A simple and robust algorithm for extracting terminology

Luís Sarmento

Linguateca

www.linguateca.pt / las@letras.up.pt

Luís Sarmento @ META Simposium - For a Proactive Translatology (Université de Montréal, Québec, Canadá, 7-9 April 2005)

Motivation

- Exponential growth of multi-lingual written information, especially in *Technical Domains*
- Need for *Terminology Resources*
  - Information Retrieval
  - Technical Writing
  - Translation
- But *Technical Knowledge* is constantly evolving and so is its *Terminology*.

Motivation

- Terminology resources
  - Short life-cycles, constant need for update
  - Expensive to produce and maintain
  - Need to keep up with emergent domains
- What we need:
  - *Rapid Terminology Resources Development*
  - Easy-to-use terminology extraction software
  - Computing-aware terminology specialists
  - "Build & Go" terminology resources

"Build & Go"

1. Obtain a specific domain corpus
   - "Do-it-yourself" / web search / specialist
2. Extract terminology (semi-automatically)
3. Validate results using corpora
   - Consult specialist, if possible...
4. Use terminology for IR, Translation, etc...
5. IF/ WHEN more terminology resources are necessary, go back to Step 1

Approaches to Extraction

- Statistical
  - Rationale: find word sequences that differ from "common-language"
  - Simple and portable but requires "common-language" corpus for comparison: *difficult*!
- Syntactic
  - Rationale: Find word sequences that have a specific POS pattern
  - Good precision and coverage, but complex and requires *"heavy" pre-processing*. Difficult to port to other languages.

Approaches to Extraction

- Morphological:
  - Rationale: find words that look like terms based on roots or suffixes.
  - Good precision for *some* domains but requires *heavy linguistic background*.
- Hybrid:
  - Rationale: try to combine any of the previous approaches and use other heuristics
  - May lead to good results but usually lacks *theoretical foundation*

But what do we need?

- The situation:
  - Large amounts of text available on-line
  - High **redundancy** – should be explored!
  - Multi-lingual corpora (comparable, not parallel)
- What is required:
  - **Fast** algorithms
    - Large amounts of text to be processed
  - High **Precision** algorithms
    - High coverage comes from redundancy
  - "**Multi-lingual**" algorithms
    - Easy to port to other languages: spare the programmers! ☺

And after all...

- We still need human intervention
  - at least domain specialists for validation
- "Fully automated" methods are never fully automated
- Human intervention in resource building is advisable and feasible
- But it cannot be too difficult/ boring
  - **Precision** is more important than coverage!

An algorithm to suit our needs

- The Corpógrafo is a complete web-based terminology extraction environment.
- We assume user intervention:
  - the "need for speed"
  - good precision
  - easy to understand!
- Need to perform reasonably well in many languages.
- We cannot afford POS tagging:
  - too complex, too slow, too expensive, too dependent

A very simple approach

- Collect N-grams from the corpus
- Ask user to check if they are terms.
- Advantages:
  - No linguistic resources needed
  - Fast and portable
- Disadvantages
  - Too noisy
  - Users obviously find it inappropriate

Example from the *Neurodemo* corpus

- Specific domain corpus (neurology)
  - Texts taken from the web (pdf, word, html)
  - 6 languages (PT,EN,FR,ES,IT,DE)
  - English section: 29192 tks.

Too noisy: only 2 terms in 15 n-grams!

N-gram	#	F (%)
of the	332	1.137
in the	243	0.832
to the	121	0.414
the cell	118	0.404
from the	71	0.243
the brain	65	0.222
nervous system	65	0.222
on the	59	0.202
and the	52	0.178
of a	51	0.174
the neuron	48	0.164
the axon	46	0.157
cell body	46	0.157
is the	42	0.143
by the	40	0.137
in a	40	0.137

Improving results

- Results can be easily improved.
- We could start by describing what a term is and trying to find n-grams that respect that description
  - Ex: a term must end with "*ology", etc..
- However, it is very difficult to say what a term might be for every domain.
- But it is much easier to say what a term is NOT!
- And it is also much more "stable"...

Excluding some N-grams

- Define 3 n-gram exclusion lists:
  - List of **non-starters**: tokens that cannot start terms
  - List of **non-enders**: tokens that cannot end terms
  - List of **non-includables**: tokens that cannot be part of the term in any position
- Find N-grams conforming to these restrictions
- Let redundancy do the rest
- Similar approach in Merkel & Andersson, 2000

But what are these lists?

- Most of the elements in these lists are:
  - prepositions
  - pronouns
  - punctuation
  - certain very frequent words
- Can be easily compiled through trial-and-error strategy
- Very stable among different knowledge domains
  - but may be easily changed, if necessary
- Easy to adapt to other non-agglutinative languages

Using simple restrictions (top 20)

N-gram (3110 found)	#	F (%)
nervous system	65	0.222
cell body	46	0.157
electrical activity	39	0.133
nerve cells	37	0.126
spinal cord	34	0.116
action potential	32	0.109
glial cells	29	0.099
synaptic cleft	20	0.068
plasma membrane	16	0.054
central nervous	16	0.054
action potentials	14	0.047
schwann cells	14	0.047
membrane proteins	13	0.044
nerve fibers	13	0.044
endoplasmic reticulum	13	0.044
nervous systems	12	0.041
developing circuits	12	0.041
amino acids	12	0.041
myelin sheath	12	0.041
nerve cell	12	0.041

N-gram (2208 found)	#	F (%)
central nervous system	16	0.054
peripheral nervous system	8	0.027
integral membrane proteins	8	0.027
nuclear pore complexes	5	0.017
name of glial	5	0.017
signaling between nerve	5	0.017
pattern of activity	5	0.017
nodes of ranvier	4	0.013
synthesis of proteins	4	0.013
induction of ltp/ltd	4	0.013
evoked nt secretion	3	0.010
can be divided	3	0.010
primary visual cortex	3	0.010
nmda receptor activation	3	0.010
– the messengers	3	0.010
action potential will	3	0.010
rate of transmission	3	0.010
primary cell walls	3	0.010
complexes of integral	3	0.010
refinement of neural	3	0.010

Using simple restrictions

- Precision increases
  - user intervention is easier: + 100 terms/ hr
- Still some problems:
  1. Some very frequent words that bring false candidates (ex: “can”)
  2. Plural/ Singular divide term occurrences
  3. Encapsulated terms still difficult to separate:
     - “nervous system” and “central nervous system”
- But still possible to improve results

Additional restrictions

- Goal: improve precision and solve some of the previous problems (1 & 2)
- If we "force" N-Grams to be NP, then, singularization is trivial in **several** languages
- Impose 1 additional restriction: N-Grams must be preceded by certain words
  - Ex: "a", "the", "one", "as", etc
  - Similar restrictions in "PT", "ES","FR", "IT"
  - Simple enough to be easily implemented
- Again: let redundancy do the rest!

Additional restrictions

N-gram (1304 found)	#	F (%)
cell body	46	0.157
nervous system	35	0.119
spinal cord	31	0.106
action potential	30	0.102
nerve cell	26	0.089
electrical activity	22	0.075
synaptic cleft	20	0.068
plasma membrane	16	0.054
glial cell	16	0.054
central nervous	15	0.051
myelin sheath	12	0.041
developing circuit	11	0.037
neural circuit	10	0.034
peripheral nervous	9	0.030
protein synthesis	9	0.030
endoplasmic reticulum	8	0.027
human brain	8	0.027
respiratory chain	7	0.023
schwann cell	7	0.023
nmda receptor	7	0.023

N-gram (943 found)	#	F (%)
central nervous system	15	0.051
peripheral nervous system	9	0.030
node of ranvier	6	0.020
integral membrane protein	6	0.020
synthesis of proteins	4	0.013
nuclear pore complex	4	0.013
induction of ltp/ltd	4	0.013
primary visual cortex	3	0.010
energy of atp	3	0.010
development of neural	3	0.010
rate of transmission	3	0.010
activation of nmda	2	0.006
evoked nt secretion	2	0.006
refinement of neural	2	0.006
xenopus retinotectal system	2	0.006
induction of ltp	2	0.006
activity-induced synaptic modification	2	0.006
cytochrome b gene	2	0.006
activity-dependent synaptic modification	2	0.006
developing neural circuit	2	0.006

How much improvement?

- Further increases in precision
  - Easier for user validation!
- Plural/ Singular division solved
- Easy to implement
- "Multi-lingual"
- Even faster: no long N-grams list to sort!
  - 29K tokens in less 2s on a standard Intel P4 machine
- But we haven't yet solved the term encapsulation problem. Possible solution:
  - use "bigger first" N-gram search strategy

Conclusion

- We have presented a simple algorithm:
  - Easy to implement by developers
  - Easy to understand by users
  - Fast enough to process large corpora (> 1M)
  - Filters can be adapted to specific domains
  - Can be easily ported to many languages
  - Improvable by simple trial-and-error methods
  - But it still needs some work to deal with encapsulated terms (in development)