selected as keywords. Adjacent keywords are then collapsed and output as a keyphrase.

According to Mihalcea and Tarau (2004), TextRank's best score on the *Inspec* dataset is achieved when only nouns and adjectives are used to create a uniformly weighted graph for the text under consideration, where an edge connects two word types only if they co-occur within a window of two words. Hence, our implementation of TextRank follows this configuration.

3.2.3 SingleRank

SingleRank (Wan and Xiao, 2008) is essentially a TextRank approach with three major differences. First, while each edge in a TextRank graph (in Mihalcea and Tarau's implementation) has the same weight, each edge in a SingleRank graph has a weight equal to the number of times the two corresponding word types co-occur. Second, while in TextRank only the word types that correspond to the top-ranked vertices can be used to form keyphrases, in SingleRank, we do not filter out any low-scored vertices. Rather, we (1) score each candidate keyphrase, which can be any longest-matching sequence of nouns and adjectives in the text under consideration, by summing the scores of its constituent word types obtained from the SingleRank graph, and (2) output the N highest-scored candidates as the keyphrases for the text. Finally, SingleRank employs a window size of 10 rather than 2.

3.2.4 ExpandRank

ExpandRank (Wan and Xiao, 2008) is a TextRank extension that exploits neighborhood knowledge for keyphrase extraction. For a given document d, the approach first finds its k nearest neighboring documents from the accompanying document collection using a similarity measure (e.g., cosine similarity). Then, the graph for d is built using the co-occurrence statistics of the candidate words collected from the document itself and its k nearest neighbors.

Specifically, each document is represented by a term vector where each vector dimension corresponds to a word type present in the document and its value is the Tf-Idf score of that word type for that document. For a given document d_0, its k nearest neighbors are identified, and together they form a larger document set of k+1 documents, $D = \{d_0, d_1, d_2, ..., d_k\}$. Given this document set, a graph is constructed, where each vertex corresponds to a candidate word type in D, and each edge connects two vertices v_i and v_j if the corresponding word types co-occur within a window of W words in the document set. The weight of an edge, $w(v_i, v_j)$, is computed as follows:

$$w(v_i, v_j) = \sum_{d_k \epsilon D} sim(d_0, d_k) \times freq_{d_k}(v_i, v_j) \quad (3)$$

where $sim(d_0, d_k)$ is the cosine similarity between d_0 and d_k, and $freq_{d_k}(v_i, v_j)$ is the co-occurrence frequency of v_i and v_j in document d_k. Once the graph is constructed, the rest of the procedure is identical to SingleRank.

3.2.5 Clustering-based Approach

Liu et al. (2009b) propose to cluster candidate words based on their semantic relationship to ensure that the extracted keyphrases *cover* the entire document. The objective is to have each cluster represent a unique aspect of the document and take a representative word from each cluster so that the document is covered from all aspects.

More specifically, their algorithm (henceforth referred to as KeyCluster) first filters out the stop words from a given document and treats the remaining unigrams as candidate words. Second, for each candidate, its relatedness with another candidate is computed by (1) counting how many times they co-occur within a window of size W in the document and (2) using Wikipedia-based statistics. Third, candidate words are clustered based on their relatedness with other candidates. Three clustering algorithms are used of which spectral clustering yields the best score. Once the clusters are formed, one representative word, called an exemplar term, is picked from each cluster. Finally, KeyCluster extracts from the document all the longest n-grams starting with zero or more adjectives and ending with one or more nouns, and if such an n-gram includes one or more exemplar words, it is selected as a keyphrase. As a post-processing step, a frequent word list generated from Wikipedia is used to filter out the frequent unigrams that are selected as keyphrases.