	Corpora			
	DUC-2001	*Inspec*	NUS	ICSI
Type	News articles	Paper abstracts	Full papers	Meeting transcripts
# Documents	308	500	211	161
# Tokens/Document	876	134	8291	1611
# Candidate words/Document	312	57	3271	453
# Candidate phrases/Document	207	34	2027	296
# Tokens/Candidate phrase	1.5	1.7	1.6	1.5
# Gold keyphrases	2484	4913	2327	582
# Gold keyphrases/Document	8.1	9.8	11.0	3.6
U/B/T/O distribution (%)	17/61/18/4	13/53/25/9	27/50/16/7	68/29/2/1
# Tokens/Gold keyphrase	2.1	2.3	2.1	1.3

Table 1: Corpus statistics for the four datasets used in this paper. A candidate word/phrase, typically a sequence of one or more adjectives and nouns, is extracted from the document initially and considered a potential keyphrase. The U/B/T/O distribution indicates how the gold standard keys are distributed among unigrams, bigrams, trigrams, and other higher order n-grams.

Step 2: Lexical unit ranking Once the candidate list is generated, the next task is to rank these lexical units. To accomplish this, it is necessary to build a representation of the input text for the ranking algorithm. Depending on the underlying approach, each candidate word is represented by its syntactic and/or semantic relationship with other candidate words. The relationship can be defined using co-occurrence statistics, external resources (e.g., neighborhood documents, Wikipedia), or other syntactic clues.

Step 3: Keyphrase formation In the final step, the ranked list of candidate words is used to form keyphrases. A candidate phrase, typically a sequence of nouns and adjectives, is selected as a keyphrase if (1) it includes one or more of the top-ranked candidate words (Mihalcea and Tarau (2004), Liu et al. (2009b)), or (2) the sum of the ranking scores of its constituent words makes it a top scoring phrase (Wan and Xiao, 2008).

3.2 The Five Keyphrase Extractors

As mentioned above, we re-implement five unsupervised approaches for keyphrase extraction. Below we provide a brief overview of each system.

3.2.1 Tf-Idf

Tf-Idf assigns a score to each term t in a document d based on t's frequency in d (term frequency) and how many other documents include t (inverse document frequency) and is defined as:

$$\text{tfidf}_t = \text{tf}_t \times log(D/D_t) \quad (1)$$

where D is the total number of documents and D_t is the number of documents containing t.

Given a document, we first compute the Tf-Idf score of each candidate word (see Step 1 of the generic algorithm). Then, we extract all the longest n-grams consisting of candidate words and score each n-gram by summing the Tf-Idf scores of its constituent unigrams. Finally, we output the top N n-grams as keyphrases.

3.2.2 TextRank

In the TextRank algorithm (Mihalcea and Tarau, 2004), a text is represented by a graph. Each vertex corresponds to a word type. A weight, w_{ij}, is assigned to the edge connecting the two vertices, v_i and v_j, and its value is the number of times the corresponding word types co-occur within a window of W words in the associated text. The goal is to (1) compute the score of each vertex, which reflects its *importance*, and then (2) use the word types that correspond to the highest-scored vertices to form keyphrases for the text. The score for v_i, $S(v_i)$, is initialized with a default value and is computed in an iterative manner until convergence using this recursive formula:

$$S(v_i) = (1-d) + d \times \sum_{v_j \epsilon Adj(v_i)} \frac{w_{ji}}{\sum_{v_k \epsilon Adj(v_j)} w_{jk}} S(v_j) \quad (2)$$

where $Adj(v_i)$ denotes v_i's neighbors and d is the damping factor set to 0.85 (Brin and Page, 1998). Intuitively, a vertex will receive a high score if it has many high-scored neighbors. As noted before, after convergence, the T% top-scored vertices are