Lexical Chains
(for Summmmarization)

Michael Strube
Heidelberg Institute for Theoretical Studies gGmbH
Heidelberg, Germany
Lexical Cohesion

A text or discourse is not just a set of sentences, each on some random topic. Rather, the sentences and phrases of any sensible text will each tend to be about the same things – that is, the text will have a quality of unity. This is the property of cohesion. . . .

Lexical cohesion is the cohesion that arises from semantic relationships between words. All that is required is that there be some recognizable relation between the words.

Source: Morris & Hirst (1991)
Lexical Chains

- lexical cohesion occurs not only between pairs of words (in pairs of sentences)
- lexical cohesion occurs also over a sequences of semantically related words (in sequences of sentences) – lexical chains
Lexical Chains

Why?

• to provide context for word sense disambiguation
  • \{hair, comb, curl, wave\} – the word sense of \textit{wave} is narrowed down to \textit{hair wave}
  • \{gin, alcohol, sober, \textit{drink}\} – the word sense of \textit{drink} is narrowed down to \textit{alcoholic drink}
Lexical Chains

Why?

• to provide context for word sense disambiguation
  • {hair, comb, curl, wave} – the word sense of wave is narrowed down to hair wave
  • {gin, alcohol, sober, drink} – the word sense of drink is narrowed down to alcoholic drink

• to provide information for determining coherence and discourse structure – when a new lexical chain begins there is a high probability that a new discourse segment begins also
Lexical Chains

Why?

• to provide context for word sense disambiguation
  • \{hair, comb, curl, wave\} – the word sense of \textit{wave} is narrowed down to \textit{hair wave}
  • \{gin, alcohol, sober, \textit{drink}\} – the word sense of \textit{drink} is narrowed down to \textit{alcoholic drink}

• to provide information for determining coherence and discourse structure – when a new lexical chain begins there is a high probability that a new discourse segment begins also

• to provide information about important entities in a document and which can be used for automatic summarization
Lexical Chains

require *lexical knowledge*

e.g.:

- thesauri (Roget’s thesaurus (4th ed., 1977) – this is a book, but an electronic version is commercially available)
  - a *dictionary* explains the meaning of words
  - a *thesaurus* aids in finding words that best express an idea or meaning
- lexical databases (WordNet, GermaNet)
- taxonomies, ontologies (e.g. ontologies derived from Wikipedia)
- ...
Roget’s Thesaurus

Class 1...

;

Class 4: Matter

I  ...

;

III Organic Matter

A  ...

;

B Vitality

;

407 Life

1. NOUNS life, living, vitality, being alive, having life, animation, animate existence; liveliness, animal spirits, vivacity, spriteliness; long life, longevity; viability; lifetime 110.5; immortality 112.3; birth 167; existence 1; bio-, organ-, -biosis.

;

2.  ...

;

408 Death  ...

• 1042 basic categories which group words by idea
• eight major semantic abstract classes on top
• related words are physically close
Determining Discourse Structure

(Morris & Hirst, 1991)

• can be applied to any domain
• computationally feasible

parameters:

• thesaural relations
• transitivity of word relations
• distance (in sentences) between words in a chain
Forming Lexical Chains

- candidate words:
  - remove closed class words
  - remove high-frequency words (such as *good, do, taking, . . .*)
- types of relations:
  1. words have common category (e.g. *apartment* and *residentialness* share category 189)
  2. a word’s category has a pointer to the other word’s category (*car* has category 273 which points to category 276, the category of *driving*)
  3. a word is either a label in the other word’s index entry, or is in a category of the other word, e.g, *blind* has category 442 in its index entry, which contains the word *see*
  4. two words are in the same group, and hence are semantically related, e.g., *blind* has category 442 *blindness*, in its index entry and *see* has category 441, *vision*, in its index entry
  5. the two words have categories in their index entries that both point to a common category, e.g., *brutal* has category 851, which in turn has a pointer to category 830; *terrified* has category 860 that likewise has a pointer to category 830
Forming Lexical Chains

- **word 1 index**
  - label 1: 521
  - label 2: 589
  - label 3: 626

- **word 2 index**
  - label 1: 860
  - label 2: 521

- **thesaurus category 521**

- **word 1 index**
  - label 1: 521
  - label 2: 589
  - label 3: 626

- **thesaurus category 860**

- **word 2 index**
  - label 1: 300
  - label 2: 860

- **&860**
Forming Lexical Chains

- candidate words:
  - remove closed class words
  - remove high-frequency words (such as *good, do, taking, ...*)

- types of relations:
  1. words have common category (e.g. *apartment* and *residentialness* share category 189)
  2. a word’s category has a pointer to the other word’s category (car has category 273 which points to category 276, the category of driving)
  3. a word is either a label in the other word’s index entry, or is in a category of the other word, e.g, *blind* has category 442 in its index entry, which contains the word *see*
  4. two words are in the same group, and hence are semantically related, e.g., *blind* has category 442 *blindness*, in its index entry and *see* has category 441, *vision*, in its index entry
  5. the two words have categories in their index entries that both point to a common category, e.g., *brutal* has category 851, which in turn has a pointer to category 830; *terrified* has category 860 that likewise has a pointer to category 830
Forming Lexical Chains

(b)
Forming Lexical Chains

- candidate words:
  - remove closed class words
  - remove high-frequency words (such as *good, do, taking, ...*)

- types of relations:
  1. words have common category (e.g. *apartment* and *residentialness* share category 189)
  2. a word’s category has a pointer to the other word’s category (e.g. *car* has category 273 which points to category 276, the category of *driving*)
  3. a word is either a label in the other word’s index entry, or is in a category of the other word, e.g. *blind* has category 442 in its index entry, which contains the word *see*
  4. two words are in the same group, and hence are semantically related, e.g., *blind* has category 442 *blindness*, in its index entry and *see* has category 441, *vision*, in its index entry
  5. the two words have categories in their index entries that both point to a common category, e.g., *brutal* has category 851, which in turn has a pointer to category 830; *terrified* has category 860 that likewise has a pointer to category 830
Forming Lexical Chains

Word 1 Index
- Label 1: 300
- Label 2: 521
- Label 3: 621

Word 2 Index
- Label 1: 23
- Label 2: 600

Thesaurus Category 521

Thesaurus Category 457

Thesaurus Category 23

&457
Forming Lexical Chains

- transitivity:
  - limits on transitivity – if \( a \) is related to \( b \), and \( b \) to \( c \), then \( a \) is also related to \( c \)
  - to avoid chains like: \{cow, sheep, wool, scarf, boots, hat, snow\}, only one transitive link is allowed
Forming Lexical Chains

• how many sentences can separate two words in a chain:
  • at most three sentences distance
  • four and more sentences distance indicates return to a previous chain

• chain strength (helps to identify whether it is possible to return to a previous chain):
  1. reiteration: the more repetitions, the stronger the chain
  2. density: the denser the chain, the stronger it is
  3. length: the longer the chain, the stronger it is
Computing Lexical Chains

keep for each word in a chain:

- word id
- sentence id
- complete chain created up to that point

lexical relationship in a chain is represented as \((u, v)^y_x\):

- \(u\): current word id
- \(v\): id of related word
- \(x\): transitive distance – 0, 1
- \(y\): identifies thesaural relationship

<table>
<thead>
<tr>
<th>Word</th>
<th>Sentence</th>
<th>Lexical Chain</th>
</tr>
</thead>
<tbody>
<tr>
<td>evade</td>
<td>15</td>
<td>((2, 1)_0^2)</td>
</tr>
<tr>
<td>feigning</td>
<td>15</td>
<td>((3, 1)_1^0 (3, 2)_1^{T1})</td>
</tr>
<tr>
<td>escaped</td>
<td>16</td>
<td></td>
</tr>
</tbody>
</table>
Algorithm Outline

1. select a set of candidate words
2. for each candidate word, find an appropriate chain relying on a relatedness criterion among members of the chains
3. if it is found, insert the word in the chain and update it accordingly
Lexical Chains and Text Structure
Lexical Chains and Text Structure

(Morris & Hirst, 1991)

• correspondence between chain boundaries and text segment boundaries
  • chain strength
  • chain length
• returns to existing chains and their correspondence to returns to previous segments
• manual application of algorithm
• results:
  • lexical chains provide a good clue for determining text structure
  • lexical chains on their own are not sufficient for determining text structure
  • chain returns are important and occur frequently
• see also (Galley et al., 2003)
Using Lexical Chains for Text Summarization
Using Lexical Chains for Text Summarization

(Barzilay & Elhadad, 1997), (Barzilay & Elhadad, 1999) contributions:

- new algorithm
- WordNet, POS tagger, NP chunker
- text segmentation algorithm

Text summarization:

1. segment text
2. construct lexical chains
3. determine strong chains
4. extract significant sentences
Automatic Sumarization

- *extractive summarization* extracts important sentences from a document and concatenates them to produce a summary.
- Earlier work relied on:
  1. Word frequency
  2. Cue phrases (*in conclusion*)
  3. Location of sentences
  4. Length of sentences
- Problems: how many sentences to include? Which notion of importance? What to do with pronouns? Location relies pretty much on domain. . . .
Lexical Chains

problem with earlier work (Morris & Hirst, 1991):

- algorithm did not distinguish between different senses of a word

hence

- lexical chain computing can be interpreted as word sense disambiguation
- greedy disambiguation (Hirst & St-Onge, 1998) apparently does not work well
- main idea here: all words in lexical chains consider all possible word senses and then choose the best global interpretation
Relevant Wordnet Relations

Two noun instances are identical, and are used in the same sense. (The house on the hill is large. The house is made of wood.)

Two noun instances are used in the same sense (i.e., are synonyms). (The car is fast. My automobile is faster.)

The senses of two noun instances have a hypernym/hyponym relation between them. (John owns a car. It is a Toyota.)

The senses of two noun instances are siblings in the hypernym/hyponym tree. (The truck is fast. The car is faster.)

Source: Silber & McCoy (2002)
Mr. Kenny is the person that invented an anesthetic machine which uses micro-computers to control the rate at which an anesthetic is pumped into the blood. Such machines are nothing new. But his device uses two micro-computers to achieve much closer monitoring of the pump feeding the anesthetic into the patient.
Mr. Kenny is the person that invented an anesthetic machine which uses micro-computers to control the rate at which an anesthetic is pumped into the blood. Such machines are nothing new. But his device uses two micro-computers to achieve much closer monitoring of the pump feeding the anesthetic into the patient.

Figure 1: Step 1, Interpretations 1 and 2
Mr. Kenny is the person that invented an anesthetic machine which uses micro-computers to control the rate at which an anesthetic is pumped into the blood. Such machines are nothing new. But his device uses two micro-computers to achieve much closer monitoring of the pump feeding the anesthetic into the patient.

Figure 2: Step 2, Interpretation 1

Figure 3: Step 2, Interpretation 2
Illustration

Figure 2: Step 2, Interpretation 1

Figure 3: Step 2, Interpretation 2

Figure 4: Step 2, Interpretation 3

Figure 5: Step 2, Interpretation 4
Mr. Kenny is the person that invented an anesthetic machine which uses micro-computers to control the rate at which an anesthetic is pumped into the blood. Such machines are nothing new. But his device uses two micro-computers to achieve much closer monitoring of the pump feeding the anesthetic into the patient.
Algorithm Details

- algorithm chooses the *strongest* interpretation eventually
  - strong: sum of the chain scores
  - chain scores: number and weight of the relations between chain members
    - reiteration and synonym: 10; antonym: 7; hypernym: 4
- algorithm computes *all* possible interpretations
- if there are too many interpretations, weak (i.e. the least strongest) interpretations are pruned
- algorithm divides document into discourse segments (Hearst, 1997) and extends chains over a segment boundary only if both segments have a common word with the same word sense
Summarization Using Lexical Chains

hypotheses:

• the most important topic in a document plays an important role in its summary
• strong lexical chains help to identify the most important topic
• advantage: words get mapped to *concepts* (chains) thus overcoming artefacts of concepts with many different realizations
Strength of a Chain

- length: number of mentions in a chain
- homogeneity index:
  1 - the number of distinct occurrences divided by the length

\[
Score(\text{Chain}) = \text{Length} \times \text{HomogeneityIndex}
\]

- strength criterion:
  \[
  Score(\text{Chain}) > \text{Average}(\text{Scores}) + 2 \times \text{StandardDeviation}(\text{Scores})
  \]
Extracting Significant Sentences

extract one sentence for each chain from the text based on the chain distribution:

• heuristic 1: for each chain choose the sentence which starts the chain
  (selects sometimes not the strongest representative sentences for a chain)
• heuristic 2: for each chain choose the sentence containing the most representative chain member
• heuristic 3: identify the (successive) text segment(s) where the chain represents the central topic
  number of chain mentions in segment(s) divided by number of all nouns in segment(s)
Extracting Significant Sentences

extract one sentence for each chain from the text based on the chain distribution:

• heuristic 1: for each chain choose the sentence which starts the chain
  (selects sometimes not the strongest representative sentences for a chain)
• heuristic 2: for each chain choose the sentence containing the most representative chain member
• heuristic 3: identify the (successive) text segment(s) where the chain represents the central topic
  number of chain mentions in segment(s) divided by number of all nouns in segment(s)

• heuristic 1 and heuristic 2 produce very similar results
• heuristic 2 gives the best results
• heuristic 3 often selects sentences from the middle of the document
Evaluation

• to evaluate automatic summarization is extremely difficult
• here:
  • 40 documents (news articles, TREC collection, avg. length 30 sentences)
  • for each document 10 summaries constructed by 5 humans (10% and 20% document length in terms of sentences)
  • 10% and 20% summaries generated with Microsoft Word 97 summarizer
  • agreement between human subjects: 96% for 10% summaries, 90% for 20% summaries – relatively high
Evaluation

- construct ideal summaries by taking majority opinion among human subjects

<table>
<thead>
<tr>
<th></th>
<th>Microsoft Prec</th>
<th>Microsoft Recall</th>
<th>Lexical Chain Prec</th>
<th>Lexical Chain Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>10%</td>
<td>33</td>
<td>37</td>
<td>61</td>
<td>67</td>
</tr>
<tr>
<td>20%</td>
<td>32</td>
<td>39</td>
<td>47</td>
<td>64</td>
</tr>
</tbody>
</table>
Limitations

• long sentences are much more likely to be extracted
• extracted sentences contain pronouns
• method does not have the means to change the length of the summary (one sentence per chain selected)
• computational complexity of chain building algorithm: exponential
Efficiently Computing Lexical Chains
Efficiently Computing Lexical Chains

(Silber & McCoy, 2002)
contributions:

• efficient algorithm for computing lexical chains: linear in space and time
• evaluation of the quality of lexical chains
Problem Outline

• (Barzilay & Elhadad, 1999) define an interpretation as a mapping of noun instances to specific senses and then of these senses to specific lexical chains
  • they essentially compute every possible combination between lexical chains and word senses and evaluate the strongest interpretation only at the end of the document
  • this has exponential complexity and requires heuristic pruning

• (Silber & McCoy, 2002) create a structure that implicitly stores every interpretation without actually creating it
  • they then provide a method for finding the best interpretation
  • they recompile WordNet into a binary format to make access to it faster
Chain Computation

- create array of *metachains*: number of noun senses in WordNet plus number of nouns in document (this is the maximum size which possibly would be needed)
- when a noun in document is encountered each sense is placed in every metachain for which it has an identity, synonym, or hypernym relation with that sense
- metachains represent every possible interpretation
- metachains are indexed with WordNet synset ids
# Chain Computation

<table>
<thead>
<tr>
<th>Index</th>
<th>Meaning</th>
<th>Chain</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>person</td>
<td>John</td>
</tr>
<tr>
<td>1</td>
<td>unit</td>
<td>Computer</td>
</tr>
<tr>
<td>2</td>
<td>device</td>
<td>Computer</td>
</tr>
<tr>
<td>3</td>
<td>organization</td>
<td>Machine</td>
</tr>
<tr>
<td>4</td>
<td>unknown</td>
<td>IBM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IBM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IBM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IBM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IBM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IBM</td>
</tr>
</tbody>
</table>

*Note: Assume the sentences “John has a computer. The machine is an IBM.” and that the nouns have the following senses (meanings): John (0), computer (1,2), machine (0,2,3), IBM (1,2,3,4), and that words are put in a chain if they have an identity relation. This table then depicts the metachains after the first step.*

Source: Silber & McCoy (2002)
Finding the Best Interpretation

2nd pass through document:

• for each noun, determine to which metachain it contributes most – based on the type of the relation and distance factors
• delete the noun from all other metachains
• after processing of document, only the interpretation with the highest scoring chains is left
• select best (highest scoring) chains from this interpretation for further processing, e.g. summarization
Finding the Best Interpretation

Step 1  For each noun instance
   For each sense of the noun instance
    Compute all scored metachains

Step 2  For each noun instance
   For each metachain to which the noun belongs
    Keep word instance in the metachain to which it contributes most
    Update the scores of each other metachain
## Scoring System

<table>
<thead>
<tr>
<th></th>
<th>One Sentence</th>
<th>Three Sentences</th>
<th>Same Paragraph</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identical word</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Synonym</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Hypernym</td>
<td>1</td>
<td>.5</td>
<td>.5</td>
<td>.5</td>
</tr>
<tr>
<td>Sibling</td>
<td>1</td>
<td>.3</td>
<td>.2</td>
<td>0</td>
</tr>
</tbody>
</table>

- for scoring chains (Silber & McCoy, 2002) use (Barzilay & Elhadad, 1999)’s method (2 standard deviations above the average)
## Runtime of the Algorithm

<table>
<thead>
<tr>
<th>Value</th>
<th>Worst Case</th>
<th>Average Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_1 = #$ of senses for a given word</td>
<td>30</td>
<td>2</td>
</tr>
<tr>
<td>$C_2 = #$ of parent/child &quot;is a&quot; relations of a word sense</td>
<td>45,147</td>
<td>14</td>
</tr>
<tr>
<td>$C_3 = #$ of nouns in WordNet</td>
<td>94,474</td>
<td>94,474</td>
</tr>
<tr>
<td>$C_4 = #$ of synsets in WordNet</td>
<td>66,025</td>
<td>66,025</td>
</tr>
<tr>
<td>$C_5 = #$ of siblings of a word sense</td>
<td>397</td>
<td>39</td>
</tr>
<tr>
<td>$C_6 = #$ of chains to which a word instance can belong</td>
<td>45,474</td>
<td>55</td>
</tr>
</tbody>
</table>

- collection of WordNet information:
  \[ n \times (\log_2(C_3) + C_1 \times C_2 + C_1 \times C_5) \]
- building the graph:
  \[ n \times C_6 \times 4 \]
- extracting the best information:
  \[ n \times C_6 \times 4 \]
- overall runtime performance:
  \[ n \times (1,548,216 + \log_2(94,474) + 45,474 \times 4) \] (worst case)
  \[ n \times (326 + \log_2(94,474) + 55 \times 4) \] (average case)
Evaluation

- linear runtime makes evaluation on larger documents possible
- evaluation question: are the lexical chains identified the concepts which humans identify as important for summaries?
- documents from different domains (10 scientific articles along with their abstracts, 10 textbook chapters along with their chapter summaries)
- length of documents between 2,200 and 26,500 words
<table>
<thead>
<tr>
<th>Document</th>
<th>Total Number of Strong Chains in Document</th>
<th>Total Number of Noun Instances in Summary</th>
<th>Strong Chains with Corresponding Noun Instances in Summary</th>
<th>Noun Instances with Corresponding Strong Chains in Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS Paper 1</td>
<td>10</td>
<td>22</td>
<td>7 (70.00%)</td>
<td>19 (86.36%)</td>
</tr>
<tr>
<td>CS Paper 2</td>
<td>7</td>
<td>19</td>
<td>6 (71.43%)</td>
<td>17 (89.47%)</td>
</tr>
<tr>
<td>CS Paper 3</td>
<td>5</td>
<td>31</td>
<td>4 (80.00%)</td>
<td>27 (87.19%)</td>
</tr>
<tr>
<td>CS Paper 4</td>
<td>6</td>
<td>25</td>
<td>5 (83.33%)</td>
<td>24 (96.00%)</td>
</tr>
<tr>
<td>CS Paper 5</td>
<td>8</td>
<td>16</td>
<td>6 (75.00%)</td>
<td>12 (75.00%)</td>
</tr>
<tr>
<td>ANTH Paper 1</td>
<td>7</td>
<td>20</td>
<td>7 (100.00%)</td>
<td>17 (85.00%)</td>
</tr>
<tr>
<td>ANTH Paper 2</td>
<td>5</td>
<td>17</td>
<td>4 (80.00%)</td>
<td>13 (76.47%)</td>
</tr>
<tr>
<td>ANTH Paper 3</td>
<td>7</td>
<td>21</td>
<td>6 (28.57%)</td>
<td>7 (33.33%)</td>
</tr>
<tr>
<td>BIO Paper 1</td>
<td>4</td>
<td>19</td>
<td>4 (100.00%)</td>
<td>17 (89.47%)</td>
</tr>
<tr>
<td>BIO Paper 2</td>
<td>5</td>
<td>31</td>
<td>5 (80.00%)</td>
<td>28 (90.32%)</td>
</tr>
<tr>
<td>CS Chapter 1</td>
<td>9</td>
<td>55</td>
<td>8 (88.89%)</td>
<td>49 (89.09%)</td>
</tr>
<tr>
<td>CS Chapter 2</td>
<td>7</td>
<td>49</td>
<td>6 (85.71%)</td>
<td>42 (85.71%)</td>
</tr>
<tr>
<td>CS Chapter 3</td>
<td>11</td>
<td>31</td>
<td>9 (81.82%)</td>
<td>25 (80.65%)</td>
</tr>
<tr>
<td>CS Chapter 4</td>
<td>14</td>
<td>47</td>
<td>5 (35.71%)</td>
<td>21 (44.68%)</td>
</tr>
<tr>
<td>ANTH Chapter 1</td>
<td>5</td>
<td>61</td>
<td>4 (80.00%)</td>
<td>47 (77.05%)</td>
</tr>
<tr>
<td>ANTH Chapter 2</td>
<td>8</td>
<td>74</td>
<td>7 (87.50%)</td>
<td>59 (79.73%)</td>
</tr>
<tr>
<td>ANTH Chapter 3</td>
<td>12</td>
<td>58</td>
<td>11 (91.67%)</td>
<td>48 (82.76%)</td>
</tr>
<tr>
<td>ANTH Chapter 4</td>
<td>13</td>
<td>49</td>
<td>11 (84.62%)</td>
<td>42 (85.71%)</td>
</tr>
<tr>
<td>ANTH Chapter 5</td>
<td>7</td>
<td>68</td>
<td>5 (71.43%)</td>
<td>60 (88.24%)</td>
</tr>
<tr>
<td>ANTH Chapter 6</td>
<td>9</td>
<td>59</td>
<td>8 (88.89%)</td>
<td>48 (81.36%)</td>
</tr>
<tr>
<td>HIST Chapter 1</td>
<td>12</td>
<td>71</td>
<td>10 (83.33%)</td>
<td>67 (94.37%)</td>
</tr>
<tr>
<td>HIST Chapter 2</td>
<td>8</td>
<td>65</td>
<td>7 (87.50%)</td>
<td>55 (84.62%)</td>
</tr>
<tr>
<td>ECON Chapter 1</td>
<td>14</td>
<td>68</td>
<td>12 (85.71%)</td>
<td>63 (92.65%)</td>
</tr>
<tr>
<td>ECON Chapter 2</td>
<td>9</td>
<td>51</td>
<td>7 (77.78%)</td>
<td>33 (64.71%)</td>
</tr>
<tr>
<td>Mean</td>
<td></td>
<td></td>
<td>79.12%</td>
<td>80.83%</td>
</tr>
<tr>
<td>Median</td>
<td></td>
<td></td>
<td>82.58%</td>
<td>85.35%</td>
</tr>
</tbody>
</table>

Note: ANTH—anthropology, BIO—biology, CS—computer science, ECON—economics, HIST—history.
Discussion

• pronouns and proper names cause problems
• WordNet does not have sufficient information on proper names
• coverage of WordNet – system defaults to word frequency if no WordNet entry is found
• lexical chains are based on nouns only; information on verbs (predicates) is disregarded
Further Applications
Further Applications

• spelling correction (Hirst & St-Onge, 1998)
• keyphrase indexing (Medelyan, 2007)
• keyword extraction (Erkan, 2007)
• more segmentation (Stokes et al., 2004)
• web directory construction (Stamou et al., 2005)
• question answering (Moldovan et al., 2002)
• . . . and many more
Homework

• read the paper on which you want to base your project (it would be preferable if this would coincide with your presentation)
• based on your knowledge of NLTK decide whether it is feasible (if you are in doubt, talk to me)
• write a specification (input, output, data structures, preprocessing tools needed, evaluation metric, data needed – I may be able to point you to or to provide you with the required data)
• send specification to me (michael.strube@h-its.org) by May 10, 2012, 1pm
• (later today) link to the slides at http://michael.kimstrube.de/teaching.php
References

Using lexical chains for text summarization.

Text summarization with lexical chains.
In Inderjeet Mani & Mark T. Maybury (Eds.), Advances in Automatic Text Summarization, pp. 111–121. Cambridge, Mass.: MIT Press.

Discourse segmentation of multi-party conversation.

TextTiling: Segmenting text into multi-paragraph subtopic passages.

Hirst, Graeme & David St-Onge (1998).
Lexical chains as representations of context for the detection and correction of malapropisms.

Medelyan, Olena (2007).
Computing lexical chains with graph clustering.

Lexical cohesion computed by thesaural relations as an indicator of the structure of text.

Silber, Gregory H. & Kathleen F. McCoy (2002).
Efficiently computed lexical chains as an intermediate representation for automatic text summarization.

A lexical cohesion based news story segmentation system.
AI Communications, 17(1):3–12.