Fuzzy Clustering for Topic Analysis and Summarization of Document Collections

René Witte[1] and Sabine Bergler[2]

[1] Institut für Programmstrukturen und Datenorganisation (IPD)
Universität Karlsruhe (TH), Germany
[2] Department of Computer Science and Software Engineering
Concordia University, Montréal, Canada

Abstract. Large document collections, such as those delivered by Internet search engines, are difficult and time-consuming for users to read and analyse. The detection of common and distinctive topics within a document set, together with the generation of multi-document summaries, can greatly ease the burden of information management. We show how this can be achieved with a clustering algorithm based on fuzzy set theory, which (i) is easy to implement and integrate into a personal information system, (ii) generates a highly flexible data structure for topic analysis and summarization, and (iii) also delivers excellent performance.

1 Introduction

Information seekers nowadays are typically overwhelmed with the multitude of documents available for any given topic online. Whereas information retrieval (IR) is adequately covered by modern Internet search engines, the user is mostly left alone with the task of sifting through the resulting set of documents. Delivering automated tools for the analysis, filtering, and pre-processing of natural language texts is an important next step in personal information management.

In this paper, we provide a fuzzy clustering algorithm for the analysis of document collections, as they could have resulted from a query posed to an Internet search engine or an intranet document server. We show how both common and distinctive topics of such a document set can be detected, which is far more informative than simple keyword extracts. Additionally, the data structure computed by our algorithm allows for the generation of several types of multi-document summaries, including a context-sensitive one. Such summaries, typically between 100 and 350 words, are more easily scanned by a human; cross-linked with the original documents they further aid a user in quickly determining relevant topics and help in deciding which documents are a good candidate to read in full.

Our research is significant for several reasons: *(1)* We provide a flexible, adaptive, context-sensitive algorithm for the analysis of natural language document collections, which is easy to implement and integrate into information systems, thereby substantially improving information management for users; *(2)* We show how fuzzy set theory can be applied to natural language processing (NLP), which adds robustness, ease of deployment, and flexibility to our approach, while at

the same time allowing for further transfer of ideas and algorithms from the field of soft computing to information management; *(3)* We show how to generate a highly flexible data structure that can be used to generate multiples types of condensed information displays that can also adapt to a user's current context; and *(4)* Results from large-scale evaluations on data from the international *Document Understanding Conference* (DUC) competition prove our approach to be highly competitive with current summarization systems.

The remainder of this paper is structured as follows: in the next section, we describe the preprocessing needed to provide the input required by our algorithm. Section 3 describes our contribution, the cluster algorithm. Some possible applications of the generated data structure, including topic detection and summarization, are outlined in Section 4. An evaluation of our approach, based on data and methods from the NIST-sponsored DUC competition, is presented in Section 5. Section 6 discusses related work, followed by conclusions in Section 7.

2 Preprocessing

Before we can describe our main fuzzy clustering technique, we have have to discuss some preprocessing steps needed to generate the required input data structures: noun phrase (NP) chunks and fuzzy coreference chains. Additionally, this section provides a first introduction to the idea of applying fuzzy set theory [1] to natural language processing.

2.1 Noun Phrases

All documents first undergo basic NLP preprocessing, including tokenization, sentence splitting, and part-of-speech (POS) tagging, which in our implementation is performed within the open source GATE *(General Architecture for Text Engineering)* framework.[3]

Noun phrases, i.e., determiner/modifier/head triples, are then computed as the main input to the following coreference resolution step. Here, we assume minimal (base) NPs, without prepositional or other attachments. Our NPs are computed by the MuNPEx chunker[4] based on part-of-speech tags and additional entity-specific grammar rules. However, since for the purpose of cluster-based coreference resolution it does not matter how NPs have been computed, an algorithm based on a full parse would work equally well.

2.2 Fuzzy Coreferences

Our technique is based on grouping NPs into *fuzzy coreference chains*. Our approach for coreference resolution is based on fuzzy set theory as the underlying formal representation. For the purpose of this paper, we will only give a brief outline of fuzzy coreference resolution. A detailed description of our algorithm is available in [2, 3].[5]

[3] For more detail on these steps, we refer the reader to the GATE user's guide at `http://gate.ac.uk/sale/tao/index.html`.

[4] Multi-lingual Noun Phrase Extractor (MuNPEx), `http://www.ipd.uka.de/~durm/tm/munpex`

[5] Additionally, [4] provides an overview of different coreference algorithms, including an independent implementation and evaluation of our approach (albeit not using fuzzy sets, i.e., a crisp implementation).

Fuzzy coreference resolution groups entities (typically NPs) into fuzzy coreference chains. Each chain contains all textual descriptions that refer to the same entity (e.g., in a text, the three descriptions *"Luke Skywalker," "he,"* and *"the young Jedi"* might refer to the same person). The central idea behind using fuzzy set theory is the uncertainty inherent in natural language processing: even for a human reader, it is not always certain whether two NPs really refer to the same entity; employing fuzzy sets allows a soft computing approach where this uncertainty is represented explicitly, rather than making decisions based on (often arbitrary) hard thresholds.

Fuzzy Coreference Chains. Fuzzy coreference chains link entities, which are typically represented by noun phrases (NPs). In this paper, we denote the set of all noun phrases within a text with the (crisp) set $\mathrm{NP} = \{\mathrm{np}_1, \ldots, \mathrm{np}_m\}$, i.e., there are m noun phrases within a document. A single fuzzy chain C is then represented by a fuzzy set μ_C, which maps the domain of all noun phrases NP to the $[0, 1]$-interval: $\mu_C : \mathrm{NP} \rightarrow [0, 1]$. Thus, each noun phrase $\mathrm{np}_i \in \mathrm{NP}$ has a membership degree $\mu_C(\mathrm{np}_i)$, indicating how certain this NP is a member of chain C. The membership degree for a single noun phrase $\mu_C(\mathrm{np}_i) \in [0, 1]$ is interpreted in a possibilistic fashion: a value of 0.0 *("impossible")* indicates that the NP cannot be a member of chain C, a value of 1.0 *("100% possibility" or "certain")* means that none of the available information indicates that the NP is not in the chain, intermediate values represent different degrees of compatibility of a noun phrase with the chain.

Example (fuzzy coreference chain). Fig. 1 shows an example for a fuzzy coreference chain C_i. Here, the noun phrases np_3 and np_6 have a very high possibility for belonging to the chain, np_1 only a medium possibility, and the remaining NPs are most likely not chain members.

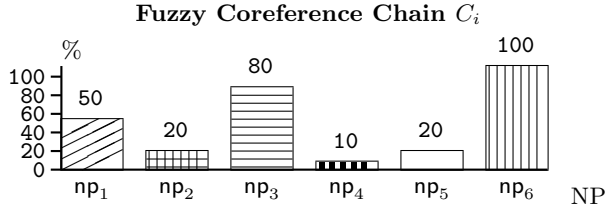

Fig. 1. Fuzzy chain C_i with membership grades for each noun phrase

The output of a fuzzy coreference algorithm is a set of fuzzy coreference chains, similarly to classical coreference resolution systems. Each chain holds all noun phrases that refer to the same conceptual entity. However, unlike for classical, crisp chains, we do not have to reject inconsistent information out of hand, so we can admit a noun phrase as a member of more than one chain, with different degrees of certainty for each. This provides an explicit representation of the uncertainty that is so common in natural language analysis.

Fuzzy chains can be converted to crisp chains using a *defuzzification* function, which allows downstream language analysis components that are not fuzzy-aware to use results of a fuzzy algorithm.

Fuzzy Coreference Resolution. Fuzzy chains are constructed through (usually knowledge-poor) *fuzzy heuristics*. Typical features used within our heuristics are *head noun, gender,* or *position*.

Our fuzzy coreference algorithm is essentially a single-link hierarchical clustering strategy. It initially creates one fuzzy chain for each NP, which forms its medoid (for example, in Fig. 1, np_6 is the chain's medoid). We then compute the degree of coreference between all NP pairs within a text, each degree normalized to a fuzzy value in the $[0, 1]$-interval. Each fuzzy degree can be interpreted as a distance between the medoid and the co-refering NP; it is added to every chain using standard fuzzy set operators. For example, in Fig. 1, at least one fuzzy heuristic must have determined a fuzzy coreference degree of 0.8 for $(\text{np}_6, \text{np}_3)$.

Finally, all chains are *merged* using a prescribed consistency degree γ. Merging combines compatible chains into merged chains (or NP clusters) using the coreference properties of symmetry and transitivity. The merge degree γ influences the size of the chains, and in effect, their precision and recall. A degree of 0 would merge all NPs into a single (yet useless) chain, while a value of 1 would lead to chains of the best possible precision, leaving out uncertain links and thereby resulting in more singletons (and lower recall).

The process of merging is now repeated for each possible value of $\gamma \in \{\gamma_1, \ldots, \gamma_n\}$,[6] leading to a *family* of coreference chains, a set of sets of chains: $\mathcal{C} = \{C_1^{\gamma_1}, \ldots, C_n^{\gamma_n}\}$. Note that a similar result can be obtained with a non-fuzzy coreference clustering strategy.

For the purpose of our algorithm described in the next section it is important that the individual chains exhibit *monotonicity*, that is, if two entities are linked within a chain of a specific certainty γ_i, they must also be linked in all chains of lower certainty $\gamma_j \leq \gamma_i$.

We use the same algorithm to create both inter- and intra-document coreference chains, only the number of enabled heuristics and various parameters differ for each. The end results are two families of coreference chains, one for intra- and one for inter-document coreferences.[7]

3 Fuzzy Coreference Graph Clustering

In this section, we describe our main contribution, a fuzzy coreference cluster graph algorithm that builds the data structure needed for identifying topics in document sets and constructing various types of summaries. This algorithm takes as input the intra- and inter-document coreference chain families computed by a coreference algorithm under different (fuzzy) clustering thresholds as described in the previous section.

[6] Since fuzzy sets are stored in horizontal representation through a set of α-*cuts*, with $[\mu]_\alpha = \{\omega \in \Omega | \mu(\omega) \geq \alpha\}$, the merge degree γ can only assume a finite number of different values, which typically correspond to the α-cut levels (e.g., $\alpha \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$).

[7] Cross-document chains do not contain links between NPs of the same document, since these links have already been computed by the intra-document step.

The first step is the construction of an initial fuzzy coreference cluster graph, as described in Section 3.1 below. Our clustering algorithm, described in Section 3.2, then works on this data structure, computing clusters that can be used to create several kinds of condensed information displays, including multi-document summaries (Section 4).

Essentially, two kinds of clusters are created by our algorithm: *common* clusters that contain NPs from two or more documents, and *distinctive* clusters that contain NPs from one document only. In other words, each cluster determines a *topic* that can be tracked through the different documents: some topics span all documents (common topic), some topics only occur in a subset or a single document (contrastive/distinctive topic).

3.1 Cluster Graph Initialization

A *fuzzy coreference cluster graph* is an undirected, weighted graph with entities (typically NPs) as nodes and weighted coreferences between these entities as edges. Essentially, it folds both inter- and intra-document coreference chains into one data structure that can then be traversed by the clustering algorithm. Thus, the algorithm's input are the intra- and inter-document coreference families, computed by one of the standard coreference clustering algorithms:

Input (cluster graph initialization). Input to the cluster initialization step is a set of sets of coreference chains $\mathcal{C} = \mathcal{C}_{\text{inter}} \cup \mathcal{C}_{\text{intra}}$ with the inter-document chains $\mathcal{C}_{\text{inter}} = \{C_1^{\gamma_1}, \ldots, C_n^{\gamma_n}\}$ and the intra-document chains $\mathcal{C}_{\text{intra}} = \{C_{n+1}^{\gamma_1}, \ldots, C_{2n}^{\gamma_n}\}$.

Note that each coreference chain C_i^{γ} contains again a set of sets of NPs, where all NPs within a subset $c \in C$ corefer with a fuzzy certainty degree of γ. We can now create the initial cluster graph.

Definition (initial cluster graph). An initial cluster graph $\mathcal{G} = (V, E)$ is constructed from the intra- and inter-document coreference families as follows. The set of graph nodes V is given by the set containing all NPs from all documents. The set of edges is derived from the set $\mathcal{C}$ containing both intra- and inter-document coreference families by iterating through all coreferences $C \in \mathcal{C}$. For each chain $c \in C$, we then iterate through all the entities $(\text{np}_i, \text{np}_j)$ within that chain and create one edge of weight γ between them. The complete algorithm is shown in Fig. 2. Note that we treat coreferences as links, that is, for a coreference chain $c_i^{\gamma} = \{\text{np}_1, \text{np}_2, \text{np}_3\}$ we add two edges with weight γ to the graph, one between np_1 and np_2 and one between np_2 and np_3.

Example (initial cluster graph). Consider three documents d_1, d_2, d_3 with two coreference families (inter- and intra-document), containing three coreference sets each for $\gamma \in \{0.6, 0.8, 1.0\}$:[8]

$$\mathcal{C}_{\text{inter}} = \{C_1, C_2, C_3\}, \mathcal{C}_{\text{intra}} = \{C_4, C_5, C_6\}$$

With the inter-document chains C_1, C_2, C_3:

$$C_1 = \{\{\text{np}_3, \text{np}_6\}, \{\text{np}_1, \text{np}_4, \text{np}_7\}, \{\text{np}_2, \text{np}_9\}\}$$
$$C_2 = \{\{\text{np}_3, \text{np}_6\}, \{\text{np}_1, \text{np}_4\}, \{\text{np}_2, \text{np}_9\}\}$$
$$C_3 = \{\{\text{np}_1, \text{np}_4\}, \{\text{np}_2, \text{np}_9\}\}$$

[8] Singletons are omitted for brevity

Require: Set of coreference families $\mathcal{C}$,
graph $\mathcal{G}$
1: **for all** $C \in \mathcal{C}$ **do**
2: **for all** $c \in C$ **do**
3: **for** i=1 to $|c| - 1$ **do**
4: $\text{np}_i \leftarrow c$.get(i);
5: $\text{np}_j \leftarrow c$.get(i+1);
6: **if** e=(np_i, np_j) $\in E$ **then**
7: $\gamma' \leftarrow \max(C_\gamma, \text{e}_\gamma)$;
8: updateEdge(np_i, np_j, γ');
9: **else**
10: addEdge(np_i, np_j, C_γ);

Fig. 2. Cluster Graph Initialization

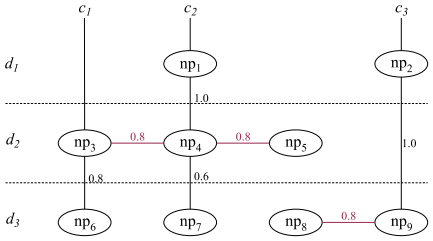

Fig. 3. Initialized fuzzy cluster graph

and the intra-document chains C_4, C_5, C_6:

$$C_4 = \{\{\text{np}_3, \text{np}_4, \text{np}_5\}, \{\text{np}_8, \text{np}_9\}\}$$
$$C_5 = \{\{\text{np}_3, \text{np}_4, \text{np}_5\}, \{\text{np}_8, \text{np}_9\}\}$$
$$C_6 = \{\}$$

Fig. 3 shows the resulting initial cluster graph. Intra-document coreference chains are drawn horizontally (in red), while cross-document chains (in black) are displayed from top to bottom. Each edge in the graph is labeled with the fuzzy *certainty* value of the coreference; in the example, np_4 and np_5 corefer with a certainty of 0.8, while np_4 and np_7 corefer with a certainty of 0.6.

3.2 The Clustering Algorithm

We can now describe the main clustering algorithm that works on the initial data structure described above. Similarly to chain merging, graph clustering is controlled by a threshold θ. In general, the lower the clustering threshold, the more entities are clustered together, resulting in fewer, but larger, clusters.

The key idea is to use the degree of coreference between entities, represented by an edge's weight, as the *inverse distance* between those entities: entities linked by an edge of weight 1.0 are closest, whereas entities with an edge of weight 0.0 (i.e., no edge) are infinitely far apart. We can now apply an agglomerative hierarchical clustering strategy, creating a dendogram data structure.

Definition (coreference graph clustering). The clustering process starts with clusters containing individual entities, i.e., each node V in the initialized graph $\mathcal{G}$ represents a cluster by itself. We now apply a hierarchical clustering strategy, where we progressively merge clusters until the algorithm terminates. Two clusters are merged if a direct edge exists between them of weight $\gamma \geq \theta$. If multiple edges exist between two clusters, we evaluate the one with the highest weight, i.e., we use a single-linkage clustering strategy. The algorithm terminates when no more edges exist between clusters. Fig. 4 shows the complete algorithm.

When the cluster algorithm terminates, the cluster graph contains clusters spanning multiple documents (common topics), as well as clusters that contain entities from a single document (distinctive topics). Of course, either set may be empty, depending on the input document set and the threshold setting.

Require: Initialized cluster graph $\mathcal{G} = (V, E)$
1: **for all** np $\in V$ **do**
2: createCluster(np); {create initial clusters}
3: **for all** $e \in E$ **do**
4: **if** $e_\gamma \geq \theta$ **then** {join clusters?}
5: $c_1 \leftarrow$ getCluster(e_{start});
6: $c_2 \leftarrow$ getCluster(e_{end});
7: mergeClusters(c_1, c_2);

Fig. 4. Fuzzy Clustering Algorithm

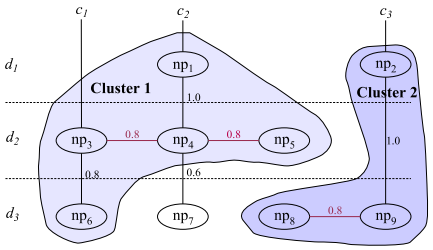

Fig. 5. Graph after running the clustering algorithm with $\theta = 0.8$

Example (final cluster graph). Fig. 5 shows the result after running the clustering algorithm on the graph in Fig. 3 with $\theta = 0.8$. This results in two large common NP clusters and the distinctive topic np_7 in document d_3 (documents d_1 and d_2 do not have any distinctive topics). For $\theta = 0.6$, however, np_7 would have been added to cluster 1, whereas a larger θ value would have created smaller clusters and more singletons.

Note that we can repeat the clustering process for each fuzzy value of θ, which results in a cluster family (or one multi-dimensional cluster). However, within this paper, we will only discuss single clusters.

4 Topic Analysis and Automatic Summarization

In this section, we discuss a number of applications for our cluster graph data structure for the analysis of document collections, including topic detection and multi-document summarization.

4.1 Common Topic Detection

The first application we discuss is the detection of a *common topic* within a collection of documents. This allows a user to easily identify the most salient information with a set of texts.

This kind of information can be immediately generated from the cluster graph data structure: Topics are identified by clusters, so by extracting clusters that span all documents (or a sufficiently large subset thereof), a system can obtain the common themes of all documents. In order to rank the topics by relevance, the *size* of each cluster can be additionally evaluated: the larger a cluster, the more important the topic contained within (we give empirical evidence for this in Section 5).

Example (common topic detection). We give a real-world example based on the DUC 2004 [5] data set `d30002`, which contains ten documents discussing *Hurricane Mitch*. Fig. 6 shows the initial cluster graph generated for this document set; after clustering, the common topics can be identified as shown in Fig. 7.

As can be seen, for a specific clustering threshold, four topic clusters have been identified, which are addressed in all ten documents. Changing the fuzzy threshold results in different clusters, either more discriminative (more and smaller clusters) for larger θ values or more lenient (fewer and larger clusters). However, we cannot show these here due to space constraints.

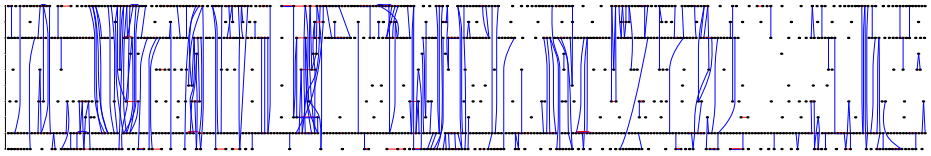

Fig. 6. Initial cluster graph for a DUC ten-document set: 2779 nodes (NPs), 3822 edges (co-references). Intra-document links (red) are aligned horizontally and cross-document links (blue) vertically (enlarge electronic version for details).

Common clusters
Hurricane Mitch in Central America (31) – Honduras (21) – the country's central coast (15) – last week's storm (12)

Fig. 7. Common topic clusters for a set of ten documents on "Hurricane Mitch" (number of NPs for common clusters in parentheses)

4.2 Multi-Document Summarization

The list of common topic identifiers presented in Fig. 7 provides a highly condensed view of a document collection. If a user is interested in obtaining more detail, but without reading each of the documents involved, he may opt to view a *multi-document summary*.

Multi-document summarization attempts to identify the most salient (shared) topics within a collection of documents. The summary, typically sentences (or sentence parts) extracted from the documents, should reflect as many common topics as space permits. Current systems (see the DUC Proceedings [5, 6]) typically achieve this by ranking sentences within a document collection, either through statistical means or (shallow) linguistic analysis and subsequent relevance scoring.

As shown above, our cluster graph data structure already represents common topics in a document set. We implemented a summarization component that generates a multi-document summary by selecting (at least) one candidate noun phrase from each cluster, in decreasing order of importance (cluster size), until a prescribed length limit has been reached or all clusters are exhausted. The candidate NPs, in turn, can be used to select the sentences they appear in as a candidate text extract.

Example (multi-document summary). Fig. 8 shows an example for a (roughly) 100-word summary generated with our approach [7, 8].

Common Topic Summary
The Honduran president closed schools and public offices on the coast Monday and ordered all air force planes and helicopters to evacuate people from the Islas de la Bahia, a string of small islands off the country's central coast. National police spokesman Ivan Mejia said the Coco, Segovia and Cruta rivers all overflowed their banks Monday along Honduras' eastern coast. The European Union on Tuesday approved 6.4 million European currency units (dlrs 7.7 million) in aid for thousands of victims of the devastation caused by Hurricane Mitch in Central America. The greatest losses were reported in Honduras, where an estimated 5,000 people died and 600,000 people – 10 percent of the population – were forced to flee their homes after last week's storm.

Fig. 8. Cluster graph generated multi-document summary

Common clusters	
Hurricane Mitch in Central America (31) – Honduras (21) – the country's central coast (15) – last week's storm (12)	
Distinctive clusters	
D_1	Gen. Mario Hung Pacheco – the shelves of some stores and some gasoline stations – mayor of Utila – a hurricane warning – the northwest Caribbean for five days
D_2	the western Caribbean on Wednesday – 165 kms – Honduras with 120 – west at only 2 mph – a resident of Guanaja Island
D_3	the center – emergency measures on the Caribbean coast of the Yucatan Peninsula – a boat – hotels – The storm's power
D_4	the storm's death toll in the region to 357 – 231 people have been confirmed dead
D_5	floods – the Guatemalan border – a state of emergency – 50 kph – late Sunday
D_6	area – the slopes of the Casita volcano in northern Nicaragua – Sunday night – a 32-square mile – addition
D_7	homes – The greatest losses – affiliate in San Miguel province – a statement – the EU
D_8	the audience – all public and private institutions and all men – the pope – a gift – six Russian cosmonauts
D_9	access to places – other countries – the recovery effort – More help – at least 300 children at the shelter for diarrhea, conjunctivitis and bacterial infections
D_{10}	Taiwan – aid and pledges of assistance – Residents – Cuba's offer – the saddest thing

Fig. 9. Differential topic analysis results based on the cluster graph generated for a set of ten documents on the "Hurricane Mitch" topic

Extractive sentence-based summarization typically involves additional techniques, i.e., replacing dangling pronominal references, eliminating duplicate noun phrases, or removing relative clauses. However, within the scope of this paper we are not concerned with this kind of post-processing, which is already widely discussed in the literature (see e.g. [9] and the DUC proceedings).

4.3 Differential Topic Analysis

So far, we concentrated on the most important, common topics within a document collection. Often, a user is also interested in what *additional*, unique information a document contains, i.e., knowledge that cannot be found in other texts.

This requires a differential topic analysis. For this, we inspect the distinctive clusters generated by our algorithm. These are clusters that span only a single document, or a (configurable) subset of all texts in a set. Each of these clusters represents a topic of a document subset that is distinctive from common topics across all documents.

Example (differential topic analysis). Fig. 9 shows an example for a differential topic analysis, on the same document set as before. For each document, a representative phrase from each of its distinctive clusters is displayed. Note that for D_1–D_3 and D_5–D_7, these topics identify a country that is distinctively discussed in this document only. D_9 quite clearly focuses on the ensuing medical emergency. Within a user interface, these clusters could again be hyperlinked to expand into a summary of the document set, similarly to the common topic summary discussed above.

4.4 Context-Based Summarization

The last type of application we address here are *focused* summaries, which are not concerned with summarizing a document (set), but rather with collecting information on an explicit interest expressed through context information. For

"Who is Stephen Hawking?"
Hawking, 56, is the Lucasian Professor of Mathematics at Cambridge, a post once held by Sir Isaac Newton. Hawking, 56, suffers from Lou Gehrig's Disease, which affects his motor skills, and speaks by touching a computer screen that translates his words through an electronic synthesizers. Stephen Hawking, the Cambridge University physicist, is renowned for his brains. Hawking, a professor of physics an mathematics at Cambridge University in England, has gained immense celebrity, written a best-selling book, fathered three children, and done a huge amount for the public image of disability. Hawking, Mr. Big Bang Theory, has devoted his life to solving the mystery of how the universe started and where it's headed.

Fig. 10. Focused summary of ten documents based on a question, generated from a cluster graph

example, a user might work on a report and needs information concerning a number of questions; or he might write an email reply and would like to see existing information relevant to the emails he is answering.

Indeed, the cluster graph also allows us to generate focused summaries by including the context information as an additional document D_0 when creating and clustering the fuzzy coreference chains. Then, all clusters that overlap with document D_0 also contain information relevant to this context. All other clusters, even if they are bigger, are discarded for this kind of summary.

Example (differential topic analysis). An example for a context-based summary is shown in Fig. 10. Here, the user stated an explicit question *"Who is Stephen Hawking?"*. The focused summary is then generated from a document set using the cluster graph data structure as discussed above. As before, elements within the clusters have to be further ranked, extracted, and post-processed to create the final summary.

5 Evaluation

We evaluated our fuzzy coreference cluster graph algorithm using the data from the DUC competition on summarization. This involved both multi-document summaries (2003/2004) and focused summaries (2005/2006) of sets between 10 and 50 documents in size. To allow a comparison with all other systems that participated in the competition, we computed the ROUGE-1 score [10] for all years. The results of our system ERSS, compared with the best, worst, baseline, and average system of each year and task is shown in Fig. 11.

In general, our algorithm performs well above the average, in some cases within a statistically insignificant difference from the top system. More detailed results, including experiments with different fuzzy values, different evaluation measures like Basic Elements (BE), as well as a comparison of our cluster algorithm with baseline ranking strategies like TF∗IDF,[9] can be found in [7, 8].

6 Related Work and Discussion

In [11] the authors define the problem of "comparative text mining" (CTM) for a given text collection as "(1) discovering the different common themes across all the collections; (2) for each discovered theme, characterize what is in common

[9] The results obtained through the cluster graph significantly outperform TF∗IDF-based ranking.

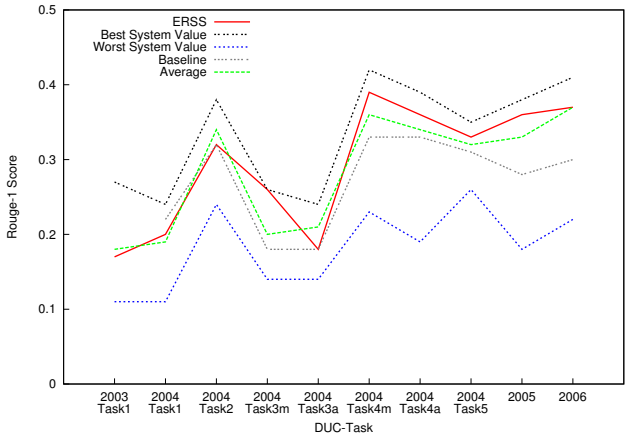

Fig. 11. Performance of the cluster graph algorithm on the DUC data 2003–2006

among all the collections and what is unique to each collection." They also apply a clustering strategy based on a cross-collection mixture model, but using only simple word-level statistics, which we believe is much less useful for creating summaries than our entity-based clustering approach.[10]

Similar work is done in the *Web Mining* community, however, they are more concerned with the static and dynamic structure of web pages (i.e., inbound/outbound links), making work like [12] more suited to information retrieval than summarization. The research area of *change summarization* is concerned with tracking a single document (or a document collection) over time and extracting new/fading topics. [13] evaluate such changes, providing the result in form of web page ranking lists.

Clustering approaches have long been applied to document analysis (see e.g. [14] for an overview), including summarization (e.g., [15]), but our work differs in that we cluster *coreferences* rather than individual (TF∗IDF-weighted) words.

7 Summary and Conclusions

Delivering tools for the automated analysis, structuring, and compression of information contained in natural language texts is an important need of end users overwhelmed with the task of manually filtering through search engine results. For the system engineer developing appropriate solutions, features such as robustness, flexibility, context-sensitivity and adaptability are essential properties. In this paper, we present the fuzzy coreference cluster graph algorithm as a possible solution that exhibits robustness and adaptability due to its reliance on fuzzy sets. It creates the highly flexible cluster graph data structure, which can also be employed for context-sensitive information filtering. It is also easy to implement and outperforms many existing summarization systems, most of which are in addition highly specific to a single task.

[10] A typical example cluster in [11] is the topic list *"port, jack, ports, will, your, warm, keep, down"*.

More work is needed on integrating information analysis and summarization algorithms, such as ours, into the desktop environments of knowledge workers. We present some ideas on this in [16], where the algorithm proposed here is implemented within a semantic desktop for building historians and architects analysing a 19th century encyclopedia written in German.

References

1. Klir, G.J., Folger, T.A.: Fuzzy Sets, Uncertainty, and Information. Prentice-Hall (1988)
2. Witte, R.: Architektur von Fuzzy-Informationssystemen. BoD (2002) ISBN 3-8311-4149-5.
3. Witte, R., Bergler, S.: Fuzzy Coreference Resolution for Summarization. In: Proceedings of 2003 International Symposium on Reference Resolution and Its Applications to Question Answering and Summarization (ARQAS), Venice, Italy, Università Ca' Foscari (June 23–24 2003) 43–50 http://rene-witte.net.
4. Angheluta, R., Jeuniaux, P., Mitra, R., Moens, M.F.: Clustering Algorithms for Noun Phrase Coreference Resolution. In: Proc. of 7èmes Journées internationales d'Analyse statistique des Données Textuelles, Louvain La Neuve, Belgium (March 10–12 2004) 60–70
5. DUC 2004 Workshop on Text Summarization, Boston Park Plaza Hotel and Towers, Boston, USA (May 6–7 2004) NIST. http://duc.nist.gov/pubs.html#2004.
6. Proceedings of the HLT/NAACL Workshop on Text Summarization, Edmonton, Canada (May 31–June 1 2003) NIST.
7. Witte, R., Krestel, R., Bergler, S.: ERSS 2005: Coreference-Based Summarization Reloaded. In: Proceedings of Document Understanding Workshop (DUC), Vancouver, B.C., Canada (October 9–10 2005)
8. Witte, R., Krestel, R., Bergler, S.: Context-based Multi-Document Summarization using Fuzzy Coreference Cluster Graphs. In: Proceedings of Document Understanding Workshop (DUC), New York City, NY, USA (June 8–9 2006)
9. Mani, I.: Automatic Summarization. John Benjamins B.V. (2001)
10. Lin, C.Y.: ROUGE: a Package for Automatic Evaluation of Summaries. In: Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004), Barcelona, Spain (July 25–26 2004)
11. Zhai, C., Velivelli, A., Yu, B.: A Cross-Collection Mixture Model for Comparative Text Mining. In: Proc. of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'04), ACM Press (2004) 743–748
12. Liu, B., Ma, Y., Yu, P.S.: Discovering unexpected information from your competitors' web sites. In: Knowledge Discovery and Data Mining. (2001)
13. Jatowt, A., Bun, K.K., Ishizuka, M.: Change Summarization in Web Collections. In: Innovations in Applied Artificial Intelligence: 17th Int. Conf. on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems. LNCS (2004) 653–662
14. Berry, M.W.: Survey of Text Mining: Clustering, Classification, and Retrieval. Springer (2003)
15. Radev, D.R., Jing, H., Styś, M., Tam, D.: Centroid-based summarization of multiple documents. Inf. Process. Manage. **40**(6) (2004) 919–938
16. Witte, R., Gerlach, P., Joachim, M., Kappler, T., Krestel, R., Perera, P.: Engineering a Semantic Desktop for Building Historians and Architects. In: Proceedings of the Semantic Desktop Workshop at the ISWC. Volume 175 of CEUR Workshop Proceedings., Galway, Ireland (November 6 2005) 138–152