*These solution notes are as submitted in January 2003*

**Information Retrieval 2003 Paper 7 Question 11 (SHT)**

(*a*) Bookwork. Lecture 2.

Some terms are more distinctive of a document's or a query's meaning than others. Weighting these distinctive terms higher when calculating document-query similarity leads to higher retrieval performance.

The tf*idf formula for term weighting works like this: From a large text collection of N documents, the following data is collated for each term i in a document j:

1. frequency of term in document: $t_{ij}$
2. Number of documents containing term i: $d_i$
3. frequency of highest-frequency term in document j: $m_j$

The weight $w_{ij}$ for term $j$ in document $i$ is:

$$w_{ij} = log \frac{t_{ij}}{m_i \cdot d_j}$$

Note: there are many variants of the tf*idf formula, most of which I will pass as correct, as long as the main idea (frequent terms in the document are more distinctive, rare terms overall are more distinctive) is clear. The normalisation with $m_j$, for instance, is not crucial, but I will subtract one point if its use is not clear to the student.

(*b*) two of the following are required. Lecture 5.

- Person's last names can be the same as common nouns or adjectives. Confusion in sentence-initial position of the common noun/adj. This problem makes use of a gazetteer for these names impractical.

  Mr. Brown; Mrs. Boxer

- Company names can be identical to full person's names:

  Peter Stuyvesant

- Names can have complex structure, such that one needs complex grammars to capture them, particularly with complex titles:

  Dr. Adam Miller III; Sam Davis Jr.; HRH The Prince of Wales

- Last names can occur without the first names, and it can be hard to distinguish them from unknown words or location names or company names.

  Brown agreed to come with us; Paris did not like what he saw.

- Mid initial ambiguous with end of sentence marker:

  it was Sandy D. Miller did not → . is end-of-sentence The party with Sandy D. Miller was a full success → . is part of mid-initial.

- Complex company names and coordination ambiguous with last names:

  Miller, Hearst & Co → "Hearst" is not to be recognised as person name in this context.

($c$) Lecture 6.

The system's input are examples of correct relations (e.g. a company name and the location of its headquarters in textual context). It uses the corpus text to iteratively find new tuples, new patterns and then new tuples again. Using the textual contexts as patterns, it finds other name-location tuples (which a human can review). Using these old and new tuples, it finds new contexts in which the two well-known parts of the tuple occur in close proximity. It has some generalisation mechanism to hypothesise different kinds of patterns from the contexts it saw (more and less general). Some of these contexts are good templates, some are not. The system evaluates how good each template is by heuristics (e.g how high is the proportion of attested good templates achieved using this particular pattern). It then uses the good templates in the next iteration, to find new tuples. A human can hand-check the new tuples and patterns at each iteration.

($d$) This is a transfer question. Touched upon in lecture 6.

Task (ii) is the task which is most straightforwardly solved with bootstrapping. Perfect conditions: close proximity of the two entities (name and job title) can be expected, and one can expect rather formulaic ways of expressing this information, which should be picked up from text of the right genre, e.g. newspaper text.

Task (i) is too easy to require bootstrapping. More information is contained in the name string itself, rather than in the immediate context of the name. Better to use machine learning on the features of the name string.

Task (iii) is to complex to be solved directly by bootstrapping from the strings. Three pieces of information have to be extracted: the relationship and the two people. The variation by which different relationships are expressed can be expected to be high, therefore there won't be enough training examples for each case in the text. Bootstrapping can be expected to have a problem with this task. Also, for many cases inference is needed, which cannot be performed directly on the string, but which requires an abstraction language (semantic representation).