CS6390 PROJECT PROPOSAL: REDIRE/GIRAF

OLIVER RICHARDSON MAKS CEGIELSKI-JOHNSON

1. Task

Our interest in this problem is two-fold; we are interested in studying both recognition and generation of paraphrases. Hence we will need to discuss our current approaches to both of these problems. 4 The paraphrase recognition task is defined as attempting to determine whether two sentences are paraphrases of each other, returning a YES/NO answer.

The paraphrase generation task is defined as being given an input sentence, and trying to generate a paraphrase from that input. Deciding correctness is not defined yet.

2. Architecture

Our current implementation is inspired by a paper by Malakasiotis[2]. The general structure of this implementation is as follows:

(1) Restructure each pair of sentences s_1 and s_2 into 10 new string transformations $t_{i,1}$ and $t_{i,2}$, maintaining the original order

(a) String of the tokens
(b) String of the stems.
(c) String of the POS tags.
(d) String of the soundex codes[1].
(e) String of all the noun tokens
(f) String of all the noun stems
(g) String of all the noun soundex codes.
(h) String of all the verb tokens.
(i) String of all the verb stems.
(j) String of all the verb soundex codes.

(2) For each string pair s_1 and s_2, and for each transformation of that string $t_{i,1}$ and $t_{i,2}$, we want to compute the similarity $d_i(t_{i,1}, t_{i,2})$, using the metrics

(a) Levenshtein (word edit)
(b) Jaro-Winkler
(c) Manhattan & Euclidean
(d) Cosine
(e) n-gram ($n = 3$)
(f) Overlap
(g) Dice coefficient
(h) Jaccard coefficient

(3) Computing the similarities between each transformation of the two strings results in 90 features.

(4) Next, for each pair of sentences s_1 and s_2, if s_1 is longer than s_2 we compute the substrings of s_1 which have the same length as s_2. For each substring s_1' generated, we compute all of the similarities between s_1' and s_2. We try to find which substring results in the max average similarity over all the similarity metrics, namely $s_1'^*$. Then for $s_1'^*$ we compute each of the similarities between it and s_2, as well as the average of all of them, resulting in 10 features. We do this for the string transformations which result in tokens, stems, POS tags, and soundex codes. This results in an additional 40 features per sentence pair (total of 130

features so far). Note that we can just swap the two strings in the discussion above if s_2 is larger than s_1.

(5) Finally, we add three additional features to the current 130. We add a boolean feature for whether s_1 contains a negation, and a feature for whether s_2 contains negation. Then finally, we add a feature for the length ratio, defined as $\frac{\min(|s_1|,|s_2|)}{\max(|s_1|,|s_2|)}$.

3. Experiments

4. Results

5. Contributions

References

[1]

[2] Prodromos Malakasiotis. Paraphrase recognition using machine learning to combine similarity measures. In *Proceedings of the ACL-IJCNLP 2009 Student Research Workshop*, pages 27–35. Association for Computational Linguistics, 2009.