Finding Related Forum Posts through Content Similarity over Intention-based Segmentation

Dimitra Papadimitriou (*University of Trento*), Georgia Koutrika (*ATHENA Research Center*), Yannis Velegrakis (*University of Trento*) and John Mylopoulos (*University of Ottawa*)

Abstract—We study the problem of finding related forum posts to a post at hand. In contrast to traditional approaches for finding related documents that perform content comparisons across the content of the posts as a whole, we consider each post as a set of segments, each written with a different goal in mind. We advocate that the relatedness between two posts should be based on the similarity of their respective segments that are intended for the same goal, i.e., are conveying the same intention. This means that it is possible for the same terms to weigh differently in the relatedness score depending on the intention of the segment in which they are found. We have developed a segmentation method that by monitoring a number of text features can identify the parts of a post where significant jumps occur indicating a point where a segmentation should take place. The generated segments of all the posts are clustered to form intention clusters and then similarities across the posts are calculated through similarities across segments with the same intention. We experimentally illustrate the effectiveness and efficiency of our segmentation method and our overall approach of finding related forum posts.

1 Introduction

Forums in the context of online user communities offer users the ability to seek solutions and make decisions regarding diverse problems by exploiting other users' experience. They also offer businesses the ability to connect and support their customer base. Existing forums range from domains like health (e.g., *Medhelp*), law (e.g.,*ExpertLaw*) and technology (e.g.,*HP support forum*). The organization of the forum posts into categories is a feature that helps users to identify more easily those posts related to a topic. However, since browsing a very large number of posts is frustrating and time-consuming, most forum sites offer keyword search capabilities. Yet, keyword search may not result in a complete set of related posts since the selection of the right keywords is not always straightforward. We believe that to better support users, an important functionality is to provide them with a number of pertinent posts once they have identified a post of interest, without having to formulate complex queries, or perform complicated, long browsing. Work towards this direction has been done for questions in Q&A archives [1], [2], [3] but not for richer-content posts.

With such a functionality, a user reading a post on a technical problem in a customer care site could find related forum posts that describe similar situations and alternative solutions. Someone with a health problem reading a medical forum post where a user is describing similar symptoms could find additional related forum posts that contain different opinions, explanations, and various courses of actions.

In this work, *we deal with the problem of finding forum posts related to a post at hand*. Relatedness has traditionally been translated into content similarity [4], [5]. Content similarity computed directly across forum posts is, unfortunately, not very effective in this case because searches are done under specific thematic categories, e.g., *printers, or hotels in New York*, in which the content of all the posts is anyway similar.

We advocate that when we are measuring the relatedness of two forum posts, treating them as composite objects instead of monolithic entities can lead to more effective comparisons. Indeed, a forum post consists of parts, each serving a different goal, i.e., expressing a different message to the reader through the text. For instance, a part may serve to describe a problem that the author has, another to provide background information in order to put the reader into context, a third to express a desire, and a fourth to reach a conclusion. We refer to these parts of a forum post as *segments*.

The relatedness of two posts can then be based on a comparison across segments that serve the same goal, i.e., they are intended for the same purpose, instead of a comparison of the two posts as wholes. The comparison among text segments with the same intention can be performed by Information Retrieval methods, such as one of the many TF/IDF or BM25 variants [6] or language-model based methods [2], or using topics generated by topic modeling techniques like LDA [3], [7], paraphrasing techniques [8] or even auxiliary external services [9], with the latter been used especially for documents with short and poor content, e.g., tweets. However in our approach, given the different intentions of the forum post author, the meaning and importance of a term is estimated based on the segment in which the term is found. Different weights for the same terms have been used across different thematic forum categories or domains [10], [11]. To the best of our knowledge, it is the first time that a weighting scheme may assign different weights to a term in posts of the same thematic category; or even within the same post.

Identifying the segments in a forum post is a challenging task. Forum posts are typically one or two paragraph long, with complete sentences. They do not follow the abbreviated style used in microblogs, but at the same time,

Y. Velegrakis has been partially supported by the COST action IC1302 (Keystone) and a "Jean d'Alembert" fellowship of the University of Paris-Saclay. D. Papadimitriou and J. Mylopoulos have been partially supported by the ERC Advanced Investigator grant N.267856 (Lucretius)

Doc A
I have an HP system with a RAID 0 controller and 4 disks in form of a JBOD. I would like to install Hadoop with a replication 4 HDFS and only 320GB of disk space used from every disc. Do you know whether it would perform ok or whether the partial use of the disk would degrade performance. Friends have downloaded the Cloudera distribution but it didn't work. It stopped since the web site was suggesting to have 1TB disks. I am asking because I do not want to install Linux to find that my HW configuration is not right.

Doc B
My boss gave me yesterday an HP Pavilion computer with Intel Matrix Storage System, a 320GB drive and Linux pre-installed. I am thinking to add an extra drive using a RAID 0 or 1. Can I do it without having to rebuild the entire system? I have already looked at the HP official web site for how to use a JBOD. But I have not found anything related to it.

Doc C
Extra RAID drives seem to be the solution to my problem but does adding RAID drives requires a reformat and rebuild of the system to improve performance?

Doc D
My HP Pavilion stops working after 15 min of activity. I called our technical department but no luck. Despite the many calls, I did not manage to find a person with adequate knowledge to find out what is wrong. All they said is bring it to up and we will see, which frustrated me. At the end I had the brilliant idea to move it to a cooler place and voila. No more problems.

Fig. 1. Four posts from a technical forum

since they are intended for interactive discussions, they are not verbose and they lack the structural constructs (e.g., sections) typically used in full-text documents to identify thematic units. Furthermore, since they are driven by the common needs of forum participants, they draw heavily their content from a common vocabulary (that depends on the nature/topic of the forum), which means that topic variation, i.e., the used vocabulary, is not a very distinctive factor for the identification of the segments. To deal with this limitation we resort to text features (characteristics) whose variation can identify a passage from one segment to another. We made this choice after realizing that the style, tone, brevity, verb tense and other grammatical characteristics can may serve as indicators of a change in the message that the author is trying to communicate. We refer to these characteristics as *features* and use the term *communication means* (CM for short) to refer to groups of such features. The idea of using communication means for capturing the intention of a segment (or intended message) is analogous to the idea of using keywords to represent a topic. Similar to the way that a variation in a weighted vector of words signals a change in the topic [12], [13], a variation in a vector of text features signals a change in the intended message.

We have developed a framework for finding related forum posts that is based on the above idea. By exploiting the communication means, the system identifies the different segments within each forum post and splits the forum post into these segments. Segments serving the same intention are identified and grouped together. Given a forum post at hand, its segments are identified and the matching score of each segment with other forum posts' segments that have the same intention is computed. To compute the segment scores, the used term weighting scheme is adjusted to consider the intention of the segment where the term is found. The segments with the highest individual scores are selected and their scores are combined to compute a score that indicates how the forum post at hand is believed to be related to other existing forum posts, and based on this score we select the top-k posts.

Note that methods that enrich text content exploiting terms, synonyms, latent topics etc. from knowledge bases such as Wikipedia, WordNet, or web search engines [9], [14], or concept graphs and complex language models [4] can still be employed in our method for the comparison among segments. We are not suggesting a new text comparison method, but we propose a method that makes the existing comparison methods more accurate.

The contributions of this paper are as follows:

- We formally introduce a novel method for finding related forum posts that treats each post as a set of segments and computes content similarity only across segments of the same intention.
- We provide a complete methodology for segment identification and for grouping the derived segments into intention clusters that exploit the text features' variation.
- We present extensive experiments with real users that confirm the existence of such segments in forum posts of different domains, and verify the effectiveness of the individual steps and decisions of our methodology, including the border selection mechanisms, the selection of features, and last but not least the functions and weights for capturing text feature variation.
- We describe a fully unsupervised multi-segment ranking technique that provides the top-k forum posts related to a reference post by considering segments with similar intentions and using content similarities within each cluster to derive an overall score between each forum post and the reference post.
- We evaluate the effectiveness of the overall approach on the recommendation of related forum posts using ratings and feedback by users in 3 different domains.

In what follows, we first present a motivating example (Sec. 2), and then introduce the problem (Sec. 3). In the sequel, we provide a brief overview of the whole approach (Sec. 4) and then we describe the individual steps. First, we describe our segmentation method (Sec. 5), and a way to identify segments of the same intention (Sec. 6). Then, we present our segment-based related forum post finding technique (Sec. 7) and position it in the context of the related work (Sec. 8). Finally, we conclude with a detailed experimental evaluation (Sec. 9).

2 Motivation

Consider a user that identified in a forum site the post A of Fig. 1 as being of interest, and would like the system to show also other posts that are of interest. Forum post B seems to be such a post since both A and B have a number of important keywords in common (e.g., RAID 0, 320GB, disk drive, HP). However, the fundamental question asked in A is whether performance will degrade ("Do you know ... performance"), while in B it is about adding an extra drive ("I am thinking to add ... system?"). Many of the keywords that the two forum posts have in common do not appear in these two parts. For instance, the keyword HP appears in the first part of A ("I have ... disc") and of B ("My boss ... pre-installed"),

that are both informative parts intended to communicate to the reader the general context of the author's situation. The keyword HP also appears in the last part of B ("I have looked ... related to it") that simply informs the reader of a related issue. None of these parts is about the main request of the respective forum post. A similar informative role has the keyword RAID at the beginning of A, while in B it has a significant role in the part intended to communicate the author's main request. Thus, despite the content similarities between A and B, B may not be of much interest to the user. On the other hand, A and C seem to have little content overlap, but the user may be interested in reading also C, since the main problem discussed in it is similar to the one discussed in A. Finally, D is very different from A in every aspect, consequently, the user would have little interest in reading it.

Thus, in order to identify posts that are likely to be of interest to a user, knowing that a reference post is of interest to him or her, one needs to identify those that are related to that reference post. As the above examples indicate, content similarity can more accurately determine relatedness if focused on parts of the forum posts that play the same role, e.g., to give the context, to describe a wish, to make a request or provide a solution. Instead, if the content similarity is computed across the documents as a whole, the results may be misleading. The main question that needs to be answered here is how these different parts can be identified and how the content similarity can be computed across these parts.

3 Preliminaries

Assume an infinite set $\mathcal{T}$ of *text units*. In its simplest form, a text unit is a word, but one can also consider undivided combinations of words, e.g., "New York", as text units.

A *document* d is a finite sequence of text units, and its *cardinality* $|d|$ is the number of text units it consists of. We will use documents to model forum posts, and for this reason we will use the terms "posts" and "documents" interchangeably. Each text unit in a document is identified by its *position*. A *segment* is a finite sequence of consecutive text units in a document, and is identified by the position of its first and its last text unit. For instance, $[n, m]$, with $n<m$, denotes the segment consisting of the text units from the n-th to the m-th position.

A document can be seen as a sequence of non-overlapping segments, the concatenation of which is the document itself. Its division into such a sequence is known as *segmentation*.

Definition 1. A *segmentation* S^d of a document d is a sequence $(s_1, s_2, \ldots, s_k)$ of segments such that for every i=1..$(k-1)$, the segments s_i=$[l, j]$ and s_{i+1}=$[m, n]$ are such that m=$j+1$, and the textual concatenation $s_1 \cup s_2 \cup \ldots \cup s_k$ is equal to d. The number k, denoted as $|S^d|$, is referred to as the cardinality of the segmentation.

We refer to the virtual point between two consecutive segments as the *border* between these segments. In a document segmentation $(s_1, \ldots, s_k)$, a *border* b_i between a segment s_i=$[l, j]$ and the subsequent segment s_{i+1}=$[m, n]$, is the position m, i.e., the position of the first text unit of the segment s_{i+1}. We will denote by $\mathcal{B}^{S^d}$ the set of borders between the segments of a segmentation S^d. Note that a segmentation S^d can be equivalently represented by its set $\mathcal{B}^{S^d}$. A segment can be as small as a text unit or as large as the document.

By nature, every piece of text is written with a goal in the mind of its author. At the moment of the text construction, the author selects words and text structure that most effectively fulfill this goal. We have experimentally verified the existence of such goals in forum posts (ref. Sec. 9.1).

The goal of a piece of text, i.e., a segment, has been written, may not be explicitly stated, but by the way it is constructed, it is reflected into the characteristics of the text. Thus, monitoring and identifying strong variations in the characteristics of a document will indicate points where the author *intends to* serve a different goal. We use $\mathcal{I}$ to denote the set of all possible intentions and a function int:$\mathcal{U}\rightarrow\mathcal{I}$ that associates every segment to its intention in $\mathcal{I}$. We refer to the text characteristics as *features*, and we will use the term *feature vector* to refer to the values of these features for a segment s. Since there is such a close correlation between the features and the intention, given that the intention is only in the mind of the author, it is natural to identify the intention using text characteristics.

Definition 2. Given a set F of n features of interest, an *intention* is identified by a feature vector, i.e., a vector of n values, one for every feature of F.

The idea of using the features to identify intentions is similar to the idea of using terms to identify topics. In the topic detection literature, the topics of the documents may not be explicitly stated but the terms used in the document are an indication of the topic, and based on this observation, a topic has been defined as a vector of terms [15].

We will use the symbol $\sim$ to indicate two highly similar intentions, and the symbol $\not\sim$ to show highly dissimilar intentions. By abuse of expression, mainly for presentation purposes, we may write that two segments have the *same*, or *different* intentions, meaning that they have highly similar or highly dissimilar intentions, respectively, where similarity can be computed using any of the many vector similarity measures in the literature. In the case of two consecutive segments of a forum that have highly dissimilar intentions, we will characterize the border between them as a *deep* border.

Problem Statement. The challenge we propose to address is as follows: given a collection $\mathcal{D}$ of documents, and a reference document d_q, find those k documents in the collection that are most likely to be related to the reference document d_q, i.e., those documents that will most likely be of interest to a user that already considers d_q being of interest. The specific task is referred to as *document matching*.

4 Intention-based Matching

To implement a document matching solution for posts, we need to be able to compute some relatedness score, referred to as the *matching score*, of every document in a document collection to a reference document. To do so, we need to compare the reference document and any other document in the collection. It is our position that the relatedness is better assessed by computing a score, not across the content of the two documents as a whole, but across their segments

that have the same intention. To achieve this, each document (including the reference document) is first divided into segments of different intentions (*segmentation phase*). The segments are then clustered together (*segment grouping phase*) so that all the segments with the same intention end up together in the same cluster. Each resulting cluster can now be seen as a representative of some specific goal that is different from that of any other cluster. Segments from the same document that may have ended up in the same cluster are concatenated into one, so that there is at most one segment from each document in each cluster (*segmentation refinement phase*). For each cluster in which the reference document has a segment, the segments, and by extension the documents, with the highest scores in the cluster are selected. The score of two same-cluster segments of two different documents can be seen as the relatedness of the two documents when considering only the specific intention that the cluster represents (*matching with respect to a specific intention phase*). The relatedness (i.e., *matching score*) of the reference document with another document is computed by a combination of their individual intention relatednesses (i.e., respective segment score) across all the clusters (i.e., intentions) considering the segments from the previous phase. Based on the matching score, the top k most related documents to the reference document can be selected (*matching with respect to all intentions phase*).

There are three main challenges in the above steps. The first is how to segment the documents since the intention is not known, neither explicitly stated in the text. The second is how to recognize whether two segments from different (or the same) documents have the same or highly similar intention, in order to be clustered together. The third is how to compute the similarity among segments of the same intention and combine these similarities to form the matching score between the documents. The following sections describe how we cope with each of these challenges.

5 Segmentation of Posts

For a document d, there are $2^{|d|-1}$ possible segmentations. Among them, we are interested in the one that is more accurately aligned with the different intentions of the text. Finding the right segmentation is a challenging task [12], [13], [16], for which there is already a large body of work, from segmentation of queries to segmentation of documents [9], [17]. In these studies, a good segmentation is one where every segment is (*i*) coherent and (*ii*) largely disconnected from its adjacent segments. Since our criterion for segmentation is the intention-based, these two properties translate to a segmentation where every segment: (*i*) conveys a single clear intention; and (*ii*) this intention is highly different from those conveyed by the adjacent segments. Equivalently, the above criteria call for segmentation with deep borders.

Definition 3. An *intention-based segmentation* S^d of a document d is a segmentation where for any segment $s \in S^d$: (*i*) $int(u_1) \sim int(u_2)$, for any subsegments $u_1, u_2 \sqsubseteq s$; and (*ii*) $int(s) \not\sim int(s')$ where s' is any adjacent segment of s.

In finding a good intention-based segmentation, there are three challenges: identify the features to use for identifying the intentions, measure the coherence within a segment alongside the depth of the borders of a candidate segmentation, and, select the best segmentation among the candidates. Sections 5.1, 5.2, and 5.3 study these issues.

TABLE 1
Features (cells) and Communication Means (rows)

Tense(CM_{tense})	present	past	future
Subject (CM_{subj})	I/we	you	it/they/(s)he
Style (CM_{qneg})	interrog.	negative	affirmative
Status (CM_{pasact})	passive	active	
Part of Speech(CM_{pos})	verb	noun	adj./adverb

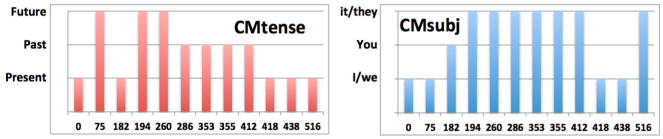
5.1 Feature Selection

First of all, we need to decide the features to use for identifying intentions. Content-based features, e.g., terms or keywords, have been used in the past for segmentation [12], [13]. Keywords have been also used by LDA for topic discovery. Since forum posts are relatively short, they tend to be very concise, which means that the basic keywords are used all over the post, making it hard to identify large topic variations. Another type of features is discourse-based features, such as pauses or voice stress, that are related to transcribed oral communication [18], but are hard to exploit in written documents such as forum posts. Since posts are intended to initiate or continue a discussion, they highly reflect the user's way of communication. Thus, it is natural to consider as features language characteristics that are related to syntax and grammar. The intuition is that a change in expression style signals a change in intention. For instance, when an author switches from the first to the third person, that is a signal of a change in the intended message.

Examples of grammar features are the verbs in some specific tense, the passive verbs, the references in the first person, etc. We classify the features into types, referred to as *communication means*. An example of a communication mean (CM) is the `Subject` that contains the features corresponding to references in the first, second and third person. In this way, each CM can be seen as a categorical variable and the features in the CM as its domain. For instance, the CM `Tense` can be seen as a categorical variable that takes the values `past`, `present` or `future`. Table 1 illustrates a number of features grouped under their respective CM. Each row in the table corresponds to a CM and each cell to a feature. One can monitor the value of a CM throughout a document (or segment).

Example 1. The top part of Fig. 2 illustrates forum post A of Fig. 1 where words indicating a value of CM_{subj} are in bold and those indicating a value of CM_{tense} are underlined. The boxes indicate certain positions in the document. Below the text, there are two bar charts that show the values of CM_{tense} and CM_{subj} throughout the document. The x-axis is the position in the document and the y-axis is the categorical value of the variable. In these bar charts, it can be seen that there is a shift in the value of the categorical variable, i.e., the CM. For instance, for CM_{tense} this takes place in positions 75, 182, 201, 285, and 418. Assuming that the time is a strong factor that can signal by itself a change in author intention, the post can be segmented into the segments shown in line (a) in Fig. 2. Line (b) shows a segmentation based on the points where there is a change in the CM_{subj} value, and line (c) based on CM_{qneg}. $CM_{pasactive}$ is not present in the post. The segmentations (d) and (e) are discussed in Example 2.

0 I have an HP system with a RAID 0 controller and 4 disks in form of a JBOD. 75 I would like to install Hadoop with a replication 4 HDFS and only 320GB of disk space used from every disc. 182 Do you know whether 201 it would perform ok or whether the partial use of the disk 259 would degrade performance. 285 Friends have downloaded the Cloudera distribution but 338 it didn't work. 355 It stopped since 371 the web site was suggesting to have 1TB disks. 418 I am asking because 436 I do not want to install Linux and then realize that 488 my hardware configuration is not the right one. 535



	Possible Segmentations
Boxes	75, 182, 201, 259, 285, 338, 355, 371, 418, 436, 488, 535
(*a*) CM_{tense}-Based	([0,75],[76,182],[183,201],[202-285],[286-418],[419-535])
(*b*) CM_{subj}-Based	([0,182],[183,201],[202,418],[419,488],[489,535])
(*c*) CM_{qneg} Shift	([0,182],[183,201],[202,438],[439,535])
(*d*) Intention-Based	([0-182],[183,418],[419-535])
(*e*) Thematic	([0-49],[50-535])

Fig. 2. CMs and Segmentations

Each CM, or combination of CMs, can be used to define a possible segmentation. We have experimented with different alternatives, either single CMs or combinations thereof. Another important factor is the domain of the categorical variables. For instance, CM_{tense} can have as a domain the $\{past, present, future\}$ or $\{past, not{-}past\}$. To select the best combination, we need to evaluate the effectiveness of each choice. To do so, we measured the diversity of the various segments in a segmentation and compared it to the diversity of the unsegmented post. For measuring diversity, we use the metrics described in the next section. More details on this selection task have been omitted, but we note that the features and the CMs that were found to be the best choice are those contained in Table 1.

5.2 Coherence and Depth Computation

Intuitively, as hinted earlier, to evaluate the quality of a segmentation we need to measure what variation is observed within a segment in terms of the user intentions and how the intentions of a segment differ from those of the adjacent segments (which would justify why the adjacent pieces of text have been placed in different segments). Thus, given a set of features, we need to be able to measure the coherence of a segment and the depth of a border.

Having a coherent segment means that in general we do not want to see large variations across the features observed in the segment, i.e., across the CMs' categorical values that have non-zero appearances. This is a measure known as *evenness* in statistics. Of course, if we select very small segments, there will be very few factors with a non-zero value. Due to the limited segment length, these values will be very similar, hence such segments will be highly coherent, yet, not really useful. To avoid this, in addition to evenness, we also need to consider the number of non-zero features, called *richness*.

The *diversity indices* consider both richness and evenness by measuring how many features have non-zero values, and at the same time how evenly are distributed among features. The value of a diversity index increases when richness and evenness increase, while decreasing in any other case.

To estimate diversity, we represent every communication mean CM_r by a distribution table (i.e., a vector) DSb_{CM_r}. Intuitively, each distribution table corresponds to a row of Table 1. The value of the element j of the table DSb_{CM_r}, denoted as $DSb_{CM_r}[j]$, indicates the number of times the value in column j of the CM r appears in the segment. For instance, a $DSb_{CM_{tense}}$ equals to $[2, 3, 0]$ means that the segment has 2 verbs in present tense, 3 in past tense and none in future tense. A well-known diversity index is Shannon's index,

$$div_{CM_r}(s_i) = -\sum_{j=1}^{|DSb_{CM_r}|} \frac{DSb_{CM_r}[j]}{All} * log(\frac{DSb_{CM_r}[j]}{All}) \quad (1)$$

where $All = \sum_{l=1}^{|DSb_{CM_r}|} DSb_{CM_r}[j]$.
The diversity values of each of the CMs in a segment s_i can be combined together to form a value for its *coherence*, which for a segment s_i can be computed by the following $coh(s)$ function that for categorical variables with at most three values takes value types less than one. (Note that higher diversity means less coherence.)

$$coh(s_i) = \frac{1}{|CM|} \sum_{r=1}^{|CM|} 1.0 - div_{CM_r}(s_i) \quad (2)$$

To measure the "depth" of a border, one can exploit the concept of coherence. A border is "deep" if the CMs in the two segments it separates are significantly different. To measure this difference, we remove the border, which in practice would mean that the segments on its left and right would become a single large segment, and we measure the coherence of this segment. That large "hypothetical" segment will have either a lower coherence than the two individuals (indicating a deep border) or a higher coherence, indicating a shallow border. Thus, the depth of a border b_i between segments s_i and s_{i+1} is:

$$depth(b_i) = \frac{|coh(s_i) - coh(s)| + |coh(s_{i+1}) - coh(s)|}{2 * coh(s)} \quad (3)$$

where the segment s is the segment resulting from the concatenation of s_i and s_{i+1}.

In previous work, the distance metrics of cosine dissimilarity, Euclidean distance, and Manhattan distance on term-based representations, have been used to decide whether two segments should remain separated or should be better merged as one. However, in the experiment section, we illustrate that term-based representations and distance metrics are not very effective for intention-based segmentation.

5.3 Border Selection

To find the best segmentation we need to select the best border positions in the document. With the ability to measure coherence of a segment and the depth of a border, we can define a measure to judge how strong or weak a border position is. A possible border b_i in position i is a good choice if each of the two segments s_i and s_{i+1} that b_i separates has a strong coherence and b_i has high depth. Based on this, we assign a score to a possible border position. The score can be computed using a weighted sum of coherence and depth, the *f-statistics* [19], or any other metric as long as

it is consistent with the above principle. We are actually computing it as the average of the three parameters, i.e.,

$$score(b_i) = (coh(s_i) + coh(s_{i+1}) + depth(b_i))/3 \quad (4)$$

There are two broad approaches to identify the borders that define an intention-based segmentation in a document. One is a top-down approach that initially considers the whole document as one segment and checks for possible positions a border can be placed in order to split the segment into two. The position is selected so that the resulting two segments have an average score that is better than the score of the borders before the split. The approach recursively splits segments as long as such borders can be found. Its main limitation is that the comparison of the depth and coherence in segments that differ significantly in terms of length may mislead the algorithm. For similar reasons, comparing two long segments may lead to incorrect decisions.

The other approach is bottom-up. It initially considers every text unit as a segment and iteratively merges consecutive segments to form longer segments. The merging of two consecutive segments is performed by simply removing the border that separates them. We propose different strategies to implement the bottom-up approach. Each strategy uses some different criteria for deciding whether to merge segments or not.

The *first strategy*, referred to as *Tile*, has also been used in thematic segmentation [12]. It iteratively passes through the whole document, and at the end of each iteration, it removes the borders that have a score smaller than a threshold. This threshold is defined as the mean score value of all the present borders but adapted by the standard deviation. This way after each iteration the score of the remaining borders increases (or remains unchanged). The process stops when no border satisfies the criterion.

The *second strategy*, referred to as *StepbyStep*, visits the borders in order, from left to the right. For each border it visits, it checks the coherence of the segment on its left. If that coherence is lower than the coherence of the whole document, the border is deleted and the segments before and after it become one. The algorithm continues until it has visited all the borders. The borders that have not been eliminated at the end specify the final segmentation.

The *third strategy* is referred to as the *Greedy*. It makes multiple passes over the document, and in each pass, it removes only one border, in particular the one with the worst score, which should also be less than some specific threshold. The algorithm stops when there is no border that can be removed either because there are no more borders or because there are no borders with a score less than the threshold. The algorithm makes locally optimal decisions, which means that it may be misled by the diversity of a single CM feature to the overall optimal solution. To avoid this, Greedy is run multiple times, one for each single CM and instead of removing the borders that the algorithm suggests to remove, it marks them for removal. After the step has been repeated for each of the CMs, those borders that have been marked for removal for the most of the times are those that are actually removed. Greedy has a higher execution time comparing to the other two algorithms due to the multiple passes it deploys, but as we will see in the experimental section, it best approximates human segmentations.

CM - Feature	Feature Vector	Intention Cluster Centroids				
		All	I_0	I_1	I_2	I_3
CM_{tense}-Present	Fs[1]	0.26	0.45	0.10	0.13	0.86
CM_{tense}-Past	Fs[2]	0.16	0.03	0.07	0.82	0.14
CM_{tense}-Future	Fs[3]	0.07	0.07	0.09	0.02	0.04
CM_{subj}-I	Fs[4]	0.22	0.29	0.12	0.28	0.47
CM_{subj}-You	Fs[5]	0.01	0.01	0.02	0.01	0.02
CM_{subj}-She/They	Fs[6]	0.25	0.39	0.10	0.26	0.80
CM_{qneg}-Interrog	Fs[7]	0.05	0.08	0.02	0.02	0.16
CM_{qneg}-Negative	Fs[8]	0.10	0.20	0.05	0.05	0.25
CM_{qneg}-Affir/ve	Fs[9]	0.26	0.36	0.11	0.30	0.85
$CM_{passact}$-Passive	Fs[10]	0.07	0.12	0.04	0.07	0.15
$CM_{passact}$-Active	Fs[11]	0.27	0.38	0.11	0.28	0.87
CM_{pos}-Verb	Fs[12]	0.27	0.39	0.11	0.27	0.88
CM_{pos}-Noun	Fs[13]	0.27	0.37	0.12	0.28	0.86
CM_{pos}-Adverb	Fs[14]	0.25	0.37	0.10	0.28	0.77
CM_{tense}-Present	Fs[15]	1.95	3.39	1.21	1.00	4.19
CM_{tense}-Past	Fs[16]	0.52	0.17	0.39	1.84	0.32
CM_{tense}-Future	Fs[17]	0.11	0.10	0.14	0.03	0.06
CM_{subj}-I	Fs[18]	0.75	0.96	0.59	0.88	1.05
CM_{subj}-You	Fs[19]	0.02	0.02	0.03	0.01	0.02
CM_{subj}-She/They	Fs[20]	1.77	2.67	1.09	1.88	3.45
CM_{qneg}-Interrog	Fs[21]	0.06	0.09	0.02	0.02	0.19
CM_{qneg}-Negative	Fs[22]	0.17	0.30	0.10	0.08	0.34
CM_{qneg}-Affir/ve	Fs[23]	2.69	3.73	1.83	3.04	4.78
$CM_{passact}$-Passive	Fs[24]	0.10	0.17	0.0	0.10	0.19
$CM_{passact}$-Active	Fs[25]	3.21	4.59	2.13	3.41	5.96
CM_{pos}-Verb	Fs[26]	4.00	5.86	2.65	4.05	7.46
CM_{pos}-Noun	Fs[27]	7.17	9.91	4.58	7.52	14.92
CM_{pos}-Adverb	Fs[28]	2.10	3.03	1.43	2.23	3.72

Fig. 3. Derived Intention Clusters after Segment Clustering

Example 2. Considering the features indicated in Table 1, the coherence and depth as defined in Section 5.2, and the score of Eq. (4), the intention based segmentation of the post of Fig. 2 is the one shown as (d) in Fig. 2. For comparison, the figure also shows as (e) the thematic segmentation generated by running Hearst's thematic segmentation method on the post [12], which highlights the significant difference between thematic and intention-based segmentation.

6 Segment Grouping

The next step in intention-based post matching is to recognize segments that are intended for the same goal (or purpose). We actually need to create groups such that segments with similar intentions end up in the same group and segments with different intentions in different groups. Since the actual intention is not known but we have modeled it through a vector of features, a natural choice for creating the desired groups is to perform clustering on the feature vectors corresponding to the intentions of the segments. Each cluster can then be seen as a representative of some communication goal. We use I to denote a cluster, and $\mathbb{C}$ to denote the set of the generated clusters.

We have found that using the feature vector as is (meaning with the absolute values of the features) is not very effective. Instead, we need to capture the relative contribution of each feature, thus we have created a vector of weights that are based on the feature values. We denote this vector with the letter F. We consider two types of weights that capture the strength of the use of each CM categorical value, i.e., of each feature. The *first type* measures the strength of the use of each CM value *within the segment*, i.e., in comparison to the frequency of the other categorical values of the same communication mean appearing in the segment. Using the notion of the distribution table DSb_{CM_r} of a communication mean CM_r introduced in Sec. 5.2, we define the vector F_s of weights, one weight for each feature. The weights

Doc A, Seg 1: I have an HP system . . . from every disk Doc B, Seg 1: My boss gave me . . . Linux pre-installed
Doc A, Seg 2: Do you know whether . . . have 1TB disks Doc B, Seg 2: I am thinking to . . . the entire system?
Doc A, Seg 3: I am asking because . . . the right one Doc B, Seg 3: I have already looked . . . related to it.

Fig. 4. Segments of forum posts A and B of Fig. 1. Segments found to belong to the same intention cluster appear together.

for a segment s are computed according to the formula: $\forall i = 1..|CM|,\ \forall j = 1..|DSb_{CM_r}|$

$$F_s[i * |DSb_{CM_r}| + j] = \frac{DSb_{CM_r}[j]}{\sum_{k=1}^{|DSb_{CM_r}|} DSb_{CM_r}[k]} \quad (5)$$

In the above formula, $|CM|$ indicates the number of different CMs we consider. For simplicity of the presentation, it also assumes that all the CMs have the same number of categorical values, i.e., in the case of Table 1, that would be that all the CMs have 3 possible categorical values, but this may not always be the case (see for instance CM_{pasact}.)

The weight $\frac{DSb_{CM_{subj}}[2]}{\sum_{k=1}^{3} DSb_{CM_{subj}}[k]}$, for instance, of the 2^{nd} value of the CM: CM_{subj}, will measure how stronger the use of the 2^{nd} person is as opposed to the 1^{st} or 3^{rd} person.

The *second type of weights* is derived from a normalization of the absolute number of occurrences of the CM categorical value *across the entire post*. For a specific categorical value, it captures the portion of the overall appearances in the whole document that correspond to the examined segment. Similarly to the weights of the first type, the vector F_s of all the weights of the second type of a segment s is computed according to the formula: $\forall i = 1..|CM|,\ \forall j = 1..|DSb_{CM_r}|$

$$F_s[i * |DSb_{CM_r}| + j] = \frac{DSb_{CM_r}[j]}{DSb^*_{CM_r}[j]} \quad (6)$$

where DSb^* denotes a distribution table that considers the whole document as a single segment. As an example, consider a document where we find five verbs in past tenses (CM_{tense}-Value 2), four of which are in the same segment. Then, the weight of this value of CM_{tense} will be high indicating for the value a significant role in the segment.

The *vector representation of each segment* is the concatenation of the two vectors corresponding to the two types of weights. Using the CMs of Table 1, the vector will have 28 elements (2 for each feature, corresponding to the two types of weights that were just introduced). Any of the well-known clustering techniques can now be applied on the weight vector representation of the segments.

We have experimented with different clustering algorithms. However, since the reason we employ clustering is to capture common patterns in the use or better the distribution of Communication Means within the segments and within the respective posts, The DBSCAN [20] algorithm, has been a good choice because: (1) it does not require to know the number of clusters in the data a priori, as opposed to distance-based clustering such as k-means, (2) it can find arbitrarily shaped clusters, and (3) it has a notion of noise.

Segmentation Refinement. It is possible that more than one segment from the same document end up in the same cluster, if they have the same intention but are not consecutive in the document, or the border selection mechanism kept a border between them due to local optimal values of segment diversity and border depth. We make one more pass over the clusters and if such cases are found, all the segments that belong to the same document in a cluster are concatenated into one. In other words, assuming the clustering $\mathbb{C}$ of the segments of a collection of documents $\mathcal{D}$, for every cluster $I \in \mathbb{C}$, a new set of segments is considered instead that is constructed as: $\{s|\ \exists d \in \mathcal{D}\text{: } \bigcup_{s' \in I\ \wedge\ s' \in S^d} s'\}$, where the symbol $\cup$ on segments indicates concatenation. As a result of this step, each document may have at most one segment in each cluster.

Example 3. Fig. 3 illustrates the results of the clustering of the segments of all the documents in the forum post dataset HP Forum (described in Sec. 9) from which the 4 documents of Fig. 1 were taken. The rows correspond to the elements of the feature vector. In white are the elements of the first type (Eq. (5)) and in gray those of the second type (Eq. (6)). Each of the columns I corresponds to a centroid of the clusters that the clustering produced. Fig. 4, on the other hand, shows which of the segments of the forum posts A and B of Fig. 1, have been clustered together, i.e., they have been assigned to the same intention.

7 Matching

To perform the document matching, i.e., to identify the documents in a collection that are related to a reference document d_q, one way is to see the document d_q as a query and then measure the relatedness of each other document d' to that query in a way similar to how IR techniques work. As already mentioned, our position is that such a task should not consider each document as a whole but should be specialized on each intention individually, and then combine the results.

Matching with respect to a specific Intention. Each cluster is the projection of every document on the specific intention that the cluster represents. Thus, to measure the *relatedness* of a document d' to the reference document d_q *with respect to a specific intention* I, it is enough to measure the relatedness of the respective segment s' of d' in the cluster I, to the respective segment s_q of d_q in that same cluster.

For computing this relatedness any text comparison, e.g., paraphrasing[8], language models [2], [3], or IR techniques may be employed. One of the best-known IR techniques is the TF/IDF. The core of the original TF/IDF method and its probabilistic variance BM25 consists of a *term weighting scheme* that weighs a term in a document considering the number of its appearances in relationship to the number of its appearances in all the other documents. We devise a version that is somewhere between the original and the BM25, and takes into consideration intentions. In particular, we start with a variance of TF/IDF that comes close to BM25 and has been implemented in MySQL 5.5.3 for full-text searching. That variance computes the weight of a term t in a document d' as

$$w(t, d') \quad = \quad \frac{log(f_{d'}(t)) + 1}{\sum_{\forall t' \in d'} (log(f_{d'}(t')) + 1) * NU(d')} \quad (7)$$

where $f_{d'}(t)$ is the frequency of a term t within the *document* d', and $NU(d')$ is the document length normalization factor that penalizes d' if the number of unique terms in the

Algorithm 1 Single Intention Matching

Input: Cluster I, Doc. Collection $\mathcal{D}$, Document $d_q \in \mathcal{D}$, Int n
Output: List of n documents and their intention matching score

$M_I \leftarrow \emptyset$
for each $s_q \in S^{d_q}$
  if $s_q \notin I$ **continue**; // See footnote[1]
  $scr \leftarrow 0$
  for each $s' \in I$
    $d' \leftarrow \{d \mid s' \in S^d\}$ // See footnote[2]
    for each $t \in s_q$
      $scr \leftarrow scr + f_{s_q}(t) * w(t, s') * log(|I| - |I^t|)/|I^t|$
      $M_I \leftarrow M \cup \langle d', scr \rangle$
Return $\{\langle d', scr \rangle \mid \langle d', scr \rangle \in M_I \wedge scr \in$ top-n scores in $M_I\}$

Algorithm 2 All Intentions Matching

Input: Document Collection $\mathcal{D}$, Document $d_q \in \mathcal{D}$, Int k, n
  Intention Clusters $\mathbb{C}$
Output: List of documents

$L \leftarrow \emptyset$, $M \leftarrow \emptyset$
for each $I \in \mathbb{C}$
  for each $s_q \in S^{d_q}$
    if $s_q \notin I$ **continue**
    $M_I \leftarrow$SingleIntentionMatching(I,$\mathcal{D}$,d_q,n)
    $L \leftarrow L \cup \{M_I\}$
for each $M_I \in L$
  for each $\langle d', scr \rangle \in M_I$
    if exists $\langle d', x \rangle \in M$, with $x \in \mathbb{R}$
      $M \leftarrow M \cup \langle d',\ scr \rangle$
    else $\langle d', x \rangle \leftarrow \langle d',\ x + scr \rangle$
Return $\{d' \mid \langle d', scr \rangle \in M \wedge scr \in$ top-k scores in $M\}$

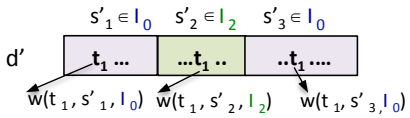


Fig. 5. Weighting for the same term in different intention clusters.

document is larger than the average number of unique terms across all the documents.

We extend the above formula in a way that the weight of a term is based on the segment it belongs (instead of the document) and the intention (i.e., cluster) that the segment has been assigned to. In particular, the weight of a term t in a segment $s' \in I$ is:

$$w(t, s') = \frac{log(f_{s'}(t)) + 1}{\sum_{\forall t' \in s'} (log(f_{s'}(t')) + 1) * NU(s', I))} \quad (8)$$

where $f_{s',I}(t)$ is the frequency of the term t within the segment s', and $NU(s', I)$ the segment length normalization factor that penalizes s' if the number of its unique terms is larger than the average segment length in that intention cluster I. With this approach, we generate weights for the terms that may be different for the same term across different segments.

Example 4. Fig. 5 illustrates our weighting approach. In a document d', with $S^{d'} = \{s'_1, s'_2, s'_3\}$, term t_1 is weighted differently when found in segment s'_1 than in segment s'_2 or s'_3. For instance, since s'_1 has been assigned to intention I_0, the weight of term t_1 is based on the terms in s'_1 and those in all other segments in I_0.

1. Due to the segmentation refinement step, there will be only one segment for which $s_q \in I$, and at most one document for which $s' \in S^d$ and $s' \in I$.

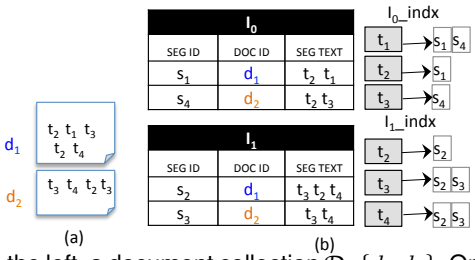


Fig. 6. On the left, a document collection $\mathcal{D}=\{d_1,d_2\}$. On the right, $\mathcal{D}$ after the segmentation, segment grouping, and indexing step.

The relatedness of a document d' to a reference document d_q with respect to an intention I, can now be computed based on the term weights. If s_q and s' are the segments of the documents d_q and d', respectively, in the intention cluster I, the relatedness is:

$$scr(d_q, d', I) = \sum_{\forall t \in s_q} f_{s_q}(t) * w(t, s') * \frac{log(|I| - |I^t|)}{|I^t|} \quad (9)$$

where $f_{s_q}(t)$ denotes the frequency of the term t in the segment s_q, $|I|$ the cardinality of the intention cluster, and $|I^t|$ the number of segments in the intention cluster I that contain the term t. The fraction $\frac{log(|I|-|I^t|)}{|I^t|}$ is actually the traditional *probabilistic inverse document frequency*, adjusted for the case of intentions. Moreover, in an application scenario where some clusters are more important than the others, different weights can be considered for each cluster turning the above sum into a weighted sum.

Note that if one of the documents d_q or d' has no segment in the intention I, then the relatedness score is by default 0.

Let $M_I(d_q)$ denote the top-n most related documents to the reference document d_q for the intention I as identified by the relatedness score. Furthermore, let $\mathcal{M}$ denote the set of all such lists for the different intentions. Note that instead of considering the top-n documents for each intention, one could consider only those that are above a specific threshold [21], however, to be fair across all the intentions that a document contains, we opted for the top-n approach. Algorithm 1 illustrates the above steps.

Matching with respect to All the Intentions. The top-n lists generated across the different intentions, i.e., the set $\mathcal{M}$ mentioned above, are used to generate the k most related documents to the reference document d_q. A new list R is created that contains every document that appears at least in one of the lists in $\mathcal{M}$. A score is associated to each such document that is the sum of the scores with which this document appears in the various lists in $\mathcal{M}$. The k elements in R with the highest score are returned as answer to the request of the matching documents to the reference document d_q. These steps are indicated in Algorithm 2.

It is important to note that a relatively small value for n (compared to the value of k) will favor documents that have high score in one list in $\mathcal{M}$ even if they do not appear in others, penalizing at the same time documents that may appear in many lists but with lower scores. A relatively high value for n compared to the value of k, on the other hand, will favor documents that appear in many lists even with not very high scores. We have empirically found that a good choice is an n equal to $2 * k$.

Indexing. In contrast to segmentation and segment grouping that are performed offline (pre-processing steps of the document collection), document matching, i.e., the retrieval of the top-k documents for a document query d_q, can be performed online due to its low response time (less than 3 millisecs for a collection with more than 1.5M posts, ref. Sec. 9.2.4). In practice, in order for Algorithms 1 and 2 to be able to generate fast the (initial) top lists in each cluster I and subsequently generate the final list, we built a fulltext index on the terms of all the segments of each segment group (cluster) I. Therefore, we are building $|\mathcal{C}|$ fulltext indices. In addition, we are building an index on the ids of the documents where the segments belong so as to be able to access faster the segments of a document query d_q. Fig. 6 graphically illustrates the two clusters (I_0, I_1) and the corresponding indices (I_0_indx, I_1_indx) that have been formed after the segmentation and segment grouping of a small document collection (d_1, d_2).

8 Related work

The detection of purposes or goals that user-generated texts are *intended for* has been a subject of research in different domains [22]. We have investigated the benefits that information systems can have by identifying and exploiting this type of information [23] with a focus on retrieval and recommendations [24]. The notion of *intention* has been used in the past in text mining but in a completely different context than ours. It has been used to label phrases such as "I want to . . . " (referred to as purchase or educational intents) in forum and social media posts [25], [26] or to characterize user clicks in web search [27], or as further description of short queries considering sources such as query logs and web search results [11]. In this work, we use intentions to identify segmentations and then use these segmentations to improve the matching task in forum posts. There exists considerable amount of work for post matching in Question Answering Communities (QAC). People seek answers to general-interest, factual or informational, questions [1]. Apart from computing the explicit content similarities of threads [2], [3] such systems may also leverage the syntactic structure of questions posted in such forums in order to match questions (e.g., [28], [29]) or thread post-reply structure (e.g., [30]). Another approach is to use different combinations of content, semantic, syntactic, and authorship-related features to classify questions as relevant or not [31], [32]. However, in question repositories, posts are plain questions. On the contrary, we suggest a method that enables the use of such techniques on elaborate forum posts that consist of multiple segments. Specifically, depending on how deep one can afford and wants to go into the content similarity, apart from traditional retrieval techniques [6], language model-based methods and semantic text comparisons [2], [3], [8], [9] could be exploited by our matching technique when the comparison of the text of the segments is performed.

[Segmentation methods] Segmentation methods are divided into 2 broad groups. The first is *topical segmentation* where adjacent pairs of text blocks are compared for overall similarity based on terms or topics [13] or lexical chains [12]. Topic text segmentation is not suitable for our case since we are interested in author intentions and not the actual topic. The second group of segmentation methods consists of *Transcribed oral-discourse* techniques used in the analysis of transcribed oral communication using linguistic criteria [18]. These are not applicable to our case, since we are dealing with written discourse.

9 Experimental results

We have evaluated all the steps of our method on the recommendation of related posts i.e., segmentation, identification of segments with the same intention and comparison of the posts based on similarity across segments of the same intention. We first needed to see whether the segmentation task we perform makes sense. Section 9.1.1 verifies the existence of segments in forum posts; while Section 9.1.2 presents the findings of the evaluation of the segmentation step of our approach contrasting alternative features, border selection mechanisms and coherence/depth functions. In the sequel, we have evaluated our overall approach comparing its effectiveness, in terms of precision, to two baseline methods that are not using any segmentation (ref. Section 9.2.2); and our approach when segmentation and grouping methods are used other than the intention-based ones (ref. Section 9.2.3). Moreover, performance/efficiency of our approach has been evaluated with experiments on data of different sizes. (Section 9.2.4).

Datasets. We used three real datasets of posts from forums in three different domains. The first had 111K posts from a product *support* forum (HP Forum, http://h30434.www3.hp.com), with an average post size of 93 terms with 2.3% unique terms (stop-words were not considered). The second dataset, had 32K posts of hotel reviews from a *travel* forum (TripAdvisor) [33]. The average post size was 195 terms with 3.2% unique content terms. And the third dataset was a dump of a well-known computer *programming* forum (StackOverFlow, http://stackoverflow.com) consisting of 1.5M (it actually consists of 4M posts but we have considered only those with an accepted answer). The average post size was 79 terms with 2.5% unique terms. In all datasets, the number of posts refers only to root posts (i.e., posts that trigger a thread); answers are not included. The percentage of unique terms verifies that in forums since users deal with issues under specific topics, the used vocabulary is limited.

Implementation. For experiments, we used an Ubuntu 0.14.04.1 machine, with 125GB memory, CPU 172 MHz. We also used MySQL 5.5.3 and code was written in Java 1.7.

9.1 Segmentation Evaluation

We conducted a user study to: (*i*) validate our observations that posts, despite their relative short size and informal writing style, *can be naturally divided into segments, with each conveying a different intention* (ref. Sec. 9.1.1.A), to understand *which are the different messages that the authors convey* (ref. Sec. 9.1.1.B), and also (*ii*) to evaluate the automatic segmentation approach (features, border selection mechanisms, coherence/depth functions) (ref. Sec. 9.1.2). For contrasting the alternative features and functions, we consider multWinDiff error; while for the border selection mechanisms we also present how the number of borders and segment coherence is affected by each of the mechanisms. Specifically for this study, we used a randomly selected sample of two of the

TABLE 2
User agreement on the segmentation task

	HP Forums	*TripAdvisor*
Offset	*Fleiss's κ/Agreement Percentage*	
$\pm$10 chars	0.20/64%	0.35/71%
$\pm$25 chars	0.41/71%	0.44/75%
$\pm$40 chars	0.68/77%	0.71/83%

datasets: 500 posts from the support and 100 posts from the travel forum.

Human Annotation Task. We had 30 participants from five countries that were all computer literate and fluent in English. All the participants had at least a bachelor's degree. Among them, there were users with PhD and PhD candidates in computer science or engineering as well as software developers and engineers. The participants were asked to read each post carefully and *divide it into coherent segments by putting a border at the end of a term after which they perceived a shift in the message that the author intended to communicate, i.e., where the text serves a different goal*. For each segment, they were asked to provide a description (label) of 1-5 keywords. In order not to bias the annotators to look for specific segments, no limit on the size of a segment or the number of segments was specified nor were labels predefined. The task was performed online through a PHP, JavaScript application that we developed for this purpose *and the outcome was 4.7K labeled segments*. The mean number of segments per post was found to be 4.2 for the HP Forum and 5.2 for the TripAdvisor.

9.1.1 Examining Human Segmentations

A. Verification of Segment Existence. The granularity of individual segmentations, as it was expected, varied. To verify that forum posts can be naturally divided into parts, we measured the annotation agreement. We considered *observed agreement percentage* (that shows how many annotators agreed over all) and *Fleiss's κ* that indicates whether the high observed agreement percentage is (or is not) due to chance agreement. We considered an offset from 10-40 characters (ref. Table 2). Within an offset of 40 characters (i.e., 3-5 terms with spaces and punctuation included), average observed agreement percentage varies from 77% to 83%, where 0 indicates complete disagreement and 100% perfect agreement. Considering the strictest character offset (i.e., 10 chars, 1-2 terms) the agreement remains high (64% to 71%). Fleiss's κ, with negative values indicating lack of agreement and with positive values from slight to perfect agreement closer to 1, varies from 0.68 to 0.74. These values indicate *considerable agreement*. Thus, *posts are indeed organized into logical units that are relatively easily recognizable by humans*.

B. Underlying Messages. Since the labels of the *different messages that the authors communicate* were not predefined, there was a large variety of keywords describing the same message and that made the analysis of the results more complex. However, a predefined list of labels would have worked as a bias while for us it was important to cross-check that the users would detect similar underlying author intentions with the ones we had observed without being directed to do so. The outcome was positive: labels such as *expectation*, *previous efforts*, *help request*, *hotel description* and *system description* were selected by the annotators. These

Technical Support Forum/Hardware-Software
- a. **Explain the problem:** problem/issue statement, general problem...
- b. **Describe previous efforts:** solution attempt, previous trial...
- c. **Explain why she wrote the post:** reason for posting, theme, target ...
- d. **Report symptoms/hypotheses:** observations, first appearance of problem...
- e. **Ask for suggestions, advice, or other requests:** help request, request for advice..
- f. **Describe the problem "environment":** system description/information, user pc..
- g. **Ask specific questions:** question, general question, first/second question...
- h. **Express thoughts/feelings:** concern, personal comment, personal thought...

Travel Site Forum/Hotels
- a. **Explain how/why user decided to book:** reason for selecting or for staying...
- b. **Judge Aspects:** location, price, staff, breakfast, other facilities...
- c. **Describe the room/hotel:** room description, general hotel description...
- d. **Declare a number of pros, cons:** pro, con, (dis)likes, strong, weak points...
- e. **Opinion/conclusion:** overall, general opinion, why (not) revisiting..
- f. **Describe to whom/why it is recommended:** for future, what to expect..

Fig. 7. Annotators' labels, grouped in categories, for the goals that the segments are intended for.

labels do not describe what a segment actually talks about, which is what topics derived from LDA would have done, for instance, but indicate why the author wrote the specific segment. Segments with similar content (i.e., considering similar terms) have been labeled differently; while segments that do not share common terms have been labeled with the same or similar labels. Fig. 7 *summarizes the most common labels* clustered into 7-8 categories for each dataset.

9.1.2 Automatic Segmentation Effectiveness

Given the posts from the two datasets that were segmented by humans in the user study, we examined how much we can approximate human performance with different automatic segmentation approaches to: (*i*) evaluate the document representation based on CMs vs on a term-based one, (*ii*) select the most appropriate border selection mechanism, and (*iii*) evaluate the coherence/depth functions.

To measure how close an automatic segmentation is to human ones, we used a well-known metric from Computational Linguistics where originally segmentation task comes from the *multWinDiff*, a variation of the traditional *winDif* error which handles different number of annotations per post [34]. The *multWinDiff* error uses overlapping windows, where the size of window equals half of the average length of reference segmentations.

A. Intention Representation: CM vs Term-based features. We tried out Hearst's segmentation algorithm that defines cohesive segments as *homogeneously lexically distributed text parts* and evaluates candidate borders using *cosine similarity on weighted terms*. We compared it to our *Tile* strategy (ref. Sec. 5.3) that uses the same mechanism for border selection as the Hearst's segmentation algorithm but it represents documents as *vectors of their CM Features* (ref. Table 1). For the border score cosine dissimilarity was used. We observed that with *Tile*, the *average error is reduced by 18% (from 0.64 to 0.46), in the HP Forum dataset and by 26% in the TripAdvisor dataset*. The significant error reduction shows that CMs represent documents better when it comes to identifying borders that reflect shifts in intention.

B. Border Selection Mechanism Effectiveness. Subsequently, we performed a comparison of our border identification mechanisms, namely *Tile*, *Greedy* and *StepbyStep*. In all cases, we used the CMs described above and the score function of Eq. 4, where coherence is determined by Shannon's diversity and sentences as text units. Sentences are usually written to express a single complete message

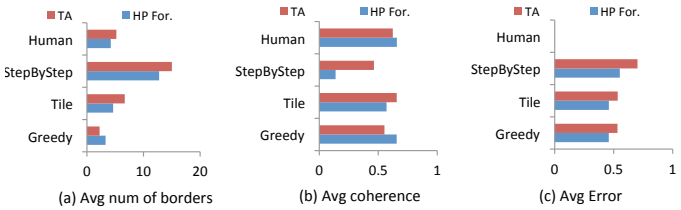


Fig. 8. Comparison of border selection mechanisms

Function	*Posts with Error decrease*	*Posts with No change*	*Posts with Error increase*	*Avg Error decrease*
Cos.Sim.	68%	19%	11.5%	-0.18
Eucl.Dist.	64.7%	8.1%	29.83%	-0.22
Manh.Dist.	43.4%	10.7%	45.8%	-0.13
Richness	46.8%	11.5%	41.8%	-0.17
Shan.Div.	79.9%	15.5%	4.7%	-0.24

Fig. 9. Error under different coherence/depth functions

and they contain all (or almost all) communication means features. Thus, they constitute natural and intuitive text units. Fig. 8(*a*) shows the average number of borders. *Tile* returns more borders per post on average while *Greedy* less than human annotators for all the data samples. *StepbyStep*, on the other hand, returns way more borders. We observe that the first two mechanisms produce the most coherent segments after human segmentations (Fig. 8(*b*)) and have *the lowest error*, i.e., they approximate better human segmentations (Fig. 8(*c*)). Thus, both *Greedy* and *Tile* look promising. We selected *Greedy* for the overall evaluation experiment.

C. Coherence and Depth Functions Comparison. We experimented with the Shannon's index and richness (for coherence) and with the distance functions: cosine dissimilarity, Euclidean distance, Manhattan distance (for depth). We found that Shannon's index diversity on CMs reduces error the most: by 24%. The table in Fig. 9 summarizes the results. For a better understanding of the results, we provide, apart from average error changes, the percentage of posts in which the segmentation of a post was approximated better, worse or the same, i.e., error reduction, increase and no change, respectively.

9.2 Overall Technique Evaluation

Our approach on the recommendation of related posts has been evaluated in the environment of (*i*) a tech support forum by users of the forum trusted as *experts* (HP Forum dataset), and (*ii*) a well-known crowd-sourcing platform (CrowdFlower) by workers there (TripAdvisor dataset), and (*iii*) in the environment of a programming forum by computer scientists and engineers that regularly use the site (StackOverFlow).

TABLE 3
Segment Granularity - Percentage of Segments

	BEFORE GROUPING			AFTER GROUPING		
	HP Forum	*Trip Advisor*	*Stack OverFl.*	*HP Forum*	*Trip Advisor*	*Stack OverFl.*
1	25.1%	19.9%	43.3%	30.7%	25.1%	53.6%
2	25.1%	23.8%	30.6%	40.5%	46.1%	41%
3	18.8%	19.8%	14%	28.4%	23.5%	6.3%
4	16.36%	13.4%	6%	0.37%	4.8%	
$5-8$	39.6%	22.9%	0.55%			

The Methods. The choice of the methods to compare with was done in order to evaluate: (*i*) how matching posts at segment granularity compares with matching at the post-as-a-whole level; and (*ii*) the effectiveness of our segmentation and clustering processes. *For comparison with methods considering posts as a whole* we used the implementation for full-text matching included in MySQL 5.5.3. that uses the weighting scheme is described in Eq. 7 and a ranking method that is a variation of BM25[2]. This method will be referred to as *FullText*. We also used matching based on LDA topics with Gibbs sampling (denoted as *LDA*) [7], [35].

For evaluating our segmentation and segment grouping processes, we examined how our matching method (ref. Algorithms 1 and 2) performs when the default segmentation into sentences is used instead of our border selection mechanism (based on intention shifts). We refer to this method as *SentIntent-MR*. We also considered our matching method using clusters of segments with similar content instead of intention clusters. Specifically, instead of the intention-based segmentation we performed a very well known segmentation based on topic shift [12] and clustering on TF/IDF vector representations of the posts. We refer to this method as *Content-MR*. Our proposed, complete, method is denoted as *IntentIntent-MR*. *MR* in the three last methods stands for *Multiple Ranking lists* and indicates the use of Algorithm 2; what changes between these methods is the type and content of clusters.

Our method produces 4 intention clusters for the HP dataset, 5 for the TripAdvisor, and 3 for the StackOverFlow dataset. We have observed that the same message can be distributed into different parts of the same post and that intention assignments are not restricted neither to their position in the text nor to the segment before or after them. Table 3 illustrates the granularity of the segmentation before and after the grouping step. In the grouping step, the information about the assigned intentions is used to refine the borders that have been derived in the segmentation step, e.g., a document with three segments assigned to $\{I_2, I_0, I_2\}$ respectively will remain with two segments. In the end, the 30.7%, 25.1%, and 53.6% of the posts of the three datasets remain undivided, i.e., with only one segment. The remaining posts contain 2-4 different messages, while right after segmentation the granularity was between 1-8 segments for the first two datasets, and 1-4 for the last one.

Posts are dynamic data and as new data arrives, it is natural that the intentions may change and may need to be updated (i.e., the clusters should be recreated taking the new posts into account). The time efficiency of clustering (ref. Sec. 9.2.4) dictates that re-running the algorithm for the whole (updated) dataset is not a major issue that would require an incremental solution. We have also investigated the way that intentions change over time by performing a comparison between the intentions in the posts of two consecutive years from the StackOverFlow dataset and noticed no significant changes.

2. For a clear and fair comparison, the same ranking method (modified accordingly as described in Section 7 to consider intention clusters) was used for the comparison among segments in our method as well.

TABLE 4
Comparison of Methods - Mean Precision

	LDA	*FullText*	*Content-MR*	*SentIntent-MR*	**IntentIntent-MR**	$Gain^{(*)}$
HP Forum	0.01	0.16	0.065	0.16	**0.26**	**+10%**
TripAdv. Forum	0.21	0.53	0.27	0.45	**0.65**	**+12%**
StackOverFlow Forum	-	0.161	-	-	**0.262**	**+10.1%**

$(^*)$Considering the best baseline, FullText

TABLE 5
Test Corpus

	HP Forum (100K)	*TripAdvisor* (33K)	*StackOverFlow* (1.5M)
Methods	All	All	2
Post pairs	5000	750	240
Evaluations	15000	2193	1440
User Agreement	0.87	0.81	0.794

9.2.1 User Evaluation

From each of the post collections described in the beginning of Section 9, we randomly selected some posts to serve as reference documents, i.e., d_q. The random selection gave us representative samples with segmentation granularity distribution very close to that of the whole datasets. For each of the sample document queries from the HP Forum and TripAdvisor datasets, users evaluated the top-5 posts returned by each method (ours and the alternative methods), while for the StackOverFlow dataset, users have evaluated the top-5 lists derived from our method and the best baseline (i.e., FullText). Every post-to-post matching, i.e., post pair, was evaluated by at least three users. We chose a binary evaluation over graded [36] since we are interested in returning to the user only highly related posts. The derived dataset is described in Table 5. The five lists (one for each method) in the environment of the tech support forum, and the two lists in the StackOverFlow were evaluated separately while for the TripAdvisor posts we performed pooling to generate a single list per query-post [37]. The user-experts evaluated the recommended forum posts in the lists having no information about how they had been generated. The inter-rater agreement (Fleiss' kappa) for the total was found to be: 0.87, 0.81, and 0.794 (for the HP Forum, TripAdvisor and StackOverFlow datasets, respectively) reflecting almost perfect agreement. The evaluations were used to estimate the mean precision: the mean of the precision values considering each information need, i.e., post query, separately. Table 4 illustrates the results that are discussed in the next subsections.

9.2.2 Comparison with Baseline Methods

Full-text comparison matches to the post at hand d_q posts that share important, according to the used weighting scheme, common terms. Table 4 shows a clear *gain of 10%, 12%, and 10.1% in mean precision* for the three datasets, respectively. Our method, *IntentIntent-MR*, retrieves the most lists with the largest number of related posts in the first two datasets (ref. Fig. 10). Moreover, for the StackOverFlow dataset, it reduces the lists with no true positives (mean precision 0) by 28.6%.

The higher precision is justified by the fact that common terms that appear in segments that are meant for a different goal often lead to false positives. On the flip side, intention-based segmentation and grouping manages to distinguish the different messages before proceeding with the comparison step. Consequently, such false positives are avoided with *IntentIntent-MR*.

Fig. 10. True Positives retrieved by the examined methods.

TABLE 6
Execution times (StackOverFlow dataset)

Avg Segmentation Time	*Total Segment Grouping Time*	*Avg Retrieval Time*
0.067 sec	3.18 min	0.029 sec

On the other hand, the *LDA method* performs worse than both our method and the *FullText* method. Specifically, Table 4 indicates *25% and 44% lower mean precision* than ours. We tried out topic-based comparisons as well since one could claim that they may exist terms correlated with different intentions that will allow such a comparison to distinguish the different intended messages without the need of segmentation. An over-simplified example would be the topics: "ink, blink, light, question" and "ink, blink, light, tried, unsuccessfully". Two documents that share the terms "ink, blink, light" would not be considered as related if they have been assigned to two different topics describing a question and a user's effort respectively. However, we see that although topics describe posts at a higher level than that of terms, they fail to compare effectively posts that already belong to the same category.

9.2.3 Comparison with Alternative Segmentation Methods

The comparison of our method with *Content-MR* (ref. Table 4) shows that forming clusters of segments that reflect different topics instead of intention clusters gives worse results (-19.5% and -39%). Consequently, term-based features can not effectively distinguish the different messages. In cases of collections with posts from different categories, *Content-MR* was found to perform better. However, the scope of this paper is matching posts within the same Forum category; therefore, we do not get into these results.

Moreover, *SentIntent-MR*, which creates *clusters of sentences* instead of clusters of segments based on the diversity of CM features (i.e., border selection step is omitted), shows performance closer to that of the *FullText* method that considers the posts as a whole and is lower than our complete method, *IntentIntent-MR*, by -10% and -20% (ref. Table 4). This comparison tells us that, without the border selection step, the segment grouping step fails to form intention clusters, thereby degrading the performance of the matching algorithm. This verifies that the diversity in CMs manages to distinguish the different messages that the authors want to communicate.

9.2.4 Scaling

We have compared the time efficiency of our method to the other four methods considering the dataset of the product forum divided into three sets of 1k, 10k, and 100k posts, respectively. Moreover, we have examined how our method behaves in a larger dataset, namely the StackOverflow.

Segmentation. Fig. 11(a) illustrates the sum of the execution times of segmenting all the posts in the collection of 100k

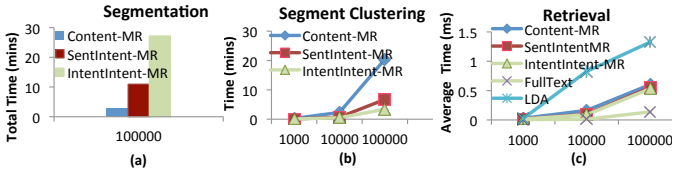


Fig. 11. Comparison of execution times (HP Forum Dataset)

in the worst-case scenario where the post segmentation is performed sequentially. The segmentation is based on: intention shifts in *IntentIntent-MR* (greedy technique), topic shifts in *Content-MR*, and segmentation into sentences for *SentIntent-MR*. *IntentIntent-MR* requires about 60% more time than *SentIntent-MR* due to the additional border selection mechanism, while *Content-MR*, which requires no preprocessing (i.e., no POS-tagging etc) takes less time. However, when the latter segmentation method is used, the matching method manages to retrieve fewer true positives (ref. Table 4). The average segmentation time of our method for the product forum posts is 0.016 sec. On the other hand, for the StackOverFlow collection (ref. Table 6), it is 0.067 sec. To run the segmentation, we first divided the dataset in 32 parts (1M lines each) and run in parallel the segmentation of 5 to 7 parts. The execution time per part was 3.7h on average and the maximum 6.99h; while in total the segmentation of the 1.5M posts lasted 23 hours. All the reported times include html and special symbols cleaning, POS tagging and CM annotation; while for the second dataset there is an additional cost for reading the data in xml format, and selecting only the root posts with accepted answers.

Clustering or Segment Grouping is run on the whole dataset. Text clustering in general is computationally expensive. However, Fig. 11(b) shows that in our case it is efficient. The reason is that in the grouping step we represent text segments by only 28 numeric features (ref. Eq. 5, 6). The same applies for *SentIntent-MR*. The execution of the latter, however, lasts more since the number of sentences is larger than the number of segments. In all cases, clustering was performed using the Weka 1.4 library. For the segment clustering of StackOverFlow dataset, we used a library that is intended for very large datasets and scales better [38]. In fact it takes only about 3 mins for the 2.93M segments derived in the segmentation step (Table 6).

Matching, i.e., the top-k list retrieval given a post-query is also very efficient. Fig. 11(c) shows that the average retrieval time in the product forum collection varies from 0.017 to 0.53 msec. The times of the methods that use multiple lists are very close. The fastest response time is that of *FullText* (less than 0.14 msec) because it accesses a single term index to get its answers. *LDA*, due to the lack of any indexing is the slowest (1.33 msec). Moreover, as Table 6 indicates, the average retrieval time in the StackOverFlow collection is only 2.9 msecs; i.e., it is less than 6 times higher although the dataset is 15 times larger.

10 Conclusions

We proposed a novel approach for matching a reference post to the k most related posts in a collection. Our method identifies and exploits post segments that convey similar author intentions. We presented several experiments regarding the right segmentation criteria, the effectiveness of the segmentation algorithms and the formation of intention clusters that prove that a rather intuitive concept, that of the author intentions to communicate a certain message, can be effectively captured by an automated process. Moreover, due to the nature of the posts, measuring the relatedness score after having distinguished the different segments/messages that the authors intend to communicate has been proved more effective than the direct comparison of the whole posts. Specifically, our approach, according to an evaluation by real users and in comparison with direct fulltext comparison, increased mean precision by 10%, 12% and 10.1% considering posts in a product support, a travel, and a programming forum.

References

[1] M. Chen, X. Jin, and D. Shen, "Short text classification improved by learning multi-granularity topics," in *IJCAI*, 2011, pp. 1776–1781.

[2] J. Jeon, W. B. Croft, and J. H. Lee, "Finding semantically similar questions based on their answers," in *Proceedings of the 28th ACM SIGIR Conference*, ser. SIGIR '05. New York, NY, USA: ACM, 2005, pp. 617–618.

[3] T. C. Zhou, C.-Y. Lin, I. King, M. R. Lyu, Y.-I. Song, and Y. Cao, "Learning to suggest questions in online forums." in *AAAI*, 2011.

[4] L. Weng, Z. Li, R. Cai, Y. Zhang, Y. Zhou, L. T. Yang, and L. Zhang, "Query by document via a decomposition-based two-level retrieval approach," in *In*. Association for Computing Machinery, Inc., July 2011.

[5] V. Govindaraju and K. Ramanathan, "Similar document search and recommendation," *Journal of Emerging Technologies in Web Intelligence*, vol. 4, no. 1, pp. 84–93, 2012.

[6] S. Robertson, S. Walker, and M. Hancock-Beaulieu, "Okapi at TREC-7: Automatic ad hoc, filtering, VLC and interactive track," *TREC '98*, pp. 199–210, 1998.

[7] D. M. Blei, "Probabilistic topic models," *Commun. ACM*, vol. 55, no. 4, pp. 77–84, Apr. 2012.

[8] J. Berant and P. Liang, "Semantic parsing via paraphrasing." in *ACL (1)*, 2014, pp. 1415–1425.

[9] H. Wen, W. Zhongyuan, W. Haixun, Z. Kai, and Z. Xiaofang, "Short text understanding through lexical-semantic analysis," in *IEEE ICDE*, 2015.

[10] Z.-Y. Ming, T.-S. Chua, and G. Cong, "Exploring domain-specific term weight in archived question search," in *Proceedings of the 19th ACM CIKM*, ser. CIKM '10. New York, NY, USA: ACM, 2010, pp. 1605–1608.

[11] H. Wu, W. Wu, M. Zhou, E. Chen, L. Duan, and H.-Y. Shum, "Improving search relevance for short queries in community question answering," *WSDM*, pp. 43–52, 2014.

[12] M. Hearst, "Texttiling: Segmenting text into multi-paragraph subtopic passages," *Computational Ling.*, vol. 23, pp. 33–64, '97.

[13] H. Misra, F. Yvon, J. M. Jose, and O. Cappe, "Text segmentation via topic modeling: an analytical study," in *CIKM*, 2009, pp. 1553–1556.

[14] X. Hu, N. Sun, C. Zhang, and T. Chua, "Exploiting internal and external semantics for clustering short texts using world knowledge," in *CIKM*, 2009, pp. 919–928.

[15] I. Hulpus, C. Hayes, M. Karnstedt, and D. Greene, "Unsupervised graph-based topic labelling using dbpedia," in *WSDM*, 2013, pp. 465–474.

[16] G. Salton, A. Singhal, C. Buckley, and M. Mitra, "Automatic text decomposition using text segments and text themes," in *ACM Hypertext*, 1996, pp. 53–65.

[17] M. Hagen, M. Potthast, B. Stein, and C. Bräutigam, "Query segmentation revisited," in *In*, ser. WWW '11. New York, NY, USA: ACM, 2011, pp. 97–106.

[18] R. J. Passonneau and D. J. Litman, "Intention- based segmentation: Human reliability and correlation with linguistic cues," in *ACL*, 1993, pp. 148–155.

[19] J. Bossart and D. P. Prowell, "Genetic estimates of population structure and gene flow: limitations, lessons and new directions," *Trends in Ecol. & Evol.*, vol. 13, no. 5, pp. 202–206, 1998.

[20] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *PODS*, 1996, pp. 226–231.

[21] R. Fagin, "Combining fuzzy information from multiple systems," in *PODS*, 1996, pp. 216–226.

[22] D. Papadimitriou, Y. Velegrakis, G. Koutrika, and J. Mylopoulos, "Goals in social media, information retrieval and intelligent agents," in *2015 IEEE 31st International Conference on Data Engineering*, April 2015, pp. 1538–1540.

[23] D. Papadimitriou, G. Koutrika, J. Mylopoulos, and Y. Velegrakis, "The goal behind the action: Towards goal-aware systems and applications (to appear)," *ACM Trans. Database Syst.*, 2016.

[24] D. Papadimitriou, "Goal-aware data management for retrieval and recommendations," in *2016 IEEE 32nd ICDE Workshops*, May 2016, pp. 216–220.

[25] Z. Chen, B. Liu, M. Hsu, M. Castellanos, and R. Ghosh, "Identifying intention posts in discussion forums." in *HLT-NAACL*, 2013, pp. 1041–1050.

[26] S. Louvigne, N. Rubens, F. Anma, and T. Okamoto, "Utilizing social media for goal setting based on observational learning," in *ICALT*, 2012, pp. 736–737.

[27] B. J. Jansen, D. L. Booth, and A. Spink, "Determining the informational, navigational, and transactional intent of web queries," *Inf. Proc. and Mangmt.*, vol. 44, no. 3, pp. 1251 – 1266, 2008.

[28] K. Wang, Z. Ming, and T. Chua, "A syntactic tree matching approach to find similar questions in community QA services," in *ACM SIGIR*, 2009, pp. 187 – 194.

[29] K. Wang, Z. Ming, X. Hu, and T. Chua, "Segmentation of multi-sentence questions: towards effective question retrieval in cQA services," in *ACM SIGIR*, 2010.

[30] A. Singh, P. Deepak, and D. Raghu, "Retrieving similar discussion forum threads: a structure based approach," in *ACM SIGIR*, 2012, pp. 135 – 144.

[31] L. Hong and B. Davison, "A classification-based approach to question answering in discusion boards," in *ACM SIGIR '09*, 2009, pp. 171 – 177.

[32] A. Shtok, G. Dror, and Y. Maarek, "Learning from the past: Answering new questions with past answers," in *WWW*, 2012, pp. 759–768.

[33] K. Ganesan and C. Zhai, "Opinion-based entity ranking," *Information Retrieval*, 2011.

[34] A. Kazantseva and S. Szpakowicz, "Topical segmentation: A study of human performance and a new measure of quality," in *HLT*, 2012, pp. 211–220.

[35] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, 2003.

[36] J. Kekalainen, "Binary and graded relevance in ir," *Inf. Processing & Management*, vol. 41, no. 5, pp. 1019 – 1033, 2005.

[37] K. Jones, C. Van Rijsbergen, B. L. Research, and D. Department, *Report on the Need for and Provision of an Ideal Information Retrieval Test Collection*, ser. British Library Research and Development reports, 1975.

[38] E. Achtert, H. Kriegel, E. Schubert, and A. Zimek, "Interactive data mining with 3d-parallel-coordinate-trees," in *SIGMOD 2013, NY, USA*, 2013, pp. 1009–1012.

Dimitra Papadimitrioul is a PhD candidate and a member of the Data and Information Management group at the University of Trento, Italy. She received her BSc and MSc degree in Computer Science from Aristotle University, Greece. She has also spent time as a visitor at HP Labs, Palo Alto. Her scientific interests include large scale information management and retrieval, text mining, and recommendations with a focus on user intentions and goals.

Georgia Koutrika is a Director of Research at ATHENA Research Center in Greece. She has worked as a senior research scientist at HP Labs, USA. Prior to HP, she has worked at IBM Almaden Research Center, and as a post-doctoral researcher at Stanford University. Her work brings together methods from databases, information retrieval, information integration, recommendations, machine learning, data mining, and big data processing. She has co-authored more than 80 papers in peer-reviewed conferences (including ACM SIGMOD and IEEE ICDE) and journals (such as ACM TODS and IEEE TKDE). She has served as a general co-chair of SIGMOD 2016 and a track (co-)chair for SIGMOD, ICDE, EDBT conferences.

Yannis Velegrakis is a faculty member at the University of Trento. He holds a PhD degree in Computer Science from the University of Toronto. His research areas of expertise is Big Data Analytics, Social Data, and Large Scale Information Management. Prior to joining the University of Trento, he held a researcher position at AT&T Research Labs, and has been a visitor at the University of California, Santa-Cruz, the IBM Almaden Research Center, and the University of Paris-Saclay.

John Mylopoulos holds a professor emeritus position at the University of Trento (Italy) and University of Toronto (Canada). His research interests include conceptual modelling, requirements engineering, data semantics and knowledge management. He is a fellow of AAAI and the Royal Society of Canada (Academy of Sciences). He has served as programme/general chair of international conferences, including IJCAI (1991), Requirements Engineering (1997, 2011), and VLDB (2004). He was recently awarded an advanced research grant from the European Research Council (ERC) for the project "Lucretius: Foundations for Software Evolution".