The Extended Lesk Algorithm

(Banerjee & Pedersen, 2003)

- **Banerjee and Pedersen (2003) present a new measure of semantic relatedness between concepts that is based on the number of shared words (overlaps) in their definitions (glosses).**

- **This measure takes as input two concepts (represented by two WordNet synsets) and outputs a numeric value that quantifies their degree of semantic relatedness. This numeric value quantifying their degree of semantic relatedness is further used for performing WSD.**

- **The measure and corresponding WSD algorithm represent a variant of the classical Lesk algorithm (that is based on gloss overlaps), a variant which extends the glosses of the concepts under consideration.**

- **This measure extends the glosses of the concepts under consideration to include the glosses of other concepts to which they are related according to a given concept hierarchy (in this case WordNet).**

The Lesk Algorithm

- **Gloss overlaps were introduced by [Lesk, 1986] to perform WSD.**

- **The Lesk algorithm assigns a sense to a target word ("cuvant tinta" sau cuvant de dezambiguizat), in a given context, by comparing the glosses of its various senses with those of the other words in the context. That sense of the target word whose gloss has the most words in common with the glosses of the neighboring words is chosen as its most appropriate sense.**

- **Example: consider the following glosses of *car* and *tire***

  car: four wheel motor vehicle usually propelled by an internal combustion engine

  tire: hoop that covers a wheel, usually made of rubber and filled with compressed air

  - ✓ **the glosses of these concepts share the content word *wheel***

- **Note that the original Lesk algorithm only considers overlaps among the glosses of the target word and those words that surround it in the given context.**

- **Limitation: dictionary glosses are short and do not provide sufficient vocabulary (content words).**

- **The <u>extended gloss overlap measure</u> introduced by Banerjee and Pedersen (2003) expands the glosses of the words being compared to include glosses of concepts that are known to be related to the concepts being compared.**

    - ✓ **Concepts that are related through explicit WordNet relations are being taken into consideration.**

The Extended Gloss Overlap Measure

- ➢ **When measuring the relatedness between two input synsets, we not only look for overlaps between the glosses of those synsets, but also between the glosses of the hypernym, hyponym, meronym, holonym and troponym synsets of the input synsets, as well as between synsets related to the input synsets through such relations as attribute, similar-to, also-see etc.**

- **NOT all of these relations are equally helpful!**

- ➢ **The optimum choice of relations to use for comparisons is possibly dependent on the application in which the overlaps-measure is being employed.**

- ➢ **Here we will be applying this measure of relatedness to the task of WSD.**

    - ✓ **Example: In the case of noun disambiguation, empirical testing has proven it is useful to compare the glosses of hyponyms and meronyms of the input synsets. Note the recommendation to use hyponym synsets, not hypernym ones, or both of them. In other words, hyponymy provides more information**

than hypernymy, although both form the IS-A relation.

Why it is important to have a measure of relatedness:

WordNet provides *explicit* semantic relations between synsets. However, such links do not cover all possible relations between synsets. For example, WN encodes no direct link between the synsets *car* and *tire*, although they are clearly related. We observe however that the glosses of these two synsets have words in common. Such overlaps provide evidence that there is an *implicit* relation between those synsets. Given such a relation, we further conclude that synsets explicitely related to *car* are thereby also related to synsets explicitly related to *tire*. Therefore:

- **This measure combines the advantages of gloss overlaps with the structure of a concept hierarchy (such as WN) to create an extended view of relatedness between synsets.**

Scoring mechanism

- **The original Lesk algorithm compares the glosses of a pair of concepts and computes a score by counting the number of words that are shared between them.**

- **Note that this scoring mechanism does not differentiate between single word and phrasal overlaps. For example, it assigns a score of 3 to the concepts *drawing paper* and *decal* having the following glosses:**

  drawing paper: paper that is specially preparedfor use in drafting

  decal: the art of transferring designs from specially prepared paper to a wood or glass or metal surface.

  Here there are three words that overlap, *paper* and the two-word phrase *specially prepared*.

- **Banerjee and Pedersen (2003) assign an n word overlap the score of n^2 (because a phrasal n-word overlap is a much rarer occurrence than a single word overlap). For the above gloss pair they would assign the score of 5 (instead of 3).**

The Banerjee and Pedersen overlap detection and scoring mechanism can be formally defined as follows:

- **When comparing two glosses, we define an overlap between them to be the longest sequence of one or more consecutive words that occurs in both glosses such that neither the first nor the last word is a function word, that is a pronoun, preposition, article or conjunction.**

- **If two or more such overlaps have the same longest length, then the overlap that occurs earliest in the first string being compared is reported.**

- **Given two strings, the longest overlap between them is detected, removed and in its place a unique marker is placed in each of the two input strings. The two strings thus obtained are then again checked for overlaps, and this process continues until there are no longer any overlaps between them.**

- **The sizes of the overlaps thus found are squared and added together to arrive at the score for the given pair of glosses.**

COMPUTING RELATEDNESS

- **The extended gloss overlap measure computes the relatedness between two input synsets *A* and *B* by comparing the glosses of synsets that are related to *A* and *B* through explicit relations provided in WordNet.**

1. We define *Rels* as a non-empty set of WN relations:

RELS $\subset$ {*r* | *r* is a relation defined in WordNet}.

For example, if $r \in$ RELS represents the hypernymy relation, then *r*(*A*) returns the gloss of a hypernym synset of *A*.

- ✓ **Note that, if more than one synset is related to the input synset by means of the same semantic relation, then the respective glosses are concatenated and returned as a unique string. If no synset is related to the input one by the given relation, then the null string is returned.**

2. Next, form a non-empty set of *pairs* of relations from the set of relations RELS. The only constraint in forming such pairs is that if the pair (r_1, r_2) is chosen, ($r_1, r_2 \in$ RELS), then the pair (r_2, r_1) must also be chosen so that the relatedness measure is reflexive. That is,

$$relatedness\ (A, B) = relatedness\ (B, A).$$

Thus, we definethe set RELPAIRS as follows:

$$\textbf{RELPAIRS} = \{(R_1, R_2) \mid R_1, R_2 \in \textbf{RELS}; \textbf{ if } (R_1, R_2) \in \textbf{RELPAIRS}, \textbf{ then } (R_2, R_1) \in \textbf{RELPAIRS}\}.$$

COMPUTING RELATEDNESS

3. Assume that *score( )* is a function that accepts as input two glosses, finds the phrases that overlap between them and returns a score as previously described.

- **Given all of the above, the relatedness score between the input synsets *A* and *B* is computed as follows:**

$$\textit{relatedness}\,(A, B) = \sum_{\forall (R_1, R_2) \in \text{RELPAIRS}} \textit{score}\,(R_1(A), R_2(B)).$$

- **Example:**

Assume that our set of relations is RELS = {gloss, hype, hypo} (where *hype* and *hypo* are contractions of hypernym and hyponym respectively). Further assume that our set of relation pairs is

RELPAIRS = {(gloss, gloss), (hype, hype), (hypo, hypo), (hype, gloss), (gloss, hype)}.

Then the relatedness between synsets *A* and *B* is computed as follows:

$$\textit{relatedness}\,(A,B) = \textit{score}\,(\textit{gloss}(A), \textit{gloss}(B)) + \textit{score}\,(\textit{hype}(A), \textit{hype}(B)) + \textit{score}\,(\textit{hypo}(A), \textit{hypo}(B)) + \textit{score}(\textit{hype}(A), \textit{gloss}(B)) + \textit{score}\,(\textit{gloss}(A), \textit{hype}(B)).$$

Application to WSD

(Usage of the defined relatedness score in WSD)

Terminology and notations:

- **word to be disambiguated = target word**
- **given the input sentence, we look at the target word within a context window or window of context (fereastra de context)**
- **within the context window we look only at the content words (cuvinte cu continut: substantive, adjective, verbe, adverbe)**
- **A window of size n will denote taking into consideration n content words to the left and n content words to the right of the target word, whenever possible. The total number of words taken into consideration for disambiguation will therefore be $2n+1$. When not enough content words are available, the entire sentence in which the target word occurs will represent the context window.**
- **Assume that the context window consists of $2n+1$ words denoted by w_i, $-n \leq i \leq +n$, where the target word is denoted w_0. Further let $|w_i|$ denote the number of candidate senses of word w_i, and let these senses be denoted by $s_{i,j}$, $1 \leq j \leq |w_i|$.**
- **Next we assign to each possible sense k of the target word a $SenseScore_k$ computed by adding together the**

relatedness scores obtained by comparing the sense of the target word in question with every sense of every non-target word in the context window. The SenseScore for sense $s_{0,k}$ is computed as follows :

$$SenseScore_k = \sum_{i=-n}^{n} \sum_{j=1}^{|w_i|} înrudire(s_{0,k}, s_{i,j}), \quad i \neq 0$$

<u>Result:</u>

- **That sense with the highest SenseScore is judged to be the most appropriate sense for the target word.**

Complexity – linear:

If there are on average α senses per word and the window of context is *N* words long, there are $\alpha^2 \times (N-1)$ pairs of sets of synsets to be compared, which increases linearly with *N*.

Implementation remarks:

- **If several senses have the same SenseScore then the first sense – according to WN – is chosen for the target word.**

- **Increasing the size of the context window does not improve disambiguation results. (The usual context window sizes are 3, 5, 7).**

- **If a word in the window is used as part of a compound (ex.: *child-doctor*), then the senses associated with that compound are the candidates (i.e. not the senses of "child", not the senses of "doctor", but the senses of "child-doctor").**

- **The choice of semantic relations depends on the part of speech of the target word. For nouns the most**

informative relations are *hyponymy* and *meronymy*; for adjectives *also-see* and *attribute*; for verbs *hyponymy*.

- **It is useful for disambiguation to use the example which is included in some of the WN glosses. (Use the gloss entirely, example included).**