Information Extraction

Defination:

IE task:

1. for each text in a set of unstructured natural language texts identify information about predefined class of **entities**, **relationships** or **events**
2. and record this information in a structured form by either:
    - Annoating the source text. (e.g. using XML tags) 标注
    - Filling in a data structure separate from the text (e.g. a template or a data-based record or "stand-off annotation") 填表

IE may also describe as:

- populating structured information repository(database) from unstructured, or free text, information source 从非结构化自由文本或信息源填充到结构化的信息源
- creating a sematically annotated text collection 创建一个语义标注的文本集合

The resulting structured data source is then used for other purpose:

- searching or analysis using conventional database quires使用传统数据库查询语句进行查找和分析
- Data-mining
- generating a summary(perhaps in another language)

Comparasion with IR:

Infomation Rerival：

Task: Given: a doucument collection and a user query Return: a (ranked) list of documents relevant to the user query

Strengths: Can search huge document collections very rappidly Intensive to genre and domain of texts Relatively straightforward to implement -challenges scaling to huge, dynamic document collections, e.g. the Web

Weaknesses: Documents are returned not information/answers, so User must further read texts to extract information Output is unstructured so limited possibilities for direct data mining/further processing

Information Extraction

Task: Given: a document collection and a predefined set of entities, relations and/or events Return: a structured representation of all mentions of the specified entities, relations and/or events

Strengths: Extracts facts from texts, not just texts from text collections Can feed other powerful applications (databases, semantic indexing engines, data mining tools)

Weakness: Systems tend to be genre/domain specific and porting to new genres and domains can be time-consuming/requires expertise Limited accuracy Computationally demanding, so performance issues on very large collections

Examples:

- Scrapping web pages to build structured databases of job postings, apartment rentals, seminar announcements, etc.
- Assisting biomedical database curators by extracting biomedical entities and relations from the scientific literature prior to entry in a human-maintained database (e.g. Flybase)
- Assisting companies in competitor intelligence gathering, e.g. management or researcher succession events, new product or project annoucements, etc.

Overview tasks:

Event Extraction:

- Task: identify all reports of event instances, typically a small set of classes.
- May be divided into 2 sub-tasks:
  - Event detection: find mentions of event in text
  - Event classification: assign detected events to one of a set of classes.
- Examples:
  - Rocket/missile launches
  - Management succession events
  - Joint venture/product announcements
  - Terrorist attacks
- Events may be simply viewed as relations. However they are typically complex relations that
  - Are temporally situated and often of relatively short duration
  - Involve multiple role players (frequently > 2)
  - Are often expressed across multiple sentences

Overview approaches:

4 categories:

1. Knowledge Engineering Approaches

Such systems use manually authored rules and can be divided into

- "Deep" - linguistically inspired "language understanding" systems
- "Shallow" - systems engineered to the IE task, typically using pattern-action rules
  - Pattern: "Mr. $uppercase-initial-word"
  - Action: add-entity(person("Mr. $uppercase-initial-word"))
  - Pattern : \$Person, $Position of $Organization

- Action: add-relation(is-employed-by($Person, $Organization))

2. Supervised Learning Approaches

Systems are given texts with manually annotated entities + relations

For each entity/relation create a training instance

k words either side of an entity mention

k words to the left of entity 1 and to the right of entity 2 plus the words in between

Training instances represented in terms of features

words, parts of speech, orthographic characteristics, syntactic info

Systems may learn 1. **patterns** that match extraction targets; 2. **classifiers** that classify tokens as beginning/inside/outside a tag type

Learning techniques include : covering algorithms, HMMs, SVMs

3. Bootstrapping Approaches: A technique for relation extraction that requires only minimal supervision.

1) System are given **seed tuples/ seed patterns** or both

2) System searches in large corpus for ① find tuples -> extract new pattern ② find patterns -> extract new tuple

3) New tuples are summed to stand in the required relation and are added to the tuple store

4) Iterate until convergence.

4. Distant Supervision Approaches: Sometimes also called "weakly labelled" approaches

1) Assumes a (semi-)structured data source, such as wikipedia infoboxes, Freebase or Wikipedia, Flybase or the Yeast Protein Database, Which contains tuples of **entities** standing in the **relation** of interest and, ideally, a pointer to a **source** text

2)Tuples from data source are used to label: ①the text with which they are associated, if available; ② documents from the web, if not

3) labelled data is used to train a standard supevised named entity or relation extraction system

Evalution + Shared task challenges:

Keys: Correct answers, produced manually for each extraction task(filled templates or SGML annotated texts); Responses: Scoring of system results, automatically

At least some portion od the answer keys are multiply produced by different humans so that **interannotater agreement** figures can be computed.

Principle metrics - borrowed from information retrieval - are:

- Precision
- Recall
- F-measure

Shared task Challenges

Are community wide exercise in which groups of research engage in a friendly competition to build systems to address a common task

key elements are:

- An agreed task definition
- Annotated text resources for training and testing
- agreed metrics for evaluation
- An agreed schedule for release of resources, system development, system evaluation and a coreference to discuss results

Shared task challenges in IE have include: MUC, ACE, TAC, BioCreative

Define the core methodology of the field and have led to significant progress

Named Entity Recognition

Task

- Entity extraction / **Named entity extraction**: for entity, identify **extent** and **type**

  Types of entities which have been addressed by IE systems include

  - Named individuals: organizations, persons, locations, books, films, ships, restaurants
  - Named kinds: Proteins, chemical compounds/drugs, disease, aircraft components
  - Times: temporal expressions – dates, times of day
  - Measures: monetary expressions 货币表示, distances/sizes, weights . . .
- **Coreference**: link together all textual references to the same real world entity ((pronouns, names/definite descriptions, abbrivated forms, orthographic variants))

Approaches:

4 categories: 1) **Rule-based/ Knowledge-engineering** 2) **Supervised learning** 3) Bootstrapping Approaches 4) Distant Supervision Approaches

Rule-based/ Knowledge-engineering

Such systems typically use: named entity lexicons and manually authored pattern/action rules or regular expression

Dominant approach in the 1990s and still in use in many IE systems today.

One such NER system, developed for participation in MUC-6, is described in Wakao et al. (1996) – will use as an example.

The Wakao et al. system recognizes organisation, person and location names and time expressions in newswire texts

System has three main stages:

- Lexical processing
- NE parsing
- Discourse interpretation

Step1: Lexical processing

Many rule-based NER systems made extensive use of specialised lexicons of proper names, such as gazetteers – lists of place names

The Wakao et al. system has specialised lexicons for

- Organisations (2600 entries)
- Locations (2200 entries)
- Person names (500 entries)
- Company designators (e.g. Plc,Corp, Ltd – 94 entries)
- Person titles (e.g. Mr, Dr, Reverend – 160 titles)

Why not use even larger gazetteers?

- e.g. Gazetteer of British Place Names claims it "provides an exhaustive Place Name Index to Great Britain, containing over 50,000 entries"

Reasons:

- Many NEs occur in multiple categories – the larger the lexicons the greater ambiguity, e.g.,
    - Ford – company vs Ford – person vs Ford – place
- the listing of names is never complete, so need some mechanism to type unseen NEs in any case

Principal lexical processing sub-steps in the Wakao et al. system are:

- Tokenisation, sentence splitting, morphological analysis
- Part-of-speech tagging – tags known proper name words and unknown uppercase-initial words as proper names (NNP, NNPS)
- Name List/Gazetteer Lookup and Tagging (organisations, locations, persons, company designators, person titles)
- Trigger Word Tagging – certain words in multi-word names function as trigger words, permitting classification of the name
    - e.g. Airlines in Wing and Prayer Airlines
    - system has trigger words for various orgs, gov't institutions, locations

Example:
Norwich Investment Bank plc. today announced ... $\longrightarrow$
Norwich$_{NNP/LOC}$ Investment$_{NNP}$ Bank$_{NNP/ORG-TRIGGER}$ plc.$_{NN/CDG}$ today$_{RB}$ announced$_{VBD}$ ...

Step2: NE parsing

The system has 177 hand-produced rules for proper names: 94 for organisation; 54 for person; 11 for location; 18 for time expressions.

A fragment of the proper name grammar:

- NP--> ORGAN_NP
- ORGAN_NP --> LIST_LOC_NP NAMES_NP CDG_NP
- ORGAN_NP --> LIST_ORGAN_NP NAMES_NP CDG_NP
- ORGAN_NP --> NAMES_NP '&' NAMES_NP
- NAMES_NP --> NNP NAMES_NP
- NAMES_NP --> NNP

E.g. Marks & Spencer

Step3 : Discourse interpretation

Coreference Resolution

① antecedent <-- --> anaphor

E.g., Alice is my sister. She is CEO of MS. (Antecedent – Alice; anaphor – she)

② unclassified PN <-- --> a variant form of a classified PN

e.g. Ford – Ford Motor Co; CAA – Creative Artists Agency

In such cases the unclassified PN may be inferred to have the same class as the classified PN. Wakao et al. use 45 heuristics 启发式方法 of this type for organisation, location, and person names.

③ unclassified PN <-- --> a definite NP (permits the PN's class to be inferred)

E.g. Kellogg ... the breakfast cereal manufacturer

Semantic Type Inference

Semantic type information about the arguments in certain syntactic relations is used to make inferences permitting the classification of PNs:

- noun-noun qualification: e.g. Erickson stocks
- possessives: e.g. vice president of ABC, ABC's vice president
- apposition: e.g. Miodrag Jones, president of XYZ

Evaluation of Wakao et al.

Evaluated on MUC-6 NE evaluation set – a blind test set of 30 Wall Street Journal Articles containing:

- 449 organisation names
- 373 person names

- 110 location names
- 111 time expressions

Strengths

• High performance – only several points behind human annotators Transparent – easy to understand what system is doing/why

Weaknesses

• Porting to another domain requires substantial rule re-engineering

• Acquisition of domain-specific lexicons

• Rule writing requires high levels of expertise

Results were:

Proper Name Class	Recall	Precision
Organisation	91%	91%
Person	90%	95%
Location	88%	89%
Time	94%	97%
Overall	91%	93%

Best system results on this evaluation were F-measure = 96.42%

- Human results were 96.68%

Supervised learning

Supervised learning approaches aim to address the portability problems inherent in knowledge engineering NER

- Instead of manually authoring rules, systems learn from annotated examples
- Moving to new domain requires only annotated data in the domain can be supplied by domain expert without need for expert computational linguist

A wide variety of supervised learning techniques have been tried, including

- Hidden Markov models
- Decision Trees
- Maximum Entropy
- Support Vector Machines
- Conditional Random Fields
- AdaBoost
- Deep Learning

Systems may learn

- patterns that match extraction targets
- classifiers that label tokens as beginning/inside/outside a tag type

Most work in recent years has followed the latter approach – called sequence labelling.

In sequence labelling for NER, each token is given one of three label types: (this scheme is called BIO or IOB)

- B_{Type} if the token is at the beginning of a named entity of type = Type (here, e.g., Type $\in$ {ORG, PER, LOC}).
- I_{Type} if the token is inside a named entity of type = Type
- O if the token is outside any named entity

For obvious reasons this scheme is called BIO or sometimes IOB sequence labelling

Example:

- Suppose we have the sentence

  [$_{ORG}$ American Airlines], a unit of [$_{ORG}$ AMR Corp.], immediately matched the move, spokesman [$_{PER}$ Tim Wagner] said.

  (Jurafsky and Martin, 2nd ed., p. 730)
- In BIO encoding this example looks like this $\longrightarrow$
- Given labelled sequences like this example sentence as training data, the task of the supervised learner to learn to predict the labelling of a new, unlabelled example.

Words	Label
American	B_{ORG}
Airlines	I_{ORG}
,	O
a	O
unit	O
of	O
AMR	B_{ORG}
Corp.	I_{ORG}
,	O
immediately	O
matched	O
the	O
move	O
,	O
spokesman	O
Tim	B_{PERS}
Wagner	I_{PERS}
said	O
.	O

Features for Sequence Labelling

Given a BIO-type encoding, each training instance (token) is typically represented as a set of features.

Features can be not only characteristics of the token itself but of neighbouring tokens as well

- usually consider tokens in a window of e.g. ± 2 or 3 tokens either side of the training instance

Features commonly used for NER sequence labelling include:

Feature	Explanation
Lexical items	The token to be labeled
Stemmed lexical items	Stemmed version of the target token
Shape	The orthographic pattern of the target word
Character affixes	Character-level affixes of the target and surrounding words
Part of speech	Part of speech of the word
Syntactic chunk labels	Base-phrase chunk label
Gazetteer or name list	Presence of the word in one or more named entity lists
Predictive token(s)	Presence of predictive words in surrounding text
Bag of words/Bag of N-grams	Words and/or *N*-grams occurring in the surrounding context

For case sensitive languages like English the orthographic pattern of a token carries significant information.

Commonly used “shape” features include:

Shape	Example
Lower	cummings
Capitalized	Washington
All caps	IRA
Mixed case	eBay
Capitalized character with period	H.
Ends in digit	A9
Contains hyphen	H-P

After a model has been learned, then at classification time the classifier extracts features from: the input string; its left predictions.

The available features for classification are those shown in the shaded area in the following figure:

Sequence Labelling Overview

- The following diagram recaps the main steps in the sequence labelling approach to NER.

One implementation of the BIO-based sequence labelling for NER is described by Carreras et al. Achieved highest score in the CONLL 2003 NER shared task challenge.

Notable aspects of their approach include:

- They divided the problem into two parts
  - **NE detection**: in a first pass over the text BIO tags are assigned without regard to type – i.e. boundaries are found for all NE's regardless of whether they are organisations,

persons, locations, etc.
    - **NE classification**: in a second pass the NE's detected in the first pass are assigned a class (organisation, person, location, etc.)
    - Two pass approach has the advantage that training data for all NE classes can be used for the NE detection task
- They used the **Adaboost** classifier
- They used all features mentioned above plus some additional ones, e.g.
    - Type pattern of consecutive words in context – functional (f), capitalized (C), lowercased (l), punctuation mark (.), quote (), other (x) – e.g. word type pattern for the phrase John Smith payed 3 euros is CClxl.

> - Overall performance on NE Detection:
> - 91.93% precision/94.02% recall on English test set
> - 85.85% precision/72.61% recall on German test set
> - Note: all common nouns are capitalised in German
> - Overall best performance on NE Classification, assuming perfect Detection:
> - 95.14% accuracy for English
> - 85.14% accuracy for German
> - Overall performance for NE Detection + Classification:
> - 84.05% precision/85.96% recall on English test set
> - 75.47% precision/63.82% recall on German test set
> - Looking at different entity classes, LOC and PER score consistently higher than ORG and MISC

Entity linking

One important application of IE is **knowlege base population (KBP**) facts are gathered from open access web sources and used to build a structuted information repository.

For KBP to work, not only must entities be detected, they must be linked to the appropriate entry in the KB, if facts are to be correctly assembled.

This leads to the Entity Linking Task: Given a text with a recognised NE mention in that text and a knowledge base (KB), such as Wikipedia, link the NEs to the matching entry in the KB if there is one, else create an entry.

Is this task difficult? – yes!!

- Wikipedia contains over 200 entries for John Smith
- There are at least 1,716 places called San Jos´e (or San Jose); 41 Springfield's in the US
- Ashoka Restaurant, ABC Taxis, . . .

Many approaches have been developed.

Approaches

Simple approach: given a text T containing an NE mention m and using Wikipedia as a KB

1. index all pages in the KB using an information retrieval system

2. build a query from T (e.g. use the sentence/paragraph/whole text) containing m and search the KB
3. from the ranked list of KB pages returned by step 2 pick the high ranked page whose name matches m and return it

Problem: doesn't work very well

More successful approaches consider **disambiguating all NEs jointly**

- Intuition: in disambiguating a text mentioning Ashoka and Sheffield, the Ashoka mentioned is likely to be in Sheffield, while the Sheffield is likely to be one containing an Ashoka restaurant.

Conclusion

- Named Entity Recognition (NER) is a core IE technology that is now relatively mature and at "usable" performance levels
- NER aims to detect and classify all mentions of named entities of a given set of entity types within a given text
- Techniques used have included:
  - knowledge engineering approaches
  - supervised learning approaches
    - a common approach here is to use BIO sequence labelling
- Open challenges include:
  - reducing the amount of training data needed via, e.g. bootstrapping techniques
  - exploiting existing structured data sources to generate "weakly labelled" training data (aka distant supervision)
  - expanding the classes of entities addressed
  - developing NERs for languages other than English

Relation Extraction

Task : identify all assertions of relations, usually binary, between entities identified in entity extraction.

May be divided into 2 sub-tasks:

- Relation detection: find pairs of entities between which a relation holds
- Relation classification: for pairs of entities between which a relation holds, determine what the relation is.

Examples

- LOCATION OF holding between
  - ORGANISATION and GEOPOLITICAL LOCATION
  - medical INVESTIGATION and BODY PART

- EMPLOYEE OF holding between PERSON and ORGANISATION
- PRODUCT OF holding between ARTIFACT and ORGANISATION
- INTERACTION holding between PROTEIN and

Relation Extraction is **chanlleging** for several reasons:

- The same relation may be expressed in several ways: Canonical 典范; Synonyms 同义词, Syntactic variations 句法变化
- Discovering relations frequently depends upon being able to follow coreference links.
- The information required may be spread across multiple sentences and discovering relations may depend upon following coreference links.
- The information to be extracted may be implied by the text, rather than explicitly asserted, and extracting it may require inference

Approaches:

Knowledge Engineering Approaches

Such systems use manually authored rules and can be divided into

- "shallow" – systems engineered to the IE task, typically using pattern-action rules

  Pattern: \$Person, $Position of $Organization" Action: add-relation(is-employed-by($Person, $Organization))

  - "deep" – linguistically inspired "language understanding" systems
    - typically parse input using broad coverage NL parser to identify key grammatical relations, like subject and object
    - use transduction rules to extract relations of interest from parser output
    - extraction rules over parser output allow a wider set of expressions to be captured than with regex's over words and NE tags alone
      - Example shows how multiple surface forms share underlying syntactic structure: here both have form subject = PER, object = ORG and verb = works

for

Strengths:High precision; System behaviour is human-comprehensible

Weaknesses: The writing of rules has no end; New rules needed for every new domain (pattern action rules for shallow approaches; transduction rules for deep approaches)

Supervised learning approaches

First question to be asked: What is to be learned?

Answer 1: rules that

- Match to all and only relation bearing sentences
- Capture substrings within the matched text that correspond to relation arguments

Answer 2: binary classifier that when applied to a sentence containing instances of the entity types between which the relation holds

- Returns 1 if the relation holds in this instance
- Returns 0 if the relation does not hold in this instance

As with NER can be divided into detection and classification stages:

- Classifier 1 (binary) determines whether a given sentence expresses any of a set of relations of interest (relation detection)
- Classifier 2 (multi-way) determines, for positive outputs from Classifier 1, which relation holds (relation classification)

Rule learning approach popular in late 1990's/early 2000's; since then most work focusses on classifier approach – we'll look at the 2nd only

In classification approaches to relation extraction:

- Assume entities to be related already tagged
- Use any algorithm for learning binary classifiers to learn to distinguish instances (typically sentences) where
    - entities co-occur and relation holds (positive instances)
    - entities co-occur and relation does not hold (negative instances)
- Key issue: iwhat features do we use to represent the instances? Features used fall into 3 broad classes:
    - Features of the named entities
    - Features from the words in the text, usually words from 3 locations
        - words between the two NE candidate arguments
        - words in a fixed window to the left of the 1st candidate
        - words in a fixed window to the right of the 2nd candidate
    - Features about the entity pair within the sentence, e.g.
        - how far the entities are apart (in words or constituents)
        - whether other NE's occur between them

- features from the syntactic structure of the sentence

Suppose we have the sentence

> [$_{ORG}$ American Airlines], a unit of [$_{ORG}$ AMR Corp.], immediately matched the move, spokesman [$_{PER}$ Tim Wagner] said.

Then features extracted for this example when classifying the tuple:

⟨ American Airlines, Tim Wagner ⟩

Entity-based features	
Entity_1 type	ORG
Entity_1 head	*airlines*
Entity_2 type	PERS
Entity_2 head	*Wagner*
Concatenated types	ORGPERS
Word-based features	
Between-entity bag of words	{ *a, unit, of, AMR, Inc., immediately, matched, the, move, spokesman* }
Word(s) before Entity_1	NONE
Word(s) after Entity_2	*said*
Syntactic features	
Constituent path	$NP \uparrow NP \uparrow S \uparrow S \downarrow NP$
Base syntactic chunk path	$NP \rightarrow NP \rightarrow PP \rightarrow NP \rightarrow VP \rightarrow NP \rightarrow NP$
Typed-dependency path	$Airlines \leftarrow_{subj} matched \leftarrow_{comp} said \rightarrow_{subj} Wagner$

Strengths: No need to write extensive/complex rule sets for each domain; Same system straightforwardly adapts to any new domain, provided training data is supplied

Weaknesses: Quality of relation extraction dependent on quality and quantity of training data, which can be difficult and time consuming to generate; Developing feature extractors can be difficult and they may be noisy (e.g. parsers) reducing overall performance

Bootstrapping Approaches

Motivation: reduce number of manually labelled examples needed to build a system

Key idea: start with a document collection D and either :

1. set of trusted tuples T (e.g. pairs of entities known to stand in the relation of interest)
2. set of trusted patterns P (i.e. patterns known to extract pairs of entities in the given relation with high accuracy)

Then, if

1. then find tuples from T in sentences S in D, extract patterns from context of sentences in S, add patterns to P and then use P to find new tuples in D and add to T; repeat until convergence
2. then match patterns from P in sentences S in D, extract tuples from pattern matches in sentences in S, add tuples to T and then use tuples in T to find new patterns in D and add to P; repeat until convergence

Diagrammatically, this can be shown as follows:

One early system employing this approach was **DIPRE** – Dual Iterative Pattern Relation Expansion – proposed by Sergie Brin (1999)

Aim: to extract useful relational tuples from the Web, of the form (PERSON, BOOK TITLE) – e.g. (Leo Tolstoy, War and Peace)

Method:

- Exploit "duality of patterns and relations"
  - Good tuples help find good patterns
  - Good patterns help find good tuples
- Starting with user-supplied tuples, iteratively
  - Use these tuples to find patterns
  - Use the patterns to find more tuples

The main loop in DIPRE is as follows:

1. $R^{'} \leftarrow$ Sample: is an approximation of the target relation (a set of tuples); Sample is a small user-supplied sample (e.g. 5 author-title pairs)
2. $O \leftarrow$ FindOccurrences($R^{'}, D$) : Find all occurrences of tuples of $R^{'}$ in D
3. $P \leftarrow$ GenPatterns(O) : Generate patterns based on the set of occurrences – want patterns to have low error rate and, ideally, high coverage (can compensate for latter with large database (e.g. the Web)
4. $R^{'} \leftarrow M_D(P)$: Update $R^{'}$ with the set of tuples from documents in D that matched by patterns in P
5. If $R^{'}$ is large enough return, else go to 2

Brin reports an experiment with finding (author,title) pairs on the web

- Patterns are defined as 5-tuples: (order, urlprefix, prefix, middle, suffix)

    - If order is true an (author, title) pair matches the pattern if there is a document in the collection (web)
        - whose URL matches urlprefix*
        - which contains text which matches the RE *prefix, author, middle, title, suffix*
        - more detailed RE's are given for author and title
    - If order is false title and author are switched
- Occurrences are defined as 7-tuples: (author, title, order, url, prefix, middle, suffix)
    - Order records the order the author and title occurred in the text
    - URL is the URL of the document the occurrence was found in
    - Prefix is the m characters (in tests m=10) preceding the author (or title)
    - Middle is text between author and title
    - Suffix is m characters following title (or author)

An algorithm for generating a pattern given a set of occurrences is described

- Algorithm inisists order and middle of all occurrences is the same they form part of the generated pattern
- Additionally pattern contains
    - longest matching prefix of the url of all the occurrences
    - longest matching suffix of the prefix of all the occurrences
    - longest matching prefix of the suffix of all the occurrences
    - See Brin (1999) for details

Patterns are assessed for specificity and rejected if their specificity is too low, i.e. if they are too general

- Specificity of a pattern is defined in terms of the product of the lengths of the pattern's middle, urlprefix, prefix and suffix
- For a pattern p, specificity(p) × n must exceed some threshold t, where n is the number of books with occurrences supporting the pattern p

- Used 24 million web pages + 5 seed tuples

Author	Title
Isaac Asimov	The Robots of Dawn
David Brin	Startide Rising
James Gleick	James Gleick
Charles Dickens	Great Expectations
William Shakespeare	The Comedy of Errors

- Yielded 199 occurrences and generated 3 patterns
- These 3 patterns produced 4047 unique (author, title) pairs
- A search over 5 million web pages yielded 3972 occurrences of these books – stopped at this point due to computational constraints
- These occurrences produced 105 patterns which in turn produced 9369 (author, title) pairs – some had bad authors and were rejected
- Using these working pairs in a final iteration resulted in 9988 occurrences, then 346 patterns and then 15257 unique books
- Manual inspection of 20 from the final list showed 19 were bonafide books and 1 was an article

Strengths: Need for manually labelled training data is eliminated

Weaknesses:

- Can suffer from semantic drift – when an erroneous pattern introduces erroneous tuples, which in turn lead to erroneous patterns
  - Introduction of confidence measures for patterns and tuples can mitigate against this problem to some extent
- Works well when significant redudancy in assertion of specific tuples an in use of specific patterns to express a relation
  - True for some domains/relations and text collections, not for others
- Issues when multiple relations hold between the same pair of entities
  - e.g. suppose someone is born, is educated and dies in the same location, then a sentence containing occurrences of person name and location name could be expressing any of three relations

Distant Supervision Approaches

As with bootstrapping approaches, distant supervision approaches aim to reduce/eliminate the need for manually labelled training data

Key idea:

- Suppose we have a large document collection D plus a structured data source (e.g. a database) R that contains
  - many instances of a relation of interest in, e.g., a relational table
  - optionally, for each relation instance a link to a document in D providing evidence for the relation

- Then we can
  - search for sentences in D containing the entity pairs that occur in relation instances (tuples) in R
  - label these sentences as positive occurrences of the relation instance
  - use the labelled sentences as training data to train a standard supervised relation extractor

One well-known approach using distant supervision is described by **Mintz** et al. (2009)

Mintz et al. use Freebase as their structured data source

Relation name	Size	Example
/people/person/nationality	281,107	John Dugard, South Africa
/location/location/contains	253,223	Belgium, Nijlen
/people/person/profession	208,888	Dusa McDuff, Mathematician
/people/person/place_of_birth	105,799	Edwin Hubble, Marshfield
/dining/restaurant/cuisine	86,213	MacAyo's Mexican Kitchen, Mexican
/business/business_chain/location	66,529	Apple Inc., Apple Inc., South Park, NC
/biology/organism_classification_rank	42,806	Scorpaeniformes, Order
/film/film/genre	40,658	Where the Sidewalk Ends, Film noir
/film/film/language	31,103	Enter the Phoenix, Cantonese
/biology/organism_higher_classification	30,052	Calopteryx, Calopterygidae
/film/film/country	27,217	Turtle Diary, United States
/film/writer/film	23,856	Irving Shulman, Rebel Without a Cause
/film/director/film	23,539	Michael Mann, Collateral
/film/producer/film	22,079	Diane Eskenazi, Aladdin
/people/deceased_person/place_of_death	18,814	John W. Kern, Asheville
/music/artist/origin	18,619	The Octopus Project, Austin
/people/person/religion	17,582	Joseph Chartrand, Catholicism
/book/author/works_written	17,278	Paul Auster, Travels in the Scriptorium
/soccer/football_position/players	17,244	Midfielder, Chen Tao
/people/deceased_person/cause_of_death	16,709	Richard Daintree, Tuberculosis
/book/book/genre	16,431	Pony Soldiers, Science fiction
/film/film/music	14,070	Stavisky, Stephen Sondheim
/business/company/industry	13,805	ATS Medical, Health care

Table 2: The 23 largest Freebase relations we use, with their size and an instance of each relation.

Source: Mintz et al. (2009)

Freebase was a free on-line database of structured semantic data

- data derived from, e.g. Wikipedia infoboxes + other open access sources
- after filtering Mintz et al. derived 1.8 million instances of 102 relations connecting 940,000 entities
- Freebase no longer available – bought by Google and now forms part of Google Knowledge Graph (partly free, partly paid access)
- Similar current sources are DBPedia and WikiData

Mintz et al. use a dump of the text from Wikipedia as their document collection

- dump consists of ≈ 1.8 million articles, averaging 14.3 sentences/article
- used 800,000 articles for training and 400,000 for testing

Distant supervision assumption: if two entities participate in a relation, any sentence that contains those two entities might express that relation.

- So, tag all sentences containing the two entity mentions as mentions of the relation

Same relation may be expressed in different ways in different sentences.

E.g. [Steven Spielberg]'s film [Saving Private Ryan] is loosely based on the brothers' story.

Allison co-produced the Academy Award- winning [Saving Private Ryan], directed by [Steven Spielberg]...

- So, combine features from multiple mentions to get a richer feature vector

- Use multiclass logistic regression as a learning framework
- At test time features are combined from all occurrences of a given entity pair in the test data and the most likely relation (or none) is assigned

Also need negative instances – an'unrelated' relation!

- to get these, randomly select entity pairs that do not appear in any Freebase relation and extract features for them
- Could be related – i.e. wrongly omitted from Freebase – but effect of these rare occurrences should be low

Mintz et al. evaluate their approach

- humans evaluate highest ranked 100 and 1000 results per relation for 10 relations
- average precision for best feature combinations just under 70% (69% for top 10; 68% for top 1000)
- these results are competitive for knowledge engineering and "normal" supervised learning systems, which struggle to get over 75% on similar tasks

Strengths: Need for manually labelled training data is eliminated; Can very rapidly get extractors for a wide range of relations

Weaknesses: Precision still lags behind best knowledge-engineered/directly supervised learning approaches; Only works if a good supply of structured data is available for the relation(s) of interest

Conclusion

ralation extraction aims to detect and classify all mentions of a give set of relations holding between specified entity types within a given text

Relations extraction is a core IE technology that is stubbornly difficult, due to the highly variable ways relations can be expressed in natural language

Techniques used have include:

- Knowledge Engineering Approaches
- Supervised Learning Approaches
- Bootstrapping Approaches
- Distant Supervision Approaches

Open challenges include:

- improving precision and recall
- Handling: relationships expressed over > 1 sentences; textual entailment
- Improving bootstrapping techniques so as to minimize "semantic drift"
- developing relation extractors for languages other than English