By Sunita Sarawagi
Presented by Rohit Paravastu and Will Wu

INFORMATION EXTRACTION
ROADMAP

- Introduction
- Entity Extraction: Rule-based Methods
- Entity Extraction: Statistical Methods
- Relationship Extraction
- Management of Information Extraction Systems
ISSUES TO BE ADDRESSED

- What is Information Extraction?
- Why do we need Information Extraction after all (applications)?
- What is/are the formal problem definition(s)?
- What are the approaches people have taken?
- What are the challenges?
In one sentence: automatic extraction of structured information from unstructured sources

- Input: unstructured sources
- Output: structured information
Type of unstructured source (input)

- Granularity
  - Records, sentences
    - Duke University, Durham, NC, 27708
  - Paragraphs, documents, etc.
- Heterogeneity
  - Machine generated pages, partially structured domain specific sources, open ended sources, etc.

Type of structure extracted (output)

- Entities, relationships, lists, tables, attributes, etc.
Structured information is much easier to handle by computers.
APPLICATIONS

- Enterprise
  - News tracking
  - Customer care
  - Data cleaning
- Personal information management
- Scientific applications
Web oriented applications
  - Citation databases
  - Opinion databases
  - Community websites
  - Comparison shopping
  - Ad placement on webpages
  - Structured web searches
OTHER KEY COMPONENTS

- Type of input resources available for extraction
  + Structured databases, labeled unstructured data, linguistic tags, etc.

- Method used for extraction
  + Rule-based, statistical
  + Manually coded, trained from examples

- Representation of output
  + Annotated unstructured text, database
CHALLENGES

- Accuracy
  - Diversity of clues
  - Difficulty of detecting missed extractions
  - Increased complexity of the structures extracted

- Efficiency (running time)

- Other systems issues
  - Dynamically changing sources
  - Data integration
  - Extraction errors
A BRIEF HISTORY

- Rooted in the Natural Language Processing (NLP) community
- Scope extended by two competitions
  - Message Understanding Conference (MUC)
  - Automatic Content Extraction (ACE)
- Now spanning
  - Machine learning
  - Information retrieval
  - Database
  - Web
  - Document analysis
ROADMAP

- Introduction
- Entity Extraction: Rule-based Methods
- Entity Extraction: Statistical Methods
- Relationship Extraction
- Management of Information Extraction Systems
Extraction handled through a collection of rules

How are rules obtained?
- Manually coded
- Learnt from example labeled sources

How to control the firings of multiple rules?
BASIC FORM OF A RULE

- Contextual pattern -> Action
  - Use the pattern to match unstructured source
  - If matched, take the action
FEATURES OF TOKENS

- **String**
- **Orthography type**
  - E.g. Capitalized word, smallcase word, mixed case word, number, special symbol, space, punctuation, etc.
- **List of dictionaries in which the token appears**
  - E.g. “DictionaryLookup = start of city”
- **Annotations attached by earlier processing steps**
RULE TYPE I - SINGLE ENTITY

- ({DictionaryLookup = Titles}{String = “.”}) {Orthography type = capitalized word}{2}) -> Person Names.
  + Matches person names such as “Dr. Jun Yang”

- ({String = “by” | String = “in”}) ({Orthography type = Number}):y -> Year=:y.
  + Matches any number following “by” or “in”
  + Could be used to extract Year entity
RULE TYPE I – SINGLE ENTITY

- A simple exercise
  - ({String = “The”}? {Orthography type = All capitalized} {Orthography type = Capitalized word, DictionaryType Company end}) -> Company name.
RULE TYPE II – MARK ENTITY BOUNDARIES

- ({String="to"} {String = “appear”} {String="in")}:jstart
  ({Orthography type = Capitalized word}{2-5})) -> insert <journal> after:jstart
  + Annotation, may be used by following processing steps
RULE TYPE III – MULTIPLE ENTITIES

- \{\text{Orthography type = Digit}\} : \text{Bedrooms} (\{\text{String = “BR”}\}) (\{\text{String = “$”}\}) (\{\text{Orthography type = Number}\}) : \text{Price} -> \text{Number of Bedrooms} = : \text{Bedrooms}, \text{Rent} = : \text{Price}
Custom policies to resolve conflicts
- Prefer rules matching a longer span
  - Prefer higher priority in case of a tie
- Merge the spans of text that overlap
  - Only if action part is the same

Rules arranged as an ordered set
- **R1**: ({String="to"} {String="appear"} {String="in"}):jstart ({Orthography type = Capitalized word}{2-5})
  -> insert <journal> after :jstart
- **R2**: {tag = <journal>})({Orthography type=word}+):jend
  {String = “vol”} -> insert </journal> after :jend
HOW ARE RULES FORMULATED?

- Manually coded by a domain expert
- Learnt automatically...
  - ...from labeled examples of entities in unstructured text
  - Trying to achieve
    - High coverage
    - High precision
    - With a small set of rules
RULE LEARNING ALGORITHMS

- \( \text{Rset} = \text{set of rules, initially empty} \)
- While there exists an entity \( x \) not covered by any rule in \( \text{Rset} \):
  + Form new rules around \( x \)
  + Add new rules to \( \text{Rset} \)
- Post process rules to prune away redundant rules
HOW TO FORM NEW RULES?

- Bottom-up rule formulation
  + Generalize a specific rule
- Top-down rule formulation
  + Elaborate a generalized rule
ROADMAP

- Introduction
- Entity Extraction: Rule-based Methods
- Entity Extraction: Statistical Methods
- Relationship Extraction
- Management of Information Extraction Systems
Decompose text into parts and model distributions to label each part jointly or independently.

Decomposition done either into:
- Tokens (single word)
- Segments (Group of words)
OVERVIEW

- Token Level Methods
  - Features
  - Labeling

- Segment Level Methods
  - Features
  - Labeling

- Grammar based Models

- Training Methods

- Inference Algorithms
‘X’ denotes the given sentence

\( x_i \) denotes each token/segment

\( Y \) is the set of labels (entity labels) for \( X \)

\( y_i \) is the label for segment \( x_i \)

\( y_i \) can be either an entity from a predefined set of entity types or “other” if it does not belong to any entity type
Token-Level Methods

- Decompose the text ‘X’ into individual words $x_i$
- Convert the sentence into set of labels $Y=\{y_i\}$
**EXAMPLES**

Here is my review of Fermat's last theorem by S. Singh

<table>
<thead>
<tr>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>Here is my review of Fermat’s last theorem by S. Singh</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>y</td>
<td>y₁</td>
<td>y₂</td>
<td>y₃</td>
<td>y₄</td>
<td>y₅</td>
<td>y₆</td>
<td>y₇</td>
<td>y₈</td>
<td>y₉</td>
<td>y₁₀</td>
<td>y₁₁</td>
</tr>
</tbody>
</table>

R. Fagin and J. Helpbern, Belief Awareness Reasoning

<table>
<thead>
<tr>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>R. Fagin and J. Helpbern, Belief Awareness Reasoning</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>y</td>
<td>y₁</td>
<td>y₂</td>
<td>y₃</td>
<td>y₄</td>
<td>y₅</td>
<td>y₆</td>
<td>y₇</td>
<td>y₈</td>
<td>y₉</td>
</tr>
</tbody>
</table>
TYPES OF TOKENS

- Two styles of encoding
  - BCEO (Begin, Continue, End, Other)
  - BIO (Begin, Inside, Other)
- Similar to Classification
OVERVIEW

- Token Level Methods
  - Features
  - Labeling
- Segment Level Methods
  - Features
- Grammar based Models
- Training Methods
- Inference Algorithms
Features

- Clues/features designed to understand the properties of a token and the context of its position in the text
- \( f: (x, y, i) \rightarrow R \)
- \( R \) can be boolean or be a probability value to show the score/possibility of a token ‘y’ being assigned to \( x_i \)
Word Features
+ $f(y,x,i) = [[ X_i \text{ equals Fagin}]].[[y = \text{Author}]]$

Orthographic Features
+ Capitalization patterns, placement of dots etc
+ $f(y,x,i) = [[x_i \text{ matches INITIAL_DOT capsWord}]].[[y = \text{Author}]]$

Dictionary Lookup Features
+ Direct matches from a set of seed examples
OVERVIEW

- Token Level Methods
  - Features
  - Labeling
- Segment Level Methods
  - Features
- Grammar based Models
- Training Methods
- Inference Algorithms
Either independent of all other tokens or dependant on the previously labeled ones

SVMs to classify them independently

- Each token in the test set treated as a data point and the features as the axes

Dependency calculation

- HMMs
- Maximum Entropy Taggers (ME Markov Models)
- Conditional Markov Models
- Conditional Random Fields
CONDITIONAL RANDOM FIELDS (CRF)

- Models a joint distribution $P(y|x)$ over the set of predicted labels for tokens in $x$
- Tractable due to Markov Random Field assumption
- A label $y_i$ only depends on the features of $x_i$ and the previous label $y_{i-1}$
- Features changes from $f(y_i, x, i)$ to $f(y_i, x, i, y_{i-1})$
\[ \Pr(y|x,w) = \frac{1}{Z(x)} \prod_{i=1}^{n} \psi(y_{i-1}, y_i, x, i) = \frac{1}{Z(x)} e^{\sum_{i=1}^{n} w \cdot f(y_i, x, i, y_{i-1})} \]

\[ \psi(y_{i-1}, y_i, x, i) = e^{\sum_{k=1}^{K} w_k f_k(y_i, x, i, y_{i-1})} = e^{w \cdot f(y_i, x, i, y_{i-1})} \]
OVERVIEW

- Token Level Methods
  - Features
  - Labeling

- Segment Level Methods
  - Features

- Grammar based Models

- Training Methods

- Inference Algorithms
SEGMENT-LEVEL METHODS

- Divide text into segments rather than individual tokens
- Useful to calculate entity dependencies
- Problem: How do we determine Segment boundaries? *Inference*

R. Fagin and J. Halpern, Belief Awareness Reasoning

<table>
<thead>
<tr>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>R. Fagin and J. Halpern</td>
<td></td>
<td>,</td>
<td>Belief Awareness Reasoning</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(l_j, u_j, y_j)</td>
<td>1, 2, A</td>
<td>3, 3, O</td>
<td>4, 5, A</td>
<td>6, 6, O</td>
<td>7, 9, T</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
OVERVIEW

- Token Level Methods
  - Features
  - Labeling
- Segment Level Methods
  - Features
- Grammar based Models
- Training Methods
- Inference Algorithms
- Features defined over segments/multiple tokens
- More easy to map exact matches to a dictionary
- Use TFIDF in features to get rid of noise in unstructured text

\[ f(y_i, y_{i-1}, x, 3, 5) = \max_{J \in \text{journals}} \text{TF-IDF-similarity}(x_3 x_4 x_5, J) \cdot [y_i = \text{journal}]. \]
SEGMENTATION MODELING

- Similar to Token label modeling
- Done on a group of tokens rather than individual tokens

\[ f(x,s) = \sum_{j=1}^{s} f(y_j, x, l_j, u_j, y_{j-1}) \]
OVERVIEW

- Token Level Methods
  - Features
  - Labeling
- Segment Level Methods
  - Features
- Grammar based Models
- Training Methods
- Inference Algorithms
GRAMMAR BASED MODELS

- A context free grammar for each entity
- For each segment, output a parse tree for each grammar
- Label entity to the segment if
  - Segment accepted by the grammar
  - Maximum score is used for labeling
Example

\[ R: S \rightarrow \text{AuthorsLF} \mid \text{AuthorsFL} \]
\[ R0: \text{AuthorsLF} \rightarrow \text{NameLF}\_\text{Separator} \text{AuthorsLF} \]
\[ R1: \text{AuthorsFL} \rightarrow \text{NameFL}\_\text{Separator} \text{AuthorsFL} \]
\[ R2: \text{AuthorsFL} \rightarrow \text{NameFL} \]
\[ R3: \text{AuthorsLF} \rightarrow \text{NameLF} \]
\[ R4: \text{NameLF}\_\text{Separator} \rightarrow \text{NameLF} \text{ Punctuation} \]
\[ R5: \text{NameFL}\_\text{Separator} \rightarrow \text{NameFL} \text{ Punctuation} \]
\[ R6: \text{NameLF} \rightarrow \text{LastName First\_Middle} \]
\[ R7: \text{NameFL} \rightarrow \text{First\_Middle LastName} \]

\[ s(R) = s(R_1) + s(R_2) + w \cdot f(R, R_1, R_2, x, l_1, r_1, r_2) \]
OVERVIEW

- **Token Level Methods**
  - Features
  - Labeling

- **Segment Level Methods**
  - Features

- **Grammar based Models**

- **Training Methods**

- **Inference Algorithms**
TRAINING ALGORITHMS

- Model the score function $s(y)$ such that the best possible set of entities are returned
- Two kinds of Training
  + Likelihood based training
  + Max-margin training
- Goal: maximise $s(y) = w \cdot f(x,y)$, given ‘y’ is the optimal set of entities
Maximises the Log likelihood of $P(y|x)$ to get the set of weights ‘$w$’ such that the probability of outputting the correct $y$ is maximised.

$$Pr(y|x) = \frac{1}{Z(x)} e^{w \cdot f(x,y)}$$

$$\max_w \sum_{\ell} (w \cdot f(x_\ell, y_\ell) - \log Z_w(x_\ell)) - ||w||^2/C$$

$$\nabla L(w) = \sum_\ell f(x_\ell, y_\ell) - E_{Pr(y'|w,x_\ell)} f(x_\ell, y') - 2w/C$$
Max Margin Training

- Minimize the weights $W$ such that margin between scores of the correct labelling $y_\ell$ and $y$ is more than $\text{err}(y, y_\ell)$

\[
\min_{w,\xi} \sum_{\ell=1}^{N} \xi_\ell + \frac{1}{2}||w||^2 \\
\text{s.t. } w \cdot f(x_\ell, y_\ell) \geq \text{err}(y, y_\ell) + w \cdot f(x_\ell, y) - \xi_\ell \quad \forall y \neq y_\ell, \ell : 1 \ldots N \\
\xi_\ell \geq 0 \quad \ell : 1 \ldots N
\]  

(3.6)
OVERVIEW

- Token Level Methods
  - Features
  - Labeling
- Segment Level Methods
  - Features
- Grammar based Models
- Training Methods
- Inference Algorithms
Inference Algorithms

- Highest scoring (MAP) labeling
  - Find $y^* = \arg\max_y w. f(x,y)$

- Expected Feature Values
  - To get the expected values of features $f(x,y_i)$
  - Find $\sum_y f(x,y)\Pr(y|x)$
MAP LABELING

- Dynamic Programming model
- Divide the sentence into two disjoint chunks S₁ and S₂.
- Take a subset S₃ from S₁ that provides enough information to evaluate both S₁ and S₂

\[ \mathcal{V}(S) = \max_{\text{label } y' \text{ of } S_3} \mathcal{V}(S_1 | S_3 = y') + \mathcal{V}(S_2 | S_3 = y') \]
Here is my review of Fermat’s last theorem by S. Singh

<table>
<thead>
<tr>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>Here</td>
<td>is</td>
<td>my</td>
<td>review</td>
<td>of</td>
<td>Fermat’s</td>
<td>last</td>
<td>theorem</td>
<td>by</td>
<td>S.</td>
<td>Singh</td>
</tr>
<tr>
<td>y</td>
<td>y₁</td>
<td>y₂</td>
<td>y₃</td>
<td>y₄</td>
<td>y₅</td>
<td>y₆</td>
<td>y₇</td>
<td>y₈</td>
<td>y₉</td>
<td>y₁₀</td>
<td>y₁₁</td>
</tr>
</tbody>
</table>

R. Fagin and J. Helpbern, Belief Awareness Reasoning

<table>
<thead>
<tr>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>R. Fagin</td>
<td>and</td>
<td>J. Helpbern</td>
<td>,</td>
<td>Belief</td>
<td>Awareness</td>
<td>Reasoning</td>
<td></td>
<td></td>
</tr>
<tr>
<td>y</td>
<td>y₁</td>
<td>y₂</td>
<td>y₃</td>
<td>y₄</td>
<td>y₅</td>
<td>y₆</td>
<td>y₇</td>
<td>y₈</td>
<td>y₉</td>
</tr>
</tbody>
</table>
Sequential Labeling

- \( V(i|y) \) be the maximum score till the position ‘I’ in the string

\[
V(i|y) = \begin{cases} 
\max_{y'} V(i - 1, y') + w \cdot f(y, x, i, y') & \text{if } i > 0 \\
0 & \text{if } i = 0.
\end{cases}
\]

- The set of entities \( Y \) that maximises \( V(n|y) \) is the optimal set of entity labels
EXPECTED FEATURE VALUES

- Techniques to estimate the expected value of the features of the tokens/segments in a sentence
- Dynamic Programming model
- Expected output $E(f(x,y)) = \sum_y f(x,y) \Pr(y | x)$
Expected Feature Values

\[ Z(x) = \sum_y e^{w \cdot f(x,y)} \]

Assuming that we know the value of \( Z \) till token \( i-1 \), we calculate the value of \( Z \) at \( i \):

Let \( \alpha(i,y) = \text{score of all labeled sequences from 1 to } i \text{ with label of } i \text{ being 'y'} \)

\[ \alpha(i,y) = \sum_{y', \in Y} \alpha(i-1,y') \cdot e^{w \cdot f(y,x,i,y')} \]

\[ Z(x) = \sum_y \alpha(n,y) \]
Let $\eta^k(i,y)$ be the equivalent of $\alpha(i,y)$ for the $k^{th}$ component in feature set $f$.

\[
\eta^k(i,y) = \sum_{y' \in \mathcal{Y}} (\eta^k(i-1,y') + \alpha(i-1,y')f_k(y,x,i,y'))e^{w \cdot f(y,x,i,y')}
\]

\[
E_{\Pr(y'\mid w)}f_k(x,y') = \frac{1}{Z_w(x)} \sum_{y} \eta^k(n,y)
\]
ROADMAP

- Introduction
- Entity Extraction: Rule-based Methods
- Entity Extraction: Statistical Methods
- Relationship Extraction
- Management of Information Extraction Systems
Given a text snippet ‘x’ and two entities E1 and E2 in the snippet, find the relationship between the entities

- A scalar prediction as opposed to a vector prediction problem in entity extraction
- Tough due to the diversity in syntactic and semantic structure of sentences
OVERVIEW

- Clues
- Relationship extraction
- Extracting entity pairs given the relation
CLUES

- **Surface Tokens**
  - Words around and in-between the entities

  (Company) Kosmix (//Company) is located in the (Location) Bay area (//Location).

- **POS tags**
  - Two noun phrases will be connected by a verb

  (Location) The University of Helsinki (//Location) hosts (Conference) ICML (//Conference) this year.

  The/DT University/NNP of/IN Helsinki/NNP hosts/VBZ ICML/NNP this/DT year/NN.
Syntactic Parse Trees

- Parse tree structure can show the relationship between prominent phrases in the sentence.
- Useful for the example “Haifa, located 53 miles from Tel Aviv will host ICML in 2010”
(ROOT
  (S
   (NP
    (NP (NNP Haifa))
    (VP (VBN located)
      (PP
        (NP (CD 53) (NNS miles))
        (IN from)
        (NP (NNP Tel) (NNP Aviv)))))
    (VP (MD will)
      (VP (VB host)
        (NP
          (NP (NNP ICML))
          (PP (IN in)
            (NP (CD 2010)))))))
Dependency Parse of a sentence

- Edge from a word ‘a’ to word ‘b’ if there exists a dependency between them
OVERVIEW

- Clues
- Relationship extraction
- Extracting entity pairs given the relation
EXTRACTION METHODS

- Feature Based
  - Flat set of features
- Kernel Based
  - Similarity calculation between trees and graphs
- Rule-based
FEATURE BASED METHODS

- Each word has a lot of properties associated
  - String form, orthography, POS tag etc.
  - Example: [[Entity 1=“Person”, Entity2=“Location”]]
- First set of features: Conjunctions of all properties of the two tokens corresponding to E1 and E2
- Most frequently co-occurring features define the relationship
FEATURE BASED METHODS

- Word Sequences
  - Unigram Features
    - [[String="host", flag="none"]]
  - Bigram Features
    - [[String="host,ICML", flags=(none,2), type="sequence"]]
  - Trigram Features
    - [[string="will,host,ICML", flags=(none,none,2), type="sequence"]]

FEATURE BASED METHODS

- Dependency Graphs
  - Similar to word sequences, but the bigrams and trigrams are formed based on the dependencies

- Parse Trees
  - Unigram features include noun phrases and verb phrases
  - New bigram and trigram features to show the path from one node to other
EXAMPLE PARSE TREE

Constituency-based parse tree

John hit the ball.
KERNEL METHOD

- Each training instance treated as a point in a graph.
- To find the relationship between two entities in a test sentence,

\[ \hat{r} = \arg\max_{r \in \mathcal{Y}} \sum_{i=1}^{N} \alpha_i r K(X_i, X). \]

- Distance measured between sentence \( x \) and \( x_i \) as \( K(x, x_i) \)
- \( K() \) is the kernel function
- Example:

\[
K(P, P') = \begin{cases} 
  0 & \text{if } P, P' \text{ have different lengths} \\
  \lambda \prod_{k=1}^{\max(|P|, |P'|)} \text{CommonProperties}(P_k, P'_k) & \text{otherwise,}
\end{cases}
\]
OVERVIEW

- Clues
- Relationship extraction
- Extracting entity pairs given the relation
EXTRACTING ENTITY PAIRS

- Given a relationship, extract corresponding entity pairs
- Useful in searching for all the occurrences of a relation ‘r’ in the corpus
- Training set
  - Entity types that can possibly correspond to that relation
  - Examples of words that can correspond to that relation
  - Manually coded patterns
LEARNING

- Create (E1,E2,r) triplets
- Prune away infrequently occurring triples
- Learn patterns from the seed examples
LEARNING PATTERNS

- Entity extraction for all the seed entities
- Extract relation patterns for these entity instances
- Challenge: differentiating between the different relationships between the two entities
- Treat each sentence containing both entities as an independent training instance and classify using SVMs
For each relation r, go through each sentence and search for entity pairs that have that relation ‘r’ in the training set

Pattern based extraction
  + Look for occurrences of particular set of words like ‘E1 is working for E2’

Keyword based
  + Prune away sentences based on keyword searches
Validation necessary to avoid snowballing of training data errors

Relationship extraction has typically 50-70% accuracy

Needs lot of special case handling dependent on the particular dataset
ROADMAP

- Introduction
- Entity Extraction: Rule-based Methods
- Entity Extraction: Statistical Methods
- Relationship Extraction
- Management of Information Extraction Systems
MAIN ISSUES

- Performance Optimization
- Handling Change
- Integration of Extracted Information
- Imprecision of Extraction
PERFORMANCE OPTIMIZATION

- **Document Selection**
  - Trade off between recall and time
    - Focused crawling
    - Searching via keywords
    - Filtering documents after fetching them using a classifier
PERFORMANCE OPTIMIZATION

- Index Search
  - Keyword queries
    - Usually for subject filtering
    - E.g. “vaccine” and “cure” -> documents containing disease outbreaks
  - Pattern queries
    - Finer grained filtering of entities of interest
    - E.g. “[Mr. | Dr. | Mrs.] Initial_Dot Capitalized_Word”
Index Design

- ... for Efficient Extraction
- Provide support for proximity queries, regular expression patterns
- Allow efficient storage of tags
  - POS
  - Phrase tags
  - Common entity tags, e.g. person/company names
- Possible solutions for regular expression
  - Suffix trees
  - q-gram index
Other Optimizations

- Efficiency in querying entity databases
- Optimizing for expensive feature evaluation
- Relational engine style frameworks
HANDLING CHANGES

- **Incremental Extraction on Changing Sources**
  - Use Unix diff or suffix tree to detect changes
  - Run extractor only on changed portions

- **Detecting When Extractors Fail on Evolving Data**
  - Defining Characteristic Patterns
  - Detecting Significant Change
Decoupled Extractions and Integration

- Binary classifier for deciding whether two input records are duplicates
  - Trained classifier, e.g. SVM
  - Manually defined rules
  - Decision tree
Decoupled Extraction and Collective Integration

- R1. Alistair MacLean
- R2. A Mclean
- R3. Alistair Mclean
INTEGRATION OF EXTRACTED INFORMATION

- Coupled Extraction and Integration
  - “In his foreword to Transaction Processing Concepts and Techniques, Bruce Lindsay”
  - Book names containing entry “Transaction Processing: Concepts and Techniques.”
  - People names containing “A. Reuters”, “B. Lindsay”, “J. Gray”
  - Authors table linking book title with people
IMPRECISION OF EXTRACTION

- Confidence Values for Single Extractions
  - Attach a probability to each possible outcome of an extraction
  - Total probability normalized to 1
IMPRECISION OF EXTRACTION

- Multi-attribute Extractions

<table>
<thead>
<tr>
<th>Id</th>
<th>House_no</th>
<th>Area</th>
<th>City</th>
<th>Pincode</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>52</td>
<td>Goregaon</td>
<td>Mumbai</td>
<td>400 062</td>
<td>0.1</td>
</tr>
<tr>
<td>1</td>
<td>52-A</td>
<td>Goregaon</td>
<td>West Mumbai</td>
<td>400 062</td>
<td>0.2</td>
</tr>
<tr>
<td>1</td>
<td>52-A</td>
<td>Goregaon</td>
<td>West Mumbai</td>
<td>400 062</td>
<td>0.5</td>
</tr>
<tr>
<td>1</td>
<td>52</td>
<td>Goregaon</td>
<td>West Mumbai</td>
<td>400 062</td>
<td>0.2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Id</th>
<th>House_no</th>
<th>Area</th>
<th>City</th>
<th>Pincode</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>52 (0.3)</td>
<td>Goregaon</td>
<td>Mumbai (0.6)</td>
<td>400 062</td>
</tr>
<tr>
<td></td>
<td>52-A (0.7)</td>
<td>West (0.6)</td>
<td>West Mumbai (1.0)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Goregaon (0.4)</td>
<td></td>
<td>(0.4)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Id</th>
<th>House_no</th>
<th>Area</th>
<th>City</th>
<th>Pincode</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>52 (0.167)</td>
<td>Goregaon</td>
<td>Mumbai (1.0)</td>
<td>400 062</td>
<td>0.6</td>
</tr>
<tr>
<td></td>
<td>52-A (0.833)</td>
<td>West (1.0)</td>
<td></td>
<td>(1.0)</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>52 (0.5)</td>
<td>Goregaon</td>
<td>West (1.0)</td>
<td>400 062</td>
<td>0.4</td>
</tr>
<tr>
<td></td>
<td>52-A (0.5)</td>
<td>Goregaon</td>
<td>Mumbai (1.0)</td>
<td>(1.0)</td>
<td></td>
</tr>
</tbody>
</table>
IMPRECISION OF EXTRACTION

- Multiple Redundant Extractions
  - Two kinds of uncertainties
    - Single source extraction uncertainty
    - Co-reference uncertainty

<table>
<thead>
<tr>
<th>Id</th>
<th>Title</th>
<th>Pr</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Last Theorem</td>
<td>0.5</td>
</tr>
<tr>
<td>2</td>
<td>“Transaction Processing: Concepts and Techniques”</td>
<td>0.95</td>
</tr>
<tr>
<td>3</td>
<td>Transaction Processing</td>
<td>0.4</td>
</tr>
<tr>
<td>4</td>
<td>Fermat’s Last Theorem</td>
<td>0.3</td>
</tr>
<tr>
<td>5</td>
<td>The Fermat’s Last Theorem</td>
<td>0.5</td>
</tr>
<tr>
<td>6</td>
<td>Transaction Processing: Concepts and Techniques</td>
<td>1.0</td>
</tr>
<tr>
<td>7</td>
<td>Transaction Processing Concepts and Techniques</td>
<td>1.0</td>
</tr>
<tr>
<td>8</td>
<td>Fermat’s Last Theorem</td>
<td>0.9</td>
</tr>
<tr>
<td>9</td>
<td>Fermat’s Last Theorem</td>
<td>0.8</td>
</tr>
</tbody>
</table>
SUMMARY

- Applications
- Rule-based and statistical methods for entity extraction
- Statistical methods for relation extraction
- Practical issues