>
</table>
Rules

Feature → NE-type, h(Feature, NE-type)

h(x,y): the strength of a rule, defined as

\[ \arg \max_{x,y} \left( \frac{Count(x, y) + \alpha}{Count(x) + k\alpha} \right) \]

where

\[ Count(x) = \sum_{y \in Y} Count(x, y) \]

\[ \alpha \] is a smoothing parameter

\[ k = \#NE-types \]

The rules ordered according to their strengths h form a decision list: the sequence of rules are tested in order, and the answer to the first satisfied rule is output.

Two separate types of rules: Spelling rules, Context rules

Is an estimate of the conditional probability of the NE-type given the feature, P(y|x)
<table>
<thead>
<tr>
<th>Rule Description</th>
<th>Entity Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full-string = New York</td>
<td>Location</td>
</tr>
<tr>
<td>Full-string = California</td>
<td>Location</td>
</tr>
<tr>
<td>Full-string = U.S.</td>
<td>Location</td>
</tr>
<tr>
<td>Contains(Mr.)</td>
<td>Person</td>
</tr>
<tr>
<td>Contains(Incorporated)</td>
<td>Organization</td>
</tr>
<tr>
<td>Full-string = Microsoft</td>
<td>Organization</td>
</tr>
<tr>
<td>Full-string = I.B.M.</td>
<td>Organization</td>
</tr>
</tbody>
</table>

Note: only one type of rules used as seed rules, and all NE-types should be covered.
The Co-training algorithm

1. Set N=5 (max. # of rules of each type induced in each iteration)
2. Initialize: Set the spelling decision list equal to the set of seed rules. Label the training set using these rules.
3. Use these to get contextual rules. (x = feature, y = label)
   1. Compute $h(x,y)$, and induce at most $N \times K$ rules
   2. all must be above some threshold $p_{\text{min}} = 0.95$
4. Label the training set using the contextual rules.
5. Use these to get $N \times K$ spelling rules (same as step 3.)
6. Set spelling rules to seed plus the new rules.
7. If $N < 2500$, set $N = N + 5$, and goto step 3.

8. Label the training data with the combined spelling/contextual decision list, then induce a final decision list from the labeled examples where all rules (regardless of strength) are added to the decision list.
Example

- (IBM, company)
  - …IBM, the company that makes…

- (General Electric, company)
  - …General Electric, a leading company in the area,…

- (General Electric, employer)
  - …joined General Electric, the biggest employer…

- (NYU, employer)
  - NYU, the employer of the famous Ralph Grishman,…
Why Separate Spelling, Context Features?

Can use theory behind co-training to explain how algorithm works.

Requirements:

1. Classification problem $f: X \rightarrow Y$
   - $f_1(x_{1,i}) = f_2(x_{2,i}) = y_i$ for $i = 1 \ldots m$

2. Can partition features $X$ into 2 types of features $x = (x_1, x_2)$
   - $f_1(x_{1,i}) = f_2(x_{2,i})$ for $i = m+1 \ldots n$
   (softer criteria requires $f_1$ and $f_2$ to minimize the disagreements $\rightarrow$ similarity)

3. Each type is sufficient for classification

4. $x_1, x_2$ not correlated to tightly (e.g., no deterministic function from $x_1$ to $x_2$)

Open question: best similarity function?

- $f_i$ must correctly classify labeled examples, and
- must agree with each other on unlabeled ex.

3. & 4. Say that features can be partitioned.
The Power of the Algorithm

- **Greedy method**
  - At each iteration method increases number of rules
  - While maintaining a high level of agreement between spelling & context rules

For n= 2500:

1. The two classifiers give both labels on 49.2% of the unlabeled data

2. And give the *same* label on 99.25% of these cases
   - The algorithm maximizes the number of unlabeled examples on which the two decision list agree.
Evaluation

- 88,962 (spelling, context) pairs.
  - 971,746 sentences
- 1,000 randomly extracted to be test set.
- Location, person, organization, noise (items outside the other three)
- 186, 289, 402, 123 (- 38 temporal noise).

Let $N_c$ be the number of correctly classified examples

- Noise Accuracy: $N_c / 962$
- Clean Accuracy: $N_c / (962-85)$
## Results

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Clean Accuracy</th>
<th>Noise Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>45.8%</td>
<td>41.8%</td>
</tr>
<tr>
<td>EM</td>
<td>83.1%</td>
<td>75.8%</td>
</tr>
<tr>
<td>Yarowsky 95</td>
<td>81.3%</td>
<td>74.1%</td>
</tr>
<tr>
<td>Yarowsky Cautious</td>
<td>91.2%</td>
<td>83.2%</td>
</tr>
<tr>
<td>DL-CoTrain</td>
<td>91.3%</td>
<td>83.3%</td>
</tr>
<tr>
<td>CoBoost</td>
<td>91.1%</td>
<td>83.1%</td>
</tr>
</tbody>
</table>
Remarks

- Needs full parsing of unlabeled documents
  - Restricted language independency
  - Need linguistic sophistication for new types of NE
- Slow training
  - In each iteration, full size of training corpus has to be re-labeled
- DFKI extensions
  - Typed Gazetteers
  - Chunk parsing only
  - Integrated into a cross-language QA system
Much work on ML-NE focuses on classifying proper names (PNs)
- Person/Location/Organization

IE generally relies on domain-specific lexicon or Generalized Names (GNs)
- Closer to terminology: single- or multi-word domain-specific expressions

Automatic learning of GNs is an important first step towards truly adaptive IE
- IE system that can automatically adapt itself to new domains
How GNs differ from PNs

- Not necessary capitalized
  - tuberculosis
  - E. coli
  - Ebola haemorrhagic fever
  - Variant Creutzfeldt-Jacob disease
- Name boundaries are non-trivial to identify
  - “the four latest typhoid fever cases”
- Set of possible candidate names is broader and more difficult to determine
  - “National Veterinary Services Director Dr. Gideon Bruckner said no cases of mad cow disease have been in South Africa.”
- Ambiguity
  - E. coli: organism or disease
  - Encephalitis: disease or symptom
NOMEN: the Learning Algorithm

1. **Input**: Seed names in several chosen categories
2. Tag occurrences of names
3. Generate local patterns around tags
4. Match patterns elsewhere in corpus
   1. Acquire top-scoring pattern(s)
5. Acquired patterns tags new names
   1. Acquire top-scoring name(s)
6. Repeat
Pre-processing

- **Text-Zoner**
  - Extract textual content
  - Strips of headers, footers etc.

- **Tokenizer**
  - Produces lemmas

- **POS tagger**
  - Statistically trained on WSJ
  - Unknown or foreign words are not lemmatized and tagged as noun
Seeds

- For each *target* category select N initial *trusted* seeds
  - Diseases:
    - Cholera, dengue, anthrax, BSE, rabies, JE, Japanese encephalitis, influenza, Nipah virus, FMD
  - Locations:
    - United States, Malaysia, Australia, Belgium, China, Europe, Taiwan, Hong Kong, Singapore, France
  - Others
    - Case, health, day, people, year, patient, death, number, report, farm

- Use frequency counts computed from corpus or some external data-base

- Many more additional categories can be defined
Positive vs. Negative Seeds

- A seed name serves as
  - a positive example for its own class, and
  - a negative example for all other classes.

- Negative examples help steer the learner away from unreliable patterns
  - Competing classes
  - Termination of unsupervised learning
Pattern generation

• Tag every occurrence of each seed in corpus
  • “…new cases of <dis> cholera </dis> this year in …”
• For each tag, generate context rule: start/left-tag
  • [new case of <dis> cholera this year]
• Generalized left-side candidate patterns:
  • [new case of <dis> * * * ]
  • [* case of <dis> * * * ]
  • [* * of <dis> * * * ]
  • [* * * <dis> cholera this year ]
  • [* * * <dis> cholera this * ]
  • [* * * <dis> cholera * * ]
Pattern generation

- For each tag, generate context rule: end/right-tag
  - [case of cholera </dis> this year in]
- Generalized right-side candidate patterns:
  - [case of cholera </dis> * * *]
  - [* of cholera </dis> * * *]
  - [* * cholera </dis> * * *]
  - [* * * </dis> this year in]
  - [* * * </dis> this year * ]
  - [* * * </dis> this * * ]

- Note: all are potential patterns
Pattern application

- Apply each candidate pattern to corpus, observe where the pattern matches
  - E.g., the pattern [* * of <dis> * * *]
  - Each pattern predicts one boundary: search for the partner boundary using a noun group NG regex:
    - [Adj* Noun+]
    - “…distributed the yellow fever vaccine to the people”
- The resulting NG can be (wrt. currently tagged corpus)
  - Positive: “…case of <dis> dengue </dis> …”
  - Negative: “…North of <loc> Malaysia </loc> …”
  - Unknown: “…symptoms of <?> swine fever </?> in …”
Identify candidate NGs

- Sets of NG that the pattern \( p \) matched
  - \( Pos = \) distinct matched NG types of correct category
  - \( Neg = \) distinct matched NG types of wrong category
  - \( Unk = \) distinct matched NGs of unknown category

Collect statistics for each pattern

\[
acc(p) = \frac{|Pos|}{(|Pos| + |Neg|)}
\]

\[
conf(p) = \frac{|Pos| - |Neg|}{(|Pos| + |Neg| + |Unk|)}
\]
Pattern selection

- Discard pattern $p$ if $\text{acc}(p) < \theta$
- The remaining patterns are ranked by
  - $\text{Score}(p) = \text{conf}(p) \times \log|\text{Pos}(p)|$
- Prefer patterns that:
  - Predict the correct category with less risk
  - Stronger support: match more distinct known names
- Choose top $n$ patterns for each category
  - $[* \text{ die of } \langle \text{dis} \rangle * * *]$
Name selection

- Apply each accepted pattern to corpus, to find candidate names (using the NG)
  - “More people die of <dis> profound heartbreak than grief.”
- Rank each name type $t$ based on quality of patterns that match it:
  \[
  \text{Rank}(t) = 1 - \prod_{p \in M_t} (1 - \text{conf}(p))
  \]
  \[M_t\] is the set of accepted patterns which match any of the instances of $t$

- Require $|M_t| \geq 2 \Rightarrow t$ should appear $\geq 2$ times
- $M_t$ contains at least one pattern predicting the left boundary of $t$ and one pattern predicting the right boundary
- Conf($p$) assigns more credit to reliable patterns
Name selection

- Accept up to 5 top-ranked candidate names for each category
- Iterate learning algorithm until no more names can be learned
  - Bootstrap by using in each new iteration the extended set of new names to re-annotate the corpus
Salient Features of Nomen

- Generalized names
- A few manually-selected seeds
- Un-annotated corpus
- Un-restricted context (no syntactic restrictions)
- Patterns for left and right contexts independently
- Multiple categories simultaneously
Experiments

- Construction of reference lists for judging recall & precision of NOMEN

Compiled from multiple sources (medical DB, Web, manual review)

<table>
<thead>
<tr>
<th>Reference List</th>
<th>Disease</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manual</td>
<td>2492</td>
<td>1785</td>
</tr>
<tr>
<td>Recall (26K)</td>
<td>322</td>
<td>641</td>
</tr>
<tr>
<td>Recall (100K)</td>
<td>616</td>
<td>1134</td>
</tr>
<tr>
<td>Precision</td>
<td>3588</td>
<td>2404</td>
</tr>
</tbody>
</table>

Score recall against recall list and precision against precision list; Distinguish type and token tests

Appearing two or more time in development corpus

Manual list + acronyms + strip generic heads
Results

• Final recall & precision for 8 categories
  • Around 70% (in case of type-based evaluation)
  • Classical PN: Recall: 86-92%, Precision: above 70%
• Multi-class learning has positive effects
  • A category is less likely to expand beyond its true territory
  • The accepted names in each category serve as negative example for all categories
  • The learners avoid acquiring patterns with too many negatives
  • In some sense, the categories self-tune each other
• Comparison with human-in-the-loop
  • “More groups” can be as good as “few groups + human reviewer”
• Using a negative category (noun groups that belong to neither category, but generic terms), then also substantial increase in performance
Research Issues

• Can a richer linguistic model improve pattern generalization?
  • More elaborate NG-grammar
  • POS/SEM instead of wildcard
  • Note: one benefit of the approach is, that it does not need sophisticated linguistics, and hence is more adaptable

• How many different classes can effectively be learned simultaneously?
  • More complex seed-determination
  • When do the different classes enter into a dead-lock situation?
  • Group learning?

• At DKFI we have already started some of these inquiries
Final Remarks

- **State-of-art in NE recognition**
  - Machine learning works
  - Core learning engines are language independent
  - Feature extraction relies on language specific properties
  - Unsupervised learning promising direction
Challenging Problems

• What level of linguistic representation works best?
  • POS-tagging or deep parsing?
  • Employ linguistic principles (e.g., X-bar, head-principle, …)
• “language alignment”
  • Is it possible to re-use a model of language X, also for processing in language Y?
• Incremental learning algorithms
  • How to perform revision of learned patterns?
• Learning of fine-grained classes
  • Ako taxonomy learning, cf. Fleischman & Hovy, Coling2002
  • NE as Word Sense Disambiguation?
• Recognition of NE-paraphrases
  • NE-centered reference resolution
  • Combination of NE from un-structured and structured sources, cf. Cohen & Sarawagi, KDD’04, Seattle
Named Entity Recognition: DFKI-Version

**NE-candidates:**
\{\text{subtree}(w) \mid \forall w \in \text{ShProT-XML}: \text{PoS}^*(w) = \text{NNP} | \text{Card} | \text{TimeEx} & \text{NP}(w)\}

**Decision list of rules (XML):**
1. Condition: spelling/context
2. Context: Syntactic criteria
   1. Appositive modifier*
   2. Complement of PP*
3. Action: NE-type

**ShProT-XML**

**ShProT+NE-XML**

**Decision list matcher**

**Gazetters (RegEx)**

**feature vector representation**

Intelligent Information Extraction,
Neumann & Xu