A Study on Presentation Slide Reuse Support

Jie ZHANG

"A good beginning makes a good ending."

"A light heart lives long."

Abstract

Presentation slides are one of the most important tools for today's knowledge workers to present knowledge, exchange information, and discuss ideas for business, education and research purposes. Presentation slide composition is an important job for these presentation composers. To create presentation slides, one common practice is to start from existing slides. One of the primary reasons of slide reuse is to repurpose existing content in existing presentation files for various events, audiences, formats, etc. For example, when many researchers and lectures create new presentation slides, they reuse the lecture notes used in university courses and the reports presented in academic conferences. In business applications, people often combine materials used in previous presentations to create a summary, and modify existing slides in order to present to different audiences.

However, browsing these existing files and searching relevant materials is a time-consuming task. It is difficult to remember where all the contents reside. People often remember only some keywords, an image, a diagram or a slide. To this end the search and retrieval method for presentation slide reuse is necessary to develop.

Detecting reused materials in presentation slides benefits many presentation-related applications; e.g., assisting composer in tracking changes in multiple versions, understanding existing presentation slides, and assembling existing slides to make new ones, etc. Although the slide retrieval method for reuse and the method to compare different versions of a presentation file have been proposed, they are either based on slide-to-slide comparison or file-to-file comparison. In many cases, only an individual element such as a sentence, a table, an image or a diagram, is copied from one file to another, but overall the slides and the files differ significantly, and thus the reuse element cannot be identified by these methods.

Many knowledge workers demonstrate presentations using slide show software such as Microsoft PowerPoint, Keynote, and OpenOffice. Although these tools provide easy ways for slide preparation by inserting texts, images, animations, etc., traditional slide show software lack in the functions of the slide structure and the content support. Many researchers propose slide generation and composition methods for presentation slides. Some of them extract presentation contents from paper, while others based on the outline wizard. But all of the proposed method and system generates presentation slides automatically, that's to say, user have no choice about

the structure of the presentations, and cannot participate in the contents and the layouts of the slides.

In order to achieve these goals of presentation slides searching, managing and design supporting, this thesis introduces the respective approaches to effective presentation slide reuse support. The fundamental approaches to these researches are to propose content-based element search methods among presentation slides, then provide a method to manage presentation slides from the perspective of individual elements. Finally, a four-stage framework for presentation design support is designed to help users composite the presentation slides.

The first research is to provide search and retrieval methods for presentation slides. Unlike previous methods, we consider processing queries on elements in slides, including text, images, diagrams, etc. The users may select an element in a slide or just input some text or specify an image in their computers as a query. We return slides that contain relevant materials and highlight the relevant parts. To improve the search quality, we propose content-based methods to handle queries involving visual elements such as images, diagrams, and charts. To process the various types of queries, we propose different scoring functions to measure the relevance of the result, hence modeling the queries as top-k searches so that the results presented to the users are ranked in decreasing relevance order.

The second research is to propose a management method from the perspective of reuse elements. Different methods are developed to detect textual and visual elements reused in a slide repository specified by users. Textual elements are divided into sentences and further decomposed to bags of words. To detect reused sentences and consider the case when slide composers make minor modifications after reusing elements, similarities are taken into account to tolerate nuances between different versions. Likewise, the bag-of-words model is adopted to find reused visual elements, and utilize similarities to handle the case when they are transformed after reuse. The techniques to tackle the efficiency challenge are also introduced. Then two ways of visualizing reused elements are proposed. The first is to show the files and slides that use the same element in a timeline. The second is to construct a network of presentation files connected by reused elements.

The third research is to introduce a framework for the presentation design support. The framework consists of four stages, the story, the unit, the page and the layout. The users may select topics to design the presentation structure first, and input

or search the contents for the topic, then allot these components into different pages, finally decide the layouts of the slides. The content-based element search is integrated in the unit stage to help user find the useful contents. In this method, user can design the structure and choose the content of the presentation slides. About this work, the preliminary work is preformed. We have design the framework and do some experiments for evaluation.

We summarize the main contributions of the research in this thesis. First, the content-base element search method help user to find interested slides for the next reuse. Second, the management by individual reuse elements will help user get the overview and detailed understanding of the presentation slides. Finally, the four-stage framework supports the presentation slide design from the structure to the contents. Experiments on the proposed methods demonstrate that our work is effective and practical.

Acknowledgements

First of all, I would like to express my appreciation to my supervisor, Prof. Yoshiharu ISHIKAWA. During my last two year Ph.D. studies, he gave me many valuable guidance and encouragement, thanks very much for his instructive advice and useful suggestions on my thesis.

I would like to express my thanks to another advisor, Prof. Toyohide WATANABE. I am very fortunate to be his student during my first three years Ph.D. study. He has given me courage and confidence to continue my research, in spite of difficulties and frustration.

I would also like to thank the members of my thesis committee: Prof. Kenji MASE, Prof. Katsuhiko TOYAMA, and Assoc. Prof. Shigeki MATSUBARA, for they put considerable time and effort into their comments on the draft.

My sincere thanks also go to Dr. Chuan XIAO, co-supervisor of my Ph.D. studies, he have instructed and helped me a lot in the past two years.

I am very grateful to the other members of Ishikawa lab at Nagoya University for their kindness and friendship. Thanks also go to my team members in Watanabe lab, especially, Ms. Wei FAN, Mr. Koichi HANAUE and Mr. Yusuke KOYANAGI, for their assistance.

Last my thanks would go to my beloved family for their loving considerations and great confidence in me all through these years. I also owe my sincere gratitude to my friends who gave me their help and time in listening to me and helping me work out my problems during the difficult course of the Ph.D. studies.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Research Background

Slide presentations are one of the most important tools for today's knowledge workers to present information, exchange idea and discuss problems for business, research and educational purposes. Composers will create many presentations to use. Instead of starting from scratch, users tend to make new presentation slides by reusing existing materials. An online survey shows that more than 97% people compose presentation slides by reusing existing slides rather than starting from scratch [2].

Reuse is fairly common in these groups. Figure 1.1 shows the extent to which the content in each group is reused(the percentage of slides that contain reused content). From this figure, we can see that presentation reuse is common among research and business applications. One of the primary reasons of slide reuse is to repurpose existing content for different events, listeners, formats and so on. For example, when many researchers and lectures create new presentation slides, they reuse the lecture notes used in university courses and the reports presented in academic conferences. In business applications, people often combine materials used in previous presentations to create a summary, and modify existing slides in order to present to different audiences.

FIGURE 1.1: Percentage of slides with reused content [1]

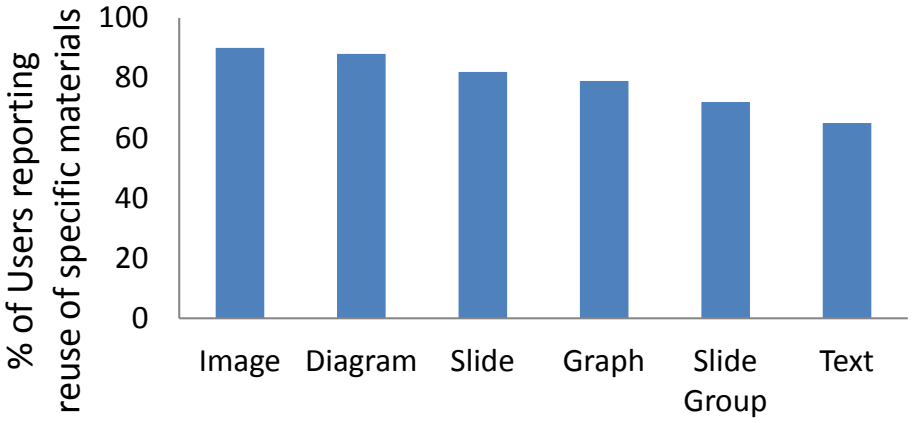


FIGURE 1.2: Reuse of different types of presentation materials [2]

Materials in presentation files that are often reused include text, graphical elements such as images and charts, and slides. In addition, presentation files are often used as data sources and templates for creating new files. Figure 1.2 shows the extent of reuse in different types of presentation materials. It can be seen that graphical elements and slides are the most frequently reused materials. However, existing systems barely offer any support for searching relevant graphical elements or slides. Presentation composers reported that they have to collect desired materials by searching a remembered element from existing presentation files, and then locate other required materials in these files.

Figure 1.3 shows the types (text-heavy, graphics-heavy, or both) of slides that contain reused elements. The "heaviness" was determined by the proportion of space on the slide taken by textual content and graphical content.

The notion of presentation slide reuse was proposed in [1]. An online survey was

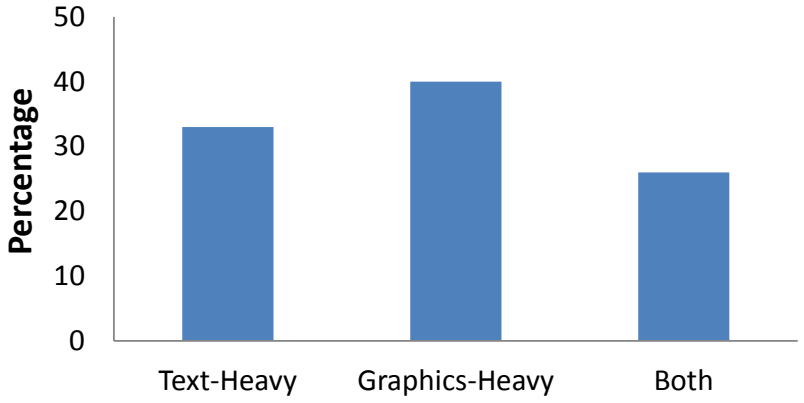


FIGURE 1.3: Percentage of different types of reuse [1]

conducted to study how often users start composing presentations from an existing presentation, and what types of materials are often reused. Based on the survey, a system was developed [2]. The users can select a slide as the query and the system recommends relevant slides stored on the users' machines. The similarities of text, image, and contextual information, i.e., file paths and names, are equally considered to compute an overall score to measure the relevance between a query and a result slide. Edit distance is employed to capture the text similarity. Image similarity is computed using the Jaccard coefficient over image IDs. Contextual similarity is also measured using Jaccard over the full paths and file names split by delimiters.

For the problem of presentation slide retrieval, many approaches focus on processing keyword queries. The notion of impression of keywords in a presentation was proposed and a search engine called UPRISE [4] was accordingly developed to retrieve relevant slides to keyword queries. Another system called SLIDIR [5] was developed to find slide images for a textual query based on machine learning technique. An XML-based system was developed [6] to solve the retrieval problem by extracting textual features to compute fuzzy relevance scores of database slides. In [7], a snippet-generation method was proposed to support the retrieval and browsing of slides based on the relationships between slides.

Besides the keyword retrieval from presentation files, an indexing and retrieval method in which slides are captured as images was also studied [8]. In addition to handling image queries, answering diagram queries is also investigated [9]. In this work, both query and database diagrams are decomposed into constituent shapes.

Then the types of shapes (e.g., rectangle, triangle, arrow, etc.) and their locations in the slide are compared to acquire the relevance score.

Prevalent presentation composition tools such as Microsoft PowerPoint and OpenOffice Impress mainly provide tools for slide composition and presentation, but they do not provide any support of showing an overview of the comparison of differences between multiple versions. For this reason, a presentation slide management system was developed to visually compare between different versions of presentation files [10]. The system compares between slides differences in text and pixel-level differences in images. It also provides an interactive visualization tool for users to examine differences between presentations. The difference between our work and this study is that our method focuses on exhibiting how individual elements are used in different slides, while their work focuses on presenting users differences between slides.

Another line of work studies presentation composition and generation methods. In [5], a presentation composition method was proposed on the basis of outline matching and implemented in a tool called Outline Wizard. Other presentation composition approaches include topic clustering [11] and hierarchical organization [12]. In addition, there are a few literatures that investigate generating slides from academic papers [13], discourse structure [14], or textbook chapters [15].

1.2 Research Objectives and Contributions

A primary challenge in slide reuse is to select desired materials from a collection of existing slides, then browse, compare and reuse or modify them. It is difficult to remember where all the contents reside. People often remember only some keywords, an image, a diagram or a slide. To this end, the search and retrieval system for presentation slide reuse was developed. Users may select a whole slide as a query and a similarity search over the text, images, and the contextual information of the slide is used to return relevant slides. It does not provide specific input for individual elements in slides and corresponding retrieval methods. Hence it is difficult for the

users to search by just text such as keywords or sentences, or graphical elements such as images, diagrams, or charts in a slide.

To this end, we investigate the problem of element search for presentation slide reuse. The users may select an element in a slide, or input just some text or specify an image in their computers as a query. We return the slides that contain relevant materials.

Although the slide retrieval method for reuse and the method to compare different versions of presentation files have been proposed, how to detect reused materials in presentation slides is still a problem. It can assist composer in tracking changes in multiple versions, understanding existing presentation slides, and assembling existing slides to make new ones. We investigate the problem of managing presentation slides from the perspective of individual elements. Different methods are developed to detect textual and visual elements reused in the presentation slide collections. In order to understand the detected reused elements, two ways of visualizing reused elements are proposed. The first is to show the files and slides that use the same element in a timeline, the second is to construct a network of presentation files connected by reused elements.

Based on the proposed methods, prototype systems are designed with user-friendly interface, which can be integrated into slide composition tools. To demonstrate the effectiveness of our methods, experiments on real presentation slide data is conducted using the prototype systems.

Many researchers propose slide generation and composition methods for presentation slides. Some of them extract presentation contents from papers, while others based on the outline wizard. But all of the proposed methods and systems generate presentation slides automatically, that's to say, user have no choice about the structure of the presentation, and cannot participate in the contents and the layouts of the slides. We investigate the problem of presentation design support and propose a design support system. The users may select topics to design the presentation structures first, and input or search the contents for the topics, then allot these components into different pages, finally decide the layouts of the slides.

Based on the proposed framework, a prototype system is designed. To demonstrate the effectiveness of our framework, experimental evaluation is conducted on the system preliminarily. We first check if all the contents are properly assigned into slides, then we make a questionnaire to see the usefulness to help users design the presentation slides.

Our contributions can be summarized as follows:

- We propose content search methods to process slide element queries for slide reuse.
- We propose an approach to presentation slide management by exploiting the notion of reused elements and develop techniques to detect and visualize reused elements in users' slide repositories.
- We propose a four-stage framework to help user design the presentation slides, and provide support methods in each stage to make the presentation slides contents fine grained and highly structured.
- We design a slide content search system integrated with a user-friendly interface to help users specify the materials they want to reuse and browse search results, and help users browse slides through reused elements.
- We conduct experiments to evaluate the effectiveness and the efficiency of the proposed methods with comparisons to existing solutions.
- We preliminarily implement a presentation design support system and evaluate the effectiveness and the usefulness of the system.

1.3 Research Map

Based on our objective and techniques related to our proposed methods, we show the research map in Figure 1.4. We use the database to store the dataset, provide

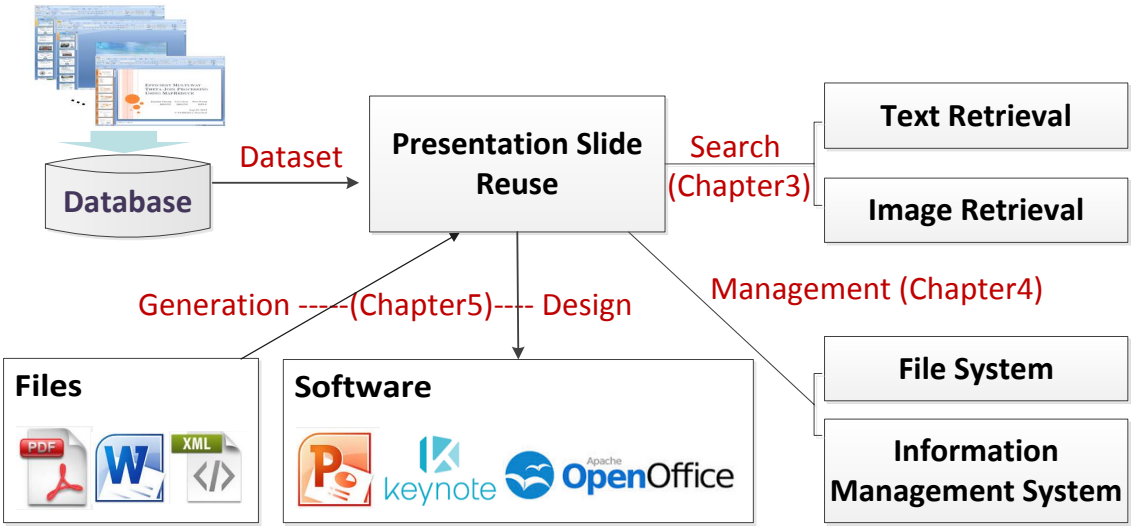


FIGURE 1.4: Research map of our work

data for the element search. The search method in Chapter 3 uses the text retrieval and image retrieval in information retrieval field. The management method in Chapter 4 is related to the file system and the information management system. Different from the existing method, we propose the reused element-based management of the presentation slides. The Microsoft Office, Keynote, OpenOffice software provide the environments to design and edit the presentation slides, but they lack the support of the structure design and content input techniques, our proposed method in Chapter 5 can provide these supports.

Many researchers focus on the presentation generation from other types of files automatically, such as the academic paper and reports. With the content retrieved, we can also provide the search support for the presentation design and generation.

1.4 Thesis Organization

The rest of the thesis is organized as follows.

In Chapter 2, we make a literature review on the related research and show our originality.

In Chapter 3, we introduce the content-based element search methods of different types of queries.

In Chapter 4, we introduce the method of managing presentation slides with reused elements.

In Chapter 5, we introduce the on-going work, the four stage presentation slide design support system.

In Chapter 6, we summarize these works and propose future studies in element search, reuse detection and design support system.

Chapter 2

Related Work

Our study on the presentation slide reuse touches upon the problems of the presentation reuse, slide retrieval, slide management, slide generation and slide composition. In this chapter, we will give the literature review about the related research separately.

2.1 Presentation Slide Reuse

Existing studies process presentation slide reuse as search or retrieval problems, "How can we help users to identify the best materials by building systems that assist in their retrieval?" [16]. According to the studies on this topic, the systems designed for reuse support need to: 1) support current practices of users [1], 2) facilitate the search process by considering what materials are remembered by users [17], and 3) represent search results to support the identification of users' desired materials [17, 18]. A deeper understanding of how users reuse presentation materials is first to design effective methods for reuse support. To this end, we investigate the problem of content search for presentation slide reuse and managing presentation slides from the perspective of individual elements.

Materials reuse is closely related to social diffusion, i.e., the spread of information spread in a social network system [19]. Since information is always spread with social

relations, people who live nearby are more likely to influence each other more than those who are distant, and interactions often occur between similar people [20]. A study by Jensen et al. [21] investigated reuse characteristics by logging the activities of 17 knowledge workers. It aims at tracking data provenance as well as evaluating the effectiveness of provenance cues for document recalls. Unlike [21], by tracking reuse elements, our goal is to facilitate content reuse in a large presentation file repository. Our methodology is also different from [21]: instead of tracking user actions using desktop instrumentation, we retrieve user's desired elements to reuse by searching in repository.

The notion of presentation slide reuse was proposed in [1]. An online survey was conducted to study how often users start composing presentations from an existing presentation, and what types of materials are often reused. Based on the survey, a system was developed [2]. The users can select a slide as the query and the system recommends relevant slides stored on the users' machines. The similarities of text, image, and contextual information, i.e., file paths and names, are equally considered to compute an overall score to measure the relevance between a query and the result. Edit distance is employed to capture the text similarity. Image similarity is computed using the Jaccard coefficient over image IDs. Contextual similarity is also measured using Jaccard over the full paths and file names split by delimiters. This method is effective to find relevant materials from the slides which were made by the users themselves, but its drawbacks are also apparent:

- The users must input whole slides as queries, while the users may only want to search for an element in a slide, such as a text paragraph, an image, or a diagram.

- Search quality is compromised due to using improper relevance functions (e.g., edit distance for text, which miss many relevant results if the order of words is changed) and retrieving images in a context-based manner. This is because they use edit distance for text similarity, while the order of words may be changed in the relevant materials, and hence yields very large edit distance. Moreover, the relevance of images is computed in a context manner by

comparing image IDs. Only images copied from one slide to another can be identified by this method. Hence if the relevant materials are bit-wise different from the query they will be completely missed.

- The similarity scores are computed for every pair of database slides and stored offline. Therefore, the queries must be slides that have been already stored in the database.

There have been a few studies on the problem of reuse detection in text documents in the last two decades, which can be divided into the following two categories:

- Overlap methods: Sliding windows are used. The window is fixed-size and shifted by one word. Identical or similar sliding windows that appear in two documents are detected and regarded as reused contents. Notable methods are: 0 *mod p* [22], winnowing [23], and Hailstorm [24].
- Non-overlap methods: Documents are split into non-overlapping text segments such as sentences or phrases. Then identical or similar text segments are detected as reused contents. This category of methods include hash breaking [25], DCT [26], qSign [27], etc.

Compared with text reuse detection, more attention has been paid on near-duplicate document detection due to the need for the reduplication task in Web search engines. Many effective and efficient methods have been proposed [28–31]. The major difference of the two problems is that only small document parts may originate from other sources, whereas in near-duplicates entire documents are almost replicated.

Our study for presentation slide reuse first provides a content-based element search for the interested slides, then gives a reused element detection method to track and compared the reused contents. Finally we propose a framework for users to design and compose the presentation slides.

2.2 Presentation Slide Search

Most studies related to academic contents focus on the slides search and retrieval. Yokota et al. [32] and Okamoto et al. [33] proposed the UPRISE (Unified Presentation Slide Retrieval by Impression Search Engine) system to retrieve lecture slides from slide and recorded video archives. Kobayashi et al. [34] proposed a method for retrieving lecture slides with UPRISE using pointer information.

Researchers proposed methods extracting important slides from unified content based on the metadata features of either a single medium or multiple heterogeneous media. Kitayama et al. [35] proposed a method to extract slides with corresponding video scenes based on the relations of slides and their roles. Le et al. [36] proposed an important slide extraction method that automatically generate digests from recorded videos of presentation demonstration. Wang et al. [37] described a process to automatically generate learning channels using the implicit semantic relationships in the lecture slides accompanied by recorded videos.

Lan et al. [38] proposed a theoretical framework to rank retrieved slides and analyze list-wise ranking algorithms. Desktop search systems rely on keywords input by users to retrieve desired materials. There are also studies on using provenance, access time and activity histories to support personal information search [21]. Tanaka et al. [39] considered the manipulation of complex objects, and proposed the notion of "element-based" generalization between the complex objects and reduction and abstraction operators. In our research, we propose the element search for the slide contents.

Finding the similarities between datasets has been studied extensively. UNIX's diff [40] is a widely used file differencing tool that highlights differences between two documents on the line level. Files are treated as ordered sequences of lines, and the longest common subsequence of lines between files is computed [41, 42]. Besides longest common subsequence, edit distance (also called Levenshtein distance) [42] is also widely adopted by file differencing tools to find lines that are inserted, deleted or substituted between different files. Similar techniques are used in plagiarism

detection [43] and biological sequences alignment [44]. SeeSoft [45] and SeeSys [46] are two tools for text file difference visualization. Viégas et al. [47] devised the history flow system that can be used to compare the edits on Wikipedia articles. Their systems focus on visualizing different versions of text documents, while the major task of our work is searching textual and graphical elements in presentation slides.

Seeing the absence of element search in existing solutions and the demand for usability and search quality, we analyze the inherent properties of various types of elements and devise corresponding methods. We propose a comprehensive solution that can support most common types of elements in presentation slides, including keywords, sentences, images, diagrams, tables, and charts. It provides a friendly input method so that users can easily specify what they want. We do not require the query to be from a slide stored in the database, whereas it is mandatory in [2]. To achieve high search quality, besides designing proper relevance functions, we choose a content-based manner for retrieving visual elements such as images and diagrams, as opposed to the context-based method used by [2]. In addition to these advantages, we develop efficient search algorithms so as to achieve very fast response speed. Next we detail our viewpoint with the framework and approaches to slide element search.

2.3 Presentation Slide Management

To locate desired materials, people often resort to information management systems and search systems [48]. As suggested by the research in the area of personal information management, there are two problems on retrieving desired materials: generating effective categorization and remembering the labels that support retrieval [49]. Since reuse of visual elements is a key factor, visual elements are often remembered and considered for presentation reuse [1].

By recording document history, knowledge workers are able to keep track of a growing amount of information. The history of a document, also called *provenance* [21], includes a list of people who have used the document or made modifications on it.

Reuse detection is useful when no provenance metadata is maintained. Drucker et al. [10] developed the slide reuse inferring techniques that can be used to support presentation version management. In addition to legitimate reuse, the detection of reused contents is also important for plagiarism detection. For example, COPS [25] is a system developed for the detection of complete or partial text copies. Near-duplicate detection techniques have been widely used. For example, data integration and cleaning [50], detecting from regulators to letters [51], and helping reviewers check if an academic paper submission closely matches any published work [23]. A variety of techniques have been proposed, including fingerprinting [22, 23, 28], locality-sensitive hashing [54] and information retrieval metrics [52, 53]. Our task is similar to near-duplicate detection. We are interested in reuse detection in a collection of presentation files that are broken into slides, each of which can be treated as a document. [2] and [10] used Edit Distance (Levenshtein Distance) similarity to search the text results, but it is sensitive to the insert and delete operations. Therefore we use Jaccard similarity in our research, since it is insensitive to the order of words, and hence more relevant results can be returned.

Many approaches have been developed for text analysis. For the problem of text summarization, the approaches can be divided into two categories: sentence-based and keyword-based. In sentenced-based approaches, most salient sentences in a document are identified [55, 56]. For example, a browsing method was proposed by Murai and Ushiama [57] to present users with a review-based recommendation of attractive sentences in fiction books. However, reading a few sentences per document is time-consuming, especially when reader are given a large number of books. In keyword-based approaches, documents are summarized by topics, which are characterized by a set of keywords [58–60].

In our management system, we process textual elements in a sentence unit. Since when the text in a slide is reused, composers may copy one or more sentences from

one slide to another, but overall the texts in both slides differ significantly. For this reason, reuse textual elements are chosen to detect on the sentence level.

Besides text summarization, various information visualization methods have been developed to visualize the result of text analysis. Basically there are two main categories of approaches: metadata-based and content-based. For metadata-based methods, a time-based visualization method was proposed in [61] to show text summarization results by a text analytic engine. In [62], a visualization method was proposed to analyze emails and visualize the relationship between email senders and receivers. For content-based methods, Viégas et al. [63] proposed to visualize keywords based on TF-IDF scores in a document collection. Strobelt et al. [64] proposed to create a compact visualization of a document using a mixture of TF-IDF-based keywords and images. Chen et al. [65] and Iwata et al. [66] proposed methods to visualize clustering results on a document collection. Other researchers have focused on representing text at the words or phrases level, e.g., WordTree [68], Phrase Net [69], TextArc [67], and FeatureLens [70].

Prevalent presentation composition tools such as Microsoft PowerPoint and OpenOffice Impress mainly focus on providing tools for slide composition and presentation, but they do not provide any support of showing an overview of the differences between multiple versions. For this reason, a presentation slide management system was developed to visually compare between slides, differences in text and pixel-level differences in images [10]. The system compares pixel-level visual and the text differences on each slide. It also provides an interactive visualization tool for users to examine differences between presentations. The difference between our work and this study is that our method focuses on exhibiting how individual elements are used in different slides, while their work focuses on presenting users differences between slides.

We investigate the problem of managing presentation slides from the perspective of individual elements. We first develop different methods to detect textual and visual elements reused in a slide repository specified by users. Then we propose two ways of visualizing reused elements. The first is to show the files that use the same

element in a timeline. The second is to construct a network of presentation files connected by common elements.

2.4 Presentation Slide Generation

Existing studies on slide-making support mainly focused on slide generation. Shibata et al. [71] proposed to convert Japanese documents to slide representations by parsing the discourse structure in documents and representing in an outline format result tree. Beamer et al. [73], Mathivanan et al. [72] and Yasumura et al. [74] separately proposed systems that generate presentation slides from academic papers. In their methods, information is extracted from a paper by the TF-IDF-based method. Sentences, figures, and tables are assigned to slides, and important phrases are identified with bullets.

However, for the textual elements in slides, these methods only concentrate on the consistency of the document structure. Kan [75] proposed a system to discover, present and align document and slide pairs. The method by Yokota et al. [32] can extract important information from slides. It assumes that the most significant description of a word in a text is the passage in which the word appears the most frequently. Hayama et al. [76] proposed a method for aligning slides and academic papers using a hidden Markov model.

Chen [77] developed a system that searches in a slide database with extracted presentation structures. It finds particular items such as images, diagrams, slides, etc. by answering queries specified with a generated ontology. On the contrary, our method allows users to specify queries with keywords, sentences, images, etc. and then search contents they are interested in. Shibata et al. [71] proposed a slide generation method by constructing discourse structure from text segments. To form the discourse structure, a clause or a sentence is considered as a unit first, and then it detects coherence relations between clauses and sentences with cue phrases, word or phrase chains, and similarities between sentences. The coherence relations are

then used to compute the depth of indentation for clauses or sentences to be put in a slide.

There are a few studies that aim at generating slides automatically from existing resources such as academic papers using text summarization techniques. Utiyama et al. [78] proposed a slide generation method that creates slides from a text segment with annotations which are called the GDA tagset. This method consider parse-tree bracketing, semantic relation and coreference in the tagset to generate slides. The semantic relation considered is mainly rhetorical relation like cause, concession and elaboration relations. Although slides are generated based on semantic relationship, the slides follow the standard format of bullet-point lists in prevalent presentation software. In order to help users better understand the contents in slides, it is necessary to diagrammatically allocate textual and graphical elements so that the discourse structure is represented effectively on slides. Miyamoto et al. [79] proposed a slide generation method from academical papers in the LaTeX format. Sentences having parallel relations are extracted from papers, and slides are generated in an itemized format using conjunctive words and phrases.

Different from automatic generation in the work above, we propose a four-stage framework to help user generate the presentation slides according to the user's choice, from the structure to the contents in each unit and page.

2.5 Presentation Slide Composition

There are several existing presentation slide composition systems that provide tools to compare different versions of the same presentation [10, 80, 81]. Other slide composition methods include topic clustering [11], outline matching [5], and hierarchical organization [12]. However, there are hardly any existing investigations on the design of recommender systems that support both slide composition and reuse.

Zellweger [82] devised a system to create multimedia documents embedded with multiple scripted path. The Pallete system proposed by Nelson et al. [83] provides

a tangible, paper-based interface for users to organize presentations. Unlike most commercial slide composition software that focus on creating single linear sequences of slides, several research systems have been developed to support multiple path in presentation. Pad [80] and CounterPoint [84] are two systems with zoomable interfaces that allow users to position slides spatially on an infinite canvas and navigate to any slide in a presentation file with hyperlinks.

Moscovich et al. [85] developed a system where users are allowed to choose between multiple paths while they are giving the talk. All of these systems aim at facilitating the process of presentation customization. Our systems target comparing and managing multiple presentation files and hence is orthogonal to these methods and can be used in conjunction with any of them.

Yasumura, et al. [74] proposed a slide composition method to generate slides in XHTML format from academic papers produced in TeX format. In their method, a paper manuscript is input first, and the number of slides for each section of the paper is computed. Appropriate layout template is then selected for each section and subsection. Wang et al. [86] investigated the problem of supporting slide composition for classroom presentations. The skeleton of a presentation is extracted from a textbook on the basis of the correspondence between existing slides and textbook contents. Hierarchical structure of keywords is represented in the skeleton in order to help users compose slides.

From the perspective of reusing existing presentation slides, some systems have been developed to support the assembling of slides for new presentations [2, 10]. In these systems, interactive interfaces are provided to visualize and align multiple presentations based on the similarities between slides. Outline Wizard [5] is a slide composition system that uses the outline-based search techniques. Given a repository of slides, it automatically extracts outlines from the hierarchical structure of the slides. Users may specify an outline, and the system retrieves slides from the repository that match users request. These systems facilitate the process of slide composition from existing ones.

All the above work disregard the user's role by how to composite the slides according to his intention. We propose a system that to help user design the structure, then input or search contents, finally allot the component to each slide according to the user's choice. Our method, therefore, focuses on the role of user's content decision and help users with the structure and component design of the presentation slides.

Chapter 3

Content-based Element Search for Presentation Slide Reuse

3.1 Introduction

Presentation slides are one of the most frequently used tools for business, education, and research purposes. One of the common actions in building new presentations is to build slides with existing ones. An online survey shows that most people compose presentation slides by reusing existing materials rather than starting from scratch [2]. One of the primary reasons of slide reuse is to repurpose existing content for different audiences, events, formats, etc. For instance, some massive open online course (MOOC) platforms such as Coursera and edX are oriented towards university students, including research students. Hence the courses they provide involve many recent advances from the research community, and the lecturers often merge the outcomes of their research into the courses. When creating presentation slides, they will reuse the lecture notes used in university courses and the reports/-tutorials presented in conferences and symposiums. In business applications, people often modify existing content for the purpose of presenting to different audiences or creating a summary by combining materials used in previous presentations.

However, browsing these files and searching relevant materials is a time-consuming task. It is difficult to remember where all the contents reside. People often remember only some keywords, an image, a diagram or a slide [1].

To this end, the search and retrieval system for presentation slide reuse was developed [2]. Users may select a *whole slide* as a query and a similarity search over the text, images, and the contextual information of the slide is invoked to return relevant slides. Moreover, a system that compares different versions of the same slides has also emerged [10]. However, neither of them provides specific input for *individual elements* in slides and corresponding retrieval methods. Hence it is difficult for users to search for just texts such as keywords or sentences, or graphical elements such as images, diagrams, or charts in a slide. In many cases, there is no appropriate slide on hand to serve as a query, but users may know what they are searching for exactly: e.g., they want to input keywords or open an image instead of starting with an existing slide that contains them. In addition, their image processing modules are *context-based* search by comparing image IDs, and thus only applicable to the images that are copied from one slide to another. If the relevant images are bit-wise different from those in the query (e.g., by a change in resolution), they will be completely missed.

About this research, we study the problem of content search for presentation slide reuse. Unlike previous methods, we consider processing queries on *elements* in slides, including text, images, diagrams, etc. Users may select an element in a slide, or input just some text or specify an image in their computers as a query. In consequence they are freer to request what they want to reuse. We return slides that contain relevant materials, and we do not require the query to be from a slide already stored in the database, whereas it is mandatory in [2].

As opposed to the context-based methods used by [2] and [10], we propose *content-based* methods to handle queries involving visual elements such as images, diagrams, and charts, hence to improve the search quality. The bag-of-words model is employed to convert the visual elements in a slide to bags of visual words. For diagram queries, we also take into account the shapes that constitute the diagram as well as

the relations between their locations. To process the various types of queries, we propose different scoring functions to measure the relevances of results, hence modeling the queries as top-k searches so that the results presented to users are ranked in decreasing relevance order. For the sake of efficiency, we devise algorithms to find the top-k answers leveraging the inverted indexes built on the database slides. Based on the proposed methods, a prototype system with a user-friendly interface is designed, and it can be integrated into slide composition tools for slide reuse. The experimental evaluation on real presentation slide data demonstrates the superior search quality of our methods to alternative solutions as well as the efficiency of our methods against the method without indexes.

Our contribution can be summarized as follows:

- We propose content search methods to process slide element queries for slide reuse.
- We design a slide content search system integrated with a user-friendly interface to help users specify the materials they want to reuse and browse search results.
- We conduct evaluation experiments for the effectiveness and the efficiency of the proposed methods with comparisons to existing solutions.

As far as we know, this is the first piece of work that systematically studies the methods to support the retrieval for most types, if not all, of elements in presentation slides.

The proposed methods are introduced in detail. Section 3.2 proposes the conceptual viewpoint with the framework and the approaches to slide content search. Section 3.3 introduces the methods to process individual element search. Section 3.4 discusses the extension to the processing of multiple element queries and slide queries. Section 3.5 introduces the user interface of the prototype system and presents experiment results as well as our analyses. Finally, Section 3.6 concludes this chapter.

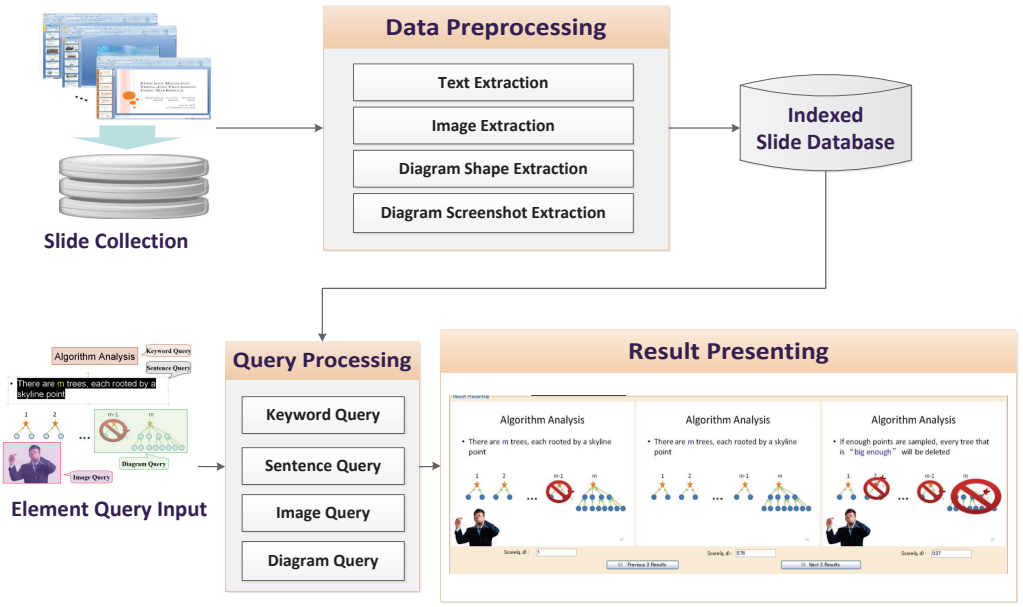


FIGURE 3.1: An overview of slide element search framework

3.2 Conceptual Viewpoint with Framework and Approaches

Figure 3.1 shows the framework of content-based slide element search, composed of four main modules: query input, data preprocessing, query processing, and result presenting. Next we introduce these modules respectively.

3.2.1 Query Input

Often reused presentation materials include textual and visual elements (e.g. graphs, diagrams, images) [2]. Our framework supports the search of an element in a slide. In addition, unlike the method in [2], the query does not need to be from a slide already stored in the database. Users begin by loading a slide file, locate a slide, and then initiate the search by selecting one of the following elements in a slide, as shown in Figure 3.2: (1) a text snippet, (2) an image, and (3) a diagram. For text queries, if the selected text consists of no more than five words, we regard it as a keyword query; otherwise it is regarded as a sentence query. For diagram queries, since they usually consist of individual shapes (e.g., quadrangle, triangle,

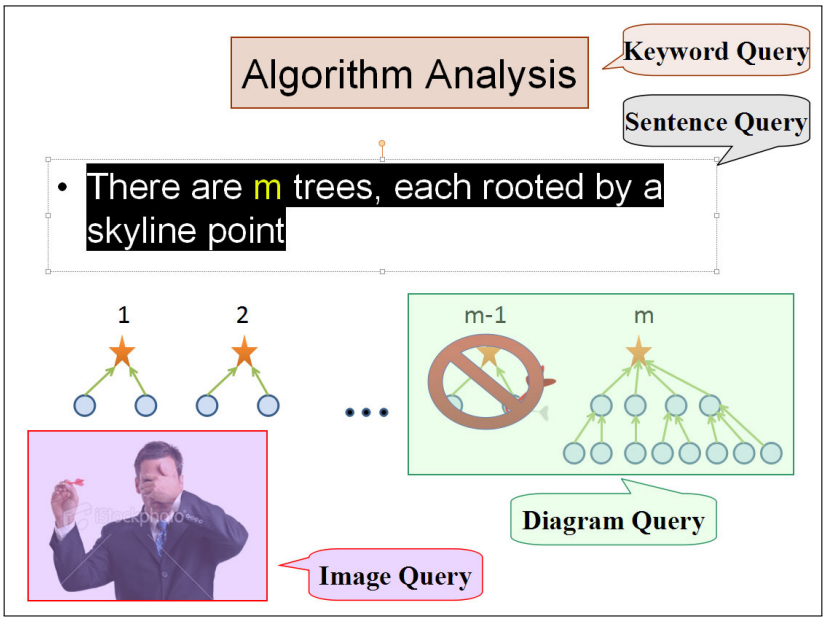


FIGURE 3.2: Element query input

arrow, etc.), we allow users to drag the cursor around a rectangle area that contains the diagram, like using a snipping tool. For other element types, a table can be searched with a text query as users may select the text in the table and submit it, and a chart can be searched with an image query.

Besides selecting directly from a slide, users may also manually type in the text to input a text query, or select an image stored in their computers to submit an image query.

3.2.2 Data Preprocessing

A database of presentation slides is built by offline and stored on disk in order to serve future queries. Users may specify the location where the repositories of slides are stored in their computers.

In this module, we extract the following elements from the database slides: texts, images, and shapes. Tables and charts are treated as texts and images, respectively. The elements are then indexed to support different types of queries. The detailed index construction will be presented in Section 3.3.

3.2.3 Query Processing

The module of query processing is divided into several parts according to the types of queries: text, image, and diagram. Text query is further divided into two subcategories, keyword query and sentence query. The methods to process the various types of queries will be introduced in Sect. 3.3.

3.2.4 Result Presentation

The search results are sorted by relevance and shown in a list of slides. We show three results at a time, and users may click the "next" button to view the next three results. Users are also notified that the remaining results may be irrelevant so that they may decide whether or not to see the remaining results. The page number and the file name are given so they can locate the slide by themselves to reuse the materials in presentation composition tools.

We design a system according to the above framework. The user interface of the system will be shown in Section 5.4.1.

3.3 Query Processing Methods

We introduce the processing methods for different types of queries, followed by the improvement on efficiency.

3.3.1 Keyword Query

For keyword query, this is a well studied problem in the area of information retrieval. The most prevalent inverted index-based approach [87] is chosen to handle this type of queries. We extract texts from the database slides. The text in each slide is tokenized and stemmed into a bag of words with white space and punctuations. Then an inverted index [87] is built, mapping each word to a list of slide IDs that

contain the word, called a postings list. For each keyword input by the users, the list of slide IDs that contain the keyword is obtained. Merging these lists will result in the slide IDs that contain all the query keywords.

To rank the result slides, we first assign weights to the input keywords by tf-idf weighting scheme:

$$w_{t,d} = (1 + \log tf_{t,d}) \cdot \log \frac{|D|}{df_t}, \tag{3.1}$$

where $tf_{t,d}$ is the term frequency of the keyword in the slide, D is the slides total number in the database, and df_t is the document frequency of the keyword in the database. Here, the subscript t and d is the term and document. Then the score of a slide with respect to the query is

$$score(q, d) = \sum_{t \in q \cap d} w_{t,d}, \tag{3.2}$$

where q denotes the set of query keywords, and d denotes the text of the slide.

3.3.2 Sentence Query

The processing of sentence query is more complicated than keyword query. Since the content in the database slides may not completely match the query, we adopt the idea of similarity search, and use the following equation to capture the score between the query and a database slide:

$$score(q, d) = \frac{|q \cap d|}{|q|}. \tag{3.3}$$

Here q and d are both bag of words, q denotes the set of query sentence, and d denotes the text of the slide. The above scoring function is similar to the Jaccard coefficient defined as the size of the intersection divided by the size of the union. The difference is that we use the size of the query rather than the union as the denominator, so that slides that approximately *contain* the query are with high scores. Compared with the edit distance-based scoring function used in [2], our scoring function is insensitive to the order of words, and hence more relevant results

can be returned. Edit distance is a measure to quantify how dissimilar a string is to another. It counts the minimum number of edit operations, including insertion, deletion, and substitution of a character, to transform one string to the other. For example, a query of "the telephone was invented by Alexander Bell in 1876", and a slide containing "in 1876, Alexander Bell invented the telephone". The score using Equation 3.3 is $7/9 = 0.78$, while the edit distance-based score is only $1 - \frac{ed(q,d)}{\max(|q|,|d|)} = 1 - \frac{40}{52} = 0.23$.

Because three results are shown at a time in the result presenting module and ranked by relevance, this problem is equivalent to a progressive top-k search where $k = 3$, 6, 9, etc. We will discuss the efficient computation of the top-k search in Section 3.3.5, and continue with other types of queries first.

3.3.3 Image Query

To retrieve images using content information, we adopt the bag of words model [88], a prevalent approach in computer vision. It represents images as bags of elementary image patches called visual words, as shown in Figure 3.3. First, a dictionary of visual words is created, called visual vocabulary. Then each database image can be represented using the words in the visual vocabulary, and indexed to support image queries. In our framework, we choose to tokenize the individual images contained in the database slides into visual words. In order to build a visual word vocabulary, we first detect interest regions in the images using Hessian-affine detector [89] widely used in visual word-related studies, which provides satisfactory performances [90] It is insensitive to affine transformations such as scaling, reflection, rotation, etc. These regions are described as 128-dimension SIFT descriptors, and then clustered by a hierarchical k-means algorithm [91], where each cluster representing a visual word. Like textual words, we also build an inverted index to map each visual word to a list of containing slide IDs.

To process image queries, the 128-dimension SIFT descriptors of the query image are generated first. To convert them into visual words, we compare them with the

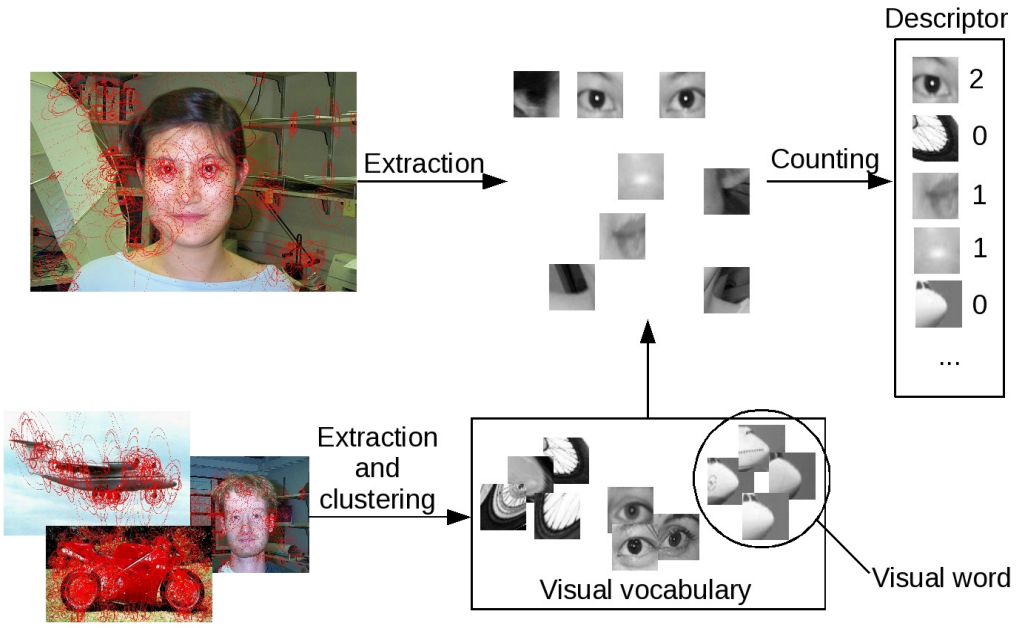


FIGURE 3.3: Bag of words model for image retrieval [3]

visual vocabulary, i.e., the set of cluster centroids found by the hierarchical k-means algorithm during the data preprocessing step, and assign each descriptor to a visual word by selecting the nearest cluster centroid. Moreover, if the Euclidean distance between the descriptor and the centroid is greater than the maximum distance from a database image descriptor to a centroid, we treat it as a visual word beyond the vocabulary, meaning it does not appear in the database slides. The query image is hence converted to a bag of visual words. In order to rank the relevance, the scoring function between a query image and a database image is defined by

$$score(q, d) = \frac{|q \cap d|}{|q \cup d|}, \tag{3.4}$$

where q and d are two bags of visual words. It is equivalent to the Jaccard coefficient, and has been adopted for near duplicate image detection [92], based on the intuition that similar images share most of their visual words.

Like sentence queries, the image queries are thus converted to top-k searches. Its efficient processing will be discussed later.

FIGURE 3.4: Example of diagram query

3.3.4 Diagram Query

Diagram queries are composed of individual shapes. For diagram queries, we take into consideration two factors: the locations of the individual shapes and the overall appearance of the diagram.

For the shapes contained, the relevance should reflect the relationship between their locations. For example, Figure 3.4(a) shows a diagram consisting of three shapes: a circle, a line, and a rectangle. Note we consider neither the text in the shapes nor their dimensions but only their *types*. Because the circle is on the bottom left of the line and the rectangle, the the shapes in a relevant result should keep approximately the same relationship. Since a database slide may contain not only the query diagram but also other shapes, it is difficult to map shapes in the query diagram to the shapes in the database slide and check their mutual relationships (the time complexity is $O(\binom{m}{n} \cdot n!)$, where m and n are the numbers of shapes in the database slide and the query diagram, respectively).

Instead, we measure the orders of shape locations in x and y-axes, respectively. Without loss of generality, we use x-axis to describe our method. The shapes in the query diagram are sorted according to the x-coordinates of their centers first. The shapes in the database slide are also sorted according to this order. Then we have two sequences of shape types, and the *longest common subsequence* measures the common part of them.

Dividing the length of the longest common subsequence by the number of shapes in the query, we have a ratio for the x-axis, capturing how much of the query sequence

is contained by the database sequence. The y-axis is processed in the same way. The two ratios are summed up and then divided by two to get the shape relevance between a query and a database slide (Equation 3.6).

$$score_{shape}(q, d) = \left(\frac{|LCS(q_x, d_x)| + |LCS(q_y, d_y)|}{2|q|} \right), \tag{3.5}$$

where q_x, q_y, d_x, and d_y denote the shape sequences of the query and the database diagrams on x and y-axes, respectively, and LCS denotes the longest common subsequence.

Example 3.1. *Consider a query diagram shown in Figure 3.4(a) and a database diagram shown in Figure 3.4(b). We use* `C`*,* `L`*, and* `R` *to denote the shape types circle, line, and rectangle. On x-axis, the shape sequences of the query and the database diagrams are* $\{$ `C`, `L`, `R` $\}$ *and* $\{$ `C`, `C`, `L`, `R` $\}$*, respectively, supposing we sort from left to right. The longest common sequences is* $\{$ `C`, `L`, `R` $\}$*. On y-axis, the shape sequences of the query and the database diagrams are* $\{$ `R`, `L`, `C` $\}$ *and* $\{$ `C`, `L`, `R`, `C` $\}$*, respectively, supposing we sort from top to bottom. The longest common sequences is* $\{$ `L`, `C` $\}$*. Thus the shape relevance score is* $\frac{3+2}{2\times3} = 0.83$*.*

For the overall appearance, the bag of words model is applied to process diagram queries. In order to build a visual word vocabulary, we take the screenshot of the each database slide as an image and extract visual words, and consequently the visual words can cover the regions of all the diagrams in the slide. Note that we only need to consider slides that contain at least one shape. An inverted index is then built on the visual words in a similar way to that of image queries. The tokenization of diagram queries is the same as that of image queries, except that we use the above-mentioned visual vocabulary for diagrams and only the user-selected area is used as the screenshot.

After converting the diagram screenshot into visual words, the query becomes the problem of finding the slide screenshots that contain approximately all the visual words of the query. We use Equation 3.6, which is essentially the same scoring

function as for sentence queries, to measure the relevance score in terms of visual appearance:

$$score_{visual}(q, d) = \frac{|q \cap d|}{|q|}. \tag{3.6}$$

The overall relevance score is the combination of the shape relevance and the visual relevance (Equation 3.7).

$$score(q, d) = score_{shape}(q, d) + score_{visual}(q, d). \tag{3.7}$$

Now we have converted the query processing for sentences, images, and diagrams to three top-k searches with different scoring functions. Next we introduce how to efficiently compute the results of these top-k searches.

3.3.5 Improving Search Efficiency

We begin with sentence queries. An immediate solution is utilizing the inverted index which has been built for keyword queries, computing the score for every slide that contains at least one word in the query, and ranking them thereafter. This method is usually prohibitively expensive due to the existence of frequent words, e.g., "the" and "of", in the query [1]. Inspired by the idea of prefix filtering [93], we sort the words in the query by a global ordering defined on the word universe. Then we check the word in the query one by one, if the the slide does not share the word, the maximum score it can achieve can be computed through the following lemma.

Lemma 3.1 (Upper bound of score for sentence query)**.** *Given a query q, the words of which are sorted by an ordering $\mathcal{O}$. If its shares none of the first i words with a slide d, the upper bound of the score is $\frac{|q|-i}{|q|}$.*

For the sake of efficiency, we choose to sort the words in the query in the order of increasing document frequency – the number of database slides that contain

[1] Although stop words such as "the" and "of" can be filtered prior to the processing, frequent words may still exist.

the word – and hence the rarest words come first in the query. An important observation is that if a slide shares a rare word with the query, it is likely that the slide approximately contains the query; otherwise, unlikely.

Example 3.2. *Consider a query* $q =$ *"*`Alexander Bell is the inventor of the telephone`*". Assuming the order of document frequency is* `Bell` $<$ `inventor` $<$ `Alexander` $<$ `telephone` $<$ `is` $<$ `of` $<$ `the`*. Then after tokenizing and sorting the query becomes* $q = \{$ `Bell`, `inventor`, `Alexander`, `telephone`, `is`, `of`, `the`, `the` $\}$*. If a slide does not contain the first two words* `Bell` *and* `inventor`*, its maximum possible score is* $\frac{6}{8} = 0.75$*.*

Algorithm 1 shows the pseudo-code of the top-k search algorithm, where I_w denotes the postings list of the word w in the inverted index, T is a set top temporary results, and $T[k].score$ denotes the score of the k-th temporary result in current state. From the above observation, we process the words in the query from the rarest side to the most frequent side (Line 2) and access the postings list of the words in the inverted index (Line 4). For each slide in the postings list, we compute its exact score and update the top-k results (Line 5 – 7). Because the scores of the unseen slides are upper-bounded by Lemma 3.1 and monotonically decreasing with an increasing i, if the upper bound is no better than the current k-th result, we pause the search and return the top-k results (Line 12).When the users click the "next" button, the search is continued and the k'-th results are computed, where $k' \in [k+1, k+3]$.

In order to notify the users that the remaining results may be irrelevant, we compare the scores of the k-th result and the $(k+1)$-th result. If $\frac{score_{k+1}}{score_k}$ is smaller than the threshold θ, the search process is paused with the notification, and then the users may decide whether or not to continue. The reason why we do not merely compare $score_k$ with a fixed threshold is that it is difficult to choose a proper threshold to separate the relevant and irrelevant results for queries with different length and contents. Instead, comparing adjacent results can detect where the relevance suddenly drops.

Algorithm 1: SentenceTop-kSearch (q, D, I)

Input : q is bag of words sorted by increasing document frequency; D is a collection of slides; I is the inverted index that maps each word to a list of slides.

Output: Top-k slides ranked by Equation 4.1.

```
1  T ← ∅;
2  for i = 1 to |q| do
3  |  w ← q[i];
4  |  foreach d ∈ I_w do
5  |  |  score(q, d) ← |q∩d| / |q|;
6  |  |  if score(q, d) > T[k].score then
7  |  |  |  UpdateResults(T, d);
8  |  |  end if
9  |  end foreach
10 |  UB ← (|q| − i) / |q|;
11 |  if UB ≤ T[k].score then
12 |  |  ReportResults (T) ;          /* pause here and report results */
13 |  end if
14 end for
```

We note that this top-k search algorithm can be applied to any bag of words model as well as weighting schemes such as BM25 and other tf-idf-based functions. In the rest of this section, we will employ the same algorithm framework to handle image and diagram queries.

For the top-k search of image queries, since an inverted index has been built on the visual words of the images of database slides, we can use it to process image queries in a similar way to the method for sentence queries. Likewise, we sort the visual words in the query in the increasing document frequency order to efficiently find the top-k answers with Equation 3.4 as the ranking function. Similarly, we have the following lemma to bound the score between a query and a database image if they do not share the first few visual words.

Lemma 3.2 (Upper Bound of Score for Image Query)**.** *Given a query q, the visual words of which are sorted by an ordering $\mathcal{O}$. If its shares none of the first i visual words with an image d, the upper bound of the score is* $\frac{\min(|q|-i,|d|)}{|q|+|d|-\min(|q|-i,|d|)}$.

According to Equation 3.4 and Lemma 3.2, we modify Algorithm 1 by changing the scoring function and the upper bound evaluation, and obtain Algorithm 2. After

Algorithm 2: ImageTop-kSearch (q, D, I)

1 Replace Line 5 in Algorithm 1 to "$score(q, d) \leftarrow \frac{|q \cap d|}{|q \cup d|}$" ;

2 Replace Line 10 in Algorithm 1 to "$UB \leftarrow \frac{\min(|q|-i,|d|)}{|q|+|d|-\min(|q|-i,|d|)}$" ;

retrieving the images that are close to the query, the slides that contain those images are shown to the users.

We also note that by exploiting the ordering of words, our method can be extended to support other common similarity/distance functions for image retrieval which are applied on sets/bags of visual words or vectors of visual word frequencies [3] (e.g., L_p-distance and cosine similarity).

Equation 3.7, the ranking function for the top-k search of image queries, consists of two parts. Considering the two factors: (1) we have an inverted index built on the visual words of screenshots, and (2) the computation of shape relevance poses more overhead ($O(mn)$ time using dynamic programming for longest common subsequences) than the computation of visual relevance ($O(m+n)$ time), our top-k search algorithm for diagram queries is designed on the basis of a biased strategy – use visual relevance to upper bound the unseen results and compute the shape relevance on-the-fly. We sort the bags of visual words in the query according to the increasing document frequency order, and then use these words to access the inverted index built on visual words. To compute the upper bound of the scores of the unseen slides, we assume that their shape relevance can achieve the highest value 1, and thus have the following lemma.

Lemma 3.3 (Upper Bound of Score for Diagram Query)**.** *Given a query q, the visual words of whose screenshot are sorted by increasing document frequency, if it shares none of the first i visual words with a slide d, the upper bound of the overall score is $\frac{|q|-i}{|q|} + 1$.*

Therefore, by scanning the visual words from the rarest side, the upper bound of scores of the unseen slides is monotonically decreasing. The above techniques constitute the top-k search algorithm for diagram queries, the pseudo-code of which is captured by Algorithm 3.

Algorithm 3: DiagramTop-kSearch (q, D, I)

1 Replace Line 5 in Algorithm 1 to "$score(q,d) \leftarrow \frac{|LCS(q_x,d_x)|+|LCS(q_y,d_y)|}{2|q|} + \frac{|q \cap d|}{|q|}$" ;

2 Replace Line 10 in Algorithm 1 to "$UB \leftarrow \frac{|q|-i}{|q|} + 1$" ;

3.4 Extensions to Multiple Element Query and Slide Query

In this section we briefly comment on the methods to deal with the scenario where users select multiple elements or a slide as the query.

To handle the query composed of multiple elements, we define the total relevance score as the sum of the respective score of each element in the query. There are two subtle cases. The first is that in Eq. 3.2, the relevance score of keywords can be larger than 1, and thus we normalize it before summing up by dividing it by the maximum relevance score of the result using only the keywords as the query. The second is that if the query contains multiple number of elements of the same type, we divide the score of this type by its number. For example, if a user selects a sentence, two images, and a diagram as the query, the total relevance score is $score(q,d) = score_s + \frac{score_{i_1}+score_{i_2}}{2} + score_d$, where $score_s$, $score_{i_1}$, $score_{i_2}$, and $score_d$ denote the relevance scores of the sentence, the first image, the second image, and the diagram, respectively.

To compute the top-k results and optimize for efficiency, our method is similar to what we use to deal with diagram queries. We utilize the inverted index of the most selective element in the query, and assume that the relevance scores of other elements can achieve the highest value 1. We define the order of selectivity as diagram, image, sentence, and then keyword, from highest to lowest. For example, considering a query composed of a sentence and an image, since image is more selective than sentence, we sort the bags of visual words in the image according to the increasing document frequency order, and then access the inverted index built on visual words. If the query share none of its first i visual words with the images in a slide d, the maximum possible total relevance score is $\frac{\min(|q|-i,|d|)}{|q|+|d|-\min(|q|-i,|d|)} + 1$.

To handle the query of a whole slide, we first find out the elements in the query slide. Tables are treated as text and charts are treated as images. Then the scenario becomes the same as a multiple element query.

3.5 Experiments

We design a slide element search system for presentation slide reuse according to the proposed framework integrated with the query processing methods. In this section, we introduce the user interface of the system and report the results and analyses of our experiment conducted on the system.

3.5.1 Prototype System

We implemented a prototype of slide element search system in C#. The user interface is shown in Fig. 3.5. It consists of three modules: the database selection module, the query input module, and the result output module.

3.5.1.1 Database Selection Module

For the first-time use of this system, users need to click the "Select Database Slides" button on the top right corner, and choose the folders that contain the database slide files. Then the data preprocessing is invoked, and the database slides will be scanned to build indexes.

3.5.1.2 Query Input Module

In this module, users can browse the folders and the slide files stored in their computers through a tree view on the top left corner. Users may select a slide file and then a slide number to open a slide, which will be shown in the center of the interface. For text and image queries, users select a segment of text or an image

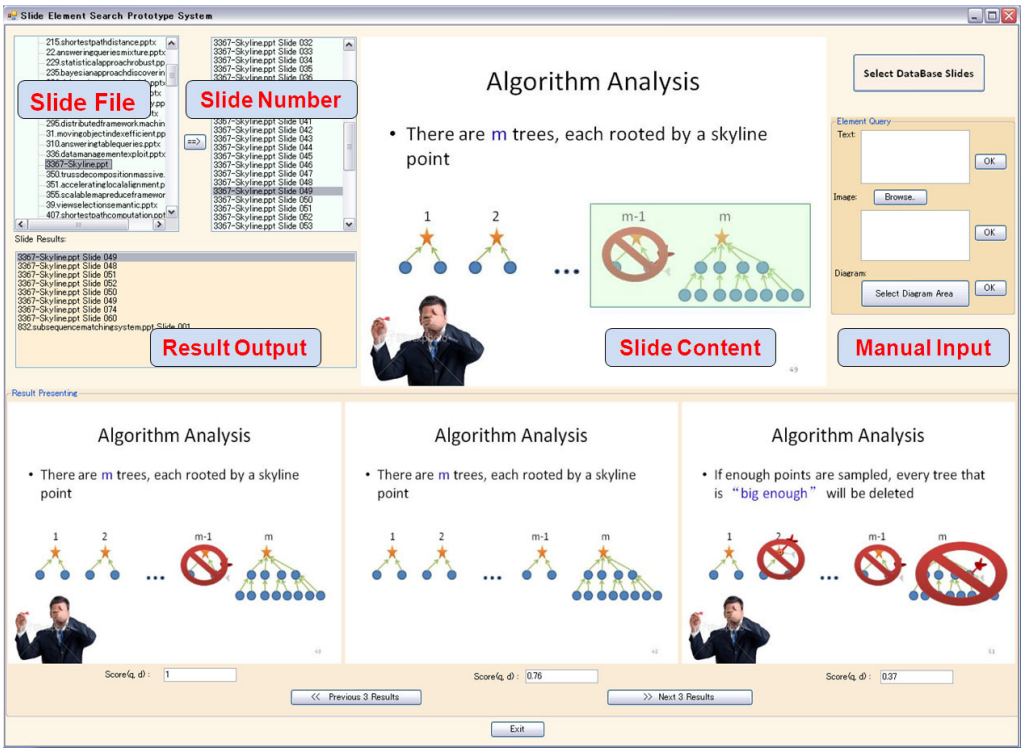


FIGURE 3.5: User interface of slide element search system

from the slide, and click the "OK" button on the right to submit the query. If there is no appropriate slide at hand, users may also manually type a text query or select an image file from their computers. For diagram query, users first click the "Select Diagram Area" button on the right, drag a rectangle area in the slide, and then click the "OK" button to submit it.

3.5.1.3 Result Output Module

Once users submit the query, the query processing is invoked, taking a scoring function and ranking the results to get top slides. The results are then presented in the bottom of the interface. On its top left corner, the slide files and the slide numbers that contain the results are listed in the order of descending relevance score. Users may double-click the file to open it with the application associated, e.g., Microsoft PowerPoint, to reuse the materials. In the bottom, three slides as a group are displayed along with their relevance scores. The top-3 results are shown first, and users may click the "Next 3 Results" button to see the next group of three results.

TABLE 3.1: Element Search Dataset statistics

Attribute	**Number**
Files	118
Slides	3989
Total words	200,095
Average slide words	33
Total slide visual words	1,147,667
Average Slide visual words	191

3.5.2 Experiment Setup

We use the slide files downloaded from the websites of the 2011[94] and 2012[95] International Conference on Very Large Data Bases (VLDB) as the dataset for evaluation. It includes 118 files of academic reports with a total of 3989 slides, and its detailed information is shown in Table 3.1.

The experiments are run on a PC with a 3.4 GHz CPU and 8.0 GB of RAM.

The following methods are involved in our experiments.

- ES is our proposed slide element search method.
- SBLK12 is a slide retrieval system for presentation slide reuse [2]. It computes the similarities of text, image, and path and file names to get an overall relevance score of a result. Edit distance is employed to capture the text similarity. The Jaccard similarity on image IDs is used to capture the image similarity. The Jaccard similarity on the word bags of path and file names is used to capture the similarity of contextual information.
- DPA06 is a slide management system developed to visually compare between different versions of the same presentation slides [10]. Edit distance is used to measure relevance in text, common image IDs are used to measure relevance in image, and the Mean Square Error of slide screenshots is used to measure the similarity in appearance.
- TTAKM13 is a method to answer diagram queries using types of shapes and their locations in a slide [9], yet it does not consider the placement relations

between the shapes. Note that users have to draw query diagrams by themselves in this method, which is a laborious task.

As SBLK12 and DPA06 are developed for whole slide queries, we make the following modifications to support element queries. For keyword and sentence queries, we modify their respective edit distance methods by enumerating all the substrings of the slide text and compute edit distance with the query. The score ($1-\frac{ed(q,s)}{\max(\text{len}(q),\text{len}(s))}$ for SBLK12 and $len(q)-ed(q,s)$ for DPA06) of each substring s is computed, and the maximum is kept as the relevance score between the query and the slide. To avoid blindly enumerating substrings, we apply the following three conditions so they may return results in reasonable amount of time: (1) A substring must start and end with word boundaries. (2) We impose a condition $\frac{1}{2} \cdot \text{len}(q) \leq \text{len}(s) \leq 2 \cdot \text{len}(q)$ on the lengths of substrings. (3) We exploit an upper bound of the score of a substring ($\frac{\min(\text{len}(q),\text{len}(s))}{\max(\text{len}(q),\text{len}(s))}$ for SBLK12 and $len(q)-|\text{len}(q)-\text{len}(s)|$ for DPA06). While processing substrings one by one, we only compute edit distance for the substrings whose score upper bounds exceed the current maximum score. For image queries, we compare with the method of SBLK12 using image IDs. Since neither SBLK12 nor DPA06 includes a module to handle diagrams, we compare with TTAKM13 on diagram queries.

3.5.3 Evaluating Search Quality

3.5.3.1 Example Query Results

We show some example query results first. We randomly choose queries and take the top-3 slides returned by different methods. Since SBLK12 and DPA06 return the same results in these examples, only SBLK12 is shown here. The examples results of the four types of queries are shown in Figure 3.6 –Figure 3.9.

The keyword query shown in Figure 3.6 consists of three keywords: `Strong`, `Simulation`, and `Properties`. The top-3 results of ES are highly relevant, with the slide title being "`Properties of Strong Simulation`". SBLK12's first two results are the

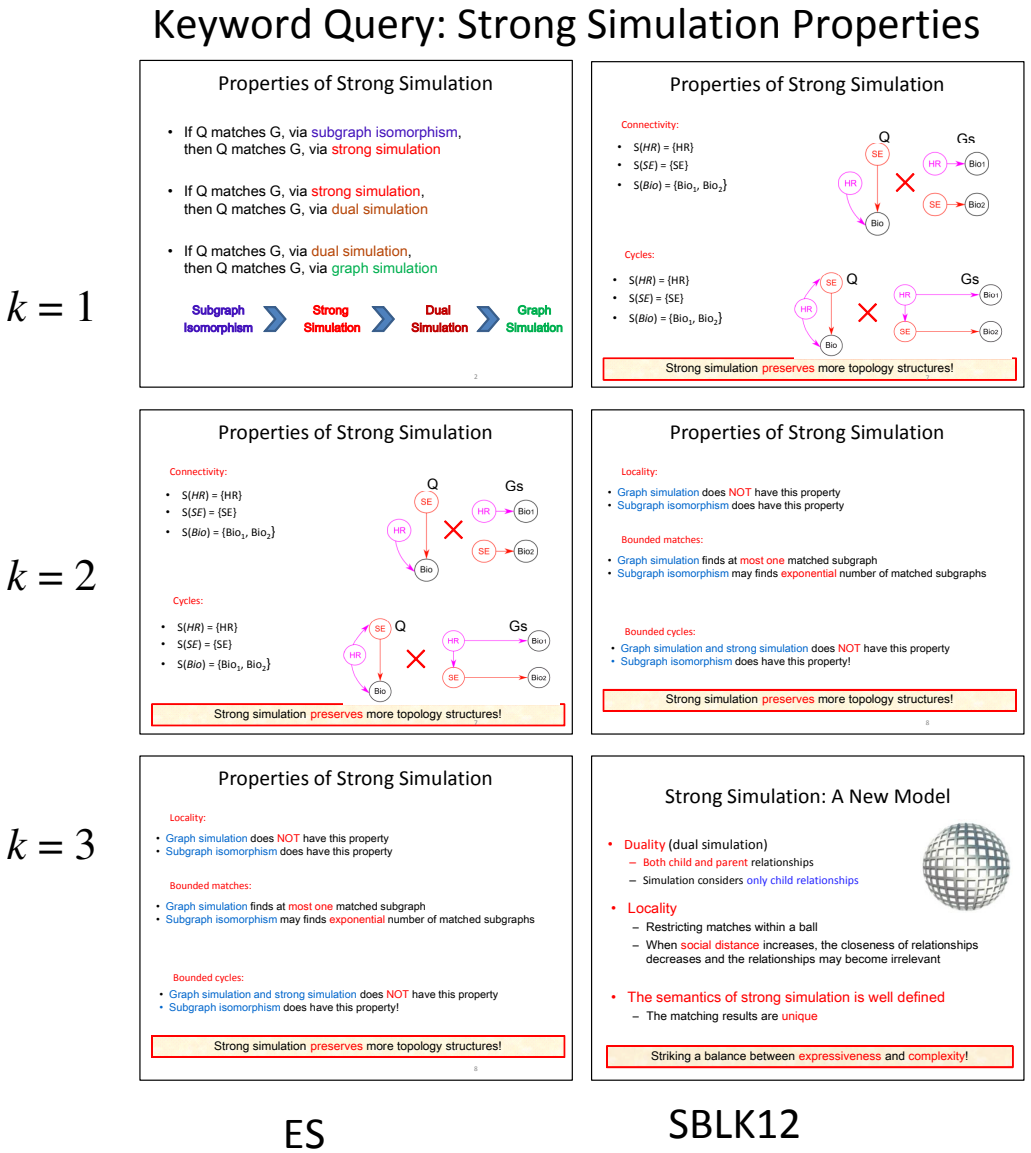


FIGURE 3.6: Example of keyword query results

second and third results of ES, respectively. However, SBLK12 returns them not because of the title but the text "`Strong simulation preserves`" on the bottom, which yields small edit distance to the query. This also explains why the first result of ES is missed by SBLK12. The third result of SBLK12 contains only two keywords `Strong` and `Simulation`, and hence are less relevant than ES's.

The sentence query shown in Figure 3.7 is "`Non-preprocessing streaming algorithm with worst-case guarantee`". The first result of ES perfectly matches the query. The second result approximates the query by only replacing "`streaming`" with "`external`". The third result shares "`streaming algorithm`" and "`with worse-case`

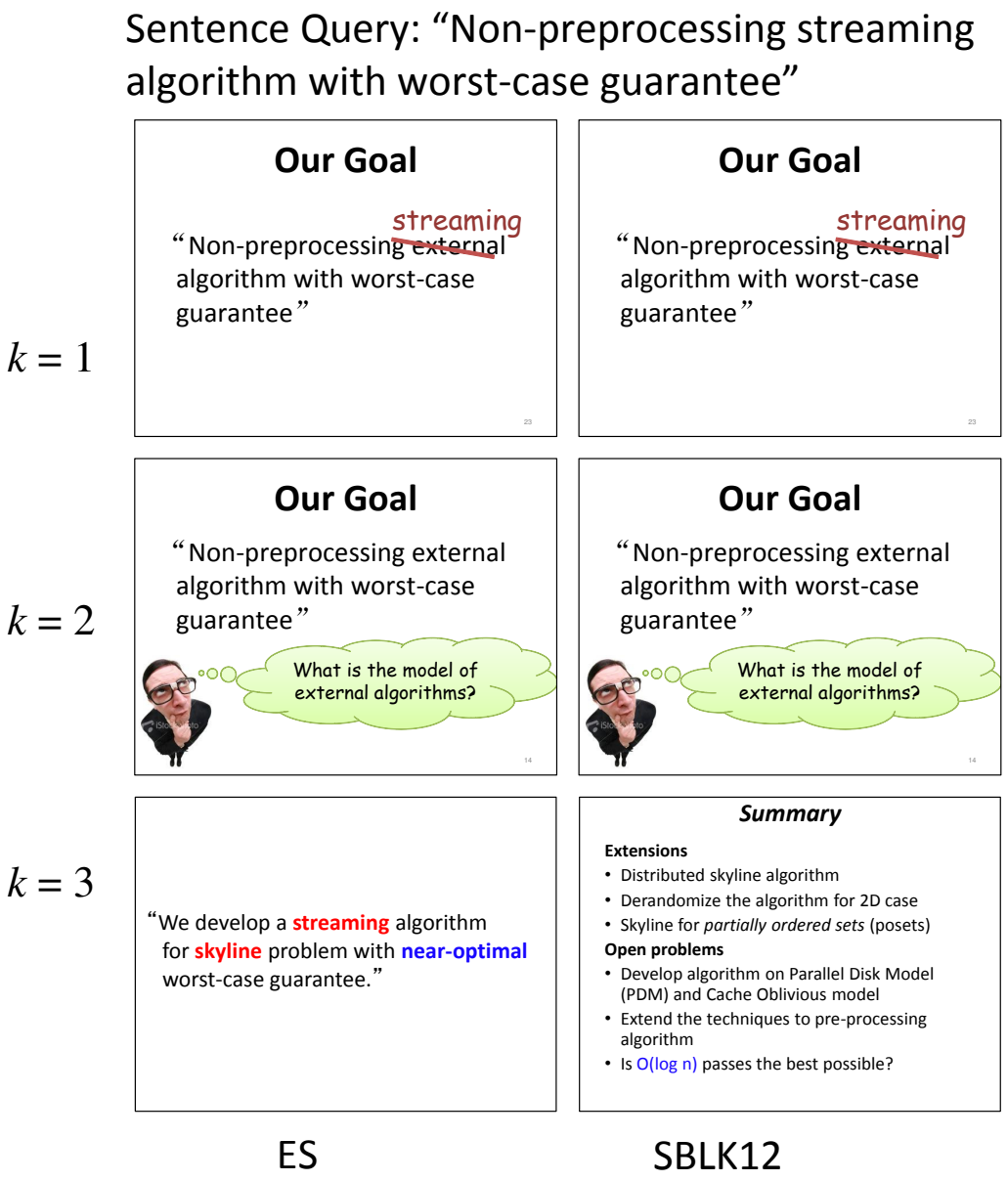


FIGURE 3.7: Example of sentence query results

guarantee" with the query, and thus is relevant as well. SBLK12 also identifies the first two results, but misses the third one due to the insertion of several words, which yields a considerable edit distance to the query. Its own third result is irrelevant, sharing only one word `algorithm` with the query.

In the image query example in Figure 3.8, all the results found by ES are relevant, being exactly or approximately containing the query image. The first results of ES and SBLK12 are the same. However, this is the sole result returned by SBLK12, because SBLK12 uses only image IDs, and this slide is the only one sharing the same image ID with the query. It is also noteworthy to mention that SBLK12 can

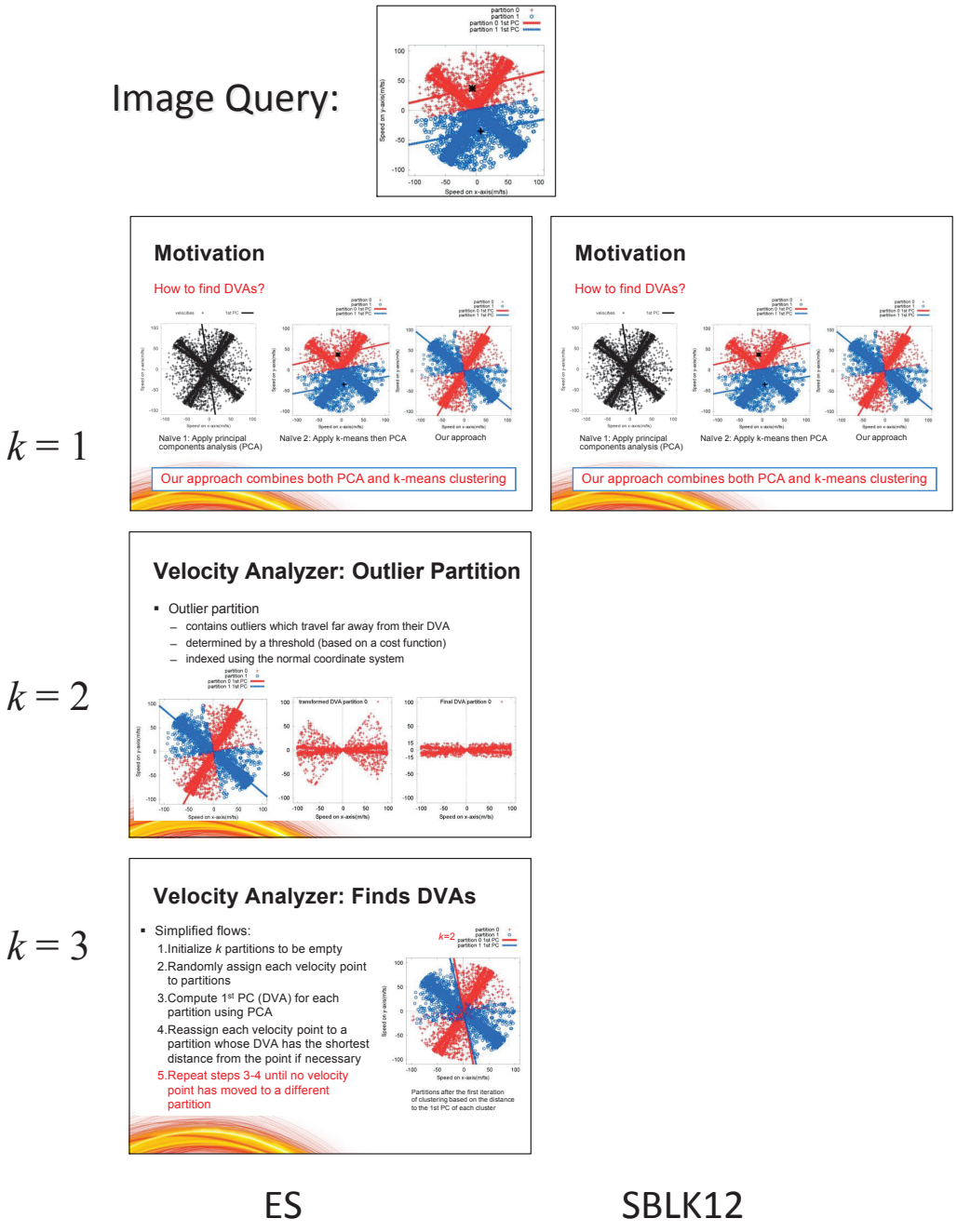


FIGURE 3.8: Example of image query results

only deal with the queries in which the image ID is given, i.e., by selecting an image from a slide. It is not applicable to the case where users select an image file from their computers as the query.

In the diagram query example in Figure 3.9, the top-2 results of ES exhibit high relevances as they exactly contain the query. The third result shows a difference with an additional red rectangle. For TTAKM13, only the first result is of high

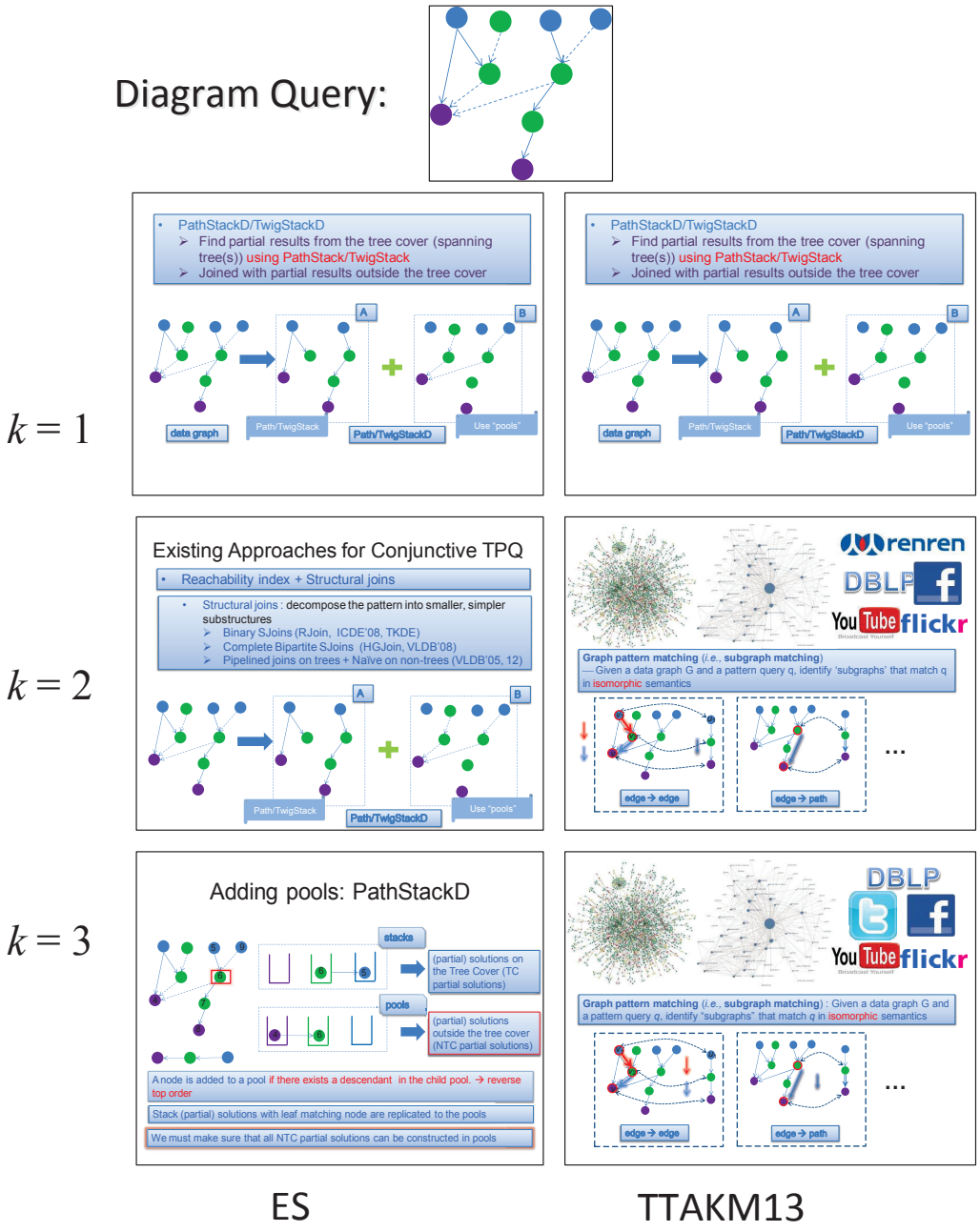


FIGURE 3.9: Example of diagram query results

relevance. Its other results contain more arrows and thus are less relevant than those returned by ES.

In all, from the example results we can see that ES returns more relevant results than the alternative methods.

3.5.3.2 Precision and Recall

We randomly select 50 queries for each type from the dataset, and they are restricted to semantically make sense. We take the top-k results, where k = 1, 3, 6, 9, and 12, indicating the results shown in the first four pages of our system. Then we measure the precision – the percentage of relevant results amid the retrieved ones, and the recall – the percentage of retrieved results amid the relevant ones, formally defined by the following equations, where R_l denotes the set of relevant slides, and R_t denotes the set of retrieved slides.

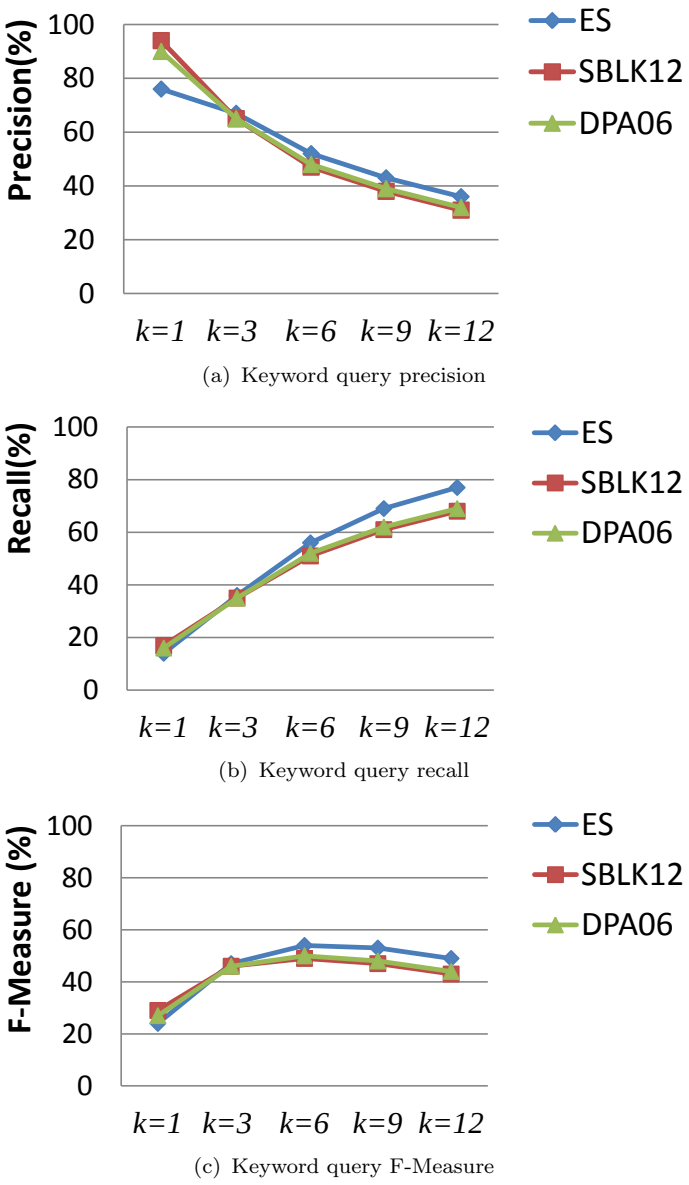
$$\text{Precision} = \frac{|R_l| \cap |R_t|}{|R_t|}, \qquad \text{Recall} = \frac{|R_l| \cap |R_t|}{|R_l|}.$$

In order to find relevant results, for keyword and sentence queries we first retrieve the slides that share at least 50% words with the query using our program. This process significantly reduces the labour of human judge and barely misses relevant results. Afterwards we manually check the retrieved slides and keep only those indeed relevant. For image and diagram queries, we check all the images and diagrams in the dataset because their numbers are small.

Figures 3.10(a), 3.10(b) and 3.10(c) show the precisions, recalls and F-Measures of ES, SBLK12, and DPA06 on keyword queries.

The three methods exhibit similar performances but ES outperforms the other two in both precision (by 5%), recall (by 9%) and F-Measure (by 6%) when k is large. This is because the keywords in the relevant results appear in a different order from the query, and additional words are also inserted between the keywords. In this case, these results are missed by SBLK12 and DPA06 due to large edit distances to the query. On the contrary, ES does not suffer from this case because its scoring function is based on the occurrences of keywords in data slides. An exception is that SBLK12 and DPA06 have better precisions than ES when $k = 1$. This will be explained in the error analysis in Sect. 3.5.5.

The search quality on sentence queries is shown in Figure 3.11(a), 3.11(b) and 3.11(c). Similar trends can be observed as we have seen on keyword queries, but



(a) Keyword query precision

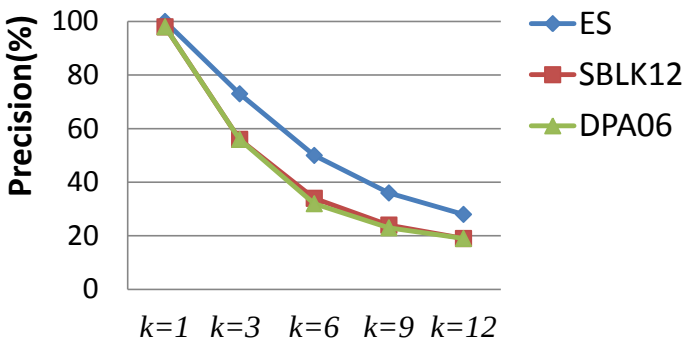
(b) Keyword query recall

(c) Keyword query F-Measure

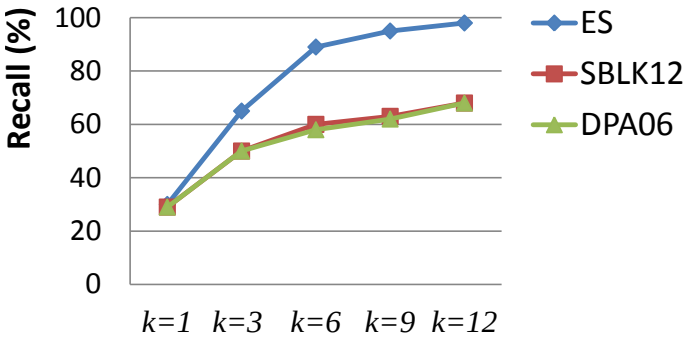
FIGURE 3.10: Experiment results of keyword search quality

the advantage of ES against other methods is more obvious: up to 18% of precision, 33% of recall and 14% of F-Measure. The reason is that sentence queries contain more words, and hence the impacts of changing word order and inserting additional words on SBLK12 and DPA06 are more apparent. We also observe that ES retrieves almost all relevant results when k is large. The experiments on text queries reveal that using tf-idf-based scoring function for keyword queries and bag-of-words-based scoring function for sentence queries yields better search quality than using edit distance.

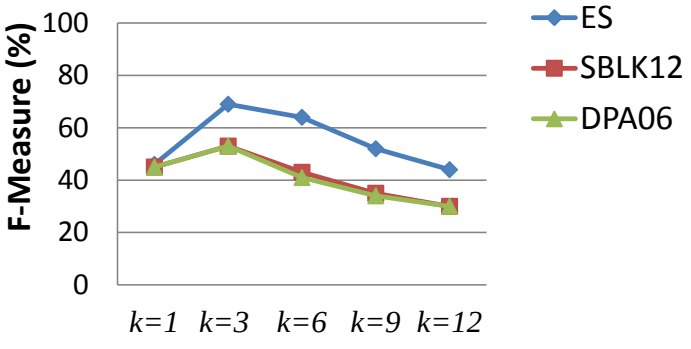
For image queries, we plot the results in Figure 3.12(a), 3.12(b) and 3.12(c) .



(a) Sentence query precision



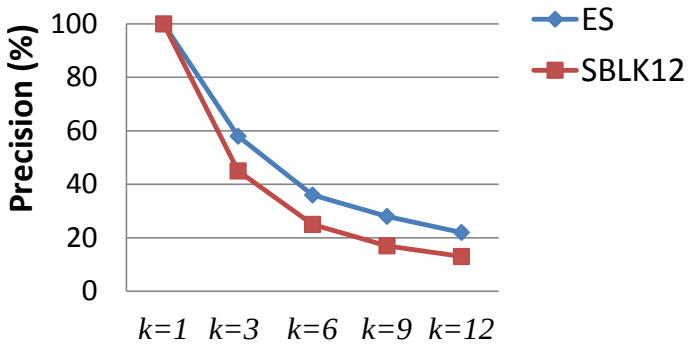
(b) Sentence query recall



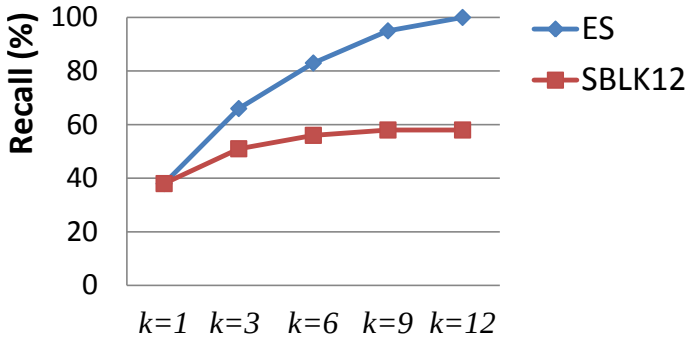
(c) Sentence query F-Measure

FIGURE 3.11: Experiment results of sentence search quality

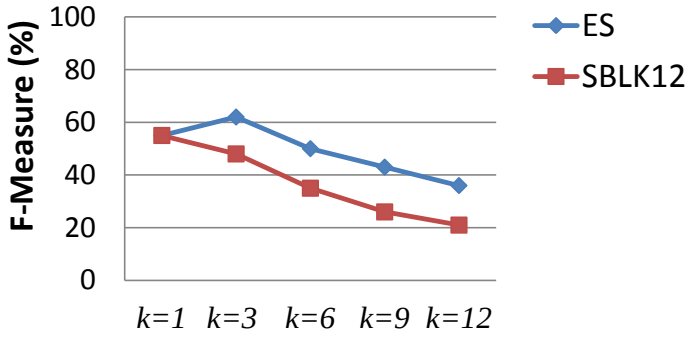
The precisions of both ES and SBLK12 drastically decrease when k moves towards larger values. This is expected because the relevant results of image queries are quite limited, and the quantities are much less than 12. Nevertheless, ES always outperforms SBLK12, and the gap can be as large as 22%. While ES outperforms SBLK12 as large as 17% in F-Measure. ES achieves 100% recall when k reaches 12, while SBLK12 only retrieves up to 58% of relevant results. The experiment result showcases the advantage of bag-of-words model over the method using only image IDs.



(a) Image query precision



(b) Image query recall

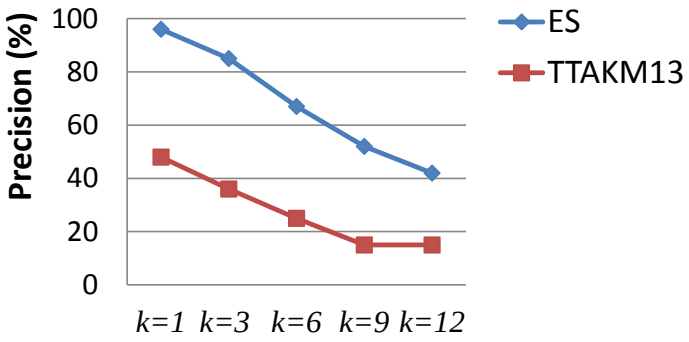


(c) Image query F-Measure

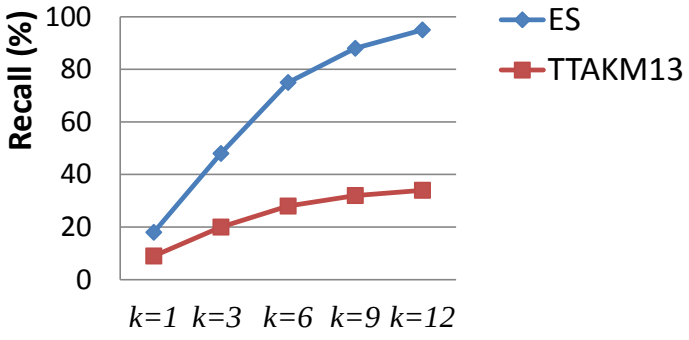
FIGURE 3.12: Experiment results of image search quality

Figures 3.13(a), 3.13(b) and 3.13(c) show the precisions and recalls on diagram queries, respectively. Several observations can be made: (1) the precisions of both methods drop with k, (2) the recalls of both methods raise with k, (3) ES outperforms TTAKM13 by up to 49% in precision, 61% in recall and 45% in F-Measure. The reason for the third observation is that we consider not only shape types but also the overall appearance, and the relationship between shape locations is utilized as well.

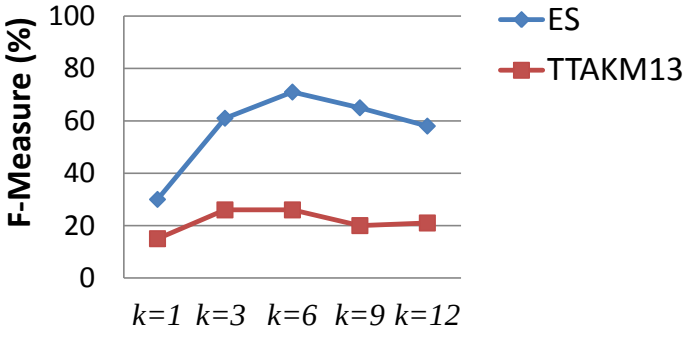
Several observations can be made as follows.



(a) Diagram query precision



(b) Diagram query recall



(c) Diagram query F-Measure

FIGURE 3.13: Experiment results of diagram search quality

- The EleSearch algorithm always get the higher precision than the compared algorithms, which is independent of the K values. Our EleSearch can retrieve more relevant slides than the exisiting algorithms. The EleSearch algorithm can also reach a relative higher recall than the compared algorithms, it retrieves most of the relevant slides in these queries.

- Due to the Edit Distance used in the keyword and sentence query, it works well when the query text length is similar to the text length of the dataset slide. It will miss the results containing the query text and other strings.

- Owing to the Image IDs used to identify the images of the slides, it will miss the results that looks similar but not absolutely same images with different Image IDs.
- As to the diagram retrieval by only autoshapes information, it cannot detect the diagram with small pictures. What's more, the same autoshapes with different text are extremely dissimilar.

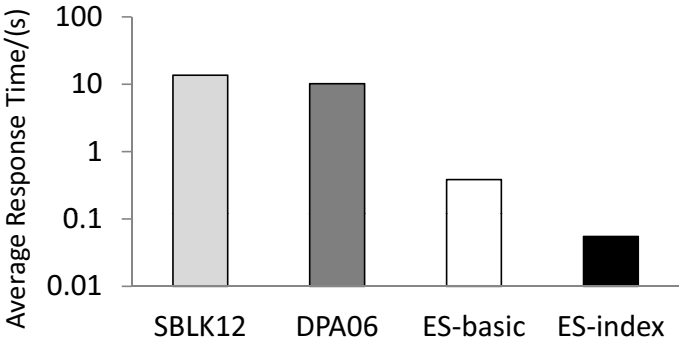
3.5.4 Evaluating Efficiency

We name the index-based top-k search method proposed in Sect. 3.3.5 ES-index. A basic algorithm that sequentially scans data slides and computes relevance scores serves as a baseline, named ES-basic. We randomly select 50 queries of each type and measure the average response time of the top-3 results.

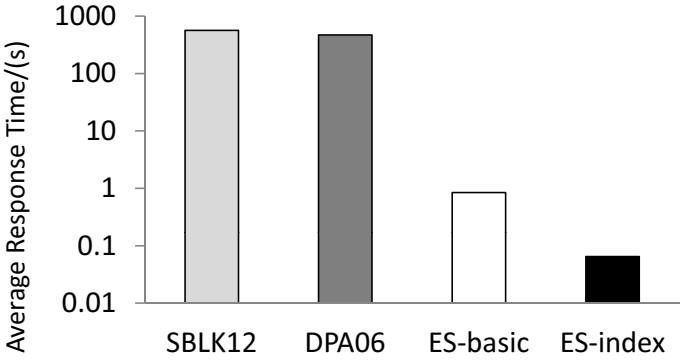
Figures 3.14(a) – 3.14(d) display the times of different methods on the four types of queries. Note that we plot the first three figures in log scale. With indexes equipped, ES-index improves runtime performance by 7.0 times on keywords and 13.0 times on sentences, in comparison with ES-basic. The efficiencies of SBLK12 and DPA06 are similar on text queries. Due to the lack of index and the costly edit distance computation, both are slower than ES-index by more than 185 times on keywords and 7200 times on sentences.

On image queries, ES-index is 6.2 times faster than ES-basic but 42.8 times slower than SBLK12. This is because we adopt a much more complicated bag-of-words model than the image ID-only method in SBLK12. Considering the trade-off in precision and recall (22% and 58%, respectively) and that ES-index returns results in only 0.2 seconds, our method's disadvantage in efficiency is in an affordable manner.

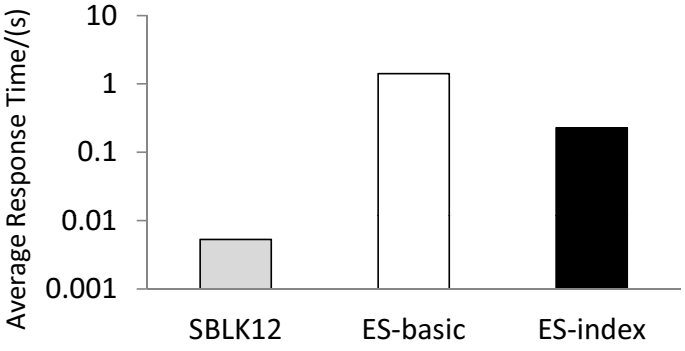
On diagram queries, ES-basic returns results in 4.0 seconds, thus rendering it not applicable for larger slide repositories. Equipped with the index on visual words,



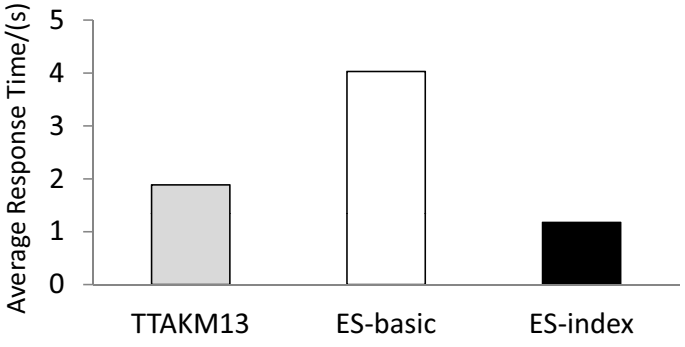
(a) Keyword query processing time



(b) Sentence query processing time



(c) Image query processing time



(d) Diagram query processing time

FIGURE 3.14: Experiment results of efficiency

ES-index responses in only 1.2 seconds, and it is 1.6 times faster than the alternative method TTAKM13.

Summary. By comparing precision, recall, and response time, we find that ES achieves the best search quality and is much more efficient than the sequential scan method.

3.5.5 Error Analysis

The top-1 results of our method are not as good as SBLK12 and DPA06 for some keyword queries. This is because ES's ranking function favors the keywords that appear frequently in a slide. There are a few subtle cases where a slide misses one keyword but contains multiple occurrences of other keywords, and ES may rank this less relevant result very high. E.g., for the query consisting of three keywords `graph`, `path`, and `query` in Figure 3.15, ES's top-1 result does not contain the keyword `query` but many occurrences of `graph` and `path`, and thus it is ranked before more relevant results containing all the three keywords. A possible remedy is to restrict that all keywords must be contained in a result.

We also note that if a sentence query consists of many frequent words in the dataset, some of ES's results are irrelevant because these slides contain almost all the words of the query but in separate locations. E.g., for the query "`proposed for efficient indexing and querying moving objects`" in Figure 3.16, one of our results contains all the words except `objects`, but the words are disjoint in locations, rendering the result irrelevant. This can be improved by using shingles [96] (contiguous subsequences of words) instead of words. E.g., consider a shingle of length 3, the query is divided into shingles "`proposed for efficient`", "`for efficient indexing`", etc. Data slides are processed in the same way, and the scoring function (Eq. 3.3) is applied on top of bags of shingles.

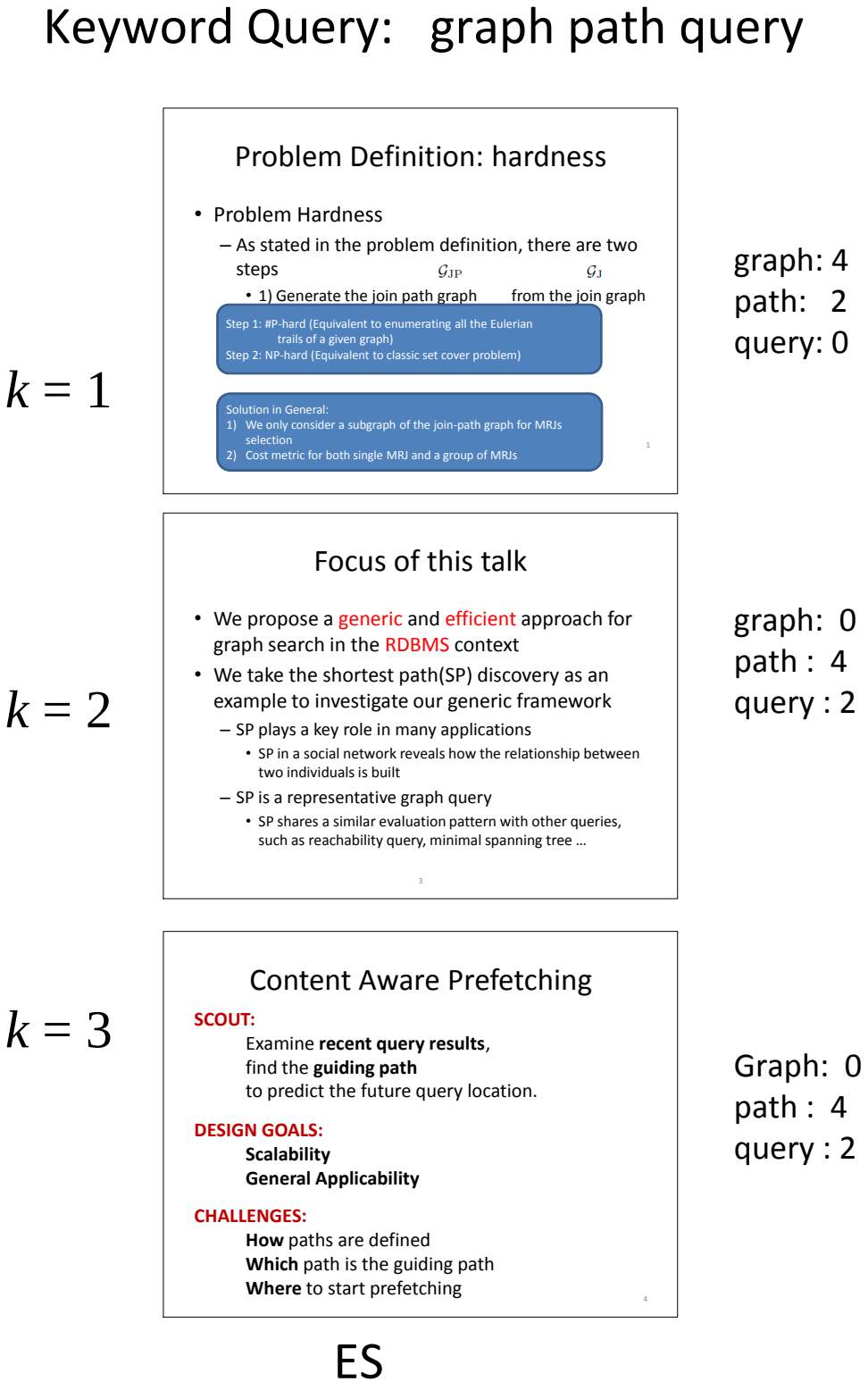


FIGURE 3.15: Error analysis of keyword example

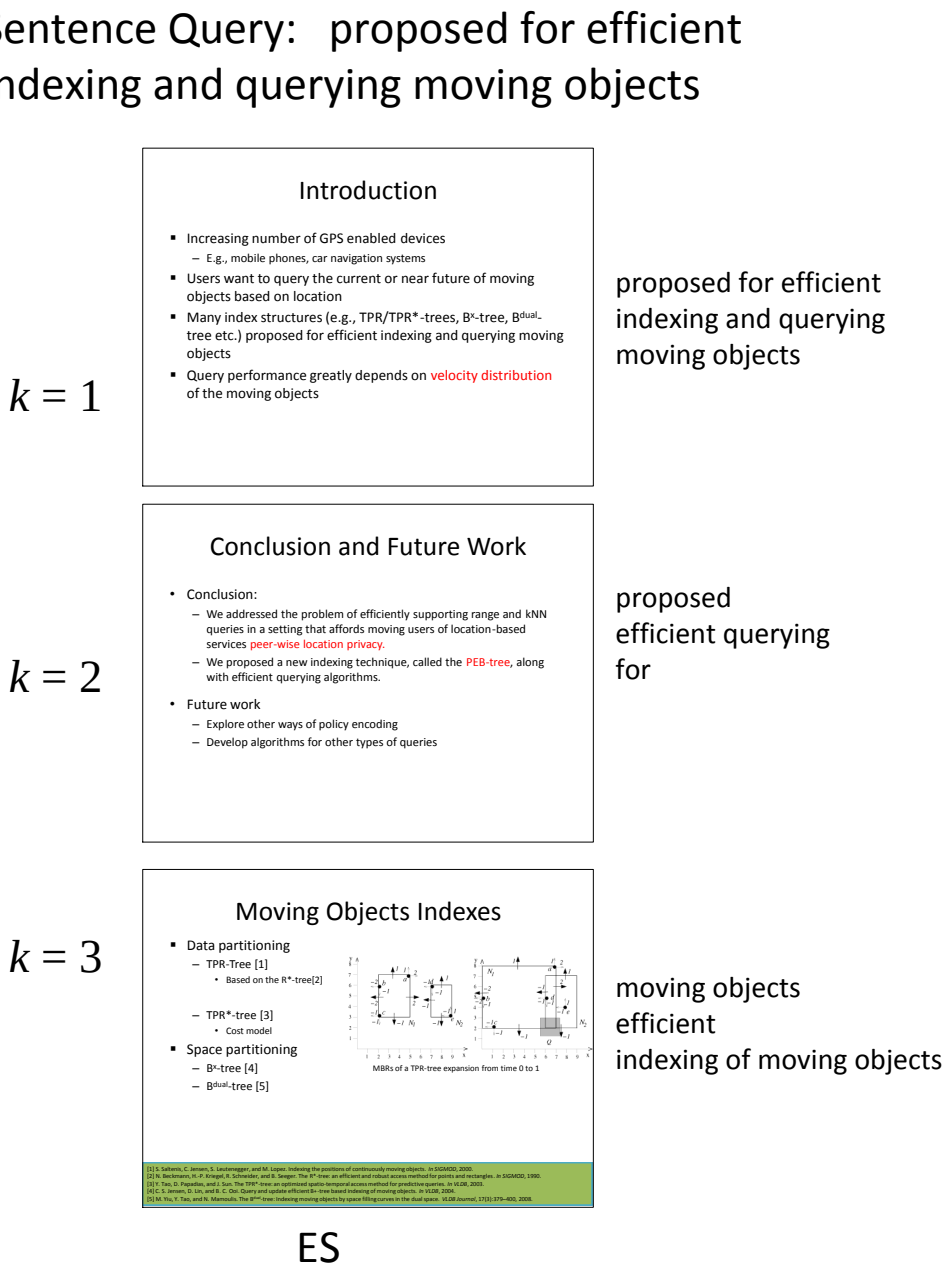


FIGURE 3.16: Error analysis of sentence example

3.6 Conclusion and Future Work

In this paper, we proposed content-based search methods for a variety of elements in presentation slides. Users can freely choose keywords, a sentence, an image, or a diagram as a query to find the materials of his interest from a collection of presentation slides. We proposed different query processing methods to improve the efficiency of answering these queries. We designed a prototype system integrated with the proposed methods along with a user-friendly interface, and conducted

experiments on top of it. The experiment results show that our proposed methods return better results than alternative methods and are much faster than the methods without indexes.

Our future work includes developing browsing methods for presentation slides based on reused elements. Users may find the origin of an element when they browse a slide. Another direction is to explore the composition methods that automatically generate slides by reusing existing materials.

Chapter 4

Managing Presentation Slides with Reused Elements

4.1 Introduction

Slide presentations are one of the most important tools for today's knowledge workers to present knowledge, exchange information, and discuss ideas. Instead of starting from scratch, slide composers tend to make new slides by reusing existing ones. One of the main reasons is to repurpose existing content for different events, listeners, formats and so on in business and educations.

Detecting reused materials in presentation slides benefits many presentation-related applications; e.g., assisting composers in tracking changes in multiple versions, understanding existing presentation slides, and assembling existing slides to make new ones [1, 10], etc. Although the slide retrieval method for reuse [2] and the method to compare different versions of a presentation file [10] have been proposed, they are either based on slide-to-slide comparison or file-to-file comparison. In many cases, only an individual element such as a sentence, a table, an image, or a diagram, is copied from one file to another, but overall the slides and the files differ significantly, and thus the reuse element cannot be identified by these methods.

In this chapter, we investigate the problem of managing presentation slides from the perspective of individual elements. We first develop different methods to detect textual and visual elements reused in a slide repository specified by users. Textual elements are divided into sentences and further decomposed to bags of words. To detect reused sentences and consider the case when slide composers make minor modifications after reusing elements, similarities are taken into account to tolerate nuances between different versions. Likewise, we adopt the bag-of-words model to find reused visual elements, and utilize similarities to handle the case when they are transformed after reuse. The techniques to tackle the efficiency challenge are also introduced. Then we propose two ways of visualizing reused elements. The first is to show the files and slides that use the same element in a timeline. The second is to construct a network of presentation files connected by reused elements.

Based on the proposed methods, we design a prototype system with a user-friendly interface, which can be integrated into slide composition tools. When users are browsing a slide, they are notified with what elements in this slide have been reused across different files in the slide repository. Users can quickly find the origin of an element, as well as when and how the element is used by other files. For all the files in the repository, users may also get an overview of their relationships via common elements. They may explore any relationship to get detailed information. To demonstrate the effectiveness of our methods, we conduct experimental evaluation on real presentation slide data using the prototype system.

Our contribution can be summarized as follows:

- We propose an approach to presentation slide management by exploiting the notion of reused elements.
- We develop techniques to detect and visualize reused elements in users' slide repositories.
- We design a system integrated with a user-friendly interface to help users browse slides through reused elements.
- We conduct experiments on real data to evaluate the proposed methods.

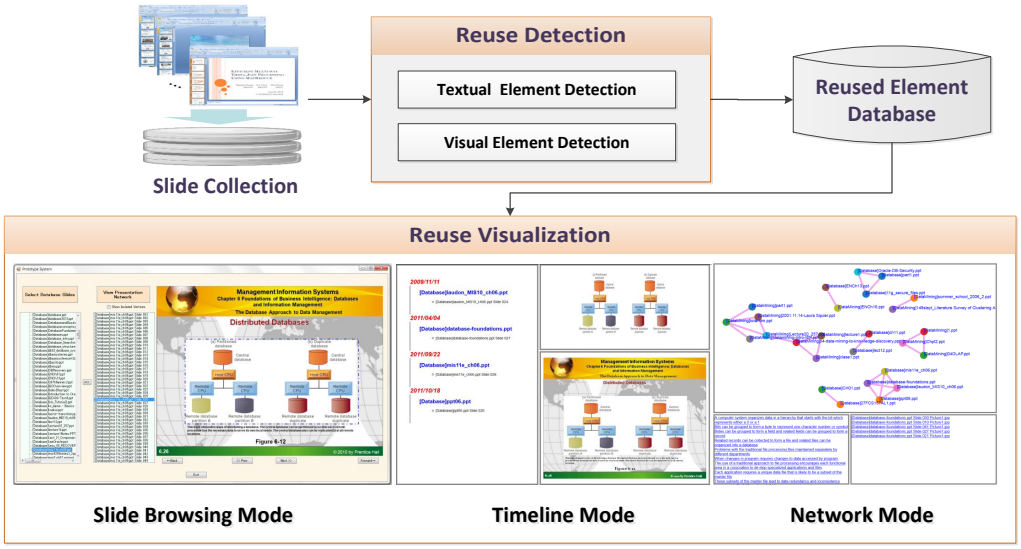


FIGURE 4.1: An overview of slide management framework

The rest of this paper is organized as follows. Section 4.2 proposes the conceptual viewpoint and introduces the framework of our approach. Section 4.3 proposes the methods to detect reused elements. Section 4.4 introduces the methods to visualize reused elements. Section 4.5 reports experiment results and our analyses. Section 4.6 concludes this chapter.

4.2 Conceptual Viewpoint with Framework and Approaches

Figure 4.1 illustrates an overview of the framework of our presentation slide management method. It is composed of two main modules: detection module and visualization module. Next we introduce the two modules respectively.

4.2.1 Detection Module

Users may specify the location where the repositories of slides are stored in their computers. A database of presentation files is then built by offline and stored on disk to serve future browsing. We extract textual and visual elements from the database slides. Textual elements include main text (including titles) and tables.

Visual elements include images, charts, and diagrams. Reused elements are detected in this module and the slides in which these elements appear are marked. The detailed reused element detection method will be presented in Section 4.3.

4.2.2 Visualization Module

This module is further divided into two submodules: timeline submodule and network submodule. In the timeline submodule, reused elements are notified to users when they are browsing a slide in the repository. Users may click an element, and all the presentation files that use the element will be shown in a timeline. Users may follow the links in the timeline to browse these slides. In the network submodule, users are shown with a presentation network, i.e., a graph connecting presentation files in the repository through reused elements. Users may quickly get an overview of the relationships amongst the presentation files. For example, the multiple versions of a presentation file form a clique because any two of them share large numbers of common elements. A summary will be shown as a hub because it is assembled by copying elements from multiple files. The details of the two submodules will be introduced in Section 4.4. In addition, we also provide users the interface to browse slides, as they have seen in prevalent presentation composition tools like Keynote and Powerpoint.

According to different types of content extracted in the data preprocessing module, we take different methods to determine the text and image content as reused. For images, we first save them as picture file then get the MD5 value to judge they are the same or different. For texts, we determine them as reused according to different threshold of similarities. The methods to process the various types of contents will be introduced in Section 4.3.

After all the content elements are certified as reused, links between presentations holding them should be created. Each presentation is treated as a node in an overview graph. If two different presentations have the same reused content, an

edge will be built between them. All the reused content will be added to the edge value to present a link. The result will be shown in the experimental section.

4.2.3 Content Source Visualization

All the presents are linked to the related presentations having the same reused contents, which will help the user to get an overview glance at all the presentations. Each edge contains many reused contents between two linked presentations. What's more, a specific content may appear more than two presentations. In this case, a collection of all the presentations holding the reused content should be presented to the user. In this module, a content source visualization is designed in timeline sequence for the user.

We design a prototype system according to the above framework. The user interface of the system will be shown in Section 5.4.1.

4.3 Detecting Reused Elements

In this section, we introduce the detection methods for textual element reuse and visual element reuse, respectively.

4.3.1 Detecting Reused Textual Elements

We first introduce the method to detect reused element in main text of presentation slides, and then discuss the case of tables.

When the text in a slide is reused, composers may copy one or more sentences from one slide to another, but overall the texts in both slides differ significantly. For this reason, we choose to detect reuse textual elements on sentence level; i.e., divide the text in each slide into sentences and then identify the sentences that have been used by multiple presentation files. In addition, considering that composers may

make modifications to the reused sentence (e.g., change the order of words, insert additional words into it, and delete a few words from it), we tokenize each sentence into a bag of words with white space and punctuations, and then adopt the idea of similarity search to find reused sentences in the presence of modifications. The Jaccard coefficient is used to capture the similarity between two sentences:

$$sim(x, y) = \frac{|x \cap y|}{|x \cup y|}, \tag{4.1}$$

where x and y are two sentences represented in bags of words, and $|x|$ denotes the cardinality of a bag x.

Example 4.1. *Considering two sentences: "`the telephone was invented by Alexander Bell in 1876`" and "`in 1876, Alexander Graham Bell invented the telephone`". After tokenization, the two sentences become* { `1876, Alexander, Bell, by, in, invented, telephone, the, was` } *and* { `1876, Alexander, Bell, Graham, in, invented, telephone, the` }*. The similarity between them is* $\frac{7}{10} = 0.7$.

We retrieve the pairs of sentences whose similarity values by Equation 4.1 are no smaller than a threshold t, and construct a *sentence reuse graph* as follows:

- Each vertex denotes a sentence in the database.
- Two vertices are connected by an edge if the similarity between the two sentences is no smaller than t.

The connected components of this graph can be computed using either a breadth-first search or a depth-first search. Since the Jaccard coefficient is a metric, sentences in the same connected component bear high similarity to each other. Thus we call the sentences in the same connected component a *reused sentence group*, and they are regarded as originate from the same sentence.

Example 4.2. *Fig. 4.2 shows an example of five sentences depicted in a graph, each vertex (ellipse) denoting a sentence. Assuming* $t = 0.5$*, we connect the pairs*

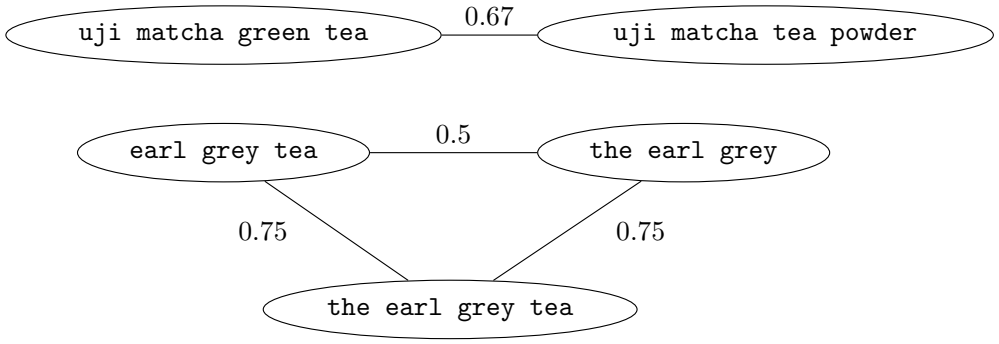


FIGURE 4.2: Example of reused sentences

of sentences that satisfy the similarity constraint, and show the similarity values next to the edges. Since there are two connected components, two groups of reused sentences are obtained from this graph.

A key issue of reused textual element detection is how to find the pairs of sentences that satisfy the constraint. A straightforward method is to compute the similarity value for every pair of sentences. If we compute Equation 4.1 by hashing the words in two bags, its time complexity is $O(W)$, where W is the number of words in a sentence. Let S denote the number of sentences in the database. The time complexity of comparing all pairs of sentences is $O(S^2W)$. It is too expensive for practical use because the value of S can be large; e.g., there are 35,932 sentences in the 200 presentation files used in our experiment. Since the problem is exactly the set similarity join problem [93] and has been studied by the database research community, we employ the ppjoin algorithm [97], a state-of-the-art method to this problem, to efficiently find the pairs of sentences that satisfy the constraint. Its basic idea is to sort the words in each bag according to a global order and exploit the threshold t. If a pair of sentences satisfy the similarity constraint, they must share at least one word in their first p words, where $p = \lfloor \max(l_x, l_y) \cdot (1 - t) \rfloor + 1$, and l_x and l_y denote the numbers of words in x and y, respectively [97]. Thus we only need to compute Equation 4.1 for the pairs that meet this condition, which can be found by indexing the first p words of each sentence. For example, consider two sentences tokenized into bags of words and sorted in alphabetical order: { `a, b, d, e, f` } and { `c, d, e, f` }, and a t of 0.5. Then the two bags must share at least one word in the first $\lfloor \max(5, 4) \cdot (1 - 0.5) \rfloor + 1 = 3$ words. The similarity

value between the two bags is $\frac{4}{6} = 0.67$, and indeed they share a common word `d` in their respective first three words.

For the case of tables, we process them separately from other text in the database, and for each table we concatenate the contents in all its cells as a sentence. Then reused tables can be identified using the above method.

4.3.2 Detecting Reused Visual Elements

We introduce the method to detect reused images in presentation slides, and then discuss the cases of charts and diagrams.

Like textual element detection, visual element detection also needs to take modification into consideration. Although composers do not often modify images with graphics editing software when copying images from one slide to another, they may transform images (e.g., by scaling and rotating) with presentation composition tools, and this will make the images bit-wise different from the original version. To address this issue, the bag-of-words model [88], a prevalent approach in computer vision, is employed to find reused images. The bag-of-words model represents images as bags of elementary image patches which are called visual words, as shown in Fig. 3.3. A dictionary of visual words called visual vocabulary is created first, and then an image can be represented using the words in the dictionary. To build a visual words vocabulary, we detect interest regions in the images with Hessian-affine detector [89] widely used in visual word-based studies. It provides good performance [90] and it is insensitive to affine transformations such as scaling, reflection, rotation, etc. These regions are described in 128-dimension SIFT descriptors and then clustered by a hierarchical k-means algorithm [91], each cluster representing a visual word. Then each image is represented in a bag of visual words.

Like detecting reused sentences, the Jaccard coefficient (Equation 4.1) is used to measure the similarity between two bags of visual words. This similarity measure has been adopted for near-duplicate image detection [92], based on the intuition that similar images share most of their visual words.

Similar to sentence reuse graph, an *image reuse graph* is constructed as follows:

- Each vertex denotes an image in the database.
- Two vertices are connected by an edge if the similarity between the images represented in bags of visual words is no smaller than a threshold t.

We call the image in the same connected component a *reused image group*, and they are regarded as originate from the same image.

The efficiency issue also exists for images. The time complexity of comparing all pairs of images to compute similarity is $O(I^2V)$, where I is the number of images in the database and V is the average number of visual words in an image. This method may not finish in reasonable amount of time for real data; e.g., there are 2,282 images in the 200 presentation files in our experiment, and an image contains an average of 480.6 visual words. Therefore we also use the ppjoin algorithm to efficiently find image pairs that satisfy the similarity constraint. The only difference from sentence reuse detection is that the algorithm is run on visual words instead of textual words.

For other types of visual elements, charts are converted to images and processed in the same way. For diagrams, since they consist of individual shapes such as rectangles, circles, and arrows, we find the topmost, leftmost, rightmost, and bottommost shapes in each slide, and convert the screenshot within this area into an image. Then the method to detect reused images can be applied.

The threshold t is the key parameter of the quality of reuse detection. In general, using smaller thresholds improves the recall but reduces precision, because more pairs of sentences or images satisfy the similarity constraint but false positives may be included as well. The effect of t will be empirically investigated in Section 4.5.3.3. In addition, we need to remove short sentences because they tend to provide a large number of false positives but almost no meaningful results for reuse detection. Small-size images should also be removed because they are usually simple graphics such as a single-color patch or a logo but not meaningful resources

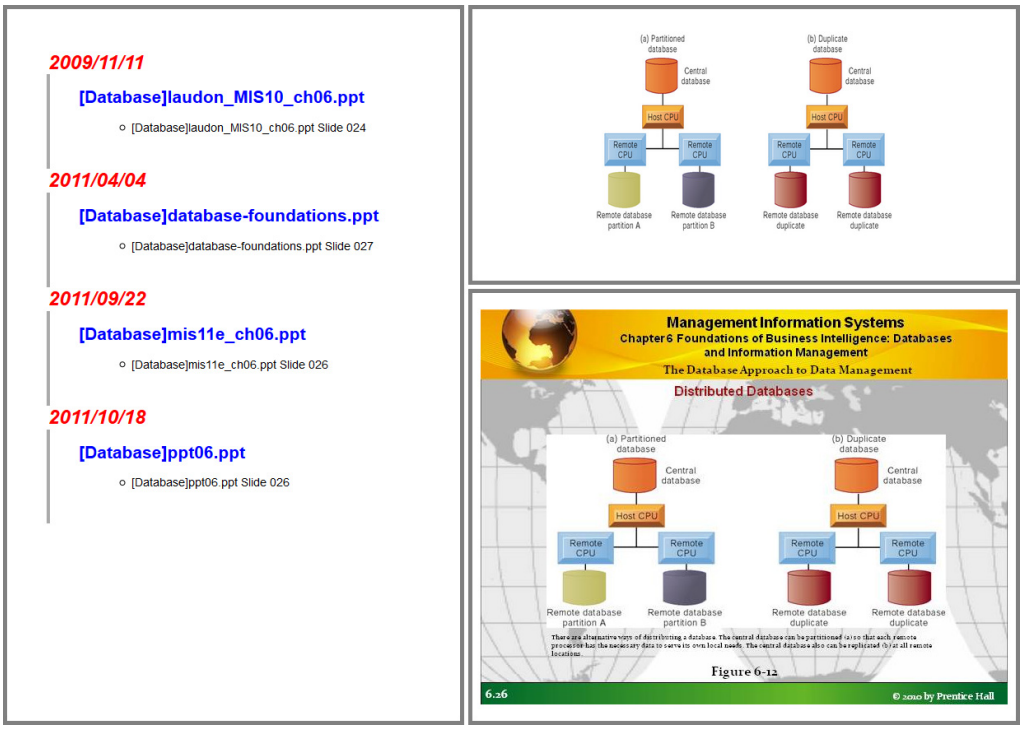


FIGURE 4.3: User interface of timeline

for reuse. To strike a balance between precision and recall, we perform reuse detection on sentences containing at least 5 words and images whose sizes are no smaller than 1KB.

4.4 Visualizing Reused Elements

In this section, we present the method to visualize the reused elements.

4.4.1 Timeline of Reused Elements

In order to help users understand how an element (a reused sentence group or a reused image group) is reused, for each reused element, we show in a timeline the following information: (1) the content of the element, (2) when it is used, and (3) in which files and slides it is used.

We use the timeline of an image as an example, and show its interface in Figure 4.3. The content of this image is displayed on the top right corner. On the left side, the

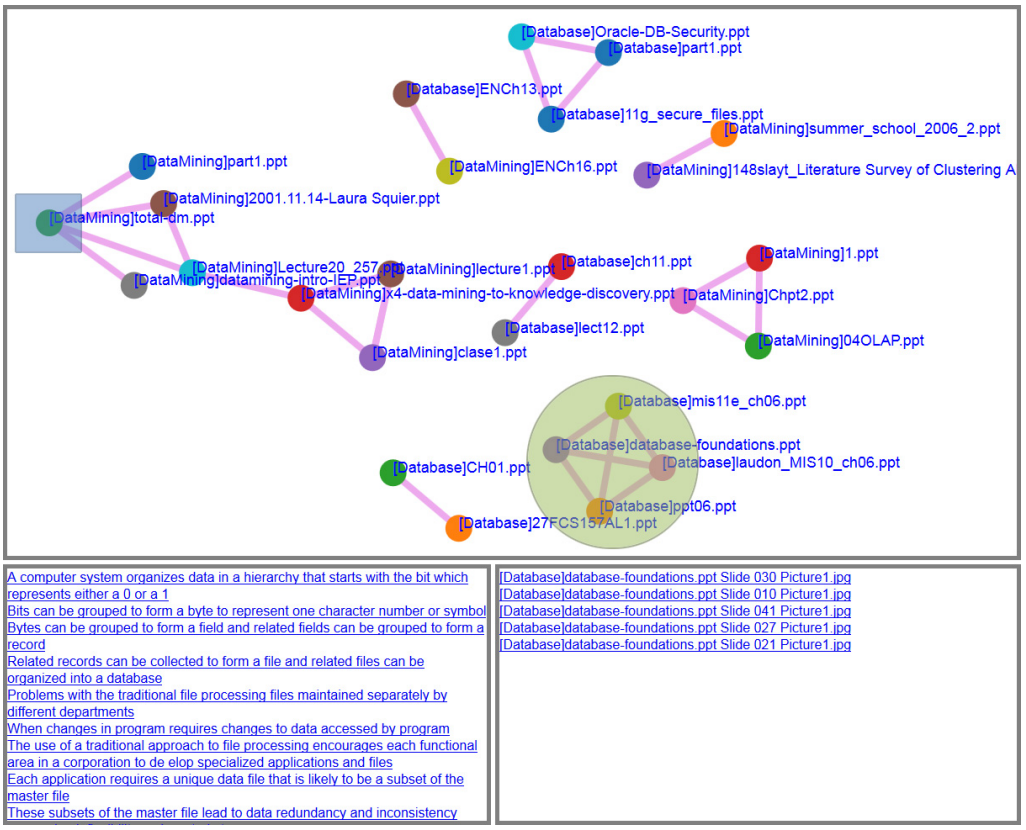


FIGURE 4.4: User interface of presentation network

slides that contain this image are listed and grouped by presentation files. The files are sorted by increasing order of last saved time, i.e., the time when the composition of presentation slides is finished. This information can be retrieved from file properties and is shown on the left side as well. Users may click any slide listed on the left, and the content of the slide will be shown on the bottom right corner. The timeline of a reused sentence group is presented in the same way. The only difference is that the sentences in a group may differ due to composers' modifications. Thus we only show the version in the origin file, i.e., the file with the smallest last saved time, on the top right corner.

4.4.2 Presentation Network based on Reused Elements

Another method of visualization is to show an overview of the relationship between presentation files in terms of reused elements. To this end, we construct a presentation network as follows:

- Each vertex denotes a presentation file.
- Two vertices are connected by an edge if the two files contain sentences (images) in the same reused sentence (image) group.

Figure 4.4 shows the interface of a presentation network. The network is displayed on the top, consisting of 26 files. The file names are given next to the vertices. Users may click any edge in the network, and the common elements between the two files will be shown on the bottom, with sentences on the left and images on the right. Then users may click a common element to view its timeline. As can be seen from the figure, the vertices in the circled area constitute a clique. They are multiple versions of the same file. In addition, the vertex in the rectangle area is a hub connected to many files. Its contents indicate it is a summary.

4.5 Experiments

We design a presentation slide management system integrated with the methods proposed in previous sections. In this section, we introduce the user interface of the system and then report the results and analyses of our experiment conducted on the system.

4.5.1 Prototype System

Our prototype system is implemented in C#, except that the visualization of the presentation network was implemented in HTML with Data-Driven Documents JavaScript Library (D3.js) [98]. The user interface is shown in Figure 4.5.

For the first-time use of this system, the user needs to click the "Select Database Slides" button on the top left corner, and choose the folders that contain the database slide files. Then the database slides will be scanned to detect reused elements. On the left side there are two columns. The left column lists file names in the database. Users may click any file, and the slide numbers in this file will be

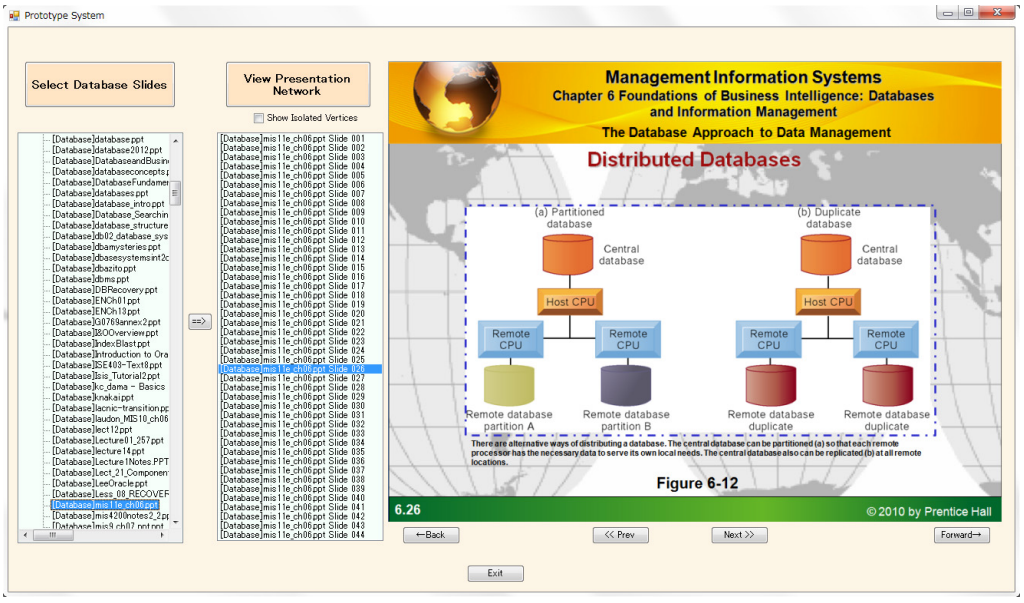


FIGURE 4.5: User interface of prototype system

listed in the right column. The main window is on the right side, and it can switch between three modes: slide, timeline, and network. Users may click any slide in the slide number column, and its content will be shown in the main window. The "Prev" and "Next" buttons help users browse slides, and the main window will show the previous slide and the next slide when they are clicked, respectively.

As shown in Figure 4.5, reused elements are highlighted in the main window when users are browsing slide contents. It will be switched to the timeline mode (Figure 4.3) when users click a reused element. Users may follow the links in the timeline and the main window can switch back to slide mode.

To view the presentation network, users need to click the "View Presentation Network" button on the top left. Then the main window will be switched to the network mode (Figure 4.4). By default, we hide vertices with no edge connected in the network. These vertices will appear if the check box "Show Isolated Vertices" on the top left corner is ticked. Users may click vertices in the network to view slides, or edges to view common elements. They may also navigate through different views using the "Back" and "Forward" buttons in the main window, like browsing Web pages.

TABLE 4.1: Dataset statistics

Attribute	**Number**
Files	200
Slides	10,327
Sentences ($\geq$ 5 words)	35,932
Images ($\geq$ 1KB)	2,282
Average number of words in a sentence	10.1
Average number of visual words in an image	480.6

TABLE 4.2: Coverage of reuse relationship

Type	**Percentage**
Textual	90.5%
Visual	17.0%
Textual and visual	15.0%
None	7.5%

4.5.2 Experiment Setup

Our dataset consists of lecture notes of database courses and data mining courses in universities in USA. Table 4.1 provides the statistics about the dataset. The experiments are run on a PC with a 3.40 GHz CPU and 8 GB of RAM.

4.5.3 Evaluating Reuse Detection

4.5.3.1 Coverage of Reuse Relationship

We first test the percentage of files having reuse relationship and show the results in Table 4.2. It can be observed that most files in the dataset have reused textual elements or their textual elements have been reused by other files. A small percentage of files have reused visual elements or visual elements have been reused by other files, and most of these files have reused/been reused textual elements as well. The percentage of the files that do not have any reuse relationship is very small. The results demonstrate the pervasiveness of reuse relationship in the dataset.

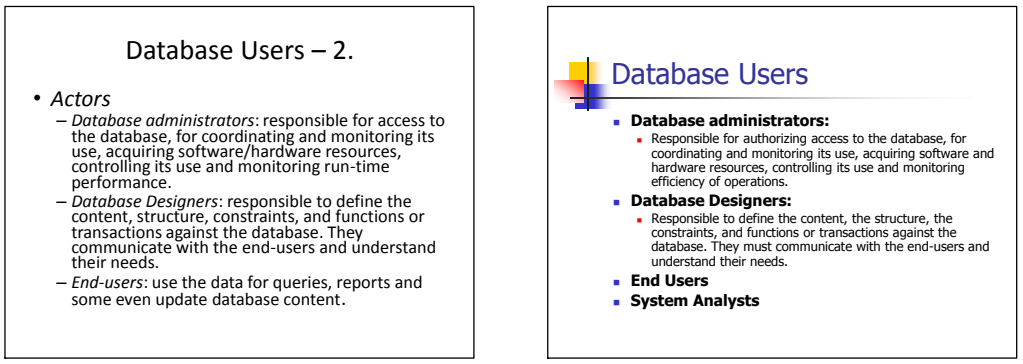


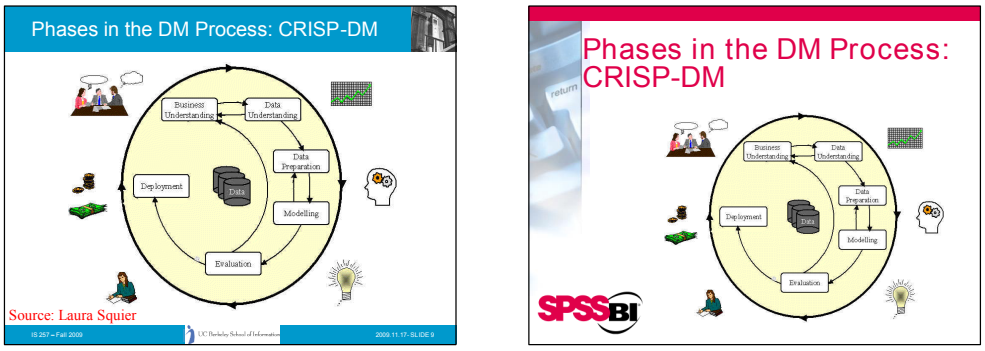
FIGURE 4.6: Example result of reused textual element detection

4.5.3.2 Example Detection Results

We show some example reuse detection results.

The example result of reused textual element detection is shown in Figure 4.6. The slide contents are displayed on the top, while the context information – file names, slide numbers, and last saved times – is given on the bottom. It can be seen that the two paragraphs on the left slide are copied to the right slide and slightly modified.

Note that this result cannot be identified by the method in either [2] or [10]. This is because in [2], the overall similarity score is the normalized sum of the text similarity (measured by edit distance), image similarity (measured by Jaccard coefficient on image IDs), and attribute similarity scores (measured by Jaccard coefficient on file names tokenized into words). Only the pairs with overall similarity no less than 0.5 are detected. In this example, the text similarity is 73%, image similarity is 0%, and attribute similarity is 20%. The overall similarity is only 31%, and hence cannot be identified. The method in [10] returns pairs of slides that satisfy the following three conditions: (1) mean square error of screenshots is no more than 100; (2) edit distance of texts is no more than 30; or (3) slide IDs match and all image IDs match. The mean square error is 5723, the edit distance is 139, and the slide IDs differ in this example. Thus it cannot be detected by the method in [10] either.



[DataMining]2001.11.14-Laura Squier.ppt Slide 017 [DataMining]Lecture20_257.ppt Slide 009

2001/11/14 0:07:46 2009/11/19 0:24:14

FIGURE 4.7: Example result of reused visual element detection

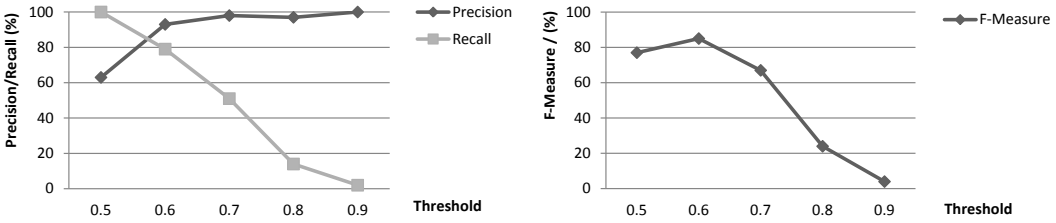
The example result of reused visual element detection is shown in Figure 4.7. It can be seen that an image is copied from left to right and then scaled. Another difference is that the slide on the original version contains more text on the bottom.

In this example, the text similarity is 36%, image similarity is 100%, and attribute similarity is 11%. The overall similarity is only 49%, and hence cannot be identified by the method in [2]. The mean square error of screenshots is 6835, the edit distance between the texts in the two slides is 61, and the slide IDs are different. Thus the method in [10] cannot detect the reuse in this pair either.

4.5.3.3 Quality of Reuse Detection Results

We study the quality of reuse detection by varying the similarity threshold t. We measure the precision – the percentage of reused results amid the retrieved ones, and the recall – the percentage of retrieved results amid the reused ones, formally defined by the following equations, where R_l denotes the set of true reused sentence/image groups in the dataset, and R_t denotes the set of reused sentence/image groups identified by our method.

$$\text{Precision} = \frac{|R_l| \cap |R_t|}{|R_t|}, \qquad \text{Recall} = \frac{|R_l| \cap |R_t|}{|R_l|}.$$



(a) Textual precision and recall (b) Textual F_1

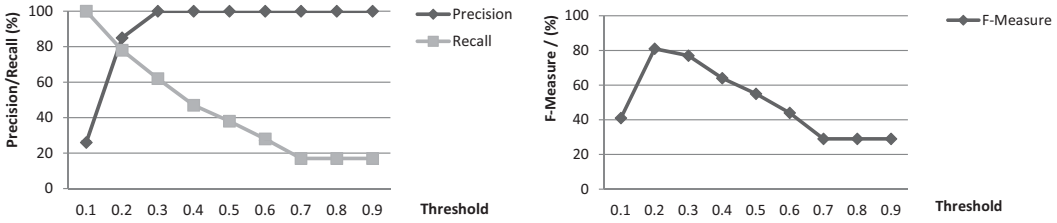
FIGURE 4.8: Experiment results of textual reuse detection

To find true reused sentence groups in the dataset, we first retrieve the sentence pairs that share at least 50% words, i.e., running our program with a threshold t of 0.5. This process significantly reduces the labor of human judge and barely misses true results. Afterwards we manually check the retrieved pair, construct reused sentence groups, and keep only true ones. To find true reused image groups in the dataset, we check all the images through thumbnails because their numbers are small.

For reuse textual element detection, we vary the threshold t from 0.5 to 0.9. Exact duplicates, i.e., reused sentences with no modifications, are removed from the results in order to test the quality of similarity search. Figure 4.8(a) shows the precision and recall. It can be observed that the precision increases with t and reaches 100% when t is 0.9. In contrast, the recall decreases with t and drops to only 2% when t is 0.9. These observations are expected because when t increases, the similarity constraint becomes stricter, and thus fewer pairs of sentences satisfy the constraint. The number of false positives is reduced, and this results in the increase of precision. On the other hand, this causes that the method misses true results, and consequently decreases the recall.

Figure 4.8(b) shows the F_1 score of reused textual element detection with varying thresholds. The general trend is that the F_1 score decreases when the threshold is rising. This is because the recall drops with increasing t and it changes more rapidly than the precision. Since our method achieves best F_1 when $t = 0.6$, we set t as 0.6 for the default setting of reused textual element detection.

For reused visual element detection, we vary the threshold t from 0.1 to 0.9, and plot



(a) Visual precision and recall (b) Visual F_1

FIGURE 4.9: Experiment results of visual reuse detection

the precision and recall in Figure 4.9(a), with exact duplicates removed. Similar trends can be observed as we have seen in the evaluation of reused textual element detection. The difference is that the precision in reused visual element detection changes more significantly. E.g., both trends are as high as 100%, but as low as 82% and 26%, respectively. This is because when the threshold is low, the bag-of-words model retrieves many similar images such as apples in different colors, but obviously they have no reuse relationship.

Figure 4.9(b) shows the F_1 score of reused visual element detection with varying thresholds. The general trend is that it first increases with t, peaks when $t = 0.2$, and drops as t keeps increasing. Therefore 0.2 is set as the default setting of the threshold of reused visual element detection.

4.5.4 Evaluating Reuse Visualization

In order to evaluate the reuse visualization method as well as the overall efficacy of our system, we conducted an experiment by recruiting five volunteers who deal with presentation slides on a daily basis and performing a trial of our system on the presentation files made by the volunteers, followed by a questionnaire and an interview.

Table 4.3 summarizes the users' opinions on our system. The scores are given on a 5-point Likert scale. As for the overall opinions, four out of five users gave 5 points to our system. The users were impressed with the design and the implementation of the system, commenting that this system would be helpful to the users who intend to track reused elements among quite a few slides.

TABLE 4.3: Users' opinions on our reuse detect system

Metric	**Average Score**
Overall rating	4.8
Identification of reused elements from slides	4.8
Interpretability of element timeline	4.8
Interpretability of presentation network	4.6
Usefulness for understanding presentation slides	4.6
Usefulness for tracking changes in slides	4.6
Usefulness for presentation slide composition	4.4
Interest in future use	4.6

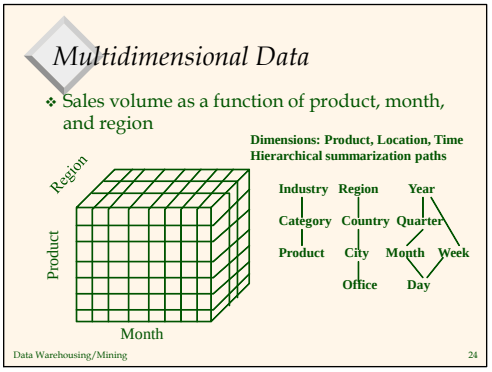
For the element timeline, the users considered the representations intuitive and easy to understand. One of them liked the apparent and impressive images shown by the system. For the presentation network, the users found the interface engaging and were impressed by the innovation of this idea. One user commented that the network shows the correlation between slides clearly and we can briefly obtain information from the network.

The users also assessed the usability of the system. They reported that the system helps understand the slides and it shows changes in slides intuitively through the element timeline. When being asked about the usefulness for presentation slide composition, they used words such as "somehow useful", "relatively high", "extremely useful", etc.

After the trial of the system, users were eager to use it in the future. They also pointed out the limitations of the system, which be discussed later in this section.

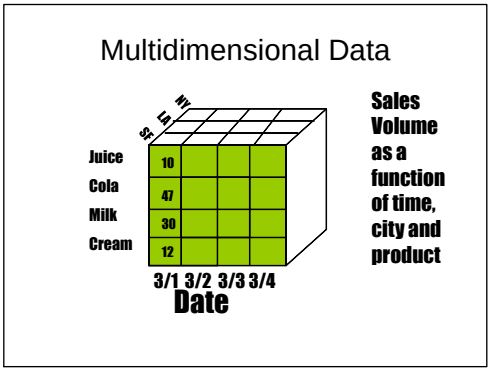
4.5.5 Error Analysis

For textual element reuse detection, an example of false positive is shown in Figure 4.10. The sentences detected are "Sales volume as a function of product, month, and region" and "Sales Volume as a function of time, city and product". The similarity between them is 0.692 but from the slides we observe that they are not reused sentences.



[DataMining]Chpt2.ppt Slide 024

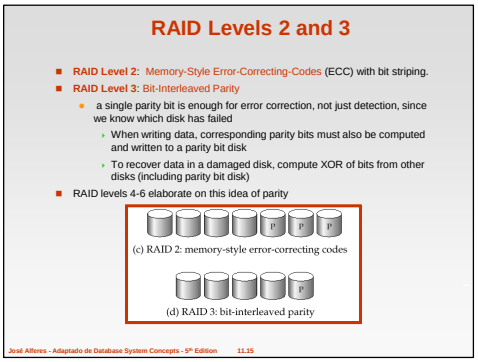
2004/02/05 7:39:50



[DataMining]Analysis technologies - day3 slides.ppt Slide 007

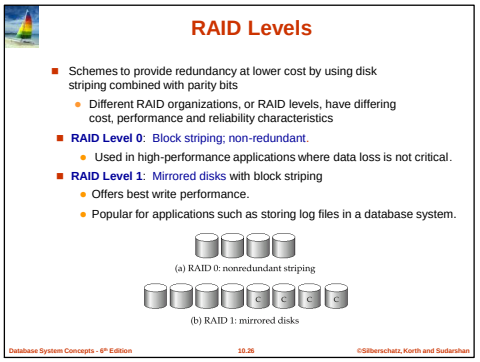
2004/06/26 0:46:44

FIGURE 4.10: False positive of reused textual element detection



[Database]ch11(1).ppt Slide 014

2007/03/07 9:18:07



[Database]ch10.ppt Slide 025

2012/03/15 0:32:19

FIGURE 4.11: False positive of reused visual element detection

The false positive of visual element reuse detection is shown in Figure 4.11. Both slides contain an image on the bottom The similarity between the visual words of the two images is 0.278. They are similar but not reused images. Both errors are due to the decreased precision of the bag-of-words model under low thresholds. A possible remedy to the above errors is to refine the results identified by similarity search with the more sophisticated machine learning techniques.

4.6 Conclusion and Future Work

In this chapter, we proposed an approach to managing presentation slides by exploiting reused slide elements. We developed different techniques to find textual and

visual elements reused in database slides. We devised interactive visualization tools to help users understand how these elements are reused and how the presentation files are related to each other. On the basis of the proposed techniques, a prototype system with a user-friendly interface is designed. Experiments were conducted on top of the system and demonstrated the effectiveness of the proposed methods.

Our future work is to explore the composition methods by reusing existing materials. Users may design the structure of the presentation and input elements or use examples to consist of the contents of the presentation. Then our method can automatically generates presentation slides.

Chapter 5

A Trial Design Support System for Presentation Slides

5.1 Introduction

As one of the most important ways of transferring information and knowledge, presentations provide users to discuss and exchange ideas together. Presentations now play an important role to promote understanding of presenters' ideas in many fields such as education and business. For example, slides for individual products may be needed to be included in a market presentation whereas for a management report slides for various projects may be required.

Many university lecturers use Web service to store, browse, and share presentation slides that are used in their lectures. Although powerful tools for slide composition have been developed and Web service has been widely used to share slides, they have a problem for preparing many lecture slides to help their students understand the contents. Lectures need to prepare slides that can promote understanding. But how to make the content of the slide fine grained and highly structured is still a problem to address. The traditional tools fail to provide any functions of handling presenters intentions. For example, the logical design of the slide structure and the interrelations between the objects. In presentation slides composition, it is

necessary for presenters to consider topics, items, and their relations. Audiences cannot understand the presentation until they see the consistent discussion points in the slide structure.

In order to cope with the above problems, we address a four-stage framework for slide design for presentation supporting. The four stages contains story, unit, page and layout steps. It can help users determine the topic units of slide structure in the story stage, make and assign objects for topic units in the unit stage, determine groups of objects in same slide in the page stage and search examples and arrange objects, and finally output the presentation slides, with the structure and component content fine grained and highly structured.

Most of the studies on the support of slide-making have focused on generating slides. [73, 74, 79] proposed systems to generate slides from academic papers. Information are extracted and summarized from academic papers based on the TF-IDF term weighting. Sentences are assigned to slides and important phrases are identified with bullets. [71] proposed to convert documents to slides by parsing the discourse structure of documents and representing in an outline format resulting tree. These approaches that only focus on a structured summary of documents according to the documents structure and generate the slides automatically disregard the importance of how to express the slide according to users' intentions. In constrast, our method focuses on the role of user's content decision and help users with the structure and component design of the slide with content fine grained and highly structured.

[75] proposed SlideSeer, a system for the alignment, discovery and presentation of such document and slides pairs. It modifies the maximum similarity in alignment in order to favor monotonic alignments, and a classifier is integrated to handle the case when slides should not be aligned. [99] detected important description of a word in a document where the word appears most frequently. Different from the terms retrieval of the important information, our method, considers the user's intention of the content alignment and their associated objects of the slide units.

From the perspective of reusing slides, [2] and [1] proposed slide composition systems that create slides from existing ones and adapt them for different events such as

conferences and lectures. Since the problem on how to support organizing slide components has been addressed [100], [101] investigated semantic relationships among slide elements and reflected a presentation strategy in the design of slides. They are similar to ours for helping presenters better organize contents of their slides. We aim at supporting presenters prepare slides with slide designed and generated by the contents fine grained and highly structured.

In this research, we investigate the problem of support for presentation slide design. Unlike previous methods, we consider the role users play in the presentation structure design and content generation. The users may select topics from a statistic list to design the structure of the presentation, then input or search the contents for each topic unit, with the help of the element search method, the user can reuse the elements returned slides. Next, users can allot components into one slide, and select or specify the layout of the slide according to their preference. Based on the proposed framework, a prototype system with a user-friendly interface is designed. The experimental evaluation on real presentation slide design and generation demonstrates the effectiveness and usefulness of the proposed framework.

Our preliminary work can be summarized as follows:

- We propose a framework to support the presentation slide design activities.
- We integrate the content-based element search method to help user reuse the existing materials, and provide different methods to support the presentation design in different stages.
- We preliminarily conduct experiments to evaluate the effectiveness and the usefulness of the proposed framework.

This chapter is organized as follows. Section 5.2 propose a framework of presentation slide design support system. Section 5.3 introduce the methods in the proposed framework. Section 5.4 implement the framework and evaluate the system preliminarily. Finally, section 5.5 presents our conclusion and future work.

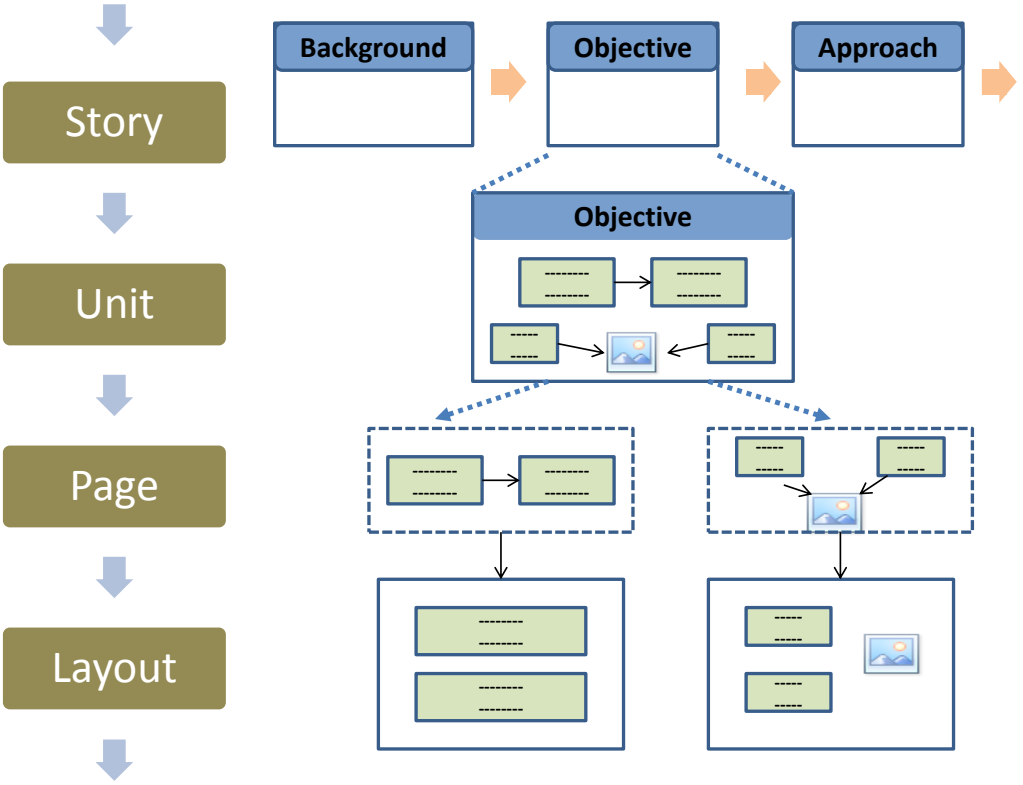


FIGURE 5.1: The four-stage framework of the presentation slide design support

5.2 Conceptual Viewpoint with Framework

Figure 5.1 illustrates an overview of the framework of our presentation slide design support system. It is composed of four stages, story, unit, page and layout modules. Next the four stages are introduced respectively.

5.2.1 Story Stage

The story is to do with *what to say*, a framework for composition, to answer what is the background, objective, idea, approach, detailed processing method, evaluation and prospects. In such perspective, it is necessary to achieve the deployment composition information, in the basic type of objective, idea, approach, method, results.

It is the phase we arrange components according to the format of one story. It is difficult to adapt its content and structure automatically. It is able to give guidance

to provide a series of templates for slides, prepare a single topic and then expand it. In this stage, the main structure of the presentation is designed. Users can select the topics from a statistics list to design the main unit of the presentation, in this stage, users can add, delete, change the order of the unit to get the best structure for presentation.

5.2.2 Unit Stage

The story section basically specifies the sequence of the unit, which means the story itself is not very long. The unit section is the real part of the input data according to the configuration of the units is defined, it is possible to save the information of the attribute value as the data link information specify. The basics of creating slides are the construction of the story and unit.

The unit stage is the phase to enter the data to construct the content of the individual topic designed in the story stage. In this stage, users can input the content by themselves, such as type some text, load an image, picture, diagram from the files. If the user have not enough materials, the content-based search method proposed in Chapter 3 is integrated in this stage. He can input some keywords, sentence, an image and a diagram to search slides which he is interested in. Then he can select some elements from the returned slides as the content of the unit. For the textual content, it is stored in text files, while for the visual content, the file path or the URL of the image, picture, diagram are stored in files.

Besides the content of the elements themselves, the attributes and the relationships to others can also be specified. For example, some texts describe a table, an image or explain a diagram, one image is in sequence of another, or opposite to or compared with another image. These relationships between each other will help the user put these elements into one slide page or successive slide pages.

5.2.3 Page Stage

In this stage, the user will allot contents into one slide. If the texts are discussing one topic, or texts are explaining a table, a diagram, an image, these components will be assigned to one slide.

In order to make the slide easy to read and understand, the user should not put too many text or images in one slide. If there are too many texts in one slide, the font size of the text will not be so large for read, at the same time, the text-heavy slides is a time-consuming task to understand.

5.2.4 Layout Stage

The layout section is pair of configuration of the integrated page section. The layout process the construction of the data in the page section.

In this stage, user will design the layout of the contents in one slide. We will provide some templates for user to select, the user can also specify the layout by himself, the font, position, size and so on.

5.3 Design Support Methods

In this section, we present the design support methods in different stages proposed in Section 5.2. In the story stage, we give some general topics in academic reports for user to select. In the unit stage, we provide the element search method to search contents in returned slides, and in the layout stage, we show some templates for user to assign the layout of the contents in one slide.

5.3.1 Story Stage

We collect main topics from the academical report from our previous dataset in Chapter 3. Then provide a list of topic to help user design the structure of the

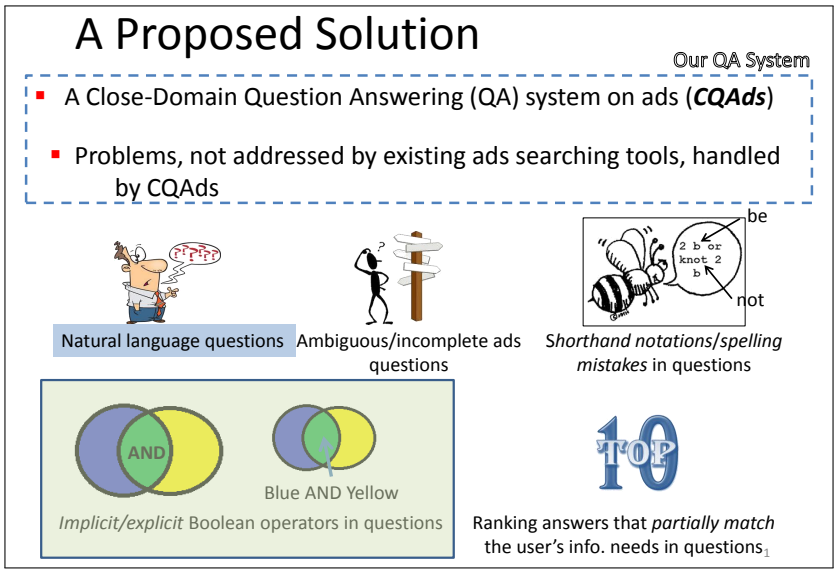


FIGURE 5.2: Select contents from the returned slides by Chapter 3

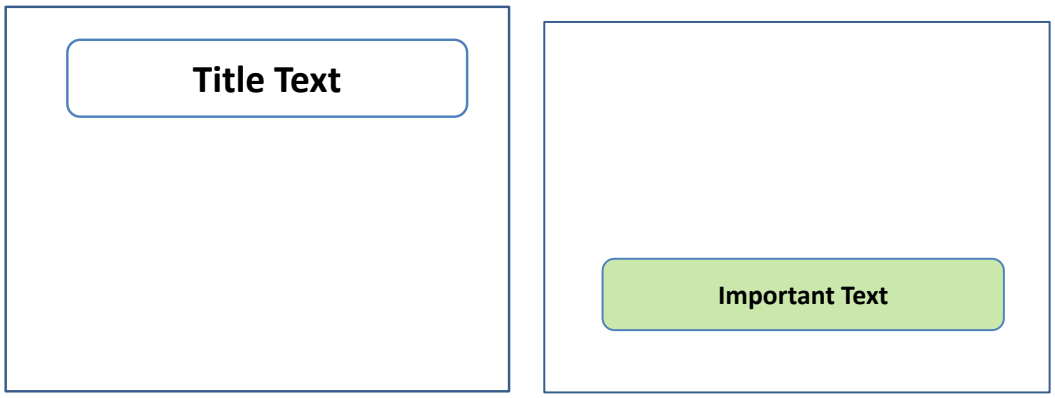
presentation. User can add, delete topics from the list to design, he can also change the order of the sequence. Hence, the structure of the presentation is designed.

5.3.2 Unit Stage

In this stage, contents of the unit of the presentation will be added to describe the topics. Users can input the contents by themselves, type some text, insert a table, select an image, a picture or a diagram from the files.

If the user have not enough contents to insert, he can also search some interested contents by input keywords, sentence, or an image using the methods proposed in Chapter 3. Then he can select some elements of the returned slides. Figure 5.2 shows some elements that user can select to reuse.

Similar to the element query input in the Section 3.2, the element in Figure 5.2 such as keywords, sentences, image, diagram will be selected as the contents to consist of the topic.



(a) Template for Title (b) Template for Text

FIGURE 5.3: Template of text layout

5.3.3 Page Stage

In this stage, the user will allot contents into one slide. If the texts are discussing one topic, or texts are explaining a table, a diagram, an image, these components will be assigned to one slide.

In order to help user put proper count of the text and image, we make a statistics on the text and image number among dataset slides.

5.3.4 Layout Stage

In the layout stage, user will design the layout of the contents in one slide. We will offer some templates for user to select, the user can also specify the layout by himself, the font, position, size and so on.

Here we give some templates for the user.

First is the title text in Figure 5.3(a) and the important text in Figure 5.3(b) .

As the general title format in the presentation slides, title will be alloted on top of the slide in a large size, while for the important texts, the text will be put in the highlighted text rectangle to show the arresting feature.

Figure 5.4 shows a template for the group texts. The texts will be shown in vertical sequence, with the summary text in a higher text level.

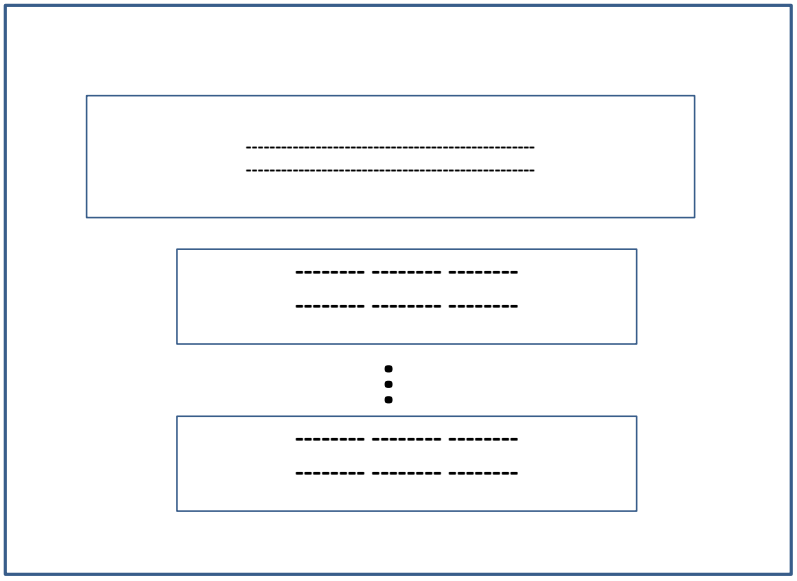


FIGURE 5.4: Template of group text layout

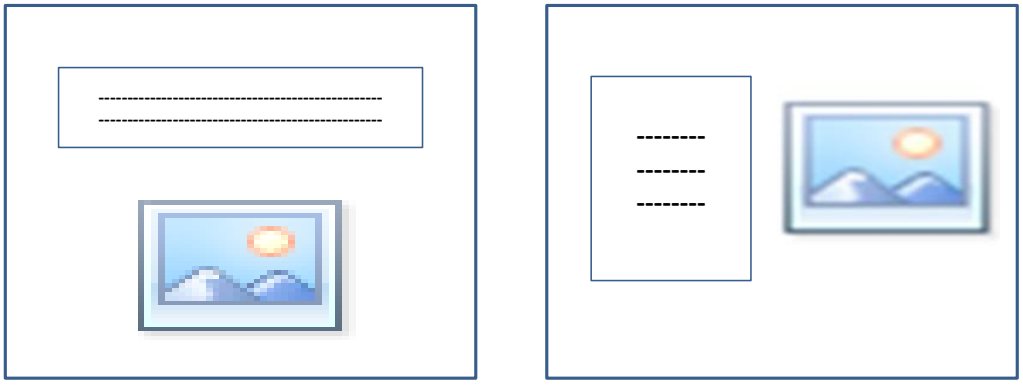


FIGURE 5.5: Template of one image layout

Figure 5.5 shows templates for text and one image assignment. The image can be put in the bottom of the slide or the right of the slide. In this way, both the text and the image will be easy to read and understand.

Figure 5.6 shows templates for two images assignment. Similar to the one image layout, the two images can be put in the bottom of the slide horizontally or on the right of the slide vertically.

In this stage, we provide some sample template for the layout. Meanwhile, the user can specify the layout at their own willings.

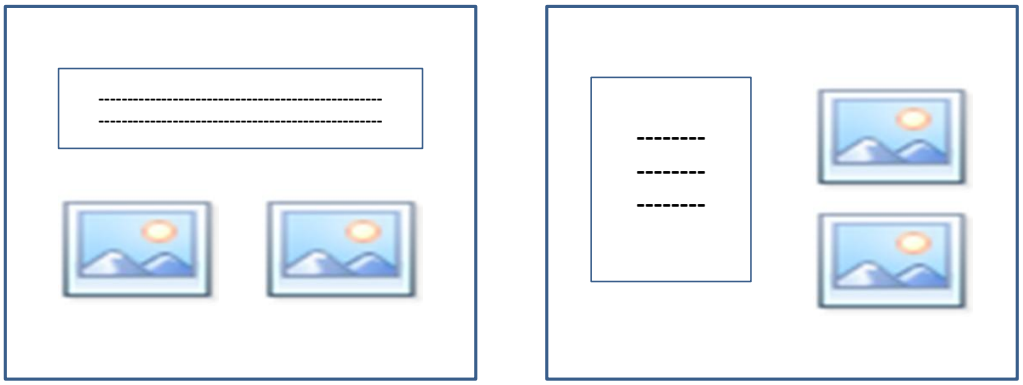


FIGURE 5.6: Template of two images layout

5.4 Experiments

We design a slide design support system according to the proposed framework integrated with the element search methods. In this section, we introduce the user interface of the system and report the preliminary experiment results conducted on the system.

5.4.1 Prototype System

We implemented a prototype of slide element search system in C#. It consists of four modules: the story stage module, the unit stage module, the page stage module and the layout stage module. Finally is the presentation slide output module.

5.4.1.1 Story Stage Module

In the story stage, users design the structure of the presentation slide. It is able to give guidance to provide a series of templates for slides, prepare a single topic and then expand it. In this stage, the main structure of the presentation is designed. Users can select the topics from a statistics list to design the main unit of the presentation, in this stage, users can add, delete, change the order of the unit to get the best structure for presentation by click the corresponding buttons shown in Figure 5.7.

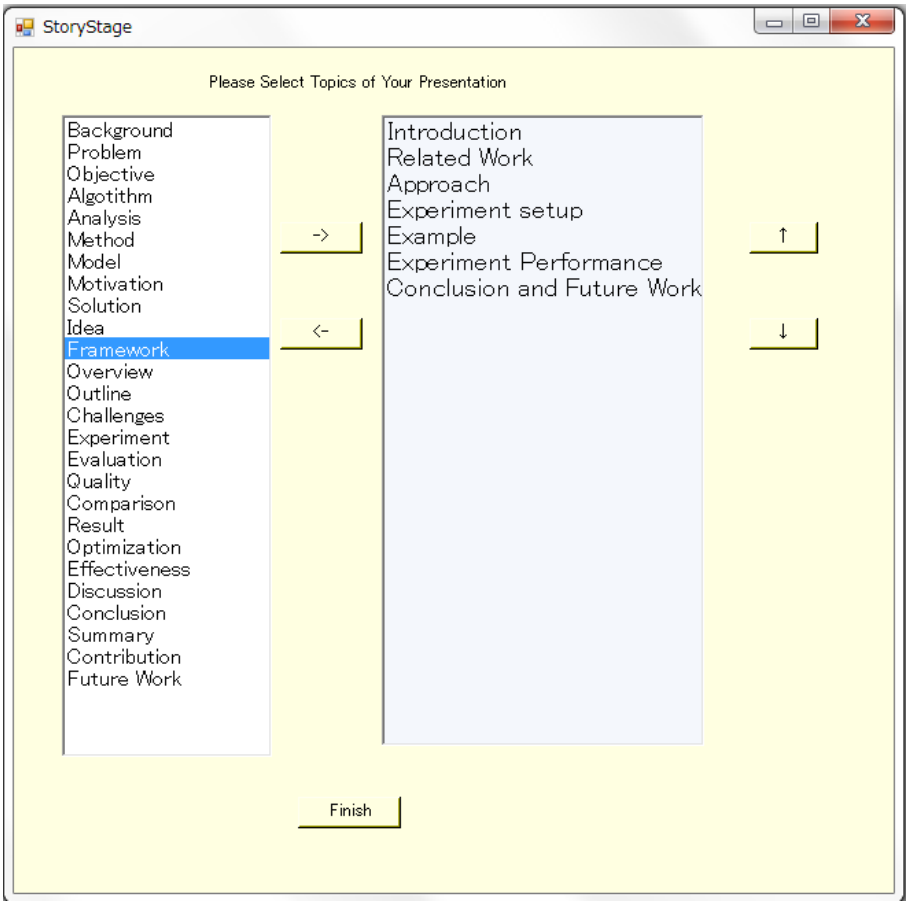


FIGURE 5.7: Interface of story stage

We list the statistic of the topics from the VLDB 2011 and VLDB 2012 presentation slides, the frequency of the topics are shown in Table 5.1.

5.4.1.2 Unit Stage Module

The unit stage is the phase to enter the data to construct the content of the individual topic designed in the story stage. In this stage, users can input the content by themselves, such as type some text, load an image, picture, diagram from the files. If the user have not enough materials, the content-based search method proposed in Chapter 3 is integrated in this stage. User can click the query button to search the interested slides.

The returned slide will be listed in the top-right corner in Figure 5.8. When user click one of them, the slide will be shown in the bottom of the interface, then user can select some keywords, sentence, image or diagram in the slide to input. If the

TABLE 5.1: Statistics of the topics on academical report presentation slides

Topic	**Count**	**Topic**	**Count**	**Topic**	**Count**
Introduction	16	Background	11	Related Work	12
Outline	218	Problem	10	Objective	8
Approach	13	Algorithm	49	Analysis	10
Method	22	Model	13	Motivation	47
Solution	6	Idea	9	Framework	31
Overview	35	Challenges	28	Function	11
Experiment	21	Evaluation	19	Experiment Setup	19
Quality	30	Example	4	Comparison	7
Result	16	Performance	9	Optimization	29
Effectiveness	10	Improvement	4	Error Results	6
Discussion	9	Contribution	25	Summary	31
Conclusion	25	Future Work	12	Conclusion and Future Work	3

elements have some relationships to others, users can specify it to help element assign.

5.4.1.3 Page Stage Module

In this stage, all the elements user input in the unit stage is shown in the top of the Figure 5.9. Users can select a group of them then put them into one same slide.

Then the pageID, elementID, the count of the text and image will be shown in the list of the page interface.

In order to get a reference to the text and image count, we give a survey on some of the slides in Table 5.2.

This will help user put proper count of text and image into one same slide.

5.4.1.4 Layout Stage Module

In this stage, according to the text and image count in one same slide, the sample templates provided in Section 5.3 will be shown to users. At the same time, users can also specify the layout according to their own choice as shown in Figure 5.10.

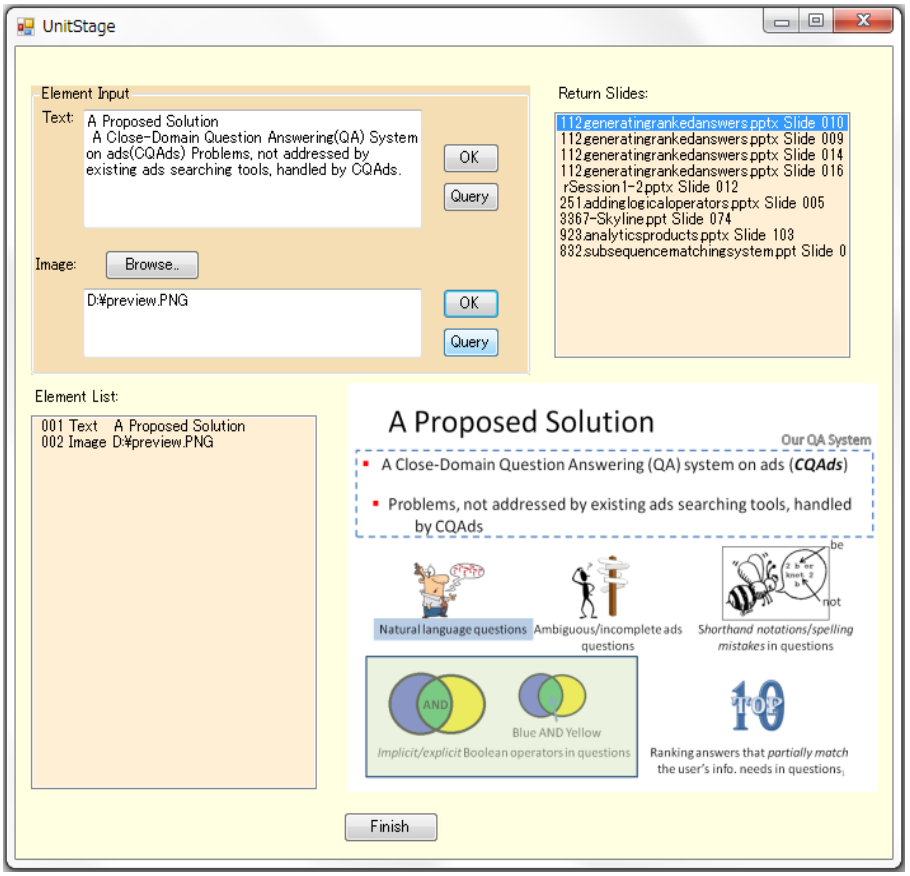


FIGURE 5.8: Interface of unit module

TABLE 5.2: Statistic of the element on slides

Textual Element	Visual Element	Slide Count
1	0	3
1	1	22
1	2	14
2	0	3
2	1	6
2	2	9
3	1	1
4	5	1
0	1	15
0	0	8

Users can click the "add" button to specify the layout of the elements in Figure 5.11. When all the elements are specified, then users can preview the slide on the bottom of the Figure 5.10.

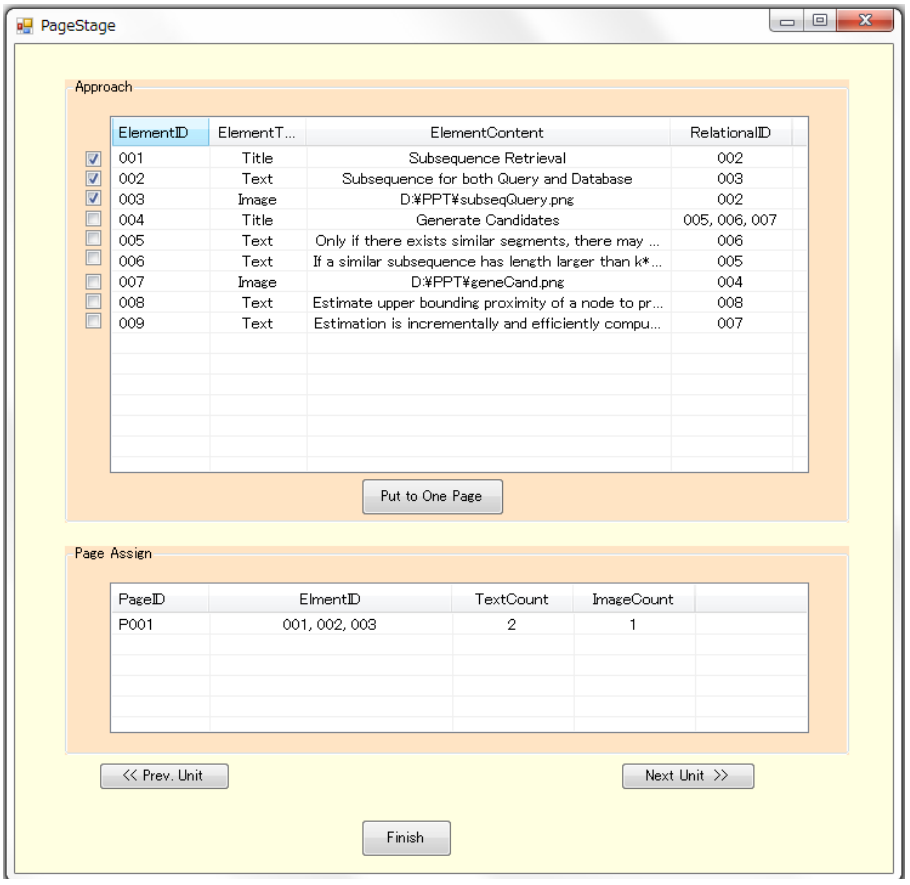


FIGURE 5.9: Interface of page stage

5.4.2 Experiment Setup

In this section, we first give the composition quality evaluation of the proposed method, then show the usefulness of the design support framework by a questionnaire.

The experiments are run on a PC with a 3.4 GHz CPU and 8.0 GB of RAM.

For the topics selection in the story stage, we count the slide title of the academic reports from the dataset in Chapter 3.5. And give some high ranking topics for the user to select.

5.4.3 Evaluating Composition Results

In this section, we will check our composition results to see whether our prototype system works well. We show some example query results first.

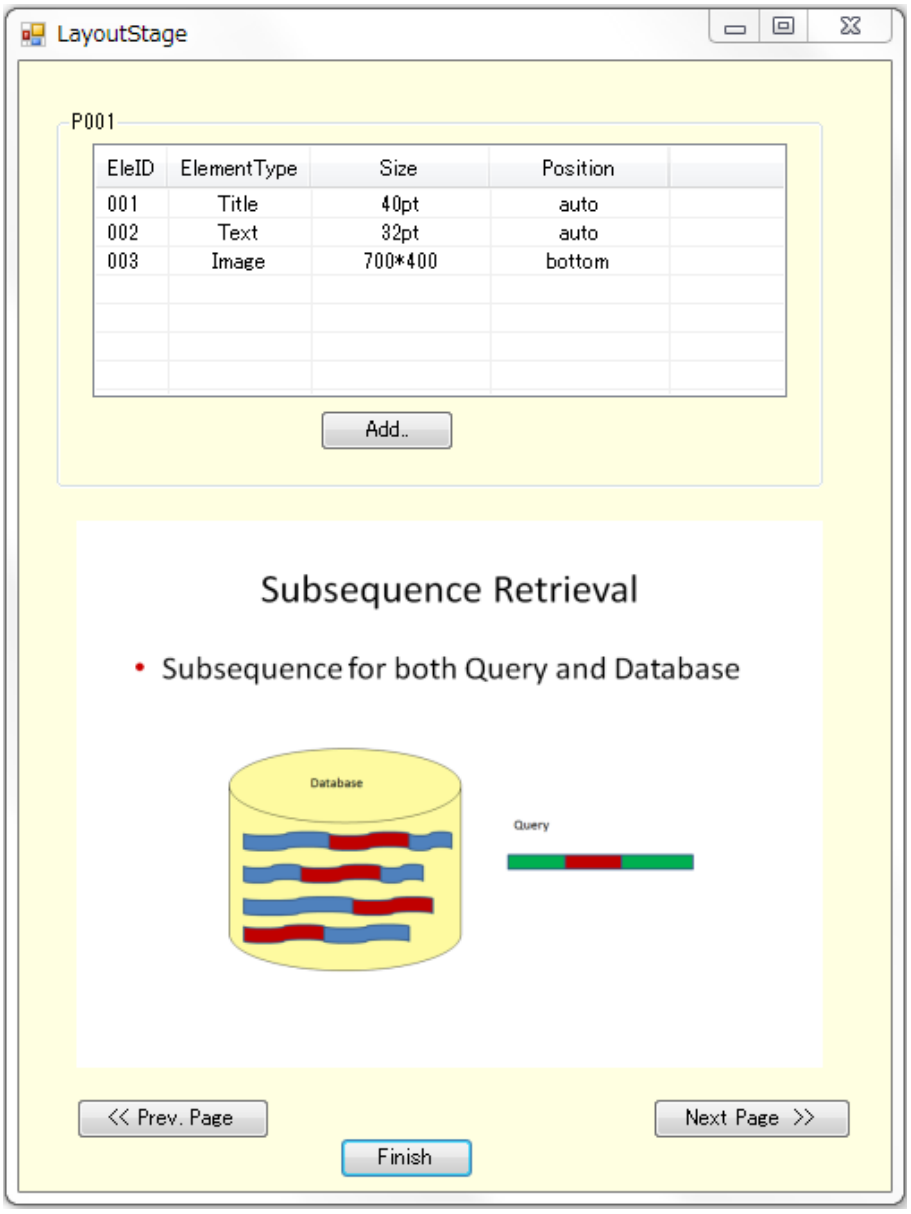


FIGURE 5.10: Interface of layout stage

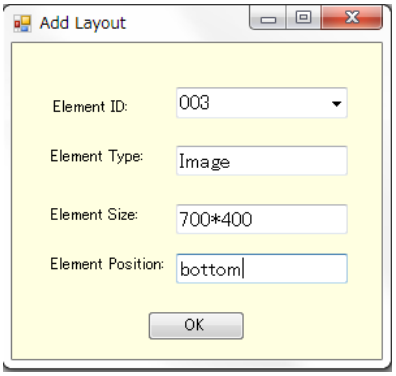
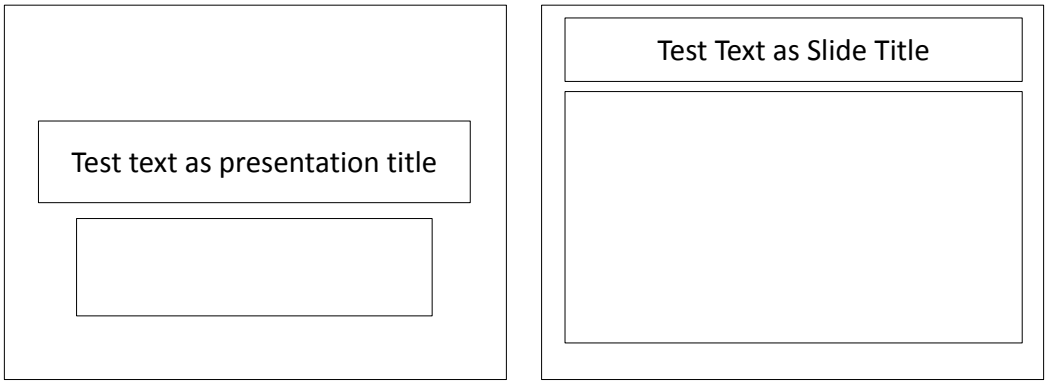


FIGURE 5.11: Interface of add layout



(a) Title for presentation (b) Title for slide

FIGURE 5.12: Example of title output

Figure 5.12 shows two results for the title as presentation title and slide title. If the slide is the first page of the presentation, the input text will be put in the middle of the slide as the presentation title. While for other slides, all the titles are put on top of the slides.

Figure 5.13 shows the example results for the text as important text and group text. With the attribute specified in the unit stage, all of the textual elements are assigned properly.

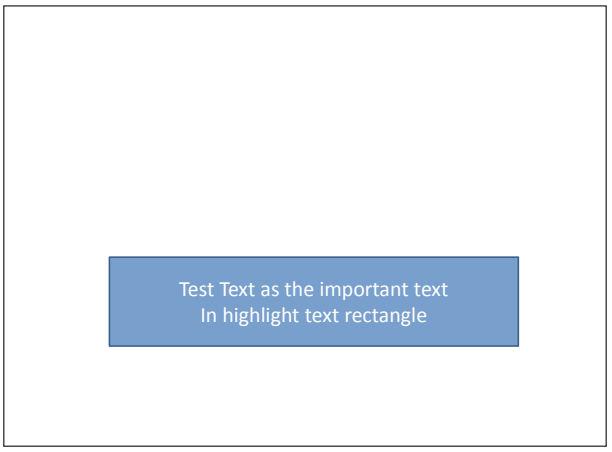
Figure 5.14 shows two output slides of one image. Both of the text and image assigned according to the templates.

Figure 5.15 shows two output slides of two images. similar to the one image template output, both of the text and images assigned according to the templates.

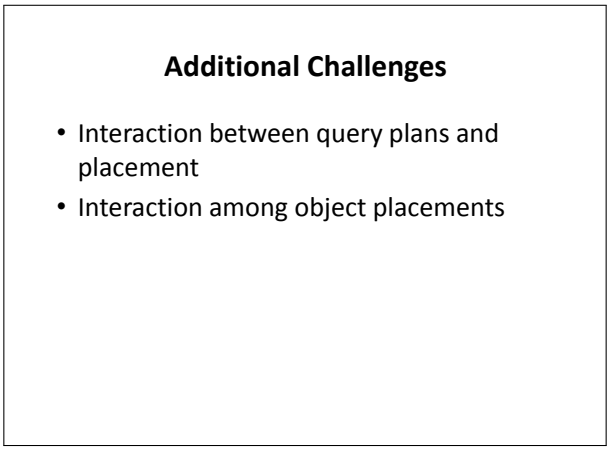
5.4.4 Evaluating Design Support System

In order to evaluate the presentation design support system, we conducted and experiment by recruiting participants and performing a trial of our system on the presentation slides made by them, followed by a questionnaire and an interview.

Table 5.3 summarizes the users' opinions on our system. The scores are given on a 5-point Likert scale. As for the overall opinions, four out of five users gave 5 points to our system. The users were impressed with the design and the implementation of



(a) Important text in slide



(b) Group text in slide

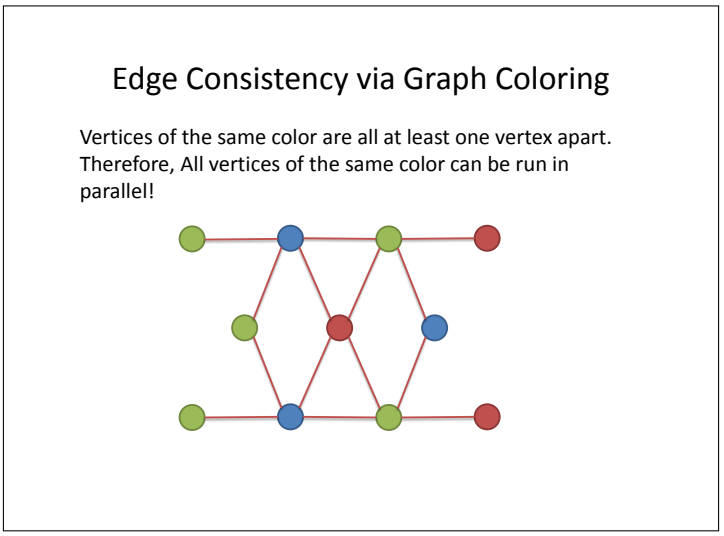
FIGURE 5.13: Example of text output

the system, commenting that this system would be helpful to the users who intend to design presentations.

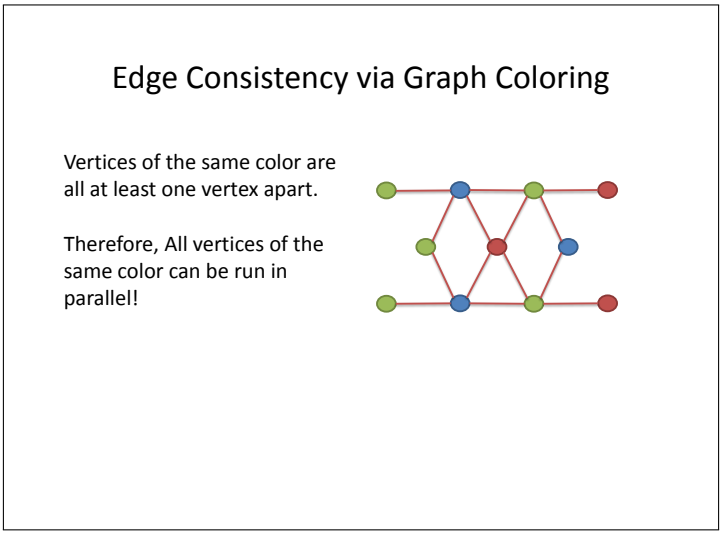
For the story stage, the users considered the presentation structures intuitive and easy to understand. One of them liked the detailed and sequence shown in this stage.

For the unit stage, the users found the interface engaging and were impressed by the innovation of this idea. One user commented that the query method helped find useful contents from returned slides clearly and we can briefly obtain information from the slides.

In the page stage, the users found it is easy to assign the elements into one page. and in the layout stage, it is said that the preview is really helpful to assign the



(a) 1image



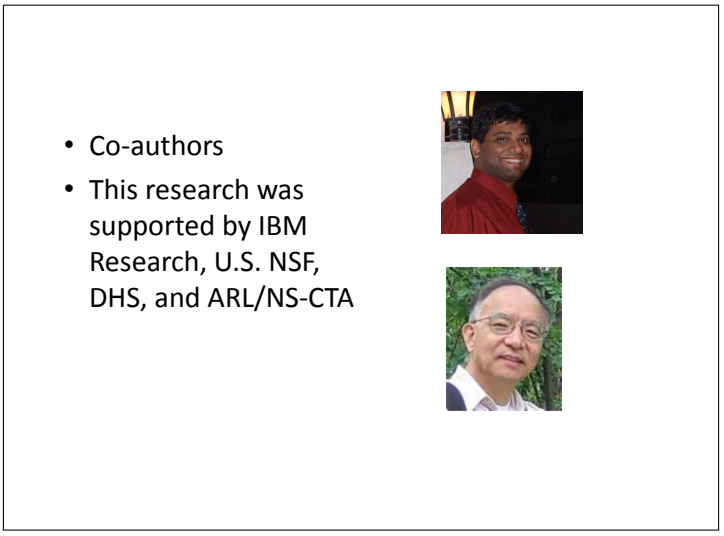
(b) 1image1

FIGURE 5.14: Example of one image layout

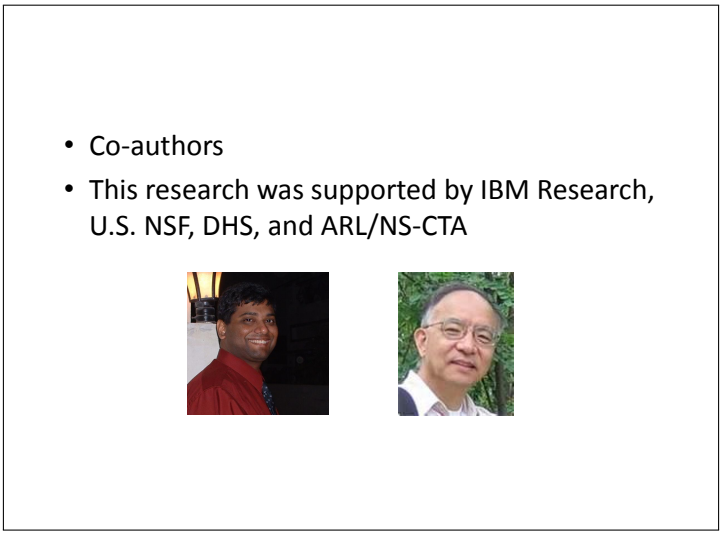
layout of the elements.

The users also assessed the usability of the system. They reported that the system helps design the structure and search the contents of the slides. When being asked about the usefulness for presentation slide composition, they used words such as "somehow useful", "relatively high", "extremely useful", etc.

After the trial of the system, users were eager to use it in the future. They also pointed out the limitations of the system, which be discussed later in this section.



(a) 2image



(b) 2image1

FIGURE 5.15: Example of two images layout

TABLE 5.3: Users' opinions on our design support system

Metric	**Average Score**
Overall rating	4.6
Usefulness of story stage	4.6
Usefulness of unit stage	4.8
Usefulness of page stage	4.6
Usefulness of layout stage	4.8
Helpfulness for design presentation structures	4.6
Helpfulness for input presentation contents	4.8
Interest in future use	4.4

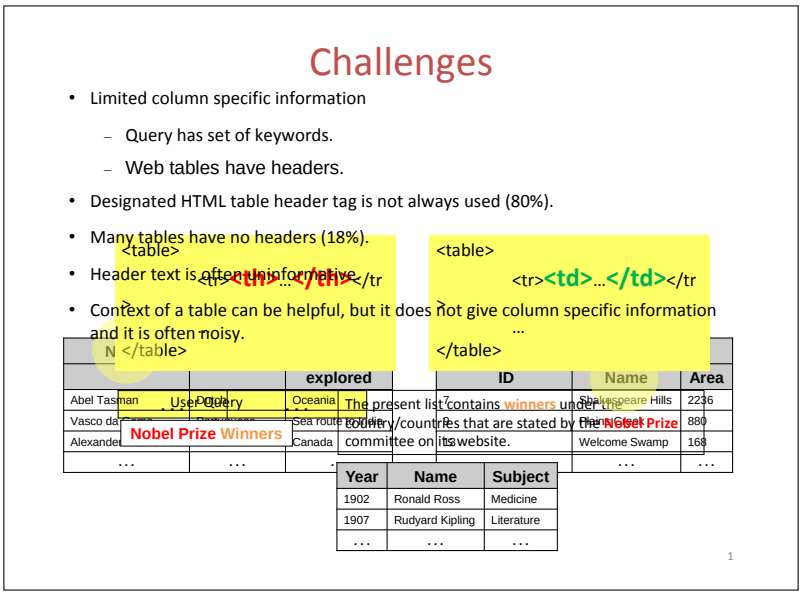


FIGURE 5.16: Failure example of slide in text

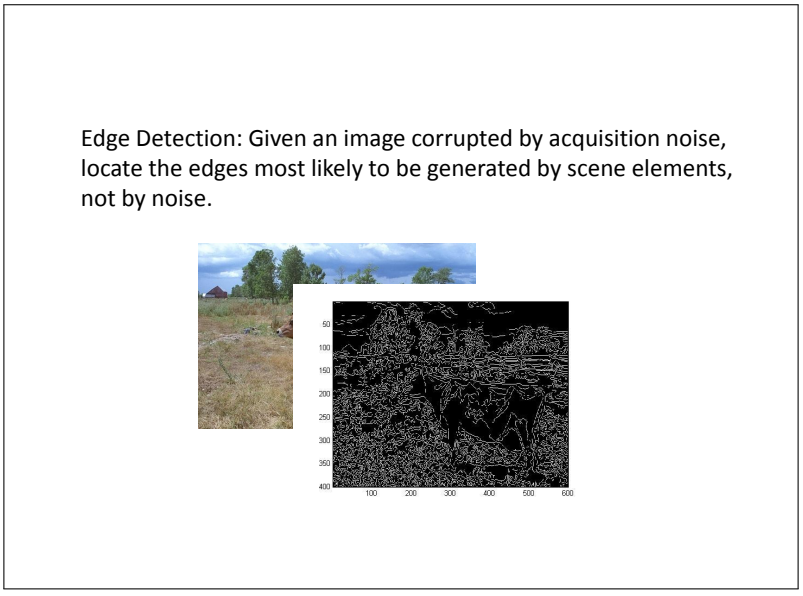


FIGURE 5.17: Failure example of slide in image

5.4.5 Error Analysis

For the layout assignment, some example of failure slides are shown in Figure 5.16 and Figure 5.17.

We can see that in this slide, not all the text are shown clearly, Since user put too many texts in one slide. Based on this failure example, we have to set some constraint about the length of the text, or the count of the elements.

We can see that in this slide, there are only one text, two image assigned. But not all the images are shown, because the user does not specify the position of each image. The image are inserted into the slide randomly, which causes one image is on top of another.

5.5 Conclusion and Future Work

In this chapter, we proposed four-stage framework for presentation design support. Users can first select the topics to design the structure of the presentation, input the contents for each unit by themselves or search some interested slides, then allot the elements into one same page and finally decide the layout of each slide. On the basis of the proposed framework, a prototype system with a user-friendly interface is designed. Preliminary experiments were conducted on top of the system and demonstrated the effectiveness of the proposed framework. The experiment results show that our proposed framework can help users to design the presentation structure, input the contents, assign the element at the willing of themselves.

Our future work includes the improvement of the system based on the failure example of the experiment results and other experiments such as the detailed experiment set up and error tests. What's more, the system should be able to back to the previous stage to make the structure and contents properly assigned.

Chapter 6

Conclusions and Future Work

In this chapter, we first summarize this thesis in Section 6.1. Next, we present several interesting extensions that serve as our future work in Section 6.2.

6.1 Conclusions

We have studied three perspectives of the presentation reuse support, the content-based element search method, the reuse detection method and the presentation design support framework. The content-based element search provide a method to find interested slides by different types of input: keywords, sentences, images and diagrams. The reuse detection method provide a new presentation slide management method to have an overview of the presentations in network and track the reused elements in timeline. The presentation design support framework provides a system to design the presentation structure, input contents, allot pages and assign layouts by reusing the existing slides.

6.1.1 Content-based Element Search

For the content-based element search for presentation slides, we proposed content-based search methods for a variety of elements in presentation slides. Users can

freely choose keywords, a sentence, an image, or a diagram as a query to find the materials of his interest from a collection of presentation slides. We proposed different query processing methods to improve the efficiency of answering these queries. We designed a prototype system integrated with the proposed methods along with a user-friendly interface, and conducted experiments on top of it. The experiment results show that our proposed methods return better results than alternative methods and are much faster than the methods without indexes.

6.1.2 Reused Element Detection

For the reused element detection, we proposed an approach to managing presentation slides by exploiting reused slide elements. We developed different techniques to find textual and visual elements reused in database slides. We devised interactive visualization tools to help users understand how these elements are reused and how the presentation files are related to each other. On the basis of the proposed techniques, a prototype system with a user-friendly interface is designed. Experiments were conducted on top of the system and demonstrated the effectiveness of the proposed methods.

6.1.3 Presentation Design Support

In order to help user design the presentation slides, we proposed a four-stage framework to help user design the structure of the presentation. Users can freely choose the topics of the presentation to design the structure, insert contents for each topic, then assign components into one page and finally decide the layout of the slides. Thus it makes the presentation content fine grained and highly structured. For the slide contents, users can input the contents by themselves or search the interested contents from the slide datasets. For the slide layout, users can use the template we provide, or specify the fonts, the sizes, and the position of the contents by themselves. On the basis of the proposed framework, a prototype system with a

user-friendly interface is designed. Preliminary experiments were conducted on top of the system and demonstrated the effectiveness of the proposed framework.

6.2 Future Work

For the three perspectives about the presentation reuse, we also consider the improvement of the current methods. We discuss them separately.

6.2.1 Content-based Element Search

For the content-based element search, our future work includes the improvement of the search method. Based on the error analysis of the experiment results, we have to define some constraints for the keyword queries. A possible remedy is to restrict that all keywords must be contained in a result. For a sentence query, in order to avoid the results contain almost all the words of the query but in separate locations, we can improve the sentence query by using shingles instead of words.

Besides the improvement of the search method on presentation slides, we can also consider other types of files, such as the Word documents, the PDF and the XML files. These files usually contains textual and visual contents, with these contents retrieved, our element search methods can be applied in these files. Our future work about content-based element search can extent our methods to these types of files.

6.2.2 Reused Element Detection

For the presentation slide management system from the perspective of reused elements, we can consider how to integrate it into composition tools for the design support. At the same time, we also have to improve the reuse detection methods. Based on the false positive of the reused element detections, we can see that these errors in the textual and visual elements detection are due to the decreased precision of the bag-of-words model under low thresholds. We have to consider the remedy to

refine the results identified by similarity search with the more sophisticated machine learning techniques.

Since the detecting methods use the similar functions in the element search methods, our reused element detecting methods can also work on other types of files.

6.2.3 Presentation Design Support

Since preliminary experiments are conducted on this work, our future work is to conduct more detailed experiments and make the system more flexible. If the users find some problem with the structure and contents, he can back to the upper stages to modify the topics or the contents. Based on the failure examples of the layout, we have to set some constraints about the counts of the textual and visual elements put into one same slide. In addition, if the users neither select the sample template nor specify the layout by themselves, the system should be able to assign all the elements in proper layouts automatically.

This design support system integrates the element search methods in Chapter 3, in our future work we can extent the element search into other types of file containing textual and visual contents, which make the data sources more abundant. While these structured files can be also designed by our system, the only difference is the file generation software and environment in the final output modules. Thus our research can be applied in more file types and applications.

Bibliography

[1] Yelena Mejova, Klaar De Schepper, Lawrence Bergman, and Jie Lu. Reuse in the wild: an empirical and ethnographic study of organizational content reuse. In *ACM CHI Conference on Human Factors in Computing Systems*, pages 2877–2886, 2011.

[2] Moushumi Sharmin, Lawrence Bergman, Jie Lu, and Ravi B. Konuru. On slide-based contextual cues for presentation reuse. In *International Conference on Intelligent User Interfaces*, pages 129–138, 2012.

[3] Pierre Tirilly, Vincent Claveau, and Patrick Gros. Distances and weighting schemes for bag of visual words image retrieval. In James Ze Wang, Nozha Boujemaa, Nuria Oliver Ramirez, and Apostol Natsev, editors, *Proceedings of the 11th ACM SIGMM International Conference on Multimedia Information Retrieval, MIR 2010*, pages 323–332. ACM, 2010.

[4] Haruo Yokota, Takashi Kobayashi, Taichi Muraki, and Satoshi Naoi. UPRISE: Unified presentation slide retrieval by impression search engine. *IEICE Transactions*, 87-D(2):397–406, 2004.

[5] Lawrence Bergman, Jie Lu, Ravi B. Konuru, Julie MacNaught, and Danny L. Yeh. Outline Wizard: presentation composition and search. In *International Conference on Intelligent User Interfaces*, pages 209–218, 2010.

[6] A. Kushki, M. Ajmal, and K. N. Plataniotis. Hierarchical fuzzy feature similarity combination for presentation slide retrieval. *EURASIP Journal on Advances in Signal Processing*, 2008:188:1–188:19, 2008.

[7] Yuanyuan Wang and Kazutoshi Sumiya. A browsing method for presentation slides based on semantic relations and document structure for e-learning. *Journal of Information Processing*, 20(1):11–25, 2012.

[8] Alessandro Vinciarelli and Jean-Marc Odobez. Application of information retrieval technologies to presentation slides. *IEEE Transactions on Multimedia*, 8(5):981–995, 2006.

[9] Seitaro Tanaka, Taro Tezuka, Atsushi Aoyama, Fuminori Kimura, and Akira Maeda. Slide retrieval technique using features of figures. In *International MultiConference of Engineers and Computer Scientists*, volume 1, pages 424–429, 2013.

[10] Steven M. Drucker, Georg Petschnigg, and Maneesh Agrawala. Comparing and managing multiple versions of slide presentations. In *ACM Symposium on User Interface Software and Technology*, pages 47–56, 2006.

[11] Ryan P. Spicer, Yu-Ru Lin, Aisling Kelliher, and Hari Sundaram. Nextslideplease: Authoring and delivering agile multimedia presentations. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 8(4): 53, 2012.

[12] Benjamin B. Bederson and James D. Hollan. Pad++: A zooming graphical interface for exploring alternate interface physics. In *ACM Symposium on User Interface Software and Technology*, pages 17–26, 1994.

[13] M. Sravanthi, C. Ravindranath Chowdary, and P. Sreenivasa Kumar. Slidesgen: Automatic generation of presentation slides for a technical paper using summarization. In *Florida Artificial Intelligence Research Society Conference*, pages 284–289, 2009.

[14] Koichi Hanaue, Yusuke Ishiguro, and Toyohide Watanabe. Composition method of presentation slides using diagrammatic representation of discourse structure. *International Journal of Knowledge and Web Intelligence*, 3(3): 237–255, 2012.

[15] Yuanyuan Wang and Kazutoshi Sumiya. A method for generating presentation slides based on expression styles using document structure. *International Journal of Knowledge and Web Intelligence*, 4(1):93–112, 2013.

[16] Derek L. Hansen and Jennifer Golbeck. Mixing it up: recommending collections of items. In Dan R. Olsen Jr., Richard B. Arthur, Ken Hinckley, Meredith Ringel Morris, Scott E. Hudson, and Saul Greenberg, editors, *Proceedings of the 27th International Conference on Human Factors in Computing Systems*, pages 1217–1226. ACM, 2009.

[17] Moushumi Sharmin, Brian P. Bailey, Cole Coats, and Kevin Hamilton. Understanding knowledge management practices for early design activity and its implications for reuse. In *Proceedings of the 27th International Conference on Human Factors in Computing Systems*, pages 2367–2376, 2009.

[18] Adrian Secord, Holger Winnemoeller, Wilmot Li, and Mira Dontcheva. Creating collections with automatic suggestions and example-based refinement. In Ken Perlin, Mary Czerwinski, and Rob Miller, editors, *Proceedings of the 23rd Annual ACM Symposium on User Interface Software and Technology*, pages 249–258. ACM, 2010.

[19] David Strang and Sarah A. Soule. Diffusion in Organizations and Social Movements: From Hybrid Corn to Poison Pills. *Annual Review of Sociology*, 24(1):265–290, 1998.

[20] Everett M Rogers, Una E Medina, Mario A Rivera, and Cody J Wiley. Complex adaptive systems and the diffusion of innovations. *The Innovation Journal: The Public Sector Innovation Journal*, 10(3):1–26, 2005.

[21] Carlos Jensen, Heather Lonsdale, Eleanor Wynn, Jill Cao, Michael Slater, and Thomas G. Dietterich. The life and times of files and information: a study of desktop provenance. In *Proceedings of the 28th International Conference on Human Factors in Computing Systems*, pages 767–776, 2010.

[22] Udi Manber. Finding similar files in a large file system. In *USENIX Winter 1994 Technical Conference*, pages 1–10, 1994.

[23] Saul Schleimer, Daniel Shawcross Wilkerson, and Alexander Aiken. Winnowing: Local algorithms for document fingerprinting. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, pages 76–85, 2003.

[24] Ossama Abdel Hamid, Behshad Behzadi, Stefan Christoph, and Monika Rauch Henzinger. Detecting the origin of text segments efficiently. In *Proceedings of the 18th International Conference on World Wide Web*, pages 61–70, 2009.

[25] Sergey Brin, James Davis, and Hector Garcia-Molina. Copy detection mechanisms for digital documents. In Michael J. Carey and Donovan A. Schneider, editors, *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, pages 398–409. ACM Press, 1995.

[26] Jangwon Seo and W. Bruce Croft. Local text reuse detection. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 571–578, 2008.

[27] Jong Wook Kim, K. Selçuk Candan, and Jun'ichi Tatemura. Efficient overlap and content reuse detection in blogs and online news articles. In *Proceedings of the 18th International Conference on World Wide Web, WWW 2009, Madrid, Spain, April 20-24, 2009*, pages 81–90, 2009.

[28] A. Broder. On the resemblance and containment of documents. In *Proceedings of the Compression and Complexity of Sequences 1997*, pages 21–29. IEEE Computer Society, 1997.

[29] Yaniv Bernstein and Justin Zobel. Accurate discovery of co-derivative documents via duplicate text detection. *Inf. Syst.*, 31(7):595–609, 2006.

[30] Moses Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing*, pages 380–388, 2002.

[31] Abdur Chowdhury, Ophir Frieder, David A. Grossman, and M. Catherine McCabe. Collection statistics for fast duplicate document detection. *ACM Trans. Inf. Syst.*, 20(2):171–191, 2002.

[32] Haruo Yokota, Takashi Kobayashi, Hiroaki Okamoto, and Wataru Nakano. Unified contents retrieval from an academic repository. In *Proceedings of the 2006 International Symposium on Large-scale Knowledge Resources*, pages 41–46, 2006.

[33] Hiroaki Okamoto, Takashi Kobayashi, and Haruo Yokota. Presentation retrieval method considering the scope of targets and outputs. In *Proceedings of the 2005 International Workshop on Challenges in Web Information Retrieval and Integration*, pages 46–53, 2005.

[34] T. Kobayashi, W. Nakano, H. Yokota, K. Shinoda, and S. Furui. Presentation scene retrieval exploiting features in videos including pointing and speech information. In *Proceedings of the 2007 International Symposium on Large-scale Knowledge Resources*, pages 95–100, 2007.

[35] Daisuke Kitayama, Akiko Otani, and Kazutoshi Sumiya. A scene extracting method based on structural and semantic analysis of presentation content archives. In *C5*, pages 128–135, 2009.

[36] Hieu Hanh Le, Thitiporn Lertrusdachakul, Tetsutaro Watanabe, and Haruo Yokota. Automatic digest generation by extracting important scenes from the content of presentations. In *19th International Workshop on Database and Expert Systems Applications (DEXA 2008), 1-5 September 2008, Turin, Italy*, pages 590–594. IEEE Computer Society, 2008.

[37] Yuanyuan Wang, Daisuke Kitayama, Ryong Lee, and Kazutoshi Sumiya. Automatic generation of learning channels by using semantic relations among lecture slides and recorded videos for self-learning systems. In *ISM 2009, 11th IEEE International Symposium on Multimedia*, pages 275–280. IEEE Computer Society, 2009.

[38] Yanyan Lan, Tie-Yan Liu, Zhiming Ma, and Hang Li. Generalization analysis of listwise learning-to-rank algorithms. In Andrea Pohoreckyj Danyluk, Lon Bottou, and Michael L. Littman, editors, *ICML*, volume 382, page 73. ACM, 2009.

[39] Katsumi Tanaka and Masatoshi Yoshikawa. Towards abstracting complex database objects: Generalization, reduction and unification of set-type objects (extended abstract). In *ICDT*, pages 252–266, 1988.

[40] J. W. Hunt and M. D. Mcilroy. An algorithm for differential file comparison. Technical Report 41, Bell Laboratories Computing Science, July 1976.

[41] Daniel S. Hirschberg. A linear space algorithm for computing maximal common subsequences. *Commun. ACM*, 18(6):341–343, June 1975.

[42] Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

[43] A. Parker and J.O. Hamblen. Computer algorithms for plagiarism detection. *Education, IEEE Transactions on*, 32(2):94–99, May 1989.

[44] Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, 1970.

[45] Stephen G. Eick, Joseph L. Steffen, and Eric E. Sumner Jr. Seesoft – a tool for visualizing line oriented software statistics. *IEEE Transactions on Software Engineering*, 18(11):957–968, 1992.

[46] S. Eick. Graphically displaying text. *Journal of Computational and Graphical Statistics*, 3(2):127–142, June 1994.

[47] Fernanda B. Viégas, Martin Wattenberg, and Kushal Dave. Studying cooperation and conflict between authors with visualizations. In *CHI*, pages 575–582, 2004.

[48] Andrea Kohlhase and Michael Kohlhase. Cpoint: Dissolving the author's dilemma. In Andrea Asperti, Grzegorz Bancerek, and Andrzej Trybulec, editors, *Proceedings of the 3rd International Conference on Mathematical Knowledge Management*, pages 175–189. Springer, 2004.

[49] M. Lansdale. The psychology of personal information management. *Applied Ergonomics*, 19(1):55–66, March 1988.

[50] Alvaro E. Monge. Matching algorithms within a duplicate detection system. *IEEE Data Eng. Bull.*, 23(4):14–20, 2000.

[51] Hui Yang and Jamie Callan. Near-duplicate detection for erulemaking. In Lois M. L. Delcambre and Genevieve Giuliano, editors, *Proceedings of the 2005 National Conference on Digital Government Research, DG.O 2005*, pages 78–86. Digital Government Research Center, 2005.

[52] Xin Chen, Brent Francia, Ming Li, Brian McKinnon, and Amit Seker. Shared information and program plagiarism detection. *IEEE Transactions on Information Theory*, 50(7):1545–1551, 2004.

[53] Rudi Cilibrasi and Paul M. B. Vitányi. Clustering by compression. *IEEE Transactions on Information Theory*, 51(4):1523–1545, 2005.

[54] Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Similarity search in high dimensions via hashing. In *VLDB*, pages 518–529, 1999.

[55] Giuseppe Carenini, Raymond T. Ng, and Xiaodong Zhou. Summarizing email conversations with clue words. In *Proceedings of the 16th International Conference on World Wide Web*, pages 91–100. ACM, 2007.

[56] Dingding Wang, Tao Li, Shenghuo Zhu, and Chris H. Q. Ding. Multi-document summarization via sentence-level semantic analysis and symmetric matrix factorization. In *SIGIR*, pages 307–314, 2008.

[57] Soichi Murai and Taketoshi Ushiama. Review-based recommendation of attractive sentences in a novel for effective browsing. *International Journal of Knowledge and Web Intelligence*, 3(1):58–69, July 2012.

[58] Mark Dredze, Hanna M. Wallach, Danny Puller, and Fernando Pereira. Generating summary keywords for emails using topics. In *Proceedings of the 2008 International Conference on Intelligent User Interfaces*, pages 199–206, 2008.

[59] Andrew Mccallum and Xuerui Wang. Topic and role discovery in social networks. In *Proceeding of the 2005 International Joint Conferences on Artificial Intelligence*, pages 786–791, 2005.

[60] Joseph Kaye, Anita Lillie, Deepak Jagdish, James Walkup, Rita Parada, and Koichi Mori. Nokia internet pulse: a long term deployment and iteration of a twitter visualization. In *CHI Extended Abstracts*, pages 829–844, 2012.

[61] Shixia Liu, Michelle X. Zhou, Shimei Pan, Yangqiu Song, Weihong Qian, Weijia Cai, and Xiaoxiao Lian. TIARA: interactive, topic-based visual text summarization and analysis. *ACM Transactions on Intelligent Systems and Technology*, 3(2):25, 2012.

[62] Adam Perer and Marc A. Smith. Contrasting portraits of email practices: visual approaches to reflection and analysis. In *Proceedings of the working conference on Advanced visual interfaces 2006*, pages 389–395, 2006.

[63] Fernanda B. Viégas, Scott A. Golder, and Judith S. Donath. Visualizing email content: portraying relationships from conversational histories. In *Proceedings of the 2006 Conference on Human Factors in Computing Systems*, pages 979–988, 2006.

[64] Hendrik Strobelt, Daniela Oelke, Christian Rohrdantz, Andreas Stoffel, Daniel A. Keim, and Oliver Deussen. Document cards: A top trumps visualization for documents. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1145–1152, 2009.

[65] Yanhua Chen, Lijun Wang, Ming Dong, and Jing Hua. Exemplar-based visualization of large document corpus. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1161–1168, 2009.

[66] Tomoharu Iwata, Takeshi Yamada, and Naonori Ueda. Probabilistic latent semantic visualization: topic model for visualizing documents. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 363–371, 2008.

[67] . URL `http://www.textarc.org/`.

[68] Martin Wattenberg and Fernanda B. Viégas. The word tree, an interactive visual concordance. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1221–1228, 2008.

[69] Frank van Ham, Martin Wattenberg, and Fernanda B. Viégas. Mapping text with phrase nets. *IEEE Trans. Vis. Comput. Graph.*, 15(6):1169–1176, 2009.

[70] Anthony Don, Elena Zheleva, Machon Gregory, Sureyya Tarkan, Loretta Auvil, Tanya Clement, Ben Shneiderman, and Catherine Plaisant. Discovering interesting usage patterns in text collections: integrating text mining with visualization. In *CIKM*, pages 213–222, 2007.

[71] Tomohide Shibata and Sadao Kurohashi. Automatic slide generation based on discourse structure analysis. In *Proceedings of the 2005 International Joint Conference on Natural Language Processing*, pages 754–766, 2005.

[72] Harish Mathivanan, Madan Jayaprakasam, K. Gokul Prasad, and T. V. Geetha. Document summarization and information extraction for generation of presentation slides. In *ARTCom 2009, International Conference on Advances in Recent Technologies in Communication and Computing*, pages 126–128. IEEE Computer Society, 2009.

[73] Brandon Beamer and Roxana Girju. Investigating automatic alignment methods for slide generation from academic papers. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 111–119, Boulder, Colorado, June 2009. Association for Computational Linguistics.

[74] Y. Yoshiaki, T. Masashi, and N. Katsumi. A support system for making presentation slides. *Transactions of the Japanese Society for Artificial Intelligence*, 18:212–220, 2003.

[75] Min-Yen Kan. Slideseer: A digital library of aligned document and presentation pairs. In *Proceedings of the 7th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 81–90. ACM, 2007.

[76] T. Hayama, H. Nanba, and S. Kunifuji. Alignment between a technical paper and presentation sheets using a hidden markov model. In *Active Media Technology, 2005. (AMT 2005). Proceedings of the 2005 International Conference on*, pages 102–106, May 2005.

[77] Cheng-Yao Chen. An integrated system supporting effective indexing, browsing and retrieval of Microsoft Powerpoint presentation database. In *ICDE Workshops*, page 16, 2006.

[78] Utiyama Masao and Hasida Koiti. Automatic slide presentation from semantically annotated documents. In *Proceedings of the Workshop on Coreference and Its Applications*, pages 25–30. Association for Computational Linguistics, 1999.

[79] M. Miyamoto, H. Sakai, and S. Masuyama. Research on automatic generation of presentation slides from a LaTeX manuscript of a paper. *Journal of Japan Society for Fuzzy Theory and Intelligent Informatics*, 18(5):752–760, 2006.

[80] Ken Perlin and David Fox. Pad: an alternative approach to the computer interface. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1993*, pages 57–64. ACM, 1993.

[81] Douglas E Zongker and David H Salesin. On creating animated presentations. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 298–308. Eurographics Association, 2003.

[82] P. T. Zellweger. Scripted documents: A hypermedia path mechanism. In *Proceedings of the Second Annual ACM Conference on Hypertext*, pages 1–14. ACM, 1989. ISBN 0-89791-339-6.

[83] Les Nelson, Satoshi Ichimura, Elin Rønby Pedersen, and Lia Adams. Palette: A paper interface for giving presentations. In *Proceeding of the CHI '99 Conference on Human Factors in Computing Systems*, pages 354–361, 1999.

[84] Lance Good and Benjamin B. Bederson. Zoomable user interfaces as a medium for slide show presentations. *Information Visualization*, 1(1):35–49, 2002.

[85] Tomer Moscovich, Karin Scholz, John F. Hughes, and David H. Salesin. Customizable presentations. Technical report, Brown University, 2004.

[86] Yuanyuan Wang and Kazutoshi Sumiya. Skeleton generation for presentation slides based on expression styles. In *Intelligent Interactive Multimedia: Systems and Services*, pages 551–560. Springer, 2012.

[87] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze. *Introduction to information retrieval*. Cambridge University Press, 2008.

[88] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *9th IEEE International Conference on Computer Vision (ICCV 2003)*, pages 1470–1477, 2003.

[89] Krystian Mikolajczyk and Cordelia Schmid. An affine invariant interest point detector. In *Proceedings of the 7th European Conference on Computer Vision, Part I*, pages 128–142, 2002.

[90] Krystian Mikolajczyk, Tinne Tuytelaars, Cordelia Schmid, Andrew Zisserman, Jiri Matas, Frederik Schaffalitzky, Timor Kadir, and Luc J. Van Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1-2):43–72, 2005.

[91] David Nistér and Henrik Stewénius. Scalable recognition with a vocabulary tree. In *2006 IEEE Computer Society Conference on Computer Vision and*

Pattern Recognition (CVPR 2006), pages 2161–2168. IEEE Computer Society, 2006.

[92] Ondrej Chum, James Philbin, and Andrew Zisserman. Near duplicate image detection: min-hash and tf-idf weighting. In Mark Everingham, Chris J. Needham, and Roberto Fraile, editors, *Proceedings of the 2008 British Machine Vision Conference*, pages 1–10. British Machine Vision Association, 2008.

[93] Surajit Chaudhuri, Venkatesh Ganti, and Raghav Kaushik. A primitive operator for similarity joins in data cleaning. In Ling Liu, Andreas Reuter, Kyu-Young Whang, and Jianjun Zhang, editors, *Proceedings of the 22nd International Conference on Data Engineering, ICDE 2006*, page 5. IEEE Computer Society, 2006.

[94] . URL `http://www.vldb.org/2011/`.

[95] . URL `http://www.vldb2012.org/`.

[96] Andrei Z. Broder. Identifying and filtering near-duplicate documents. In *Combinatorial Pattern Matching, 11th Annual Symposium, CPM 2000*, pages 1–10, 2000.

[97] Chuan Xiao, Wei Wang, Xuemin Lin, Jeffrey Xu Yu, and Guoren Wang. Efficient similarity joins for near-duplicate detection. *ACM Trans. Database Syst.*, 36(3):15, 2011.

[98] . URL `http://d3js.org/`.

[99] Shiraki N. Kurohashi, S. and M. Nagao. A method for detecting important descriptions of a word based on its density distribution in text. 38(4):845–854, 1997.

[100] Watanabe Toyohide, Ishiguro Yusuke, and Koichi Hanaue. Automatic composition of presentation slides, based on semantic relationships among slide components. In *Intelligent Interactive Multimedia Systems and Services*, volume 11 of *Smart Innovation, Systems and Technologies*, pages 261–270. 2011.

[101] Hanaue Koichi and Watanabe Toyohide. Supporting design and composition of presentation document based on presentation scenario. In Gloria Phillips-Wren, LakhmiC. Jain, Kazumi Nakamatsu, and RobertJ. Howlett, editors, *Advances in Intelligent Decision Technologies*, volume 4 of *Smart Innovation, Systems and Technologies*, pages 465–473. Springer Berlin Heidelberg, 2010.

List of Publications

Journal Papers

- Jie ZHANG, Chuan XIAO, Toyohide WATANABE and Yoshiharu ISHIKAWA, "Content-Based Element Search for Presentation Slide Reuse", *IEICE Trans. Information and Systems*, Vol. E97-D, No. 10, pp. 2685–2696, 2014.

- Jie ZHANG, Chuan XIAO, Sheng HU, Toyohide WATANABE and Yoshiharu ISHIKAWA, "Managing Presentation Slides with Reused Elements", *International Journal of Information and Education Technology*, Vol.6, No.3, pp.170–177, March 2016.(Accepted)

- Jie ZHANG, Chuan XIAO, Toyohide WATANABE and Yoshiharu ISHIKAWA, "Detecting Reused Elements in Presentation Slides", *WIT Transactions on Information and Communication Technologies*, 2015.(Accepted)

International Conference/Workshop Papers

- Jie ZHANG, Chuan XIAO, Sheng HU, Toyohide WATANABE and Yoshiharu ISHIKAWA, "Managing Presentation Slides with Reused Elements", *International Conference on Computer Technology and Development, Nov.7-9, 2014, Hongkong.*

- Jie ZHANG, Chuan XIAO, Toyohide WATANABE and Yoshiharu ISHIKAWA, "Detecting Reused Elements in Presentation Slides", *International Conference on Computer Engineering, Nov. 9-10, 2014, Shenzhen, China.*

Domestic Conference Papers

- Jie ZHANG, Chuan XIAO, Toyohide WATANABE and Yoshiharu ISHIKAWA, "A Slide Element Retrieval Method for Presentation Reuse", *IEICE Data Engineering*, Vol.114, No.204, pp.69-74, Sept.2014.
- Jie ZHANG, Toyohide WATANABE, "Extraction of Events Knowledge from News Articles", *Educational Engineering, the Society of Instrument and Control Engineers*, Vol.35, pp.43-46, Sept.2012.
- Jie ZHANG, Wei FAN and Toyohide WATANABE, "Finding Frequent Patterns from Data Streams by Time Stamps", *2010th Tokai-Section Joint Conference on Electrical and Related Engineering*, F1-4, Aug.2010.