INFORMATION EXTRACTION

By Sunita Sarawagi
Presented by Rohit Paravastu and Will Wu

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?

WHAT IS INFORMATION EXTRACTION…?

- In one sentence: automatic extraction of *structured information* from *unstructured sources*
  - *Input: unstructured sources*
  - *Output: structured information*

INPUT AND OUTPUT

- Type of unstructured source (input)
  - Granularity
    - **Records, sentences**
      - <inst> Duke University </inst>, <city> Durham </city>, <state> NC </state>, <zipcode> 27708 </zipcode>
    - 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.

WHY INFORMATION EXTRACTION?

- Structured information is much easier to handle by computers

APPLICATIONS

- Enterprise
  - News tracking
  - Customer care
  - Data cleaning
- Personal information management
- Scientific applications

APPLICATIONS (CONT'D)

- 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

OVERVIEW

- Extraction handled through a collection of rule
- 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

- ({Orthography type = Digit}):Bedrooms ({String=“BR”}) ({}*) ({String =“$”}) ({Orthography type = Number}):Price -> Number of Bedrooms = :Bedrooms, Rent = :Price

ORGANIZING COLLECTION OF RULES

- 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

- Rset = set of rules, initially empty
- While there exists an entity **x** not covered by any rule in Rset
  - **Form new rules around x**
  - Add new rules to 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

STATISTICAL METHODS

- 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

NOTATION

- ‘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

i	1	2	3	4	5	6	7	8	9	10	11
x	Here	is	my	review	of	Fermat's	last	theorem	by	S.	Singh
y	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}	y_{11}

R. Fagin and J. Helpbern, Belief Awareness Reasoning

i	1	2	3	4	5	6	7	8	9
x	R.	Fagin	and	J.	Helpbern	,	Belief	Awareness	Reasoning
y	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9

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) ⟶ R
- R can be boolean or be a probability value to show the score/possibility of a token ‘y’ being assigned to x_i

FEATURES

- Word Features
  - f (y,x,i)= [[X_i equals Fagin]].[[*y*= Author]]
- Orthographic Features
  - Capitalization patterns, placement of dots etc
  - f(y,x,i)=[[x_i matches INITIAL_DOT capsWord]].[[*y*=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

TOKEN LABELING

- 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 $f(y_i, x, i, y_{i-1})$

CONDITIONAL RANDOM FIELDS

$$\Pr(\mathbf{y}|\mathbf{x},\mathbf{w}) = \frac{1}{Z(\mathbf{x})}\prod_{i=1}^{n}\psi(y_{i-1},y_i,\mathbf{x},i) = \frac{1}{Z(\mathbf{x})}e^{\sum_{i=1}^{n}\mathbf{w}\cdot\mathbf{f}(y_i,\mathbf{x},i,y_{i-1})}$$

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

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. Helpbern, Belief Awareness Reasoning

i	1	2	3	4	5	6	7	8	9
x	R. Fagin		and	J. Helpbern		,	Belief Awareness Reasoning		
(l_j, u_j, y_j)	$1, 2, A$		$3, 3, O$	$4, 5, A$		$6, 6, O$	$7, 9, T$		

OVERVIEW

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

FEATURES

- 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

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

SEGMENTATION MODELING

- Similar to Token label modeling
- Done on a group of tokens rather than individual tokens
  - f(x,s)=$\sum_{j=1}^{|s|} f(y_j, x, l_j, u_j, y_{j-1})$

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

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

GRAMMAR BASED MODELS

- Example

R: S $\rightarrow$ AuthorsLF | AuthorsFL
R0: AuthorsLF $\rightarrow$ NameLF_Separator AuthorsLF
R1: AuthorsFL $\rightarrow$ NameFL_Separator AuthorsFL
R2: AuthorsFL $\rightarrow$ NameFL
R3: AuthorsLF $\rightarrow$ NameLF
R4: NameLF_Separator $\rightarrow$ NameLF Punctuation
R5: NameFL_Separator $\rightarrow$ NameFL Punctuation
R6: NameLF $\rightarrow$ LastName First_Middle
R7: NameFL $\rightarrow$ First_Middle LastName

$$s(R) = s(R_1) + s(R_2) + \mathbf{w} \cdot \mathbf{f}(R, R_1, R_2, \mathbf{x}, l_1, r_1, r_2)$$

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.f**(x,y) , given ‘y’ is the optimal set of entities

LIKELIHOOD TRAINER

- 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.

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

$$\max_{\mathbf{w}} \sum_{\ell} (\mathbf{w}\cdot\mathbf{f}(\mathbf{x}_\ell,\mathbf{y}_\ell) - \log Z_{\mathbf{w}}(\mathbf{x}_\ell)) - ||\mathbf{w}||^2/C$$

$$\nabla L(\mathbf{w}) = \sum_{\ell} \mathbf{f}(\mathbf{x}_\ell,\mathbf{y}_\ell) - E_{\Pr(\mathbf{y}'|\mathbf{w},\mathbf{x}_\ell)}\mathbf{f}(\mathbf{x}_\ell,\mathbf{y}') - 2\mathbf{w}/C$$

MAX MARGIN TRAINING

- Minimize the weights W such that margin between scores of the correct labelling y_l and y is more than **err(y,y_l)**

$$\min_{\mathbf{w},\xi} C\sum_{\ell=1}^{N}\xi_\ell + \frac{1}{2}||\mathbf{w}||^2$$

$$\text{s.t. } \mathbf{w}\cdot\mathbf{f}(\mathbf{x}_\ell,\mathbf{y}_\ell) \geq \text{err}(\mathbf{y},\mathbf{y}_\ell) + \mathbf{w}\cdot\mathbf{f}(\mathbf{x}_\ell,\mathbf{y}) - \xi_\ell \quad \forall \mathbf{y} \neq \mathbf{y}_\ell, \ell : 1\cdots N$$

$$\xi_\ell \geq 0 \; \ell : 1\cdots N \qquad (3.6)$$

OVERVIEW

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

INFERENCE ALGORITHMS

- Highest scoring (MAP) labeling
  - Find $y^* = \text{argmax}_y$ **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

$$\mathcal{V}(S) = \max_{\text{label } \mathbf{y}' \text{ of } S_3} \mathcal{V}(S_1|S_3 = \mathbf{y}') + \mathcal{V}(S_2|S_3 = \mathbf{y}')$$

EXAMPLES

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

i	1	2	3	4	5	6	7	8	9	10	11
x	Here	is	my	review	of	Fermat's	last	theorem	by	S.	Singh
y	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}	y_{11}

R. Fagin and J. Helpbern, Belief Awareness Reasoning

i	1	2	3	4	5	6	7	8	9
x	R.	Fagin	and	J.	Helpbern	,	Belief	Awareness	Reasoning
y	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9

MAP LABELING

- Sequential Labeling
  - *V*(i|y) be the maximum score till the position 'I' in the string

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

- 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 $E(\mathbf{f(x,y)}) = \sum_y \mathbf{f(x,y)} Pr(\mathbf{y|x})$

EXPECTED FEATURE VALUES

- $Z(x)=\Sigma_y\ e^{w.f(X,y)}$
- Assuming that we know the value of Z till token i-1, we calculate the value of Z at i
- Let α(i,y) = score of all labeled sequences from 1 to i with label of i being 'y'
- $\alpha(i,y)=\sum_{y' \in Y} \alpha(i-1,y')\ e^{\mathbf{w}.f(y,\mathbf{x},i,y')}$
- $Z(x)=\Sigma_y \alpha(n,y)$

EXPECTED FEATURE VALUES

- Let $\eta^k(i,y)$ be the equivalent of $\alpha(i,y)$ for the k^{th} component in feature set f

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

$$E_{\Pr(\mathbf{y}'|\mathbf{w})} f_k(\mathbf{x},\mathbf{y}') = \frac{1}{Z_{\mathbf{w}}(\mathbf{x})} \sum_y \eta^k(n,y)$$

ROADMAP

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

RELATIONSHIP EXTRACTION

- 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.

CLUES

- 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”

PARSE TREE FOR EXAMPLE

```
(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)))))))))
```

CLUES

- 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-occuring features define the relationship

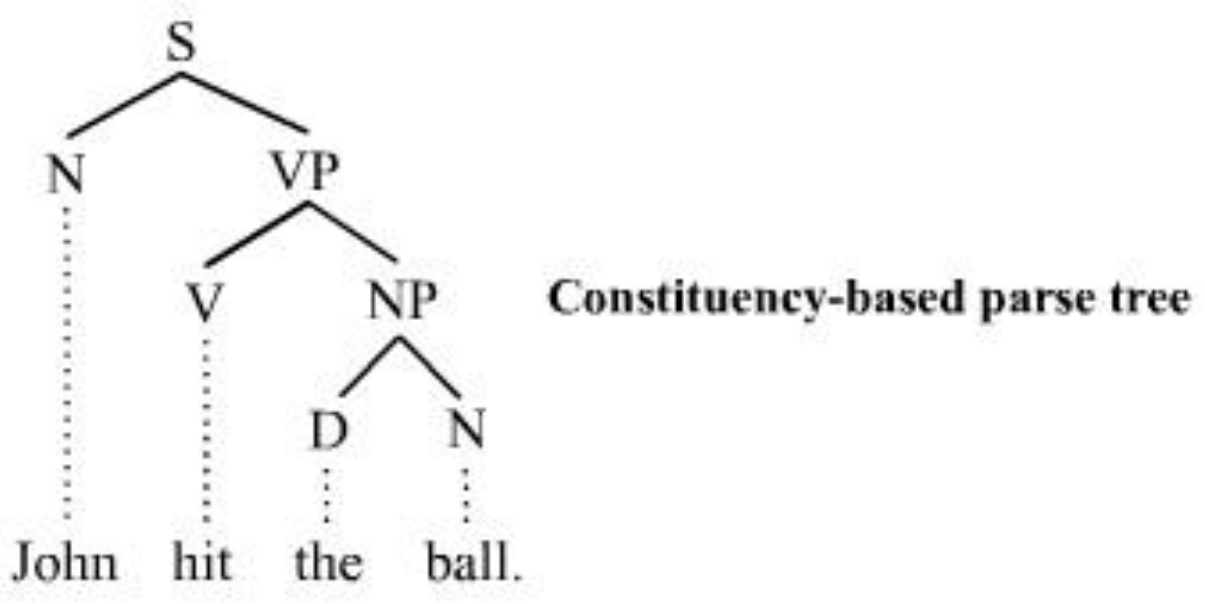
FEATURE BASED METHODS

- 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



KERNEL METHOD

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

$$\hat{r} = \underset{r \in \mathcal{Y}}{\operatorname{argmax}} \sum_{i=1}^{N} \alpha_{ir} K(X_i, X).$$

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

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

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 possbily 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

USING THE MODEL ON CORPUS

- 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

SUMMARY

- 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”

PERFORMANCE OPTIMIZATION

- 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

PERFORMANCE OPTIMIZATION

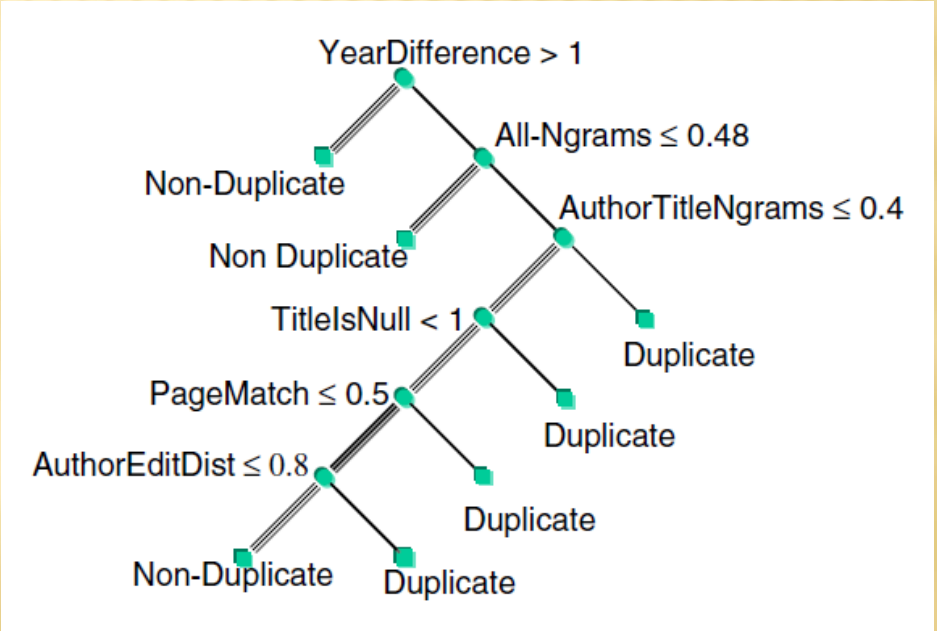
- Other Optimizations
  - Efficiency in querying entity databases
  - Optimizing for expensive feature evaluation
  - Relational engine style frameworks

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

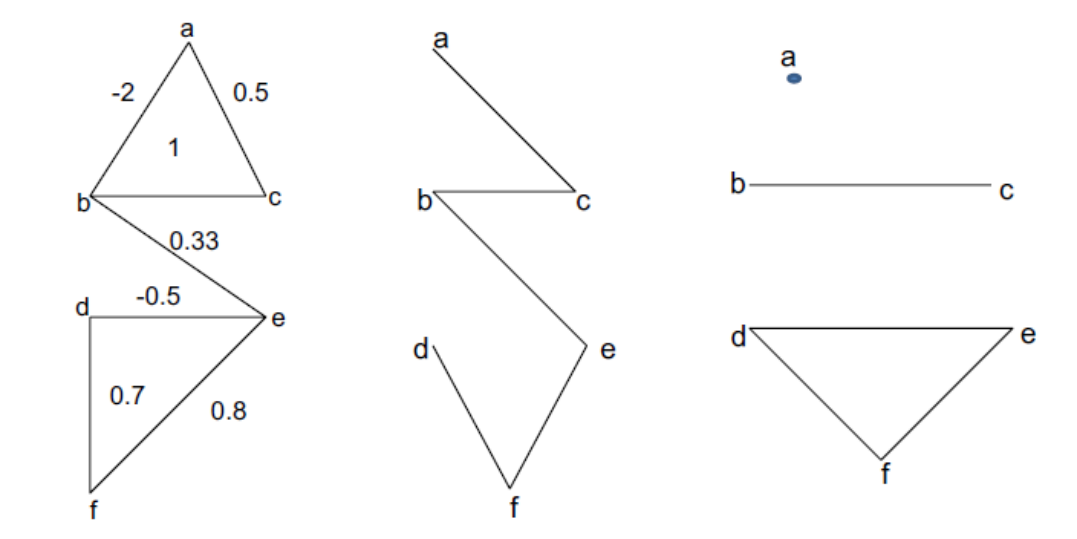
INTEGRATION OF EXTRACTED INFORMATION

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



INTEGRATION OF EXTRACTED INFORMATION

- 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

Id	House_no	Area	City	Pincode	Prob
1	52	Goregaon West	Mumbai	400 062	0.1
1	52-A	Goregaon	West Mumbai	400 062	0.2
1	52-A	Goregaon West	Mumbai	400 062	0.5
1	52	Goregaon	West Mumbai	400 062	0.2

Id	House_no	Area	City	Pincode
1	52 (0.3)	Goregaon	Mumbai (0.6)	400 062
	52-A (0.7)	West (0.6)	West Mumbai	(1.0)
		Goregaon (0.4)	(0.4)	

Id	House_no	Area	City	Pincode	Prob
1	52 (0.167)	Goregaon	Mumbai (1.0)	400 062	0.6
	52-A (0.833)	West (1.0)		(1.0)	
1	52 (0.5)	Goregaon (1.0)	West	400 062	0.4
	52-A (0.5)		Mumbai (1.0)	(1.0)	

IMPRECISION OF EXTRACTION

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

Id	Title	Pr
1	Last Theorem	0.5
2	"Transaction Processing: Concepts and Techniques"	0.95
3	Transaction Processing	0.4
4	Fermat's Last Theorem	0.3
5	The Fermat's Last Theorem	0.5
6	Transaction Processing: Concepts and Techniques	1.0
7	Transaction Processing Concepts and Techniques	1.0
8	Fermat's Last Theorem	0.9
9	Fermat's Last Theorem	0.8

SUMMARY

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