Extraction of Semantic Relations between Concepts with KNN Algorithms on Wikipedia

A. Panchenko[1,2], S. Adeykin[2], A. Romanov[2] and P. Romanov[2]

[1] Université catholique de Louvain, Center for Natural Language Processing

[2] Bauman Moscow State Technical University, Information Systems dept.

May 10, 2012

Plan

Semantic Relations

In the context of this work, semantic relations are:

- **synonyms** (equivalence relations):
  $\langle car, SYN, vehicle\rangle, \langle animal, SYN, beast\rangle$
- **hypernyms** (hierarchical relations):
  $\langle car, HYPER, Jeep\ Cherokee\rangle, \langle animal, HYPER, crocodile\rangle$
- **co-hypernyms** (have a common parent):
  $\langle Toyota\ Land\ Cruiser, COHYPER, Jeep\ Cherokee\rangle$

Formally:

- $r = \langle c_i, t, c_j\rangle$ – a **semantic relation**
- $c_i, c_j \in C$ – **concepts**, such as "*radio*" or "*receiver operating characteristic*"
- $t \in T$ – **relation type**, such as **synonym** or **hypernym**
- $R \subseteq C \times T \times C$ – a set of **semantic relations**
- $R \subseteq C \times C$ – a set of **untyped semantic relations**

Semantic Relations Can Be Found In ...

Thesauri: a graph $G = (C, R)$

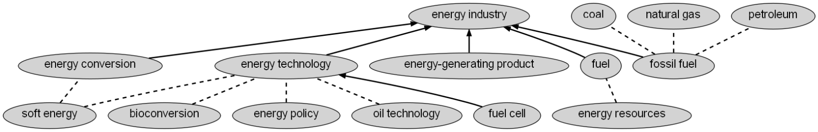


Figure: A part of information-retrieval thesaurus EuroVoc.

$T = \{NT, RT, USE\}$
$R =$

- ⟨energy-generating product, NT, energy industry⟩
- ⟨energy technology, NT, energy industry⟩
- ⟨petrolium, RT, fossil fuel⟩

Other semantic resources: ontologies, semantic networks, synonymy rings, subject headings, etc.

Applications

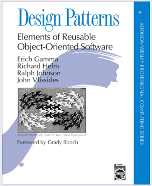
Semantic relations are successfully used in **NLP/IR applications**:

- Query Expansion and Suggestion (Hsu et al., 2006)
- Word Sense Disambiguation (Patwardhan et al., 2003)
- QA Systems (Sun et al., 2005)
- Text Categorization Systems (Tikk et al, 2003)

Problem

- Existing resources are often **not suitable** for a given...
    - NLP/IR application
    - Domain
    - Language

Example: a book store



"Design Patterns: Elements of Reusable Object-Oriented Software" $\Leftrightarrow$ *"Gang of Four Book"* $\Leftrightarrow$ *GOF*

- How to show in the results the book for the query *"GOF"*?

Problem

- **Manual** construction of semantic resources:
  - (+) Precise result
  - (–) Very expensive and time-consuming
  - (–) Inapplicable in most of the cases
- **Existing** relation extraction methods:
  - (+) No manual labor
  - (–) Do not precise enough
- $\Longrightarrow$ Development of **new** relation extraction methods.

State of the Art

Existing relation extraction **methods** are based on...

- **lexico-syntactic patterns** (Snow, 2004)
  - (+) high precision
  - (−) low recall
  - (−) manually crafted extraction rules
  - (−) rules are language-dependent
- **distributional analysis** (Grefenstette, 1994; Curran and Moens, 2002)
  - (+) no manual labor
  - (−) low precision

Semantic similarity measures based on Wikipedia (Strube and Ponzetto, 2006; Gabrilovich and Markovitch, 2007; Zesch, Muller, and Gurevych, 2008):

- (+) high precision and recall
- (+) cover the key domains and languages
- (+) constantly updated by users
- (−) were not used for relation extraction

Contributions

- A semantic relation extraction **method** based on:
  - Wikipedia abstracts
  - two measures of semantic similarity – Cos, Overlap
  - two algorithms – KNN, MKNN
- A relation extraction **system** Serelex:
  - Open Source license LGPLv3
  - `https://github.com/AlexanderPanchenko/Serelex`

Outline

Semantic Relation Extraction Method

Input:

- C – a set of words
- D – a set of definitions for C
- k – number of nearest neighbors

Output:

- $R \subset C \times C$ – a set of semantically related words

Algorithms

- KNN
- MKNN (Mutual KNN)

Similarity Measures

- Cos – Cosine between definition vectors
- Overlap – Number of common lemmas in definitions

Data and Preprocessing

Data:

- a set of definitions D of a set of English words C
- a **definition** $d \in D$ is a text of the first paragraph of a Wikipedia article with title $c \in C$
- source of the articles – `DBPedia.org`

Preprocessing:

- POS tagging and lemmatization (TreeTagger)
- Removing stopwords
- 327.167 definitions (237 MB)
- 775 definitions for a test (824 KB)

```
axiom; in#IN#in traditional#JJ#traditional logic#NN#logic ,#,#, an#DT#an
axiom#NN#axiom or#CC#or postulate#NN#postulate is#VBZ#be a#DT#a
...is#VBZ#be not#RB#not proved#VVN#prove ...
```

Semantic Similarity Measures

Calculate semantic similarity of a pair of words $c_i, c_j \in C$ as similarity of their definitions $d_i, d_j \in D$

Overlap – Number of common lemmas in definitions

- $similarity(c_i, c_j) = \frac{2|(d_i \cap d_j|}{|d_i| + |d_j|}$
- $|d_j|$ – number of words in definition $d_j \in D$

Cos – Cosine between definition vectors

- $similarity(c_i, c_j) = \frac{\mathbf{f}_i \cdot \mathbf{f}_j}{||\mathbf{f}_i|| \cdot ||\mathbf{f}_j||}$
- f_{ik} – frequency of lemma c_k in definition d_i
- $\mathbf{f}_i = (f_{i1}, \ldots, f_{in})$

KNN Algorithm

```
R = ComponentAnalysis(C, D, k, isMutualKNN)
Input: C — concepts, D - definitions of concepts, k - number of nearest
neighbors, isMutualKnn - if true then MKNN, else KNN
Output: R - set of semantic relations <c_i,c_j> in C X C
// Calculation of pairwise similarities between words all concepts C
Rmatrix = void
for i=0; i<count(C); i++ {
      for j=i; j<count(C); j++ {
            // Calculation of semantic similarity of two concepts
            s_ij = similarity(D(i), D(j))
            // Saving most similar concepts
            if( count(Rmatrix(C(i))) < k || s_ij > min(Rmatrix(C(i)) ){
                  Rmatrix(C(i)).addOrReplaceMin(C(j))
            }
      }
}
```

MKNN Algorithm

```
R = ComponentAnalysis(C, D, k, isMutualKNN)
Input: C – concepts, D - definitions of concepts, k - number of nearest
neighbors, isMutualKnn - if true then MKNN, else KNN
Output: R - set of semantic relations <c_i,c_j> in C X C
// Calculation of pairwise similarities between words all concepts C
Rmatrix = void
for i=0; i<count(C); i++ {
        for j=i; j<count(C); j++ {
                // Calculation of semantic similarity of two concepts
                s_ij = similarity(D(i), D(j))
                // Saving most similar concepts
                if( count(Rmatrix(C(i))) < k || s_ij > min(Rmatrix(C(i)) ){
                        Rmatrix(C(i)).addOrReplaceMin(C(j))
                }
        }
}
// Calculation of semantic relations
R = void
foreach c_i in Rmatrix {
        foreach c_j in Rmatrix(c_i) {
                if(!isMutualKNN || Rmatrix(c_j) contains c_i){
                        R.add(<c_i, c_j>)
                }
        }
}
return R
```

- Time complexity is $O(|C|^2)$
- Space complexity is $O(k|C|)$

Example of KNN and MKNN

computer	apple	fruit	mango	
-	0.7	0.0	0.0	computer
0.7	-	1.0	0.8	apple
0.0	1.0	-	0.9	fruit
0.0	0.8	0.9	-	mango

Nearest neighbors $(k = 2)$:

- computer: apple
- apple: fruit, mango, computer
- fruit: apple, mango
- mango: fruit, apple

KNN:

$\langle apple, computer \rangle, \langle apple, fruit \rangle, \langle apple, mango \rangle, \langle fruit, mango \rangle$

MKNN:

$\cancel{\langle \text{apple, computer} \rangle}, \langle apple, fruit \rangle, \langle apple, mango \rangle, \langle fruit, mango \rangle$

Relation Extraction System Serelex

- http://github.com/AlexanderPanchenko/Serelex
- Language: C++
- Libraries: STL, boost
- Cross-platform: Windows/Linux, 32/64-bit
- Interface: console
- License: LGPLv3

Empirical estimation of performance:

- 755 definitions – 3 seconds
- 41.729 definitions – 14 min (Overlap,MKNN,$k = 5$), 120min (Cos, MKNN, $k = 5$)
- 327.168 definitions – 3 days 3 hours 47 minutes
- Server configuration: Linux 2.6.32-cs-kernel with Intel® Xeon® CPU E5606@2.13GHz

Extracted Relations

An example of extracted relations...

- between a set of 775 concepts
- with MKNN, k=2
- with Overlap measure

$$R = \{$$
$$\langle acacia, pine \rangle, \langle aircraft, rocket \rangle,$$
$$\langle alcohol, carbohydrate \rangle, \langle alligator, coconut \rangle,$$
$$\langle altar, sacristy \rangle, \langle object, library \rangle,$$
$$\langle object, pattern \rangle, \langle office, crew \rangle,$$
$$\langle onion, garlic \rangle, \langle saxophone, violin \rangle,$$
$$\langle saxophone, clarinet \rangle, \langle tongue, mouth \rangle,$$
$$\langle watercraft, boat \rangle, \langle watermelon, berry \rangle,$$
$$\langle weapon, warship \rangle, \langle wolf, coyote \rangle,$$
$$\langle wood, paper \rangle, \ldots$$
$$\}$$

Number of Extracted Relations

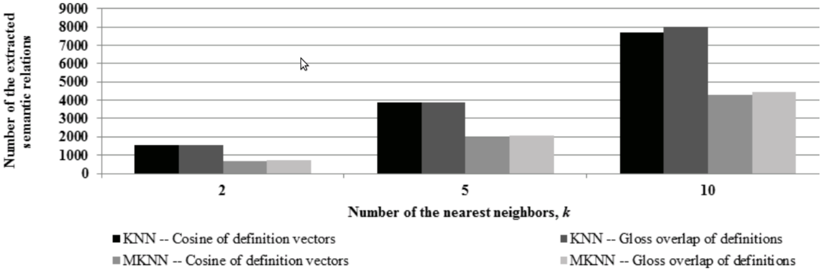


Figure: Dependence of the number of extracted relations $|R|$ on the number of nearest neighbors k.

Precision of Relation Extraction

Algorithm	Similarity Measure	Extracted	Correct	Precision
KNN	Cos	1548	1167	0.754
KNN	Overlap	1546	1176	0.761
MKNN	Cos	652	499	0.763
MKNN	Overlap	724	603	**0.833**

Table: Precision of relation extraction for 775 concepts with the KNN and MKNN (k=2).

Alternative Relation Extraction System

- SEXTANT (Grefensette, 1992) – open-vocabulary extraction, precision $\approx 75\%$
- PMI-IR (Turney, 2001) – TOEFL synonymy test (1 of 4), precision $\approx 74\%$
- WikiRelate! (Strube and Ponzetto, 2006) – the most similar system
  - does not extract relations
  - correlation around 0.59 with human judgements
  - different similarity measures
  - source codes are not available
  - uses Wikipedia category lattice
- Explicit Semantic Analysis (Gabrilovich and Markovich, 2007)
- Wikipedia/Wiktionary (Zesch, Muller, and Gurevych, 2008)
- PF-IBF (Nakayama et al., 2007)

Conclusion:

- We proposed and analyzed a **method** for semantic relation extraction from texts of Wikipedia with algorithms KNN and MKNN and two semantic similarity measures.
- The **best results** (precision of 83%) were obtained with the method based on MKNN and Overlap measure.
- We presented an **open source system**, which efficiently implements the proposed method.
- **Characteristics** of the proposed method:
  - computationally efficient
  - can be used to extract relations between 3.8 million of concepts in English Wikipedia
  - the only language-dependent resources are stoplist, part-of-speech tagger, and lemmatizer

Future Work:

- **Using the developed method** to extract relations between Russian, French, and German words.
- **Improving the precision** of the extraction by clustering of the obtained semantic relation graph.