Entity Recognition in the Text

Jelena Jovanović

Email: jeljov@gmail.com

Web: http://jelenajovanovic.net

Outline

- Major challenges in the domain of entity recognition
- Main approaches to entity recognition in text
  - List lookup approaches
  - Rule-based approaches
  - Machine learning based approaches
  - Approaches that combine m. learning and knowledge bases (e.g., Wikipedia)
- Useful Web resources (frameworks and services)

Major challenges in entity recognition

- Entity spotting - identification of the pieces of text that represent entities
  - Chunking – correctly selecting the sequence of words that represents an entity
    - entities can be represented with a single word (e.g., *MIT*) or a sequence of words (*Massachusetts Institute of Technology*)
  - Determining if a particular text segment really represents an entity (i.e., it is not a false positive)
    - particularly difficult when words that might represent an entity are placed at the beginning of a sentence (e.g., May, Galaxy, …)

Major challenges in entity recognition

- Determining the type of an entity

Group (Team) vs. Location:

“**England** won the World Cup” vs.

“The World Cup took place in **England**”

Company vs. Artefact:

“having shares in **BBC**” vs. “watching **BBC**”

Location vs. Organisation:

“she met him at **Heathrow**” vs. “the **Heathrow** authorities…”

Major challenges in entity recognition

- Recognizing segments of text that refer to the same entity
  - Problems: different ways of referencing the same entity; e.g.:
    - John Smith; Mr Smith; John
    - *UMBC*; *University of Maryland Baltimore County*

- Maintenance of lists/dictionaries with entity names
  - Such lists/dictionaries are required for a majority of todays’ entity recognition systems

Main approaches to entity recognition in text

- List lookup approaches
  - Rely on the use of domain specific dictionaries and gazetteer lists
- Rule-based approaches
  - Approaches that rely on shallow parsing of text
  - Approaches that rely on regular expressions
- Approaches based on machine learning
- Approaches based on m. learning and knowledge bases
- Hybrid approaches
  - They combine two or more of the aforementioned approaches
  - Most frequently applied in practice

LIST LOOKUP APPROACHES

List lookup approaches

- Capable of recognizing entities whose names are present in the available dictionaries/lists

- Typically applied when our task is domain specific and we have (or can assemble) lists of entity names
  - E.g., list of company names or list of experts from a particular domain

List lookup approaches

- Two kinds of methods for matching entity names:
  - *Exact matching* – requires exact (complete) matching between terms in the text and names from the lists/dictionary
  - *Approximate matching* – extends exact matching with techniques for approximate comparison of strings
    - E.g., Levenshtein distance (edit distance) - the minimum number of operations required to transform one string into the other; possible operations include insertion, deletion and change of one character

      Lev (machine, marine) = 2
      - deletion of ‘c’
      - replacement of ‘h’ with ‘r’

List lookup approaches

Gazetteer

- A tool that makes use of names lists to recognize entities in texts
  - gazetteer lists are plain text files with one data item (name) in each line
  - each list consists of names of a certain group of entities, such as names of cities, companies, days in a week,...
  - the index file is used for accessing individual lists
  - each token in the analyzed text is matched against names in the gazetteer lists; when a match is found, the token is annotated with the major and the minor type of the list where the match was found
  - An example: “Belgrade”

    Annotation: majorType = location, minorType = city

List lookup approaches

- Advantages:
  - Simplicity,
  - Speed (often better than for the other approaches),
  - Language independence,
  - Easily adaptable to new/different types of text
- Disadvantages:
  - Creation and maintenance of lists
  - Not able to recognize entities in the case of weak/partial matching of names from the lists and terms from the text
  - Do not consider the context of terms, and thus are incapable of resolving ambiguous terms

RULE-BASED APPROACHES

Rule based approaches: shallow parsing

- The main idea:
  - identify frequently occurring language forms (terms and phrases), and associate such language forms with their meaning;
  - derive a template (pattern) for each recognized language form;
  - formalize the templates so that the process can be automated; formalization is typically done using a rule-modeling language

Rule based approaches: shallow parsing

An example: recognition of entities of the type location:

CapWord + {City, Forest, Center}

e.g. Sherwood Forest

Cap Word + {Street, Boulevard, Avenue, Crescent, Road}

e.g. Portobello Road

“to the” COMPASS “of” CapWord

e.g. *to the south of Boston*

“based in” CapWord

e.g. *based in Boston*

CapWord “is a” (ADJ)? GeoWord

e.g. *Boston is a friendly city*

Rule based approaches: shallow parsing

- Well-known Hearst patterns for recognizing entities of different types

such NP as {NP,}* {or | and} NP

... works by such authors as Herrick, Goldsmith, and Shakespeare

NP {,} including {NP,}* {or | and} NP

All common-law countries, including Canada and England ...

NP {,} especially {NP,}* {or | and} NP

... most European countries, especially France, England, and Spain.

M. Hearst. Automatic Acquisition of Hyponyms from Large Text Corpora. In Proc. of the 14th Int'l Conference on Computational Linguistics, Nantes France, 1992 (link).

Rule based approaches: shallow parsing

- Using rules to formalize linguistic patterns
- For example, JAPE* is a rule language that allows for defining rules of the following form:

  template => action

  - Left hand side of the rule comprises one or more templates to be matched against the text;
  - Right hand side of the rule consists of statements that specify how the matched text will be annotated; they can also define different operations over annotations

*JAPE is a part of the GATE Java framework for text analysis

Rule based approaches: shallow parsing

- An example rule formalized using JAPE

```
Rule: Location_1 //CapWord + {City, Forest, Center}
(
  {Token.kind == word, Token.category == NP,
                            Token.orth == ″upperInitial″}
  {Token.kind == ″space″}
  ( {Token.string == ″City″} |
    {Token.string == ″Forest″} |
    {Token.string == ″Center″}
   )
):loc
 -->
 :loc.Location = {rule = ″Location_1"}
```

Rule based approaches: shallow parsing

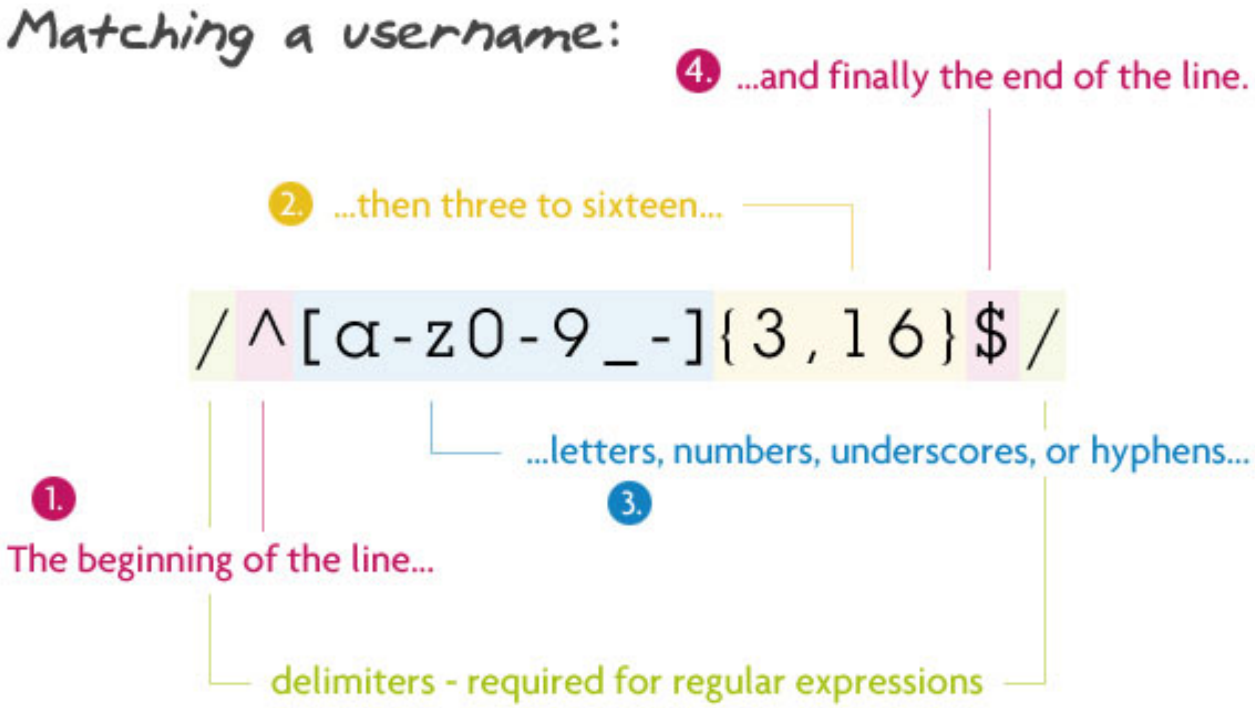
Challenges associated with the Shallow Parsing approach

- Creation of “trustworthy” templates
- Interpretation of the first word in a sentence
  - Is it capitalized just because it is the first word in a sentence or it is also a part of an entity name?
  - E.g., [All American Bank] vs. All [State Police]
- Structural ambiguity
  - E.g., [Cable and Wireless] vs. [Microsoft] and [Dell]
  - [Center for Computational Linguistics] vs.

    message from [City Hospital] for [John Smith]

Rule-based approaches: regular expressions

- Particularly suitable for detecting entities whose textual representation has to follow a well-defined structure
- An example: regular expression for recognizing someone's username



Matching a username:
4. ...and finally the end of the line.
2. ...then three to sixteen...
/^[a-z0-9_-]{3,16}$/
...letters, numbers, underscores, or hyphens...
3.
1.
The beginning of the line...
delimiters - required for regular expressions

Source: http://net.tutsplus.com/tutorials/other/8-regular-expressions-you-should-know/

APPROACHES BASED ON MACHINE LEARNING

Machine learning based approaches

- Supervised learning methods are typically used
  - The task of entity recognition is treated as a classification task
- The main idea:
  - Learn to distinguish entities of different types by identifying features that characterize entities of each particular type;
  - Such features are derived from the terms that represent entities in the text as well as the context of those terms
- Precondition:
  - The availability of sufficiently large training dataset, i.e., a corpus of annotated documents

Supervised M. Learning for Entity Recognition

We will consider the main elements of a supervised m. learning process when applied to the entity recognition in text:

- training dataset, i.e., annotated corpus of documents
- attributes/features for the m. learning model
- algorithm(s) to use
- evaluation of the built model

Training dataset (corpus)

An example illustrating the kind of text annotations required for training a supervised m. learning model:

Unlike <PERSON>Robert</PERSON>, <PERSON>John Briggs Jr</PERSON> contacted <ORGANIZATION>Wonderful Stockbrockers Inc</ORGANIZATION> in <LOCATION>New York</LOCATION> and instructed them to sell his <NUMBER>100</NUMBER> shares in <ORGANIZATION>Acme</ORGANIZATION>

Obviously, preparation of a training dataset (corpus) is a laborious task…

Training dataset (corpus)

- Luckily, some organizations, groups and individuals have published datasets (corpora) that can be used for training purposes
  - Contests organized in the scope of research conferences
    - Message Understanding Conference (MUC): MUC06 i MUC07 datasets
    - Conference on Computational Natural Language Learning (CoNLL): CoNNL-2002 i CoNNL-2003 datasets
  - Professional associations
    - Linguistic Data Consortium maintains a catalog of liguistic datasets
  - Research groups and individuals
    - Twitter NER – dataset that was used for training a model that recognizes entities in tweets (read more about it here)
    - GeneTag dataset – dataset published by the US National Center for Biotechnology Information

Selection of attributes/features

- Selection of features
  - Depends on the type of the text we analyze (e.g., tweets vs. newspaper articles vs. scientific papers)
  - Has a high influence on the system performance; it has the same if not greater impact than the selection of m. learning algorithm

Selection of attributes/features

- A wide range of features can be used:
  - Features related to individual words:
    - word length;
    - first capital letter;
    - all capital letters;
    - part of speech (POS) role;
    - the frequency of the word’s occurrence in the training set;
    - position in the sentence,…
  - Features related to the word’s context/surrounding:
    - width of the surrounding;
    - the types of words (POS) in the surrounding, …

Selection of attributes/features

The chosen set of features is used for representing individual words and/or phrases of the analyzed text

<u>A simple example</u>

Let’s suppose that we have selected the following features:

- Boolean attribute that indicates if a word starts with a capital letter
- Numerical attribute that represents the word’s length
- Nominal attribute representing the word written in lower case

Then, the sentence: “The apple sign makes Apple laptops easily recognizable.”

will have the following representation:

<true, 3, "the">, <false, 5, "apple">, <false, 4, "sign">, <false, 5, ”makes">, <true, 5, ”apple">, … , <false, 12, ”recognizable">

Selection of the learning algorithm

- Most frequently used mach. learning algorithms for the entity recognition task *
  - Decision trees
  - Hidden Markov Models (HMM)
  - Maximum Entropy classification
  - Support Vector Machines (SVM)
  - Conditional Random Fields (CRF)

*this is to serve just as an information, as these models are far more complex than those we have covered in the course

Evaluation of the model

- For the evaluation purposes, typical classification metrics are used, namely:
  - Precision, Recall, and F measure
- Software frameworks for the comparison (benchmarking) of different entity recognition tools:
  - NERD (Named Entity Recognition and Disambiguation): http://nerd.eurecom.fr/
  - GERBIL (General Entity Annotator Benchmark): http://gerbil.aksw.org/gerbil/

Alternatives to supervised m. learning

- The problem: preparation of a sufficiently large annotated corpus required for the training purposes, is a rather demanding task
- Therefore, semi-supervised and unsupervised m. learning approaches are increasingly explored
  - they do not require annotated corpora
  - they used to have lower performance than supervised models, but their performance is getting better and better

Semi-supervised m. learning

- A popular semi-supervised learning technique is *Bootstrapping*
  - It involves just a small degree of supervision, such as a set of seeds, for starting the learning process
- An example: let’s consider a system aimed at recognizing diseases in the text
  - initially, the user provides a small number of examples (disease names);
  - the system analyzes the text and tries to identify some contextual clues (features) common to the given examples; it then tries to find other instances of diseases by looking for similar contexts;
  - the learning process is reapplied to the newly found examples, so as to discover new relevant contexts (features);
  - by repeating this process, a large number of diseases and a large number of contexts will eventually be gathered.

Semi-supervised m. learning

Recommendation:

Lecture titled

Semi-supervised Learning Approaches

given by *Tom Mitchell* during

Autumn School 2006: Machine Learning over Text and Images

URL: http://videolectures.net/mlas06_mitchell_sla/

APPROACHES THAT COMBINE M. LEARNING AND KNOWLEDGE BASES (KB)

Approaches that combine m. learning and KBs

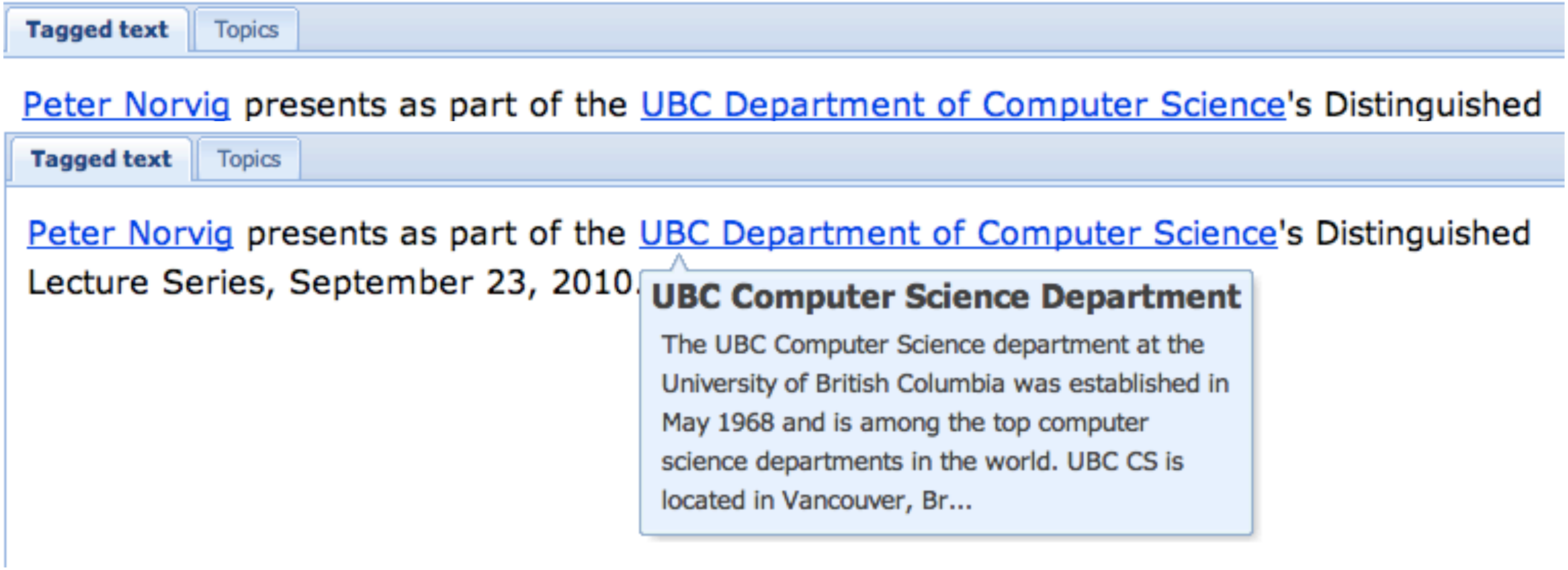
- Combination of supervised m. learning (classification) and knowledge stored in Web-based KBs
- Most frequently used KBs: Wikipedia, Freebase, DBpedia
- Specific advantage of these approaches: they allow for *entity disambiguation*, i.e., uniquely identifying the recognized entities

Approaches that combine m. learning and KBs

- Example output of a “traditional” entity recognition system:

Peter Norvig presents as part of the UBC Department of Computer Science's Distinguished Lecture Series, September 23, 2010.

- Results of a system that makes use of a KB (Wikipedia):



Peter Norvig presents as part of the UBC Department of Computer Science's Distinguished

Peter Norvig presents as part of the UBC Department of Computer Science's Distinguished Lecture Series, September 23, 2010.

UBC Computer Science Department

The UBC Computer Science department at the University of British Columbia was established in May 1968 and is among the top computer science departments in the world. UBC CS is located in Vancouver, Br...

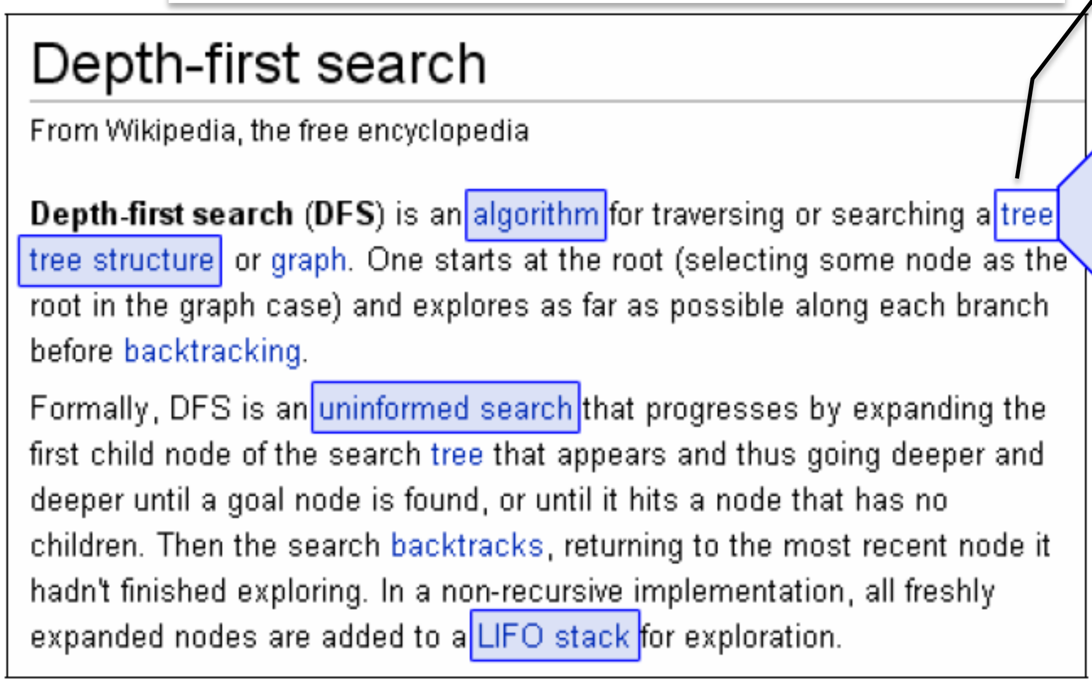
Approaches that combine m. learning and KBs

- An additional advantage of this type of approach is easier creation of the training set

- For instance, if Wikipedia is used as a KB:
  - Each term that has an embedded Wikipedia link is treated as a potential entity; we'll refer to such terms as *anchors*
  - Each *anchor* provides a few training instances:
    - positive example: link destination (i.e., Wikipedia page), that is, the "true" meaning of the given anchor in the given context
    - negative examples: all other potential destinations, i.e., all other possible meanings of the considered anchor

Creation of a training dataset by making use of the Wikipedia's internal links – an illustration

For the term (anchor) *tree,* there are 26 possible destinations (i.e., meaning); this results in 1 positive and 25 negative examples for training the algorithm

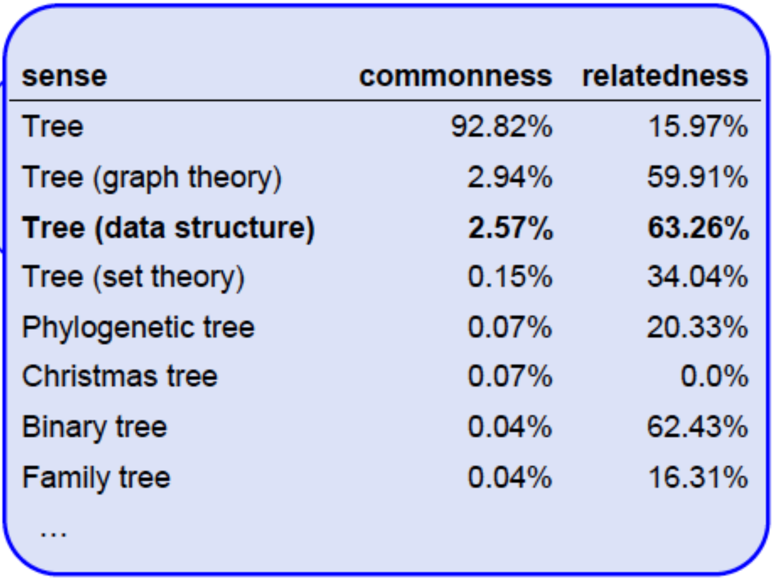


Depth-first search

From Wikipedia, the free encyclopedia

Depth-first search (DFS) is an algorithm for traversing or searching a tree, tree structure, or graph. One starts at the root (selecting some node as the root in the graph case) and explores as far as possible along each branch before backtracking.

Formally, DFS is an uninformed search that progresses by expanding the first child node of the search tree that appears and thus going deeper and deeper until a goal node is found, or until it hits a node that has no children. Then the search backtracks, returning to the most recent node it hadn't finished exploring. In a non-recursive implementation, all freshly expanded nodes are added to a LIFO stack for exploration.



sense	commonness	relatedness
Tree	92.82%	15.97%
Tree (graph theory)	2.94%	59.91%
Tree (data structure)	**2.57%**	**63.26%**
Tree (set theory)	0.15%	34.04%
Phylogenetic tree	0.07%	20.33%
Christmas tree	0.07%	0.0%
Binary tree	0.04%	62.43%
Family tree	0.04%	16.31%
…		

In this way, starting from, for instance, 500 Wikipedia articles one can generate a training set of >50,000 instances

The entity recognition process

The main steps of the entity recognition process:

1) *Entity spotting and candidate selection* – identification of terms that might represent entities in the text (entity-mentions), and selection of candidates from a KB for each entity-mention

2) *Disambiguation* – selection of the “best” entity, from the set of candidate entities, for each entity-mention*;*

3) *Filtering* – pruning the results with the aim of eliminating irrelevant entities

The Entity Spotting phase

- The objective of this phase is twofold:
  - to identify ‘mentions’ in the input text, i.e., the parts of the text (single words or phrases) that represent entities;
  - to identify a set of candidate entities from a KB (e.g., Wikipedia or DBpedia) for each mention

The Entity Spotting phase

- An example

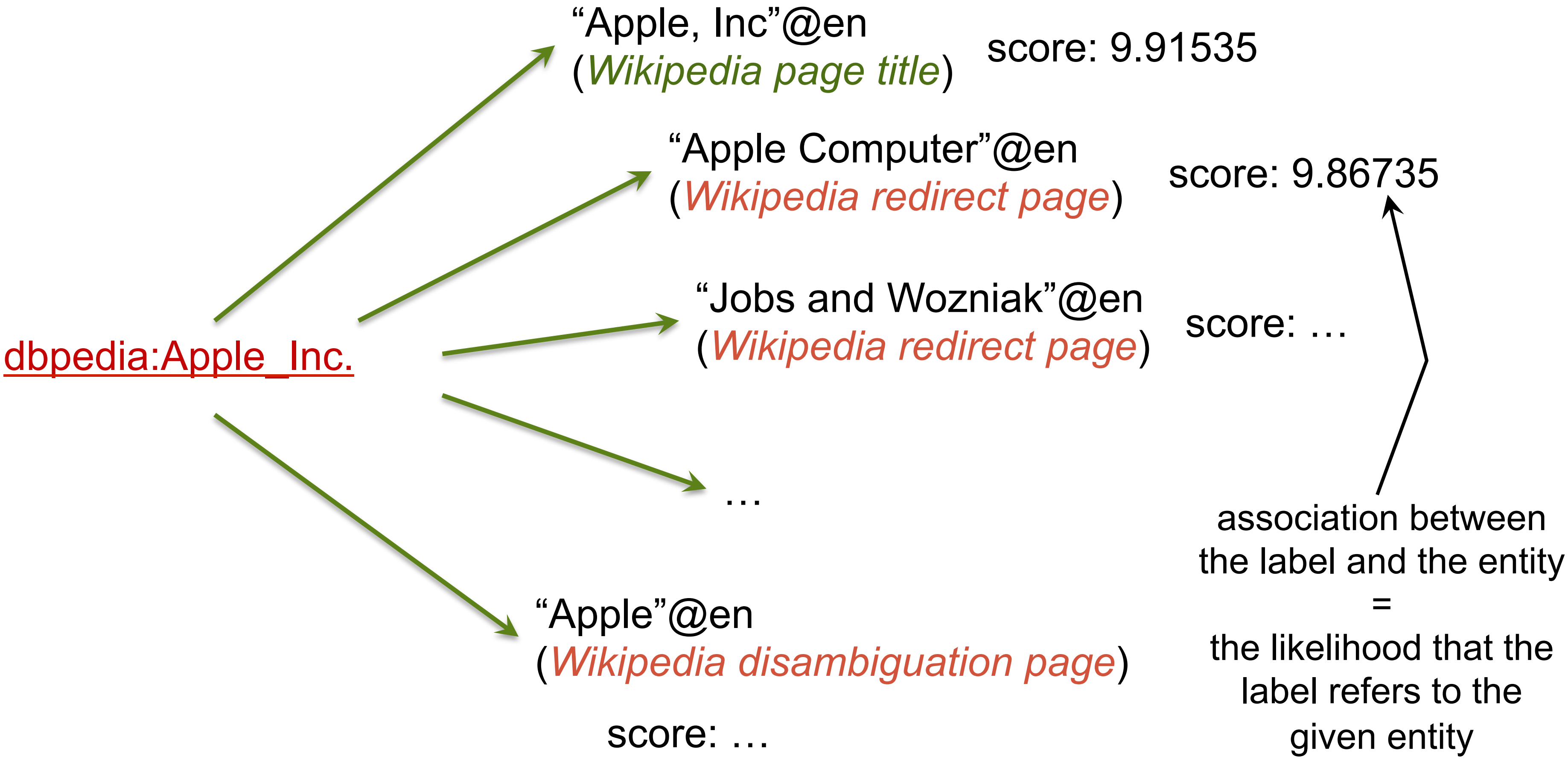
“They performed Kashmir, written by Page and Plant.
Page played unusual chords on his Gibson.”

dbpedia:Kashmir – a valley between Pakistan, India and Ladakh
dbpedia:Kashmir_(band) – a Danish rock band
dbpedia:Kashmir_(song) – 1975 song by rock band Led Zeppelin
dbpedia:Kashmir,_Iran – a village in Iran
…

The Entity Spotting phase (2)

- Typically, the tasks of this phase are performed as *dictionary look-up* tasks
  - a dictionary is typically created through the extraction of entity labels and descriptions from a specific knowledge base
  - Wikipedia is often used as the source of labels and descriptions
  - dictionary entries might be enriched with statistics computed over the content of the knowledge base

Example: DBpedia Lexicalization dataset



dbpedia:Apple_Inc.

"Apple, Inc"@en
(*Wikipedia page title*)

score: 9.91535

"Apple Computer"@en
(*Wikipedia redirect page*)

score: 9.86735

"Jobs and Wozniak"@en
(*Wikipedia redirect page*)

score: …

…

"Apple"@en
(*Wikipedia disambiguation page*)

score: …

association between the label and the entity
=
the likelihood that the label refers to the given entity

Available at: http://dbpedia.org/Lexicalizations

The Disambiguation phase

- The objective: for each entity-mention, select the entity/entities that properly reflect(s) the semantics of the mention
  - the selection is done from, often numerous, candidate entities identified in the spotting phase

- Continuing with the same example

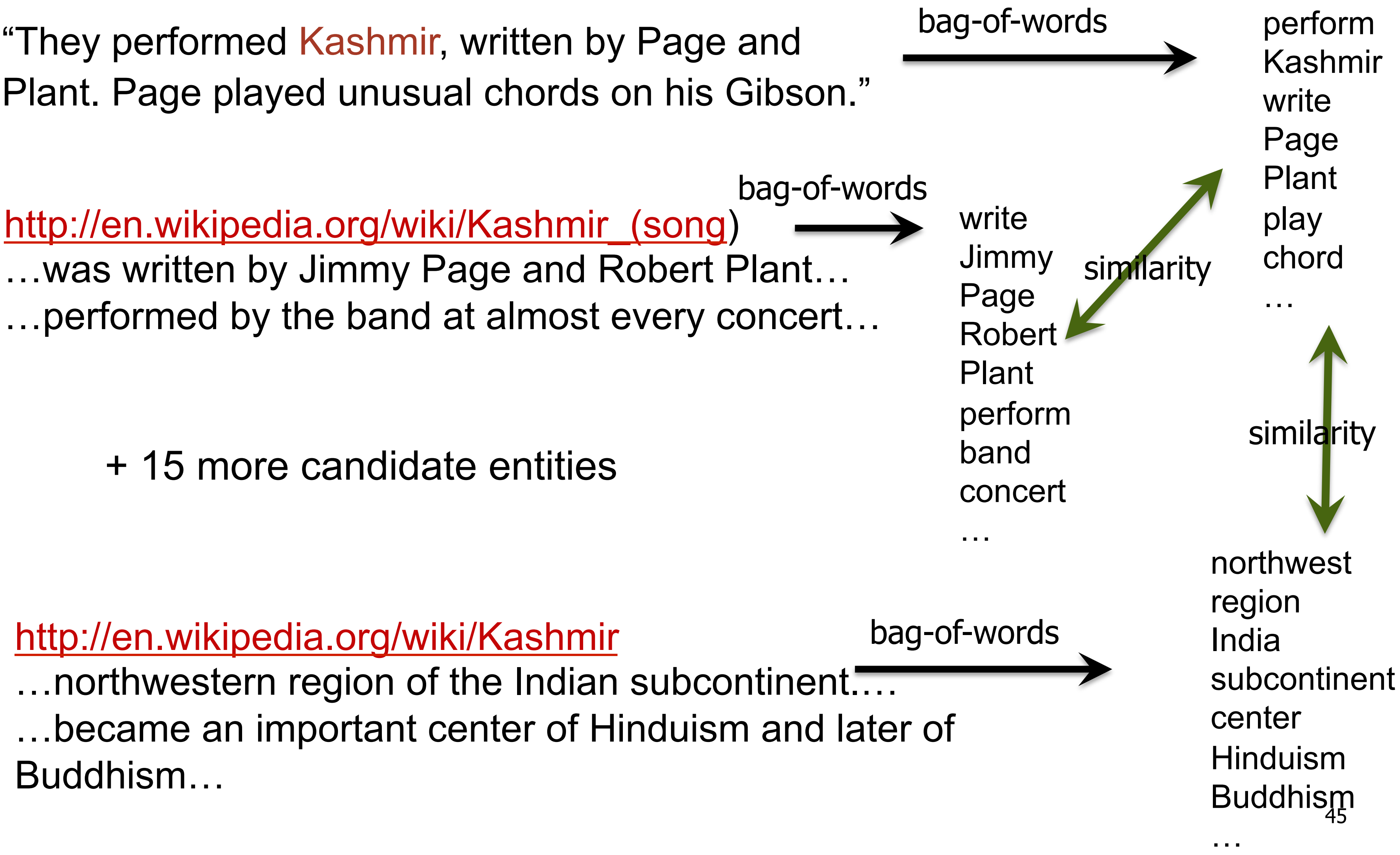
  "They performed Kashmir, written by Page and Plant. Page played unusual chords on his Gibson."

dbpedia:Kashmir – a valley between Pakistan, India and Ladakh
dbpedia:Kashmir_(band) – a Danish rock band
dbpedia:Kashmir_(song) – 1975 song by rock band Led Zeppelin
dbpedia:Kashmir,_Iran – a village in Iran
…

Disambiguation: Context-based approach

- Often used approach for entity disambiguation
- Relies on a comparison of the context of an entity-mention, and the context of the candidate entities
- Typically, context is represented as a bag-of-words, and the comparison is done using some similarity measure
  - E.g., Cosine similarity, weighted Jaccard coefficient, Wikipedia links-based measure

Context-based approach: an example



“They performed Kashmir, written by Page and Plant. Page played unusual chords on his Gibson.”

bag-of-words

perform
Kashmir
write
Page
Plant
play
chord
…

http://en.wikipedia.org/wiki/Kashmir_(song)
…was written by Jimmy Page and Robert Plant…
…performed by the band at almost every concert…

bag-of-words

write
Jimmy
Page
Robert
Plant
perform
band
concert
…

similarity

similarity

+ 15 more candidate entities

http://en.wikipedia.org/wiki/Kashmir
…northwestern region of the Indian subcontinent.…
…became an important center of Hinduism and later of Buddhism…

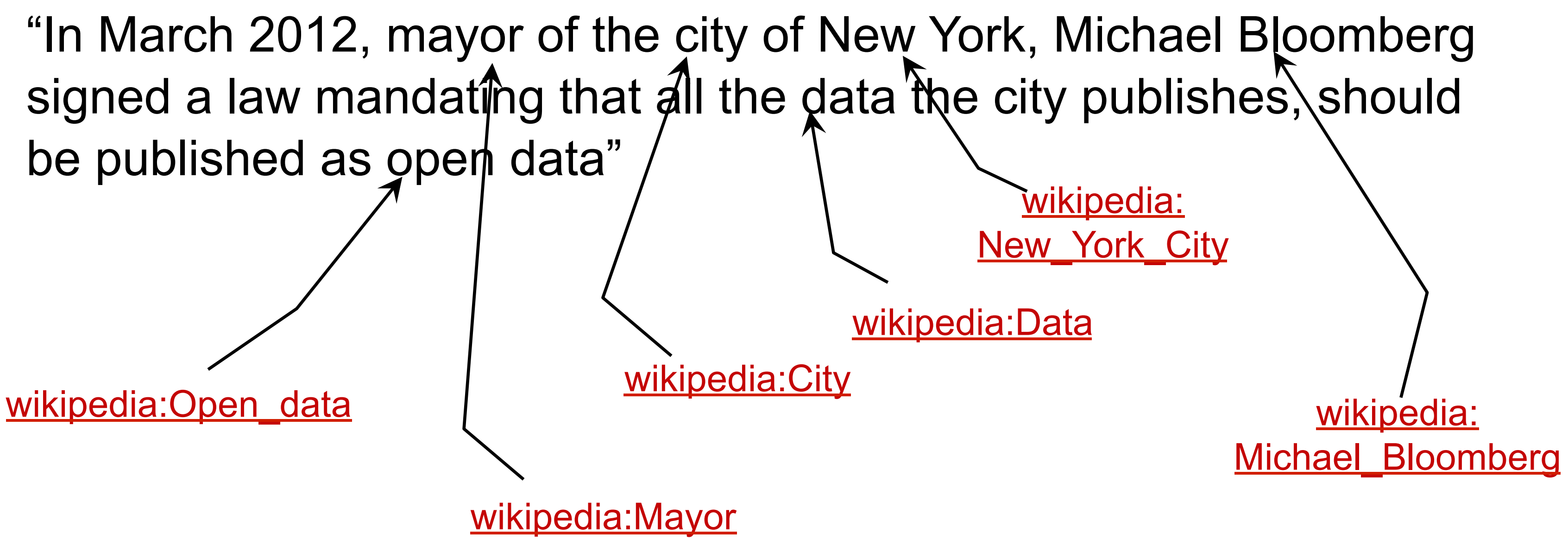
bag-of-words

northwest
region
India
subcontinent
center
Hinduism
Buddhism
…

The Pruning Results phase

- The objective is to remove annotations that would be of no interest to the user
  - e.g., overly general annotations or those that are only marginally related to the main topic of the text

- Example



"In March 2012, mayor of the city of New York, Michael Bloomberg signed a law mandating that all the data the city publishes, should be published as open data"

Tools that implement the described process

- Wikipedia Miner – offers a range of services:
  - *wikify* – identifies Wikipedia entities mentioned in the given text
  - *compare* – computes relatedness between the two given Wikipedia entities
  - *suggest* – suggests entities that are semantically related/similar to the given entities
- TagMe – offers the following services:
  - *tagging* – recognizes Wikipedia entities mentioned in the give text
  - *spotting* – identifies relevant terms/phrases in the given text (but does not establish links with the corresponding Wikipedia entities)
  - *relating* – computes semantic relatedness of the two given entities

USEFUL WEB RESOURCES

Useful Web resources

Wikilinks Corpus

- The largest public dataset for training supervised m. learning algorithms for the task of recognizing Wikipedia entities in the text
- URL: http://www.iesl.cs.umass.edu/data/wiki-links
- Some basic facts about this corpus:
  - 10 million Web pages
  - 3 million Wikipedia entities
  - 40 million uniquely identified entity mentions
  - published by Google Research on March 8, 2013.
- Read more about this dataset in the following article: Learning from Big Data: 40 Million Entities in Context

Useful Web resources

Software tools that perform entity recognition in the text

- AlchemyAPI: http://www.alchemyapi.com/tools/
- Open Amplify: http://www.openamplify.com/quickstart
- Text Razor: http://www.textrazor.com/
- TextWise: http://www.textwise.com/
- TagMe: http://tagme.di.unipi.it/
- Wikipedia Miner: http://wikipedia-miner.cms.waikato.ac.nz/
- Denote: http://denote.io/

Useful Web resources

- State-of-the-art Java frameworks for text analysis and meaning extraction
  - Stanford CoreNLP: http://nlp.stanford.edu/software/corenlp.shtml
  - Apache OpenNLP: http://opennlp.apache.org/
  - Apache Stanbol: http://stanbol.apache.org/
  - GATE: http://gate.ac.uk/
  - LingPIPE: http://alias-i.com/lingpipe/

(Anonymous) questionnaire for your critique, comments, suggestions:

http://goo.gl/cqdp3I