*Journal of Universal Computer Science, vol. 18, no. 5 (2012), 650-675*
*submitted: 5/8/11, accepted: 28/2/12, appeared: 1/3/12 © J.UCS*

# Wikipedia-Based Semantic Interpreter Using Approximate Top-k Processing and Its Application[1]

Jong Wook Kim [2]
(Teradata Corporation, El Segundo, CA, USA
jongwook.kim@teradata.com)

Ashwin Kashyap, Sandilya Bhamidipati
(Technicolor, Palo Alto, CA, USA
{ashwin.kashyap, sandilya.bhamidipati}@technicolor.com)

**Abstract:** Proper representation of the meaning of texts is crucial for enhancing many data mining and information retrieval tasks, including clustering, computing semantic relatedness between texts, and searching. Representing of texts in the concept-space derived from Wikipedia has received growing attention recently. This concept-based representation is capable of extracting semantic relatedness between texts that cannot be deduced with the bag of words model. A key obstacle, however, for using Wikipedia as a semantic interpreter is that the sheer size of the concepts derived from Wikipedia makes it hard to efficiently map texts into concept-space. In this paper, we develop an efficient and effective algorithm which is able to represent the meaning of a text by using the concepts that best match it. In particular, our approach first computes the approximate top-$k$ Wikipedia concepts that are most relevant to the given text. We then leverage these concepts for representing the meaning of the given text. The experimental results show that the proposed technique provides significant gains in execution time without causing significant reduction in precision. We then explore the effectiveness of the proposed algorithm on a real world problem. In particular, we show that this novel scheme could be leveraged to boost the effectiveness in finding topic boundaries in a news video.

**Key Words:** Wikipedia, concept, semantic interpretation
**Category:** H.3.1, H.3.3, M.4

## 1 Introduction

The bag of words (BOW) model has been shown to be very effective in diverse areas which span a large spectrum from traditional text-based applications, such as clustering and classification, to web and social media. While there have been a number of models in information retrieval using the bag of words, including boolean [Salton et al. 1983], probability [Croft and Harper 1979] and fuzzy [Ogawa et al. 1991] ones, the vector-space model [Salton et al. 1975] is the most commonly used in the literature. In the word-based vector model, given a

[1] This work is based on an earlier work: Efficient wikipedia-based semantic interpreter by exploiting top-k processing, in proceedings of the 19th ACM international conference on information and knowledge management , © ACM, 2010. http://doi.acm.org/10.1145/1871437.1871736

[2] Corresponding author

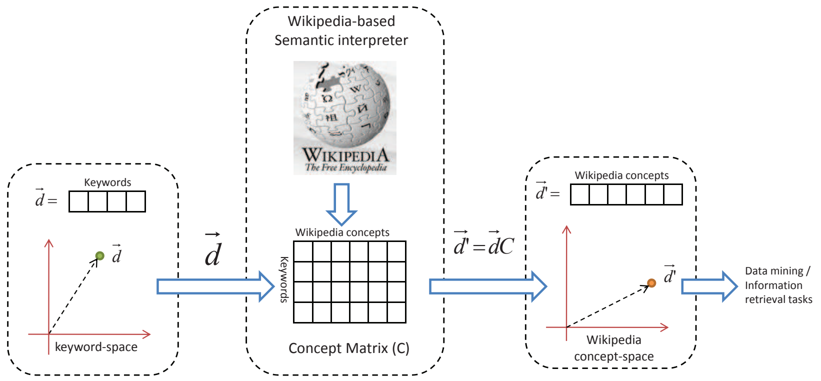


Figure 1: A Wikipedia-based semantic interpreter maps documents from the keyword-space into the Wikipedia concept-space: resulting document vectors in the *concept-space* can be used in diverse applications, instead of the original document vectors in the *keyword-space*.

dictionary, $\mathcal{U}$, with $u$ distinct words, a document is represented as $u$-dimensional vector, $\mathbf{d}$, where only those positions in the vector that correspond to the document words are set to $> 0$ and all others are set to 0.

Although the BOW-based vector model is the most popular scheme, it has limitations: these include sparsity of vectors and lacking semantic relationship between words. One way to tackle these limitations is to analyze the keywords of the documents in the corpus to extract latent concepts that are dominant in the corpus, and models documents in the resulting latent concept-space. While these techniques, including LSI [Deerwester et al. 1990], NMF [Lee and Seung 2000] and LDA [Blei et al. 2003] have produced impressive results in text-based application domains, they still have a limitation in that the resulting latent concepts are different from human-organized knowledge, and thus they cannot be interpreted by human knowledge.

A possible solution to resolve this difficulty is to enrich the individual documents with the background knowledge obtained from existing human-contributed knowledge; i.e., Wikipedia [Wikipedia], WordNet [Miller et al. 1990] and ODP [Open Directory Project]. Especially, Wikipedia is one of the largest free encyclopedias on the Web, containing more than 4 million articles in the English version. Each article in Wikipedia describes a concept (topic), and each concept belongs to at least one category. Wikipedia uses redirect pages, which redirects a concept to another concept, for synonymous ones. On the other hand, if a concept is polysemous, Wikipedia displays possible meanings of polysemous concepts in disambiguation pages. Due to its comprehensiveness and expertise, Wikipedia has been applied to diverse applications, such as word disambiguation [Gabrilovich et al. 2007], clustering [Carmel et al. 2009], classification [Wang and Domeniconi 2008], topic detection [Schonhofen 2006], user profile creation [Ramanathan and Kapoor 2008] and link analysis [Milne et al. 2008], where it is used as a *semantic interpreter which reinterprets original documents based on the concepts of Wikipedia*. As shown in Figure 1, such semantic reinterpretation maps original documents from the *keyword-space* into the *concept-space* [Gabrilovich et al. 2009]. Generally, the mapping between the original keyword dictionary and the concept is performed by (a) computing the similarity score between each concept in Wikipedia and the original document and (b) replacing the keywords in the original document with these matched concepts and the corresponding similarity scores. In the literature, this process is commonly defined as the matrix multiplication between the original document-keyword matrix and the keyword-concept matrix (Figure 1). Such a Wikipedia-based semantic reinterpretation has the potential to ensure that documents mapped into the Wikipedia concept-space are semantically informed, significantly improving the effectiveness on various tasks, including text categorization [Gabrilovich et al. 2009] and clustering [Hu et al. 2009].

### 1.1 Our Contributions

While obtaining the concept-keyword matrices from Wikipedia is challenging, there are also plenty of techniques for discovering them [Gabrilovich et al. 2007, Gabrilovich et al. 2009]. The main obstacle in leveraging the Wikipedia as a semantic interpreter stems from efficiency concerns. Considering the sheer size of Wikipedia articles (more than 4M concepts), reinterpreting original documents based on all possible concepts of Wikipedia can be prohibitively expensive. Therefore, it is essential that the techniques used for such a semantic reinterpretation be fast.

More importantly, reinterpreting original documents with all possible Wikipedia concepts imposes an additional overhead in the application level, since reinterpreted documents will be represented in the augmented concept-space that corresponds to a very high dimension. Most applications do not require documents to be represented with all possible Wikipedia concepts, since they are not equally important to the given document. Indeed, insignificant concepts tend to be noisy. Thus, in this paper, we aim to find the best $k$ concepts in Wikipedia that match a given document, and semantically reinterpret it based on such $k$ concepts. In particular, our contributions in this paper include the following:

- We develop a novel algorithm that efficiently reinterprets documents based on the concepts of Wikipedia. The presented algorithm in this paper is able to approximately but efficiently compute the most significant $k$-concepts in Wikipedia for a given document, and use these concepts to map an original document from the keyword-space into the concept-space. The experimental results show that the proposed algorithm significantly improves efficiency while maintaining high precision. Processing time gains against existing solutions can be pushed orders of magnitude for a 95% precision.

- Having evaluated the efficiency of the proposed technique, we also explore the usage of the proposed algorithm on a real world problem. In particular, we use the proposed method to effectively identify topic boundaries in a news video. Naturally, the news video consists of a series of stories which are independent of each other. Since many of stories in a news video are not clustered based on topical similarity, the whole contents themselves are not useful in guiding users for accessing the relevant piece of information. Thus, there is an impending need for mechanisms that effectively detect topic boundaries in a news video in order to provide proper navigational support for users. We experimentally show that the proposed scheme significantly improves the accuracy of topic boundary detection tasks in a news video.

We note that although we rely on Wikipedia as the background knowledge in this paper, the proposed technique can be used as a general framework for a semantic interpretation, since it can be equally applicable to various external knowledge, such as ODP [Open Directory Project], WordNet [Miller et al. 1990] and Yahoo-directory [Yahoo!].

## 2 Problem Definition

In this section, we formally define the problem and introduce the notation we will use to develop and describe our algorithms.

### 2.1 Semantic Reinterpretation with the All Possible Wikipedia Concepts

Let $\mathcal{U}$ be a dictionary with $u$ distinct words. The concepts in Wikipedia are represented in the form of a $u \times m$ *keyword-concept matrix*, $C$, where $m$ is the number of concepts which correspond to articles of Wikipedia and $u$ is the number of distinct keywords in the dictionary. Let $C_{i,r}$ denote the weight of the $i$-th keyword, $t_i$, in the $r$-th concept, $c_r$. Let $C_{-,r} = \mathbf{c_r} = [w_{1,r}, w_{2,r}, \cdots, w_{u,r}]^T$ be the $r$-th concept vector. Without loss of generality, we assume that each *concept-vector*, $C_{-,r}$, is normalized into a unit length.

Given a dictionary, $\mathcal{U}$, a document, $d$, is represented as a $l$-dimensional vector, $\mathbf{d} = [w_1, w_2, \cdots, w_u]$.

Given a *keyword-concept matrix*, $C$, and a document vector, $\mathbf{d}$, *a semantically reinterpreted document vector with all possible Wikipedia concepts*, $\mathbf{d'} = [w'_1, w'_2, \cdots, w'_m]$, is defined as (Figure 1)

$$\mathbf{d'} = \mathbf{d}C.$$

By definition of matrix multiplication, the contribution of the concept $c_r$ in the vector $\mathbf{d'}$ is computed as follows:

$$w'_r = \sum_{1 \le i \le u} w_i \times C_{i,r} = \sum_{\forall_{w_i \neq 0}} w_i \times C_{i,r}.$$

### 2.2 Semantic Reinterpretation with the Top-*k* Wikipedia Concepts

As we mentioned in the introduction, computing $\mathbf{d'}$ with all possible Wikipedia concepts may be prohibitively expensive. Thus, our goal in this paper is to *reinterpret a document with the best k concepts in Wikipedia that are relevant to it.*

Let $\mathbf{d'} = [w'_1, w'_2, \cdots, w'_m]$ be a semantically reinterpreted document. We then define $S_k$ to be a set of $k$ concepts, such that the following holds:

$$\forall_{c_r \in S_k,\ c_p \notin S_k} \ \ w'_r \ge w'_p.$$

In other words, $S_k$ contains $k$ concepts whose contributions to $\mathbf{d'}$ are greater than or equal to the others. Then, *a semantic reinterpretation of* $\mathbf{d}$ *based on the top-k concepts in Wikipedia that best match it is defined as* $\mathbf{d'} = [w'_1, w'_2, \cdots, w'_m]$ *where*

– *if* $c_r \in S_k$,
$$w'_r = \sum_{1 \le i \le u} w_i \times C_{i,r} = \sum_{\forall w_i \neq 0} w_i \times C_{i,r}$$

– otherwise, $w'_r = 0$.

### 2.3 Problem Definition: Semantic Reinterpretation with the Approximate Top-*k* Wikipedia Concepts

Exactly computing the best $k$ concepts that are relevant to a given document often requires to scan an entire keyword-concept matrix which is very expensive. Thus, in order to achieve further efficiency gains, we relax $S_k$ as follows: given a document, $\mathbf{d}$, let $S_{k,\alpha}$ be a set of $k$ concepts such that approximately $\alpha k$ answers in $S_{k,\alpha}$ belong to $S_k$, where $0 \le \alpha \le 1$. Then, our objective in this paper is defined as follows:

*Problem 1 Semantic reinterpretation with* $S_{k,\alpha}$. Given a keyword-concept matrix, $C$, a document vector, $\mathbf{d}$, and the corresponding approximate best $k$ concepts, $S_{k,\alpha}$, a semantic reinterpretation of $\mathbf{d}$ based on the approximate top-$k$ concepts in Wikipedia that best match it is defined as $\mathbf{d'} = [w'_1, w'_2, \cdots, w'_m]$ where

– *if* $c_r \in S_{k,\alpha}$,
$$w'_r \approx \sum_{1 \le i \le u} w_i \times C_{i,r} = \sum_{\forall w_i \neq 0} w_i \times C_{i,r}$$

– otherwise, $w'_r = 0$.

In other words, the original document, $d$, is approximately mapped from the *keyword-space* into the *concept-space* which consists of the approximate $k$ concepts in Wikipedia that best match a document $d$. Thus, the key challenge to this problem is how to efficiently identify such approximate top-$k$ concepts, $S_{k,\alpha}$. To address this problem, in this paper, we present a novel ranked processing algorithm to efficiently compute $S_{k,\alpha}$ for a given document.

## 3 Naive Solutions to $S_k$

In this section, we describe naive schemes for exactly computing the top-k concepts, $S_k$, of a given document.

### 3.1 Scanning the Entire Data

One obvious solution to this problem is to scan the entire $u \times m$ *keyword-concept matrix*, $C$, multiply the document vector, $\mathbf{d}$, with each concept vector, $C_{-,r}$, sort the resulting scores, $w'_r$ (where $1 \leq r \leq m$), in descending order, and choose only the $k$-best solutions. A more promising solution to this problem is to leverage an *inverted index*, commonly used in IR systems, which enables to scan only those entries whose corresponding values in the keyword-concept matrix are greater than 0. Both schemes would be quite expensive, because they waste most of resources in processing unpromising data that will not belong to the best $k$ results.

### 3.2 Threshold-Based Ranked Processing Scheme

There have been a large number of proposals for ranked or top-k processing. Among them, the threshold-based algorithms, such as TA, FA, and NRA, are the most well-known methods. These algorithms assume that given sorted-lists, each object has a single score in each list and an aggregation function, which combines independent object's scores in each list, is monotone such as *min*, *max*, *(weight) sum* and *product*. These monotone scoring functions guarantee that a candidate dominating the other one in its sub-scores will have a combined score better than the other one, which enables early stopping during the top-$k$ computation, to avoid scanning all the lists. Generally, TA (and FA) algorithm requires two access methods: random-access and sorted-access. However, supporting random-access to a high-dimensional data, such as document-term matrix, would be prohibitively expensive. Therefore, in this paper, we employ NRA as a base framework, since it requires only a sorted-access method, and thus is suitable for high-dimensional data, such as a concept matrix $C$.

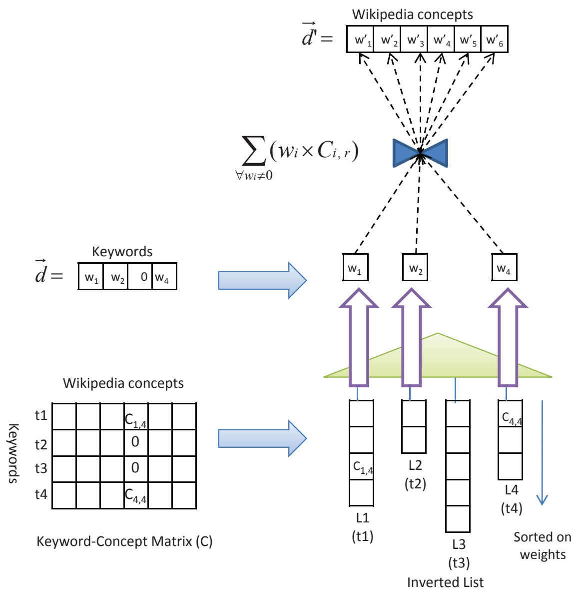


Figure 2: A general framework of the Wikipedia-based semantic interpreter which relies on ranked processing schemes.

### 3.2.1 Sorted Inverted Lists for the Concept Matrix

To support sorted accesses to a $u \times m$ *keyword-concept matrix*, $C$, an inverted index that contains $u$ lists is created (Figure 2). For each keyword $t_i$, the corresponding list $L_i$ contains a set of $\langle r, C_{i,r}\rangle$s, where $C_{i,r}$ is the weight of the keyword, $t_i$, in Wikipedia concept $c_r$. As shown in Figure 2, each inverted list maintains only concepts whose weights are greater than 0. This inverted list is created in decreasing value on weights to support sorted accesses.

### 3.2.2 NRA-based Scheme for Computing $S_k$

From the definition of $w'_r$ in the previous section, it is clear that the score function is monotone in the $u$ independent lists since it is defined as a *weight sum*. Given an original document $\mathbf{d} = [w_1, w_2, \cdots, w_u]$, NRA visits the input lists in a round-robin manner and updates a threshold vector $\mathbf{th} = [\tau_1, \tau_2, \cdots, \tau_u]$ where $\tau_i$ is the last weight read on the list $L_i$. In other words, a threshold vector consists of the upper bounds on the weights of unseen instances in input lists.

After reading an instance $\langle r, C_{i,r}\rangle$ in the list, $L_i$, the possible worst score of the $r$-th position in the semantically reinterpreted document vector, $\mathbf{d'} = [w'_1, w'_2, \cdots, w'_r, \cdots, w'_m]$, is computed as

$$w'_{r,wst} = \sum_{i \in KN_r} w_i \times C_{i,r} \ ,$$

where $KN_r$ is a set of positions in the concept-vector, $C_{-,r}$, whose corresponding weights have been read before by the algorithm. Note that since each list is sorted on weights instead of concept IDs, the concept-vector $C_{-,r}$ is only partially available. On the other hand, the possible best score of $r$-th position in $\mathbf{d'}$ is computed as follows:

$$w'_{r,bst} = \sum_{i \in KN_r} w_i \times C_{i,r} + \sum_{j \notin KN_r} w_j \times \mu_j \ .$$

In summary, the possible worst score is computed based on the assumption that the unseen entries of the concept-vector will be 0, while the possible best score assumes that all unseen entries in the concept-vector will be encountered after the last scan position of each list.

NRA maintains a cut off score, $min_k$, which equals to the lowest score in the current top-$k$ candidates. NRA would stop the computation when a cut off score, $min_k$, is greater than (or equal to) the highest best-score of concepts not belonging to the current top-$k$ candidates. Although this stopping condition always guarantees to produce the correct top-$k$ results (i.e., $S_k$ in our case), such stopping condition is overly pessimistic, assuming that all unknown values of each concept vector would be read after the current scan position of each list. This, however, is not the case especially for the sparse *keyword-concept* matrix where unknown values of each concept vector are expected to be 0 with a very high probability. Therefore, NRA may end up scanning the entire lists, which would be quite expensive.

## 4 Efficiently Interpreting a Document with Wikipedia Concepts

In this section, we describe the proposed algorithm which efficiently interprets a document using Wikipedia. The proposed algorithm consists of two phases: (1) computing the approximate top-$k$ concepts, $S_{k,\alpha}$, of a given document and (2) mapping an original document into the concept-space using $S_{k,\alpha}$.

### 4.1 Phase 1: Identifying the approximate top-$k$ concept, $S_{k,\alpha}$

As described earlier, the threshold-based algorithms are based on the assumption that given sorted-lists, each object has a single score in each list. The possible scores of unseen objects in NRA algorithm are computed based on this assumption. This assumption, however, does not hold for the sparse keyword-concept matrix where most of entries are 0. Thus, in this subsection, we first describe a method to estimate the scores of unseen objects with the sparse keyword-concept matrix, and then present a method to obtain the approximate top-$k$ concepts of a given document by leveraging the expected scores.

#### 4.1.1 Estimating the Bounds on the Number of Input Lists

Since the assumption that each object has a single score in each input list is not valid for a sparse keyword-concept matrix, in this subsection we aim to estimate a bound on the number of input lists where each concept is expected to be found during the computation. A histogram is usually used to approximate data distributions (i.e., probability density function). Many existing approximate top-$k$ processing algorithms maintain a histogram for each input list and estimate the scores of unknown objects by convoluting histograms [Arai et al. 2007, Theobald et al. 2004]. Generally, approximate methods are more efficient than exact schemes. Nevertheless, considering that there are a huge number of lists for the *keyword-concept* matrix, maintaining such histograms and convoluting them in run-time for computing possible aggregated scores is not a viable solution. Thus, in order to achieve further efficiency, we simplify the data distribution of each inverted list by relying on the binomial distribution: i.e., the case in which an inverted list contains a given concept or the other one in which it does not. As will be shown in the next section, such simplified data distribution does not cause a significant reduction in the quality of the top-$k$ results, due to the extreme sparsity of the concept matrix.

Given a keyword $t_i$ and a keyword-concept matrix $C$, the length of the corresponding sorted list, $L_i$, is defined as

$$|L_i| = |\{C_{i,r} \mid C_{i,r} > 0 \;\; where \;\; 1 \leq r \leq m\} \,|.$$

Given a $u \times m$ keyword-concept matrix, $C$, the likelihood that an instance $\langle r, C_{i,r} \rangle$ is in $L_i$ is initialized as

$$\frac{|L_i|}{m}.$$

Generally, the threshold-based algorithms sequentially scan the each sorted list. Let us assume that the algorithm sequentially scans the first $f_i$ instances from the sorted list $L_i$, and the instance $\langle r, C_{i,r} \rangle$ was not seen during the scans. Then, we can compute the likelihood, $P_{\langle r, f_i \rangle}$, that an instance $\langle r, C_{i,r} \rangle$ will be found in the unscanned parts of the list $L_i$ (i.e., the remaining $(|L_i| - f_i)$ instances) as follows:

$$P_{\langle r, f_i \rangle} = \frac{|L_i| - f_i}{m - f_i}.$$

Note that $P_{\langle r, f_i \rangle}$ will be 1 under the assumption that each object has a single score in each input list (i.e., $|L_i| = m$). However, the keyword-concept matrix is extremely sparse, and thus, in most cases, $P_{\langle r, f_i \rangle}$ is close to 0.

Let us be given a document, $d$, and a corresponding $u$-dimensional vector, $\mathbf{d} = [w_1, w_2, \cdots, w_u]$. Furthermore, given $\mathbf{d}$, let $\mathcal{L}$ be a set of sorted lists such that:

$$\mathcal{L} = \{L_i \mid w_i > 0 \;\; where \;\; 1 \leq i \leq u\}.$$

In other words, $\mathcal{L}$ is a set of sorted lists whose corresponding words appear in a given document $d$. Other lists not in $\mathcal{L}$ do not contribute to the computation of the semantically reinterpreted vector, $\mathbf{d'}$, because their corresponding weights in the original vector $\mathbf{d}$ equal to 0 (Figure 2).

Let us further assume that the occurrences of words in a document are independent of each other, which has long been used by many applications due to its simplicity. Given a set of sorted lists, $\mathcal{L}$, let us ask the question, "What is the probability, $P_{found\_exact(\mathcal{L},r,n)}$, that the concept $c_r$, which was not seen in any list so far, will be found in exactly $n$ lists in $\mathcal{L}$ afterward?". While exactly computing $P_{found\_exact(\mathcal{L},r,n)}$ is not impossible, this requires significant amount of computations. A reasonable approximation can be obtained by approximating each $P_{\langle r,f_i\rangle}$ associated with $L_i \in \mathcal{L}$ with a Poisson distribution [Devore 1999]. Then, $P_{found\_exact(\mathcal{L},r,n)}$ can be approximately computed as follows:

$$P_{found\_exact(\mathcal{L},r,n)} \approx poisson(n;\lambda) = e^{-\lambda}\frac{(\lambda)^n}{n!},$$

where,

$$\lambda = \sum_{L_i \in \mathcal{L}} P_{\langle r,f_i\rangle}.$$

Furthermore, we can compute the $P_{found\_upto(\mathcal{L},r,n)}$, the probability that a fully unseen concept $c_r$ will be found in up to $n(\leq |\mathcal{L}|)$ lists in $\mathcal{L}$ during the computation as follows:

$$P_{found\_upto(\mathcal{L},r,n)} = \sum_{0\leq q\leq n} P_{found\_exact(\mathcal{L},r,q)}.$$

Note that $P_{found\_upto(\mathcal{L},r,|\mathcal{L}|)}$ always equals to 1.

As described earlier, our objective in this subsection is to compute "What is the bound, $b_r$, on the number of lists where a fully unseen concept, $c_r$, will be found, given an acceptable precision rate, $\alpha$?". In order to compute this bound, we chose the smallest value $b_r$ satisfying

$$P_{found\_upto(\mathcal{L},r,b_r)} \geq \alpha.$$

In other words, the probability that a fully unseen concept, $c_r$, will be found upto $b_r$ sorted input lists is higher than an acceptable precision rate, $\alpha$. We note that NRA algorithm presented in the previous section corresponds to the case where an acceptable precision rate, $\alpha$, equals to 1, and thus $b_r$ would be $|\mathcal{L}|$.

#### 4.1.2 Computing Expected Score for Fully Unseen Object

Once we estimate the number of lists where any fully unseen concept will be found, we can compute the corresponding expected score. Given a current threshold vector $\mathbf{th} = [\tau_1, \tau_2, \cdots, \tau_u]$ and an original document vector $\mathbf{d} = [w_1, \cdots, w_u]$, we first define $W_d$ as a list of length $u$ which is sorted in descending order of $\tau_i \times w_i$ where $1 \leq i \leq u$. Then, the expected score of the fully unseen concept $c_r$ is bounded by

$$w'_{r,exp} \leq \sum_{1 \leq h \leq b_r} W_d[h] \ = (W_d[1] + W_d[2] + \cdots + W_d[b_h]),$$

where $W_d[h]$ corresponds to the $h$-th largest value in $W_d$.

Note that the expected score of any fully unseen concept, $c_r$, will equal to the possible best score in NRA algorithm, when the bound, $b_r$, on the number of input lists where $c_r$ will be found is same with $|\mathcal{L}|$. However, the sparsity of the keyword-concept matrix guarantees that the expected scores are less than the possible best scores in most cases.

#### 4.1.3 Computing Expected Score for Partially Unseen Object

Each list in an inverted index is sorted on weights rather than concept IDs, which results in a partially available (seen) concept-vector of a given concept, $c_r$, during the top-$k$ computation. Thus, we also need to estimate the expected scores of partially seen objects. Let $d_r$ be a partially seen document. Furthermore, let $KN_r$ be a set of positions in the concept-vector, $C_{-,r}$, whose weights have been seen before by the algorithm. Then, the expected score of partially seen concept $c_r$ is defined as follows:

– $If\ |KN_r| \geq b_r$, then

$$w'_{r,exp} = \sum_{i \in KN_r} (C_{i,r} \times w_i).$$

– Otherwise,

$$w'_{r,exp} \leq \sum_{i \in KN_r} (C_{i,r} \times w_i) + \sum_{|KN_r|+1 \leq h \leq b_r} W_d[h].$$

#### 4.1.4 The Algorithm

Figure 3 describes the pseudo-code for the proposed algorithm to efficiently compute the approximate top-$k$ concepts, $S_{k,\alpha}$, of a given document. The algorithm first initializes the set of the approximate top-$k$, $S_{k,\alpha}$, the cut off score, $min_k$, and the set of candidates, $Cnd$. The threshold vector, $\mathbf{th}$, is initially set to $[1, 1, \cdots, 1]$. Initially, the expected score of any fully unseen concept is computed, as described in Subsection 4.1.2 (line 1-5).

Generally, the threshold algorithms visit input lists in a round-robin manner. In case where the input lists have various lengths, however, this scheme can be

**Input**: an original document vector $\mathbf{d} = [w_1, w_2, \cdots, w_u]$
an acceptable precision rate $\alpha$, and a top $k$,
**Output**: an approximate top-$k$ concepts $S_{k,\alpha}$

1: $S_{k,\alpha} \leftarrow \emptyset$ // approximate top-k results
2: $min_k \leftarrow 0$ // cut off score
3: $Cnd \leftarrow \emptyset$ // set of candidates
4: $\mathbf{th} = [1, 1, \cdots, 1]$ // threshold vector
5: $unseen_{exp} \leftarrow$ compute the expected score of fully unseen concept
6: **while** $|Cnd| = k$ *and* $unseen_{exp} \leq min_k$ **do**
7: // read a new instance and update the threshold
8: $i \leftarrow$ `Select-Next-List`$(\mathcal{L})$
9: $\langle r, C_{i,r} \rangle \leftarrow$ read an instance from $L_i$
10: $w'_{r,wst} \leftarrow$ update the worst-score with $\langle r, C_{i,r} \rangle$
11: $Cnd \leftarrow$ update the candidate set with $\langle r, w'_{r,wst} \rangle$
12: $min_k \leftarrow$ choose the cut of score from $Cnd$
13: $\mathbf{th}[i] = C_{i,p}$
14: // remove unpromising concepts from the candidate set
15: **for each** $\langle p, w'_{p,wst} \rangle \in Cnd$ **do**
16: $w'_{p,exp} \leftarrow$ compute the expected score of partially seen concept
17: **if** $w'_{p,exp} < min_k$ **then**
18: $Cnd = Cnd - \langle p, w'_{p,wst} \rangle$
19: **end if**
20: **end for**
21: $unseen_{exp} \leftarrow$ compute the expected score of any fully unseen concept
22: **end while**
23: $S_{k,\alpha} \leftarrow Cnd$
24: **return** $S_{k,\alpha}$

Figure 3: Phase 1: Pseudo-code for computing the approximate top-$k$ Wikipedia concepts of a given document

inefficient, as resources are wasted for processing unpromising objects whose corresponding scores are relatively low, but are read early because they belong to short lists. To resolve this problem, in this paper, we visit input lists in a way to minimize the expected score of a fully unavailable concept. Intuitively, this enables the algorithm to stop the computation earlier by providing a higher cut off score, $min_k$. Given an original document vector, $\mathbf{d} = [w_1, w_2, \cdots, w_u]$, and a current threshold vector, $\mathbf{th} = [\tau_1, \tau_2, \cdots, \tau_u]$, to decide which input list will be read next time by the algorithm, we are looking for a list $L_i$ (line 8) such that:

$$\forall_{L_h \in \mathcal{L} - \{L_i\}} \ \ w_h \times \tau_h < w_i \times \tau_i.$$

The list satisfying the above condition guarantees to minimize the expected score of any unavailable concept, and thus provides the early stopping condition to the algorithm.

For a newly seen instance $\langle r, C_{i,r} \rangle$ in the list $L_i$, we compute the corresponding worst score, $w'_{r,wst}$, as described in NRA algorithm and update the candidate list with $\langle r, w'_{r,wst} \rangle$ (line 9-11). The cut off score, $min_k$, is selected such that $min_k$

**Input**: an original document vector $\mathbf{d} = [w_1, w_2, \cdots, w_u]$
an approximate top-$k$ concepts $S_{k,\alpha}$, identified in Phase 1
**Output**: a semantically reinterpreted document,
$\mathbf{d'} = [w'_1, w'_2, \cdots, w'_m]$, in the concept-space

1: $\mathbf{d'} = [0, 0, \cdots, 0]$ // a semantically reinterpreted vector
2: **for each** $\langle p, w'_{p,wst} \rangle \in S_{k,\alpha}$ **do**
3: $w'_{p,exp} \leftarrow$ compute the expected score of partially seen concept
4: $\mathbf{d'}[p] = w'_{p,exp}$
5: **end for**
6: **return** $\mathbf{d'}$

Figure 4: Phase 2: Pseudo-code for mapping an original document from the keyword-space into the concept-space

equals to the $k$-th highest value among the worst scores in the current candidate set, $Cnd$ (line 12). If there are less than $k$ elements in the current candidate set, $min_k$ is set to 0. Then, the threshold vector is updated (line 13).

Between line 15 and 20, we remove unpromising concepts from the candidate set, which will not be in the top-$k$ results with a high probability. For each concept, $c_p$, in the current candidate set, we compute the corresponding expected score, $w'_{p,exp}$, as described in this section. Note that each concept in the current candidate set corresponds to a partially seen concept. If the expected score, $w'_{p,exp}$, of the partially seen concept, $c_p$, is less than the cut off score, we remove the pair, $\langle p, w'_{p,wst} \rangle$, from the current candidate set, since this concept is not expected to be in the final top-$k$ results with a high probability (line 18). In line 21, the expected score of any fully unseen concept is computed. The top-$k$ computation stops only when the current candidate set contains $k$ elements and the expected scores of fully unseen concepts are likely to be less than the cut off score (line 6).

### 4.2 Phase 2: Mapping a Document from the Keyword-Space into the Concept-Space

Once the approximate top-$k$ concepts of a given document are identified, next step is to map an original document from the keyword-space into the concept-space. Figure 4 describes the pseudo-code for mapping an original document from the keyword-space into the concept-space using $S_{k,\alpha}$.

Initially, a semantically reinterpreted vector, $\mathbf{d'}$, is set to $[0, 0, \cdots, 0]$ (line 1). Since the algorithm in Figure 3 stops before scanning full input lists, the concept-vectors of the concepts in $S_{k,\alpha}$ are partially available. Therefore, for each concept in $S_{k,\alpha}$, we need to estimate the expected scores with the partially seen concept-vectors, as explained in this section (line 3). Then, the corresponding entries in the semantically reinterpreted vector, $\mathbf{d'}$, are updated with the estimated scores (line 4). Finally, the algorithm returns a semantically reinterpreted document vector, $\mathbf{d'}$ (line 6).

## 5 Evaluation

In this section, we describe the experiments we carried out to evaluate the proposed method. We first show that the proposed algorithm can efficiently and effectively compute the approximate top-$k$ Wikipedia concepts of a given document. In particular,

- we show that the proposed algorithm achieves a significant time gains against existing solutions in identifying the top-$k$ Wikipedia concepts that match a given document, and
- we verify the precision of top-$k$ Wikipedia concepts, $S_{k,\alpha}$, obtained by using the proposed algorithm.

Note that, in the next section, we also verify the usefulness of the proposed scheme on a real world problem. First we describe the experimental setup and then we discuss the results.

**Experimental setup:** In the experiments, we compute the top-$k$ results using following alternatives:

- case *IF*: Here, the exact top-$k$ concepts, $S_k$, are computed based on the inverted file based method.
- case *NRA*: In this case, the threshold based algorithm (NRA) is employed to compute the exact top-$k$ concepts, $S_k$, of a given document.
- case *Sparse_topk*: This is the proposed scheme in this paper. In this case, the approximate top-$k$ concepts, $S_{k,\alpha}$, are computed based on the technique presented in the paper.

For the *keyword-concept matrix*, $C$, we downloaded the Wikipedia dump which was released at September 2009. This data contains $\sim$ 4M articles. After discarding redirect pages and articles containing insufficient non-stopwords, we generated the $2,394,723 \times 812,926$ *keyword-concept matrix*, $C$. For document vectors, $\mathbf{d}$, we used the news articles collected from Wikinews [Wikinews]. This data contains 100 documents. The results reported in this section are the averages of all runs for 100 documents. We ran all experiments on a Linux machine with 3.1 GHz of CPU and 64 GB of memory.

### 5.1 Execution Times in Computing Approximate Top-k Wikipedia Concept

Before verifying the effectiveness of the proposed scheme, we first report the execution time results in identifying the top-$k$ concepts of Wikipedia that match

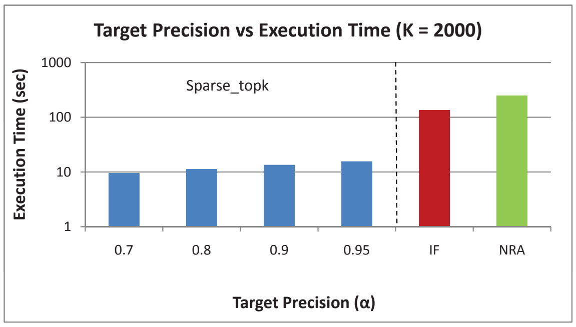


**Figure 5:** The execution times on various target precisions ($\alpha$)

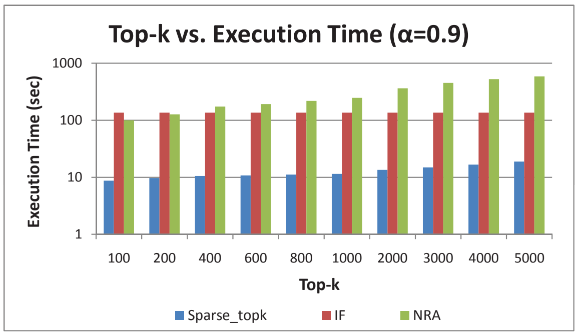


**Figure 6:** The execution times on various top-$k$.

a given document. First of all, Figures 5 and 6 show the execution times for varying target precisions, $\alpha$, and $k$ respectively. Note that the proposed approach, *Sparse_topk*, can vary the precision rates of the top-$k$ results depending on the application needs, while the precision of *IF* and *NRA* schemes always equals to 1. Key observations based on Figure 5 and 6 can be summarized as follows:

- The first thing to note in the figures is that in most cases, *NRA*-based scheme is even worse than the inverted file based method, when input lists are extremely sparse, such as a *keyword-concept matrix*, $C$. This is because with *NRA*, the early stopping condition is overly pessimistic, which results in the expensive overhead of maintaining candidate lists and computing the possible best- and worst- scores for each candidate object. These results verify the need for the approach proposed in this paper, when input lists are sparse.
- In Figures 5, the target precision is varied from 0.7 to 0.95 and $k$ is set to 2000. As can be seen in this figure, the processing time of the proposed scheme decreases, as the target precision rate decreases. This is because lower target precision rates result in lower expected scores of fully or partially

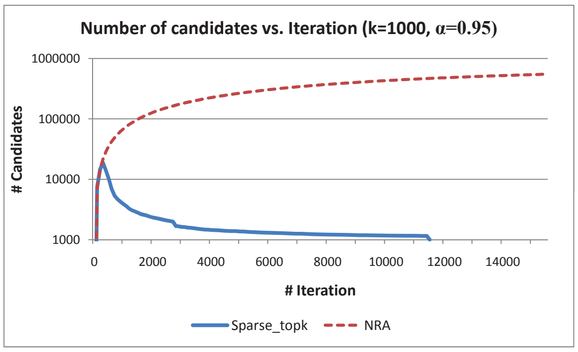


Figure 7: The number of candidates during the top-$k$ computation using the proposed approach, $Sparse\_topk$, and the existing scheme, $NRA$

unseen objects, which leads to early stopping for the top-$k$ computation. Furthermore, the effect of $\alpha$ on the execution time of the proposed approach is very slight. This indicates that the proposed algorithm in the paper trade-offs between precision and execution time effectively.

– In Figure 5, depending on the permissible precision rate, the proposed algorithm significantly outperforms the naive approaches. The processing time gains against naive solutions can be 10X, with precision rates of up to 95%.

– Figure 6 plots the case where the target precision, $\alpha$, is set to 0.9 and $k$ is varied from 100 to 5000. As expected, the processing times of the proposed scheme and $NRA$ increase as the number of top-$k$ results increases. At all $k$ values, the proposed scheme, $Sparse\_topk$, significantly outperforms the existing solutions, $IF$ and $NRA$. The effect of $k$ on the execution time of $NRA$ scheme is very high. On the other hand, the proposed approach in this paper is slightly affected by $k$. This shows that the proposed scheme is highly effective on the number of top-$k$ results.

Figure 7 tracks the number of candidates during the top-$k$ computation, using different schemes, $NRA$ and $Sparse\_topk$. As the figure shows, at the very beginning of run, the number of candidates significantly increases in both algorithms. However, as the number of iterations increases, the number of candidates in the proposed scheme, $Sparse\_topk$, significantly drops and becomes stable, while the number of candidates in $NRA$ still increases. This is because the proposed algorithm can more effectively prune unpromising candidates, which will not be in the top-$k$ answers, than $NRA$ scheme. This shows that the approximation based algorithm, such as $Sparse\_topk$, is more suitable for the top-$k$ processing with sparse input lists than the exact-based algorithm, such as $NRA$.

Table 1: Observed precision and average concept weight error for varying target precision ($\alpha$); $k = 2000$

| Target precision | 0.7 | 0.8 | 0.9 | 0.95 |
|---|---|---|---|---|
| Observed precision | 0.871 | 0.892 | 0.921 | 0.963 |
| Error | 0.0123 | 0.0107 | 0.0083 | 0.0070 |

### 5.2 Verifying the Approximate Top-k Results

In this subsection, we verify that the proposed algorithm in the paper can approximately identify top-$k$ concepts and correctly estimate corresponding concept weights. Table 1 shows the observed precision and average concept weight error of the top-$k$ results obtained by using the proposed algorithm in this paper. The observed precision, $P$, is defined as

$$Observed\_Precision = \frac{|S_{k,\alpha} \bigcap S_k|}{|S_{k,\alpha}|}.$$

In the experiments, $k$ is set to 2000 and target precision level, $\alpha$, is varied from 0.7 to 0.95. As can be seen in the table, the observed precisions are always higher than the target precisions.

In Table 1, the average concept weight error is defined as

$$\frac{1}{|S_{k,\alpha}|} \times \sum_{c_r \in S_{k,\alpha}} (|w_r^{exact} - w_r^{est}|),$$

where

- $w_r^{exact}$ is the concept weight based on exact algorithms, such as *IF* and *NRA*, and
- $w_r^{est}$ is the estimated one computed by the algorithm proposed in this paper.

As shown in the table, the average concept weight errors decrease as the target precisions increase. This is expected because the higher target precision results in more scanning of input lists during the top-$k$ computation.

In order to study the degree of alignment between ranks, we measured the Spearman's rank correlation coefficient [Devore 1999]. Given $n$-pairs of ranks $(x_1, y_1), (x_2, y_2) \cdots (x_n, y_n)$, Spearman's rank correlation coefficient, $\rho$, is defined as

$$\rho = 1 - \frac{6 \times \sum_{1 \leq i \leq k} (x_i - y_i)^2}{n \times (n^2 - 1)},$$

and measures how closely related $x$ observations are with the $y$ observations. The correlation value of 0.0 means that two ranks are not linearly related, while

**Table 2:** Target precision vs. Spearman's rank correlation coefficient

| Target precision ($\alpha$) | 0.7 | 0.8 | 0.9 | 0.95 |
|---|---|---|---|---|
| k = 1000 | 0.667 | 0.774 | 0.919 | 0.939 |
| k = 2000 | 0.743 | 0.835 | 0.936 | 0.952 |
| k = 3000 | 0.794 | 0.860 | 0.941 | 0.954 |
| k = 4000 | 0.834 | 0.881 | 0.943 | 0.955 |
| k = 5000 | 0.854 | 0.894 | 0.943 | 0.957 |

the value of 1.0 indicates that there is a perfect linear relationship between the two ranks.

Table 2 shows the Spearman's rank correlation coefficients between ranks obtained by the naive schemes (such as *IF* and *NRA*) and by the proposed approach. In the experiments, $k$ is varied from 1000 to 5000 and target precision are varied from 0.7 to 0.95. In Table 2, as the target precision increases, the correlation coefficient increases. The correlation coefficient values of $> 0.9$ can be achieved, when the target precision set to higher than 0.9. Considering that the perfect correlation agreement between two ranks corresponds to the coefficient value of 1.0, this correlation is quite high. Furthermore, Table 2 indicates that the correlation coefficient values align well to the target precisions, which verifies that the expected score based stopping condition can be effectively used for the top-$k$ computation with sparse lists.

## 6 Application: Topic Boundary Detection in a News Video

In the previous experiments, we evaluated the efficiency of the proposed algorithm. In this section, we aim to explore the usefulness of the proposed semantic interpreter. In particular, we exploit the proposed scheme to improve the effectiveness in finding topic boundaries in a news video.

### 6.1 Motivation

The use of the Web to track real world events is continuously growing, as more individuals start relying on the Web as their primary information source. Similarly, traditional media outlets, such as print and broadcast, are trying to reach consumers through the Web. In order to reach consumers through the Web, the number of news sites on the Web is also continuously increasing. Yet, due to the continuously increasing sizes and complexities of data shared by these sites, it is also becoming more and more difficult for users to navigate through such sites to access the relevant piece of information. While the navigation problem is a challenge for all types of data shared by these news sites, the problem is most evident with news video data. In these days, it is common that most of traditional news providers share news videos (which were already broadcasted on TV) on their Web sites. Naturally, a news video consists of a series of stories that are independent of each other. Since many of stories in a news video are not clustered based on topical similarity, the whole contents themselves are not useful in guiding users for accessing the relevant piece of information. Thus, it is important to effectively detect topic boundaries in a news video. Knowledge about these topic boundaries then can be an effective asset for indexing, retrieval, and presentation of appropriate information units to users.

### 6.2 Topic Boundary Detection Using Wikipedia-based Semantic Interpreter

Topic boundary detection is an important problem that has implications in various application domains, including text documents [Choi 2000], discussion board [Kim et al. 2005], blogs [Qi et al. 2006], and videos [Boreczky et al. 1996]. Especially, in video, shot (or segment) boundaries are usually detected by comparing various features (e.g. objects, color histograms) of consecutive frames or neighborhoods of frames to identify major content changes [Boreczky et al. 1996]. However, for a news video consisting of independent stories, the challenge is not to identify shot (or segment) boundaries, but to discover semantically coherent information units. Thus, in this section, we introduce an effective mechanism to detect boundaries of a new video based on topical similarity. In particular, the presented scheme *reinterprets the contents of news videos with the background knowledge obtained from Wikipedia, and then identifies topic boundaries by leveraging these reinterpreted contents.*

Figure 8 provides an overview of identifying topic boundary in a news video by leveraging Wikipedia-based semantic interpreter:

- Given a new video, first we extract a stream of closed caption data and create sentence sequences.
- We compute the semantically reinterpreted vectors by mapping the original sentence sequences from the keyword-space into the Wikipedia concept-space. Then, we identify topic boundaries of a given news video by relying on these semantically reinterpreted vectors.

We now explain and describe each of these steps in detail.

#### 6.2.1 Creating Sentence Sequences using Closed-Caption Stream

It is common that most of news videos contain closed-caption data, which are texts displayed on a screen to provide additional information. In this paper, we treat closed caption data as a stream of $n$ sentences, $s_1, s_2, \cdots, s_n$. Then, the

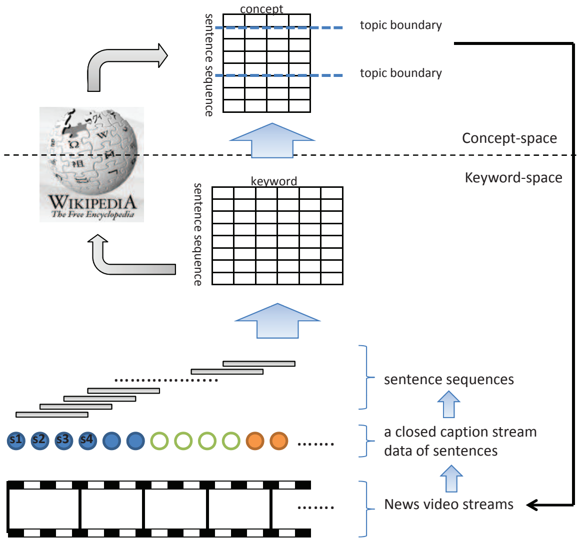


Figure 8: An overview of identifying topic boundaries in a news video using the proposed Wikipedia-based semantic interpreter

topic boundary detection problem in a news video can be defined as searching for the special sentences, which correspond to the entry points that introduce new stories. Once the topic boundary detection based on closed-caption data is completed, the physical boundaries can be determined by the shots which are associated with these entry sentences (Figure 8).

In order to create sentence sequences, we use the sliding window technique that has been extensively used in diverse applications. The sliding window can be used at different levels, such as paragraph, sentence, and keyword. As we treat closed-caption data of a news video as a stream of sentences, the sentence-level sliding window has to be applied to the stream. Let $p$ be the size of the sliding window. Given a closed-caption stream containing $n$ sentences, a sliding window of size $p$ is passed over the stream of $n$ sentences. This process produces $(n-p)$ sentence sequences such that the $r$-th sentence sequence, $s_{\langle r,r+p-1\rangle}$, is formed by concatenating $p$ consecutive sentences, $s_r, s_{r+1}, \cdots, s_{r+p-1}$. Without loss of generality, we assume that each sentence sequence vector is normalized into a unit length.

#### 6.2.2 Mapping Closed Caption Data into the Concept-Space with the Approximate Top-k Wikipedia Concepts

Most sentences in a stream of closed-caption data are too short to be meaningful by themselves. In addition, the varying sizes of sentences in the stream make it difficult to effectively capture the importance of the keyword in terms of weights associated to it. Therefore, in this section, we leverage Wikipedia-based semantic interpreter to reinterpret the original closed-caption text and then identify topic boundaries by relying on these reinterpreted text.

Given a $r$-th sentence sequence vector, $\mathbf{s}_{\langle r,r+p-1\rangle}$, let $S_{k,\alpha}$ be a set of $k$ Wikipedia concept identified by using the algorithm presented in Figure 3. Then, a corresponding semantically reinterpreted vector, $\mathbf{s'}_{\langle r,r+p-1\rangle} = [w'_1, w'_2, \cdots, w'_m]$ with the Wikipedia concepts in $S_{k,\alpha}$ is computed as

– $if$ $\mathbf{c_r} \in S_{k,\alpha}$,

$$w'_r = \sum_{1 \leq i \leq l} w_i \times C_{i,r}$$

– otherwise, $w'_r = 0$.

Once we map the original sentence sequence vectors from the keyword-space into the Wikipedia concept-space, we then identify topic boundaries of a given news video by relying on these semantically reinterpreted vectors. We note that semantically reinterpreted vectors with Wikipedia concepts can be effectively used with existing topic boundary detection algorithms, such as [Allan et al. 2003, Choi 2000, Qi et al. 2006].

### 6.3 Evaluating the effectiveness of topic boundary detection

In this subsection, we describe the experiments we carried out to show that the proposed Wikipedia-based semantic interpreter with the approximate top-$k$ concepts significantly improves the effectiveness of topic boundary detection task in a new video.

**Experimental Setup:** We evaluated the proposed approach with real data set: We crawled 312 CNN news transcripts. The ground truth of the topic boundaries in these data set is manually identified. For the *keyword-concept matrix*, $C$, we downloaded the same Wikipedia dump used in the previous section. In the experiments, we report results obtained by using following alternatives:

– case $\mathtt{TS_{keyword}}$: in this case, we determine topic boundaries by relying on the algorithm presented in [Allan et al. 2003] without reinterpreting the original contents with Wikipedia concepts.

– case $\mathtt{TS_{k-concept}}$: in this case, as described in this section, we first reinterpret the contents of the news transcripts with the approximate top-$k$ Wikipedia concepts and identify topic boundaries by using the same algorithm [Allan et al. 2003].

In order to verify the effectiveness of the proposed algorithm, we compute the precision and recall.

**Table 3:** Precision and recall of $\mathtt{TS_{k-concept}}$ on varying $\alpha$ ($k = 5000$)

| | $\alpha$ | | | | Exact |
|---|---|---|---|---|---|
| | 0.7 | 0.8 | 0.9 | 0.95 | Top-$k$ |
| Precision | 0.733 | 0.751 | 0.773 | 0.775 | 0.780 |
| Recall | 0.685 | 0.704 | 0.727 | 0.730 | 0.730 |

**Table 4:** The performance comparison of $\mathtt{TS}_{k-concept}$ and $\mathtt{TS}_{keyword}$ ($\alpha = 0.9$)

| | $\mathtt{TS_{k-concept}}$ | | | | | $\mathtt{TS_{keyword}}$ |
|---|---|---|---|---|---|---|
| | $k = 1000$ | $k = 2000$ | $k = 3000$ | $k = 4000$ | $k = 5000$ | |
| Precision | 0.697 | 0.727 | 0.764 | 0.773 | 0.773 | 0.624 |
| Recall | 0.703 | 0.716 | 0.724 | 0.727 | 0.727 | 0.645 |

**Result and discussion :** Table 3 shows the performance comparison, when $\alpha$ varies from 0.7 to 0.95. In the experiments, the number of Wikipedia concepts ($k$) is set to 5000. As expected, the precision and recall values increase, as $\alpha$ increases. The best performance is achieved when the exact top-$k$ concepts, $S_k$, are used to reinterpret the original closed-caption text. However, as evaluated in the previous section, this scheme performs very poorly in terms of the execution time and thus is not a viable solution.

Table 4 shows the performance comparison of $\mathtt{TS_{k-concept}}$ and $\mathtt{TS_{keyword}}$. In the experiments, we set $\alpha = 0.9$ and $k$ varies from 1000 to 5000. Key observations based on Table 4 can be summarized as follows: As shown in the table, $\mathtt{TS_{k-concept}}$ significantly outperforms $\mathtt{TS_{keyword}}$ in precision and recall, which verifies that the proposed Wikipedia-based semantic interpreter with the approximate top-$k$ concepts can indeed boost the effectiveness in finding topic boundaries in a news video. Moreover, $\mathtt{TS_{k-concept}}$ observes significant improvements in precision and recall, as the number of Wikipedia concepts used to reinterpret the original contents increases from 1000 to 3000. Beyond these, the precision and recall rates become more stable, implying that insignificant concepts do not contribute to the computation of topic boundaries in a news video, because their corresponding weights in concept vectors are close to 0. These results imply that the top-$k$ based Wikipedia semantic interpreter can be effectively used in real world applications, instead of the bag of keyword model.

## 7 Related Work

Recently, there has been growing research on exploiting Wikipedia, the largest encyclopedia, for diverse applications. ESA (Explicit Semantic Analysis), which is a Wikipedia-based semantic interpreter, represents the meaning of a text segment (such as sentences, paragraphs, and documents) in the concept-space that corresponds to Wikipedia articles [Gabrilovich et al. 2009]. The concept-based representation enables extraction of semantic information between text segments that cannot be obtained with the bag of words model. Evaluation results in [Gabrilovich et al. 2009] show that using ESA, the significant performance gains against the state-of-art methods can be achieved in various applications, such as text categorization and computing the semantic similarity between text segments. *Hu et al.* [Hu et al. 2009] proposes a general framework that leverages Wikipedia to enhance clustering performance. The method in [Hu et al. 2009] maps input documents into the concept-space of Wikipedia, and then computes the similarity scores between documents in the concept-space. A drawback of using Wikipedia as a semantic interpreter, however, is that the sheer size of the concept-space derived from Wikipedia makes it hard for the efficient reinterpretation of texts. To resolve this problem, most existing techniques rely on limiting the number of distinct keywords that each concept can contain [Gabrilovich et al. 2009, Hu et al. 2009]. However, this can cause an input text to be incorrectly mapped into the concept-space. However, unlike [Gabrilovich et al. 2009, Hu et al. 2009], the algorithm presented in this paper maps a text segment into the concept-space by exploiting the entire information deduced from Wikipedia. The ODP [Open Directory Project], which contains a real world semantic hierarchy generated by humans, has been used successfully in the various areas. *Zhu and Dreher* [Zhu and Dreher 2009] proposed a method to leverage the ODP data to boost precision of Web search.

The most well-known methods for ranked query processing are the threshold-based algorithms [Fagin 1999, Fagin et al. 2003, Nepal and Ramakrishna 1999, Guntzer et al. 2000]. The threshold-based algorithm assumes that given sorted-lists, each object has a single score in each list. More importantly, except for some very recent efforts, such as [Kim and Candan 2009], it assumes that the underlying scoring function, which combines each object's score in each list, is monotone. In the literature, many variants of TA algorithm have been proposed, including [Chakrabarti et al. 2006, Cheng et al. 2007, Ilyas et al. 2003, Tsaparas et al. 2003]. With the development of the web, there have been studies to determine the ranking of objects based on the score of text data which are related with the specific objects [Chakrabarti et al. 2006, Cheng et al. 2007]. [Chakrabarti et al. 2006] maintains the upper- and lower-bound scores of partially seen objects and uses these bounds to decide the stopping condition. An approximate-based algorithm relies on the probabilistic confidence to terminate earlier than the original threshold-based algorithms [Theobald et al. 2004, Arai et al. 2007]. *Bansal et al.* [Bansal et al. 2008] introduced an approximation algorithm to aggregate the score of terms in the presence of hierarchies. In their approach, each term score is accumulated into the more commonly used term, which is located at a higher level of the hierarchies, and a probabilistic framework is used for early stopping. The algorithm presented in this paper is also based on an approximation method. However, our method is different with [Theobald et al. 2004, Arai et al. 2007] in that it relies on the simplified data distribution of input lists to achieve efficiency in the top-$k$ computation with sparse input lists.

## 8 Conclusion

In this paper, we proposed a semantic interpreter for efficiently reinterpreting original documents based on the concepts of Wikipedia. Our proposed approach enables to efficiently identify the most significant $k$-concepts in Wikipedia for a given document and leverage these concepts to semantically reinterpret an original document by mapping it from keyword-space to the concept-space. Experimental results show that the proposed technique significantly improves efficiency of semantic reinterpretation without causing significant reduction in precision. We also show that the top-$k$ based Wikipedia semantic interpreter boosts the effectiveness in topic boundary detection in a news video.

## References

[Allan et al. 2003] Allan, J., Wade, C., Bolivar, A.: "Retrieval and novelty detection at the sentence level"; Proceedings of the 26th ACM SIGIR International Conference. 2003.

[Arai et al. 2007] Arai, B., Das, G., Gunopulos, D., Koudas, N.: "Anytime measures for top-k algorithms"; Proceedings of the 33th International Conference on Very Large Data Bases. 2007.

[Bansal et al. 2008] Bansal, N., Guha, S., Koudas, N.: "Ad-hoc aggregations of ranked lists in the presence of hierarchies"; Proceedings of the 28th ACM SIGMOD International Conference. 2008.

[Blei et al. 2003] Blei, D., Ng, A., Jordan, M.: "Latent dirichlet allocation"; Journal of Machine Learning Research 3, (Jan 2003), 993-1022.

[Boreczky et al. 1996] Boreczky, J.S., Rowe, L.A.: "Comparison of video shot boundary detection techniques"; Proceedings of Storage and Retrieval for Image and Video Databases. 1996.

[Carmel et al. 2009] Carmel, D., Roitman, H., Zwerdling, N.: "Enhancing cluster labeling using wikipedia"; Proceedings of the 32nd International ACM SIGIR Conference. 2009.

[Chakrabarti et al. 2006] Chakrabarti, K., Ganti, V., Han, J., Xin, D.: "Ranking objects by exploiting relationships: computing top-K over aggregation"; Proceedings of the 26th ACM SIGMOD International Conference. 2006.

[Cheng et al. 2007] Cheng, T., Yan, X., Chang, K.C.C: "EntityRank: searching entities directly and holistically"; Proceedings of the 33th International Conference on Very Large Data Bases. 2007.

[Choi 2000] Choi, B.: "Advances in independent linear text segmentation"; Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference. 2000.

[Croft and Harper 1979] Croft, W.B., Harper, D.J. : "Using probabilistic models of document retrieval without relevance information"; Journal of Documentation 35, 4 (1979), 285-295.

[Deerwester et al. 1990] Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: "Indexing by latent semantic analysis"; Journal of the American Society for Information Science 41, 6 (1990), 391-407.

[Devore 1999] Devore, J.L.: "Probability and statistics for engineering and the science"; International Thomson Publishing Company, (1999).

[Fagin 1999] Fagin, R.: "Combining fuzzy information from multiple systems"; Journal of Computer and System Sciences, 58, 1, (Feb 1999), 83-99.

[Fagin et al. 2003] Fagin, R., Lotem, A., Naor, M.: "Optimal aggregation algorithms for middleware"; Journal of Computer and System Sciences, 66, 4, (2003), 614-656.

[Gabrilovich et al. 2007] Gabrilovich, E., Markovitch, S.: "Computing semantic relatedness using Wikipedia based explicit semantic analysis"; Proceedings of the 20th International Joint Conference on Artificial Intelligence. 2007.

[Gabrilovich et al. 2009] Gabrilovich, E., Markovitch, S.: "Wikipedia-based semantic interpretation for natural language processing"; Journal of Artificial Intelligence Research 34, 1 (Jan 2009), 443-498.

[Guntzer et al. 2000] Guntzer, U., Balke, W., Kiebling, W.: "Optimal aggregation algorithms for middleware"; Proceedings of the 26th International Conference on Very Large Data Bases. 2000.

[Hu et al. 2009] Hu, X., Zhang, X., Lu, C., Park, E.K, Zhou, X.: "Exploiting Wikipedia as external knowledge for document clustering"; Proceedings of the 15th ACM SIGKDD International Conference. 2009.

[Ilyas et al. 2003] Ilyas, I.F., Aref, W.G., Elmagarmid, A.K.: "Supporting top-K join queries in relational databases"; Proceedings of the 29th International Conference on Very Large Data Bases. 2003.

[Kim and Candan 2009] Kim, J. W., Candan, K.S.: "Skip-and-Prune: Cosine-based top-K query processing for efficient context-sensitive document retrieval"; Proceedings of the 29th ACM SIGMOD International Conference. 2009.

[Kim et al. 2005] Kim, J.W., Candan, K.S., Donderler, M.E.: "Topic segmentation of message hierarchies for indexing and navigation support"; Proceedings of International World Wide Web Conferences. 2005.

[Kim et al. 2010] Kim, J.W., Kashyap, A., Li, D., Bhamidipati, S.: "Efficient Wikipedia-based semantic interpreter by exploiting the top-k processing"; Proceedings of the 19th Conference on Information and Knowledge Management. 2010.

[Lee and Seung 2000] Lee, D.D, Seung, H.S.: "Algorithms for non-negative matrix factorization"; Proceedings of Advances in Neural Information Processing Systems. 2000, 556-562.

[Miller et al. 1990] Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Mille, K.J.: "Introduction to WordNet: An on-line lexical database"; International Journal of Lexicography 3, 4 (1990), 235-244.

[Milne et al. 2008] Milne, D., Witten, I.H.: "Learning to link with wikipedia"; Proceedings of the 17th ACM Conference on Information and Knowledge Management. 2008.

[Nepal and Ramakrishna 1999] Nepal, S., Ramakrishna, M.V.: "Query processing issues in image(multimedia) databases"; Proceedings of the 15th International Conference on Data Engineering. 1999.

[Ogawa et al. 1991] Ogawa, Y., Morita, T., Kobayashi, K.: "A fuzzy document retrieval system using the keyword connection matrix and a learning method"; Fuzzy Sets and Systems 39, 2 (Jan 1991), 163-179.

[Open Directory Project] Open Directory Project, http://www.dmoz.org.

[Qi et al. 2006] Qi, Y., Candan, K.S.: "CUTS: CUrvature-based development pattern analysis and segmentation for blogs and other text streams"; Proceedings of the ACM Conference on Hypertext and Hypermedia. 2006.

[Ramanathan and Kapoor 2008] Ramanathan, K., Kapoor, K.: "Creating user profiles using wikipedia"; Proceedings of the 28th International Conference on Conceptual Modeling. 2008.

[Salton et al. 1975] Salton, G., Wong, A., Yang, C.S. : "A vector space model for automatic indexing"; Comm. ACM 18, 11 (Nov 1975), 613-620.

[Salton et al. 1983] Salton, G., Fox, E.A., Wu, H. : "Extended boolean information retrieval"; Comm. ACM 26, 11 (Nov 1983), 1022-1036.

[Schonhofen 2006] Schonhofen, P.: "Identifying document topics using the wikipedia category network"; Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence. 2006.

[Theobald et al. 2004] Theobald, M., Weikum, G., Schenkel, R.: "Top-k query evaluation with probabilistic guarantees"; Proceedings of the 30th International Conference on Very Large Data Bases. 2004.

[Tsaparas et al. 2003] Tsaparas, P., Koudas, N., Palpanas, T.: "Ranked join indices"; Proceedings of the 19th International Conference on Data Engineering. 2003.

[Wang and Domeniconi 2008] Wang, P., Domeniconi, C.: "Building semantic kernels for text classification using wikipedia"; Proceedings of the 14th ACM SIGKDD International Conference. 2008.

[Wikinews] Wikinews, the free news source. http://en.wikinews.org.

[Wikipedia] Wikipedia, the free encyclopedia. http://www.wikipedia.org/.

[Yahoo!] Yahoo! Directory. http://dir.yahoo.com.

[Zhu and Dreher 2009] Zhu, D., Dreher, H.,: "Discovering semantic aspects of socially constructed knowledge hierarchy to boost the relevance of Web searching"; Journal of Universal Computer Science, 15, 8, (Feb 2009), 1685 - 1710.