Lexical chains for summarization

a summary of Silber & McCoy’s work
by Keith Trnka
Outline

• Lexical chains
• Silber & McCoy’s algorithm for computing lexical chains
• Silber & McCoy’s evaluation of lexical chains for text summarization
Lexical chains

• Intuitively, lexical chains represent a recurring noun concept within a document that may be referred to by different nouns*

• Thus, it deals with the problem of word sense disambiguation (WSD) and other problems
  – WSD - given a word and multiple meanings for the word, choose the correct meaning (sense)

* to my knowledge this has not been proven; it is only an intuition
Lexical chains

- Manually annotated example (from Barzilay & Elhadad ‘97 Appendices A, B):

  When Microsoft Senior Vice President Steve Ballmer first heard his company was planning to make a huge investment in an Internet service offering movie reviews and local entertainment information in major cities across the nation, he went to Chairman Bill Gates with his concerns. After all, Ballmer has billions of dollars of his own money in Microsoft stock, and entertainment isn't exactly the company's strong point. But Gates dismissed such reservations. Microsoft's competitive advantage, he responded, was its expertise in Bayesian networks. Asked recently when computers would finally begin to understand human speech, Gates began discussing the critical role of "Bayesian" systems. Ask any other software executive about anything Bayesian and you're liable to get a blank stare.

  Is Gates onto something? Is this alien-sounding technology Microsoft's new secret weapon? Bayesian networks are complex diagrams that organize the body of knowledge in any given area by mapping out cause-and-effect relationships among key variables and encoding them with numbers that represent the extent to which one variable is likely to affect another. Programmed into computers, these systems can automatically generate optimal predictions or decisions even when key pieces of information are missing.
Lexical chains

• How to compute them?
• An *interpretation* is a set of lexical chains in which each noun occurrence is mapped to exactly one chain
• Could generate all possible interpretations and find the “best” one
  – the one with the lengthiest lexical chains
Silber & McCoy’s algorithm

• Problem: all possible interpretations is exponential, takes too long
• Solution: don’t compute them explicitly
Silber & McCoy’s algorithm

• Each lexical chain is associated with a concept and each concept can be mapped to a core meaning (word sense)

• Implicitly build all possible chains for each sense in WordNet
  – note: senses in WordNet are represented as synonym sets (*synsets*)
  – called *meta-chains*
  – a single meta-chain represents all possible lexical chains for that core meaning
Silber & McCoy’s algorithm

- Given a meta-chain identified by a word sense $S$ and the next noun in the input, for each sense $NS$ of the noun, add it if:
  - they have the same sense
  - they are synonyms
  - $S$ is a kind of $NS$ ($NS$ is a hypernym of $S$)
  - $NS$ is a kind of $S$ ($NS$ is a hyponym of $S$)
- these last two are computed from WordNet; it has a tree of “is a” relationships
Silber & McCoy’s algorithm

- How do we identify nouns in the input?
  - use a part of speech (POS) tagger before computing lexical chains
Silber & McCoy’s algorithm

• How do we find the best interpretation?
  – for each noun occurrence in the document, find which chain it “fits” best in
    • use the notion of giving a score to each chain by giving a score to each link in the chain and summing them
    • the score of each noun occurrence in a chain is the score of the strongest backward-looking relationship to other words in the chain
Silber & McCoy’s algorithm

– the score of each noun occurrence in a chain is the score of the strongest backward-looking relationship to other words in the chain
  • the first one in the chain is given maximal score
  • other scores are computed based on:
    – relationship type
      » same word, synonym, hypernym, sibling in the hypernym/hyponym tree
    – distance
      » 1 sentence, 2 sentences, same paragraph, etc.
Silber & McCoy’s algorithm

- Given the constant nature of WordNet and implementation details, computation of the best interpretation is linear in the number of nouns in the source document.
Silber & McCoy’s algorithm

• So what do we do with our interpretation?
  – compute the most important chains: strong chains (Barzilay & Elhadad ‘97)
    • chains with a score at least two standard deviations above the mean score

• And what do we do with our strong chains?
  – generate a summary!
Lexical chains (cont’d)

- Barzilay & Elhadad:
  - One method is to find the first sentence referring to each strong chain and include them (in order) in the summary
  - Another is to find the first sentence using only the *representative nouns* in a lexical chain
    - representative nouns are noun senses in a lexical chain that best identify the chain
Silber & McCoy’s evaluation

• How to evaluate the lexical chain algorithm’s potential for summarization?
  – We could generate summaries and have humans evaluate them, but that would mingle evaluation of our summary algorithm and lexical chains algorithm.
Silber & McCoy’s evaluation

• For background, consider the summarization process:

```
Source document → Intermediate representation → Summary
```
Silber & McCoy’s evaluation

• A summary should contain the important points of the document
• Therefore, it will share properties with the document
  – the cosine distance between term vectors might be low, for instance
Silber & McCoy’s evaluation

- Idea: convert both to the intermediate representation and compare the representations

Source document

Summary

Intermediate representation
To evaluate lexical chains, we compute two similarity measures:

- percentage of noun senses in the summary that are in the strong lexical chains of the document
- percentage lexical chains of the document that have a noun sense in the summary

Note: noun senses in the summary are determined by the lexical chains algorithm
Silber & McCoy’s evaluation

• Results
  – 24 documents with summaries were used
    • varied document type (paper, book chapter) and discipline (computer science, anthropology, etc)
  – percentage of noun senses in the summary that are in the strong lexical chains of the document
    • about 80%
  – percentage lexical chains of the document that have a noun sense in the summary
    • about 80%
Silber & McCoy’s evaluation

• Results (cont’d)
  – two documents in particular scored very poorly
    • without them the percentages were both about 85%
  – Why?
    • I’ll hopefully figure it out later, but here are some
deficiencies that I would like to look into:
      – pronoun resolution
      – non-noun concepts (or at least words that aren’t tagged
        as nouns)
      – noun compounds (for example, Monte Carlo algorithms)
References / FYI

  – Obtained from CiteSeer (http://citeseer.nj.nec.com/cs)

  – Original lexical chains work

  – Obtained via McCoy’s website (http://www.cis.udel.edu/~mccoy/)