published as: Martin Malchow, Matthias Bauer, Christoph Meinel: Enhance Lecture Archive Search with OCR Slide Detection and In-Memory Database Technology, In Proc. of IEEE 18th International Conference on Computational Science and Engineering (CSE2015), 21-23 October, 2015, Porto (Portugal)

2015 IEEE 18th International Conference on Computational Science and Engineering

Enhance Lecture Archive Search with OCR Slide Detection and In-Memory Database Technology

Martin Malchow, Matthias Bauer, Christoph Meinel
Hasso Plattner Institute (HPI)
University of Potsdam
Potsdam, Germany
Email: {martin.malchow, matthias.bauer, christoph.meinel}@hpi.de

***Abstract*—On the Web there are a lot of frequently used video lecture archives which have grown up fast during the last couple of years. This fact led to a lot of lecture recordings which include knowledge for a variety of subjects. The typical way of searching these videos is by title and description. Unfortunately, not all important keywords and facts are mentioned in the title or description if they are available. Furthermore, there is no possibility to analyze how important those detected keywords are for the whole video. Another lecture archive specific virtue is that every regular university lecture is repeated yearly. Normally this will lead to duplicate lecture recordings. In search results doubling is disturbing for students when they want to watch the most recent lectures from the search result. This paper deals with the idea to resolve these problems by analyzing the recorded lecture slides with Optical Character Recognition (OCR). In addition to the name and description the OCR data will be used for a full text analysis to create an index for the lecture archive search. Furthermore, a fuzzy search is introduced. This will solve the issue of misspelled search requests and OCR detection defects. Additionally, this paper deals with the performance issues of a full text search with an in-memory database, issues in OCR detection, handling duplicate recordings of lectures repeated every year. Finally, an evaluation of the search performance in comparison with other database ideas besides the in-memory database is performed. Additionally, a user acceptability survey for the search results to increase the learning experience on lecture archives was performed. As a result, this paper shows how to handle the big amount of OCR data for a full text live search performed on an in-memory database in reasonable time. During this search a fuzzy search is performed additionally to resolve spelling mistakes and OCR detection problems. In conclusion this paper shows a solution for an enhanced video lecture archive search that supports students in online research processes and enhances their learning experience.**

I. Introduction

The first lecture archives have established in the beginning of 2000 like the Berkeley Internet Broadcasting System (BIBS) [1] at the Berkeley Multimedia Research Center. Which is now known as webcast.berkley[1]. Another lecture archive approach has been started in 2002 at the University of Trier with the tele-TASK [2] system. Over the years the tele-TASK web portal has grown and currently provides over 5500 lectures which should be searchable to provide users easy access to the information they are searching for. Due to, lectures nature they will be rerecorded every year. This behavior has to be considered while processing the search result. Furthermore, lectures contain a lot of information on the slides. This information have to be considered during the search to create a more accurate result than just by analyzing names and descriptions of lectures. In order to realize these ideas the slides have to be analyzed to extract this additional information. Then, all the text has to be indexed with an appropriate full text index. As users are impatient when a device reacts slowly [3] the search speed has to be fast. This is one of the biggest challenges as a lot of data has to be processed with the content of all lecture slides from all lectures. In order to handle this speed problem an in-memory database will be used and special indexes will be created to reach the required speed up.

This paper is structured into 4 parts. The parts of the paper are “Related work”, “Approach”, “Evaluation” and “Results and Future Work”. In the following Section II, related work in this research area is described. Mainly research in the area of OCR search in lecture archives, OCR detection and difficulties in lecture duplicated are considered.

This section is followed by the approach of OCR slide search using an in-memory database for processing the data. As in-memory database the HANA is used leading to HANA specific SQL queries for processing the search. These queries lead to a search offering different features. One feature is fuzzy search which leads to reasonable result even when the user misspelled a word or the OCR detection recognized a word not completely accurately. Another feature is to use a full text search over multiple fields in the database to find the most reasonable lectures by title, description and OCR data. Beside this approach a currently used search algorithm which only analyzes lecture names and descriptions will be presented and the architecture of the lecture archive search with the in-memory database will be shown.

In the evaluation in Section IV it will be discussed if the speed up of an in-memory database used in this approach is powerful enough and if an improvement to the state of the art search server Elasticsearch is possible. Furthermore, this section includes a user study to compare if the search results from the new search approach are more appropriate for users of the lecture archive platform than before. Therefore survey participants were asked to decide for five different search phrases which of both results is more appropriate.

Finally, the paper is concluded with the approach result and ideas for future work in this area. Considered areas for future work are the use of Automatic Speech Recognition (ASR), semantic approaches to search similar and corresponding terms and different optimization ideas to speed up the search.

[1] http://webcast.berkeley.edu/

978-1-4673-8297-7/15 $31.00 © 2015 IEEE
DOI 10.1109/CSE.2015.19

II. Related Work

During the last years, there have been several companies offering commercial lecture archives like Panopto[2], Cisco Media Experience Engines[3] or Echo360[4]. Some of these services already offer OCR analysis of recorded lectures to search keywords. Examples for this OCR keyword search are Cisco Pulse [4] and Panopto Smart Search [5]. These products analyze the slides and detect keywords. Nevertheless, only the keywords can be searched. The companies also describe it as keyword search or search for a specific word in a lecture. The idea of this paper is to go a step further and create a "search engine like"-search in the OCR data of all recorded lectures in the archive.

Important work for the OCR lecture search is the algorithm for OCR detection in lecture slides [6]. This algorithm made video data accessible. But there are still some errors because bullet points can not be processed easily. This is caused by the missing sentence structure for credibility. Nevertheless, this algorithm has great results and fault detection can be handled by using a fuzzy search algorithm.

Another important research area used in this approach is fuzzy search [7]. Using fuzzy search enables users to find information for certain topics even when a word is slightly misspelled. This will avoid frustrations and awkward search of spelling errors. Furthermore, while searching OCR data fuzzy search offers additional functionality. Due to, wrongly detected words by the automatic OCR algorithm searching the OCR data is a complex task. There are a many unpredictable wrong detections possible for every word. The fuzzy search solves the problem satisfactorily by finding typographically close terms to the search term. This leads to considerable results even when users and/or OCR detection do not use proper terms.

An important research topic for this approach is the search engine in general. An established open source search engine based on Apache Lucene Core[5] is Elasticsearch which also offers, besides numerous other features, a full text search in the indexed data [8]. The performance of Elasticsearch in context of the search in OCR slide data to find lectures is analyzed more detailed in the following sections of the paper. An existing approach using Apache Lucene for OCR lecture search is TalkMiner [9].

In video archives there is a problem with certain videos that seem to have duplicates in the search results [10]. But these videos are often not actual duplicates. They are just lectures repeated every year. Therefore, algorithms for filtering duplicate videos automatically on the server cannot be applied in a lecture archive environment. Another approach has to be developed so that users get important lectures without seeing only results of one lecture over several years of recording.

III. Approach

The current lecture archive search is based on a weighted search of the name and description of lectures, which leads to reasonable search results with one-term searches. Nevertheless, it has some drawbacks when it comes to word group search. The basic algorithm for the weighted approach with a scoring model operates like described in the following sentences. First, an additional index table is created in a database. This index contains all terms and a field in which lecture this term occurs and at which position in the name or description of the lecture. This index reduces the computational effort per search request but it has to be updated when a new lecture is added. Now every found lecture gets points for matching results like shown in Table 1. For elimination of the doubled results from lectures of different semesters this algorithm divided the points by the age of the lecture. So, older results will be displayed later in the ranking if there are lectures which are close to this lecture. Nevertheless, this approach will show doubled lectures from different semesters in various circumstances and removes relevant lectures from the top search result which were archived in the past and not recorded several times just because they are older.

TABLE I. Scoring System for Weighted Search

Lecture Name	
Exact name match:	50 points
Name starts with:	30 points
Contains name match:	20 points
Lecture Description	
Exact description match:	20 points
Description starts with:	10 points
Contains description match:	5 points

A. System Architecture

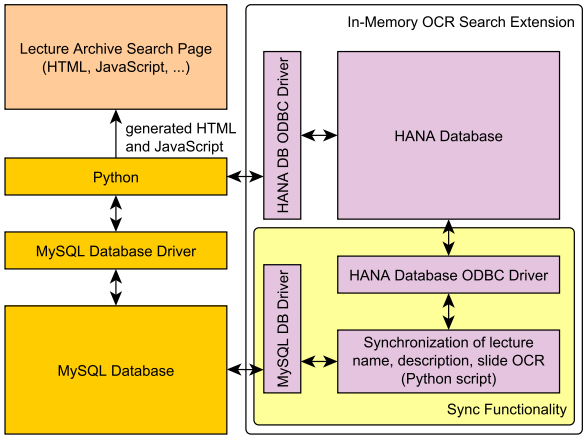

Fig. 1. Architecture of the lecture archive with extended HANA Database Functionality

The current system architecture of the lecture search is shown on the left side of Figure 1. The core of the current weighted search is a Python script loading lecture data from the MySQL database, processing the data and displaying the results on the lecture archive search page. This MySQL database is used because of legacy reasons. Due to the use of the Python framework Django[6] the Django database model is used to execute queries. This model currently does not support the HANA database. To avoid changing the complete source

[2]http://www.panopto.com/
[3]http://www.cisco.com/
[4]http://echo360.com/
[5]https://lucene.apache.org/core/
[6]https://www.djangoproject.com/

code of the lecture archive the MySQL database is still used and only search-relevant data will be synchronized with the in-memory database HANA.

On the right side of Figure 1 the extension for the OCR search with the in-memory database HANA [11] is shown. To activate and insert data into the in-memory database an additional Python script for synchronizing the lecture name, description and OCR data was written. This script uses a MySQL Driver and a HANA ODBC Driver to communicate with both databases. This synchronization is shown in the bright yellow box in the Figure 1.

When a HANA search is going to be performed the Python script is accessing the synchronized HANA Database by the HANA ODBC Driver. The Python script handles the lecture IDs from the result set and gather additional lecture data from the MySQL database. This merged result will be displayed on the lecture archive search page.

B. Preparing OCR Data

First, all lectures are going to be analyzed so that the OCR data can be created with the algorithm described in [6]. This leads to recorded OCR data which consist of several detected lines per slide for all slides of a lecture. In the following step the OCR data out of one slide line is going to be combined with an additional space. This additional space is necessary to avoid combination of words to one word. This procedure is done for every slide by simply adding the lines in the right order. As a result a full text for the lecture slide is created and made machine-processable. Now it is possible to do a full text search for most suitable slides. As the main focus of the paper is to find lectures in a lecture archive efficiently another aggregation step is necessary. Now the complete texts of every slide have to be combined again with an additional space. This results in a long OCR detected text for a lecture containing combined bullet points.

C. Search Matching Results in the Database

In the first approach a simple SQL query is used in a MySQL database to find suitable lectures for an entered word or word group in the OCR result. This SQL query finds lectures which contain the word or word group in the OCR result and counts the number of occurrences and set it into relation to the number of words. The result is going to show the importance of the search results. Nevertheless, this approach has a couple of drawbacks. Considering that the 5500 lectures in the concerned archive will slow down this way of search significantly to nearly 80 seconds per request like it is shown in the measurement in Section IV-A. This is unacceptable when taking into mind that users expect an instant reaction to their search request [3].

Furthermore, this approach can only find word groups in the OCR data if they are found in the submitted order. If a user searches for “TCP UDP” an occurrence of the string “advantages of TCP in comparison to UDP” can not be detected by this approach because linked words can only be found if they occur with the same linking again. This would lead to search results with limited coverage of the available data.

Another drawback is that the OCR analysis is not completely accurate. This leads to the situation that words on the slides which were not recognized correctly by the OCR detection are not found and a lecture containing interesting information of a topic will be missed in a search.

The final drawback of this approach is that there is no handling of doubled lectures from different semesters. The resulting order is random which leads to very old lectures possibly shown first instead of the newest lecture, which would lead to unsatisfied and outdated search results for the users.

This first approach for the full text OCR search seems to be inefficient and not useful for the search. Nevertheless, it shows which issues have to be solved to include OCR data into the search result. These approaches will be described in the following sections.

D. Handle Speed Issues of OCR Search and Word Linking Problem

To handle the full text OCR search in a reasonable time for users an in-memory database is used. In our case the SAP HANA [11] which already offers text processing. To work efficiently in an in-memory database a table is stored as column table and not as it is known from other databases as row table. Currently only search relevant data are stored in the in-memory database. These data consist of the lecture IDs from the currently used MySQL database, lecture names, the lecture descriptions and lecture OCR data. Currently a synchronization is necessary to keep the data from the MySQL database up to date with the data in the in-memory database. Furthermore, HANA can automatically create an index for text processing to decrease the search time. To create a column table with the necessary fields and the creation of the index the SQL statement in Listing 1 have to be executed.

Listing 1. HANA SQL statement to create a column table for lecture search with index creation for full text search

```
CREATE COLUMN TABLE "LECTURES" (
  "LECTURE_ID" INTEGER CS_INT NOT NULL ,
  "NAME" NVARCHAR(500),
  "DESCRIPTION" NCLOB,
  "OCR_TEXT" NCLOB,
  PRIMARY KEY ("LECTURE_ID"));

CREATE FULLTEXT INDEX "NAMEIDX" ON
"LECTURES" ("NAME") SYNC;
CREATE FULLTEXT INDEX "DESCRIPTIONIDX" ON
"LECTURES" ("DESCRIPTION") SYNC;
CREATE FULLTEXT INDEX "OCRIDX" ON
"LECTURES" ("OCR_TEXT") SYNC;
```

After the data were filled to the database a full text search for OCR data can be performed with the statement in Listing 2. The engine searches on the OCR table for exact matches of the “SearchTerm” and automatically creates a score which is going to be ordered by importance.

Listing 2. HANA SQL statement to do an exact full text search with scoring the result

```
SELECT "LECTURE_ID", SCORE() AS SCORE
```

```
FROM "LECTURES" WHERE
CONTAINS ("OCR_TEXT", 'searchterm')
ORDER BY SCORE DESC
```

The SQL keyword “CONTAINS” additionally offers a phrase search. This functionality solves the problem of the linked word search. Instead of searching one term a word group or phrase can be searched similar to popular search engines. Additionally it is possible to operate with double quotes, e.g. "search me" to search for a phrase which is matching without any terms in between. Furthermore, using a “-” can exclude results which contains a special word. As example, searching for “search -term” will show all lectures containing “search” and do not mention “term”.

E. Handle OCR Defects with Fuzzy Search

As discussed in Section III-C the OCR search is not completely precise. Moreover, users do not always search terms without any spelling mistakes. Nevertheless, the lecture search should be able to show reasonable results even when there are spelling mistakes in the OCR or user search data. The so called “Fuzzy Search” solves the “String-to-String Correction Problem”[12] based on the Levenshtein distance [13] which calculate how far strings from each other by analyzing the computational effort to change a string into another. In the in-memory database HANA a fuzzy search is already included. To find results with the fuzzy search an SQL query like it is shown in Listing 3 has to be executed with the corresponding search term to find a lecture. The fuzzy attribute “0.8” visible in the listing describes up to which fuzzy score results are displayed. Smaller numbers will find more blurry results and bigger numbers more exact results.

Listing 3. HANA SQL statement to do an fuzzy full text search with scoring the result

```
SELECT "LECTURE_ID", SCORE() AS SCORE
FROM "LECTURES" WHERE CONTAINS
("OCR_TEXT", 'searchterm', FUZZY(0.8))
ORDER BY SCORE DESC
```

The fuzzy search is able to find good results within the OCR search. Nevertheless, currently the lecture search is still to slow. This is because the index is just created for full text search. As a solution HANA offers the search option for creating an index for fuzzy search to speed up fuzzy full text search. To activate it the SQL statement in Listing 4 has to be executed to create the search index optimized for fuzzy search. Using this special fuzzy search index speeds up the lecture search with OCR data dramatically to 220ms per search. The search speed is going to be discussed in detail in Section IV-A.

Listing 4. HANA SQL statement creation of a full text fuzzy search index

```
CREATE FULLTEXT INDEX "NAMEIDX" ON
"LECTURES" ("NAME") SYNC
FUZZY SEARCH INDEX ON;
CREATE FULLTEXT INDEX "DESCRIPTIONIDX" ON
"LECTURES" ("DESCRIPTION") SYNC
FUZZY SEARCH INDEX ON;
CREATE FULLTEXT INDEX "OCRIDX" ON
"LECTURES" ("OCR_TEXT") SYNC
FUZZY SEARCH INDEX ON;
```

F. Freestyle Search to find multiple lecture texts

This section describes how to merge the search results for lecture name, lecture description and OCR data for the lecture search. The SQL statement for this select operation is shown in Listing 5. This search is known as freestyle search and uses “term frequency - inverse document frequency” (tf-idf) to analyze the importance of a document with a set of data [14]. In case of the lecture search with the in-memory database the score of a lecture depending on the name, description and slide OCR data is created.

Listing 5. HANA SQL statement to perform a search over several lecture information fields with fuzzy search

```
SELECT "LECTURE_ID", SCORE() AS SCORE
FROM "LECTURES" WHERE CONTAINS
(("NAME", "DESCRIPTION", "OCR_TEXT"),
'searchterm', FUZZY(0.8))
ORDER BY SCORE DESC
```

G. Handling Doubled Lectures from Different Semesters

A final step for selecting lectures out of the in-memory database is to filter double lectures from different years. Usually, lectures are almost the same every year with minor improvements in fact of new research results. Based this assumption the newest lecture with the same title and lecturer should be shown in the search result only. Furthermore, lectures which were just recorded once should not be unimportant in the search order like it happens with the old search approach described in the beginning of Section III. For realization there is the presumption that all lectures have a unique name and the same name when they are repeated in other semesters. Hence, we can find same lectures by name comparison. Additionally, we can find out easily the newest lecture because the lecture IDs are count up by auto increment. So, the newest lecture of same lectures will be detected by the maximum lecture ID. The adapted SQL statement to realize this behavior is shown in Listing 6. The basic SQL statement is already known from Section III-F. An additional `SELECT` is added to find all double lectures by name and select the lecture with the highest ID. At the end the search result will be joined with the results of the newest lecture search of this series by an `INNER JOIN`. This increase the search time to an justifiable value of approximately 300ms.

Listing 6. HANA SQL statement to perform a lecture search selecting the newest lecture

```
SELECT SCORE() AS SCORE,
T1."NAME", T1."LECTURE_ID"

FROM "LECTURES" AS T1 INNER JOIN
 (SELECT "NAME", MAX("LECTURE_ID")
  AS "LECTURE_ID" FROM
    "LECTURES"
  GROUP BY ("NAME")
 ) AS T2

ON T1."LECTURE_ID" = T2."LECTURE_ID"
WHERE CONTAINS
((T1."NAME", "DESCRIPTION", "OCR_TEXT"),
'searchterm', FUZZY(0.8)) ORDER BY SCORE DESC
```

H. Combine In-Memory Database Results with the Existing Database

Finally, the lecture IDs found by the in-memory database have to be connected to the MySQL data to receive additional information like lecture URL, lecture duration, comments, e.g. Depending on the amount of lectures found the search time increases. This behavior is going to be discussed in the following experiment Section IV-A

IV. Evaluation

The quality of the described search approach is measured by time of the request and by fineness of the search result. The speed is compared with the OCR approach of a MySQL Database and the in-memory database in the following section IV-A. Additionally, the search result is valuated completely by 78 unique users in a survey comparing the results of the weighted search described in Section III and the in-memory database approach. The result is going to be discussed in Section IV-B.

A. Database Request Time

The request time is compared with a normal database request counting the number of occurrences of a word in the OCR data to decide the importance of the search term in different lectures and the in-memory database approach. The in-memory database approach is divided into subsets with and without taking care of repeated lecture appearance as well as with and without adding the time to select lecture information from the MySQL database. In Figure 2 only differences of the in-memory approach are perceptible even when the in-memory subsets are included. On the x-axis the number of search results are displayed which are limited by the SQL query and the y-axis shows the required execution time for this database request with these amount of results. It is ascertainable that the number of requests has no impact on the execution time. This behavior is caused by the search algorithm. To find the top results performs an overall text analyzes for all lectures and reducing this result set to a limited number does not interfering the overall execution time. Even when there are small timing differences which were caused by other processes and database caching the processing time of the text search is independent from the number of requested results.

Figure 2 shows clearly that a full text search with a standard MySQL database is not manageable. Therefore, the different steps of the in-memory database approach have been analyzed to verify if the timing results are reasonable for the user. In the following Figure 3 the in-memory and Elasticsearch timing results are visualized. As mentioned in the beginning the number of results has no influence on the execution time for a search query in traditional databases or in-memory databases. Nevertheless, it has an influence when taking into account that additional lecture data has to be collected from the database due to the mentioned legacy reasons. In case of this experiment after processing the SQL query to analyze the search result, MySQL queries have to be executed to gather lecture information needed for visualization on the lecture archive website. This additional gathering of data exceeds the execution time like it is shown in Figure 3. Additionally, Figure 3 shows the differences with and without removing

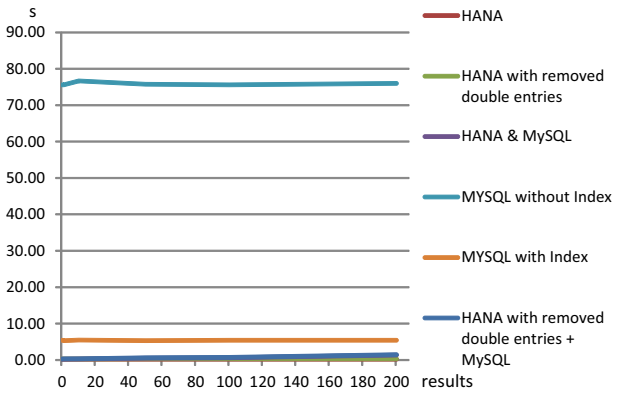

Fig. 2. Request time for MySQL content search compared with HANA in-memory index search

duplicate lectures. In case the difference is less than 100ms it is not necessary to spare this functionality in a productive system. Nevertheless, this result shows that a search should not show all matching lectures instantly because of the response time of more than a second. As a solution only the top 5 or top 10 lecture matches should be shown and if users want to see more results they interact with the website and get a visual feedback that the search for more lectures is executed with an animated progress image as example. In comparison to the in-memory database search the Elasticsearch execution time is related to the number of selected results. In the beginning the Elasticsearch engine is faster for up to 20 search results. If more search results are requested the HANA is obviously faster. Furthermore, the timing results for Elasticsearch filtering lectures from different years and getting additional information from the MySQL database are not measured due to the worse execution compared to the HANA. Executing these additional tasks will increase the search time significantly. Finally, also the search results of the Elasticsearch in the OCR data seem to be worse compared to the HANA result. Especially, Elasticsearch seems to search for keywords in the OCR data only without taking into account that words of a search term are more worth if they are found next to each other. In addition no search engine search behavior is implemented implicitly in Elasticsearch. As an example when a student wants to search for a term where another term is shall be excluded using a "-" is not processed in the expected way by Elasticsearch without additional implementation effort. Due to the mentioned reasons Elasticsearch is not considered in the following evaluation of this paper.

B. Fineness of Search Results

To analyze the quality of the in-memory search approach for lecture archives a survey was created where the weighted search described in Section III and the in-memory approach results are compared by users. The 5 search terms with the appropriate result are shown in the tables. 78 users answered all survey questions. Sometimes there are more then 78 evaluations for a question. The reason is that some users did not answer all 5 questions. The survey is structured in 5 questions. Every question shows 2 search results with 4 matches for a specified search term. Now the user has to decide which

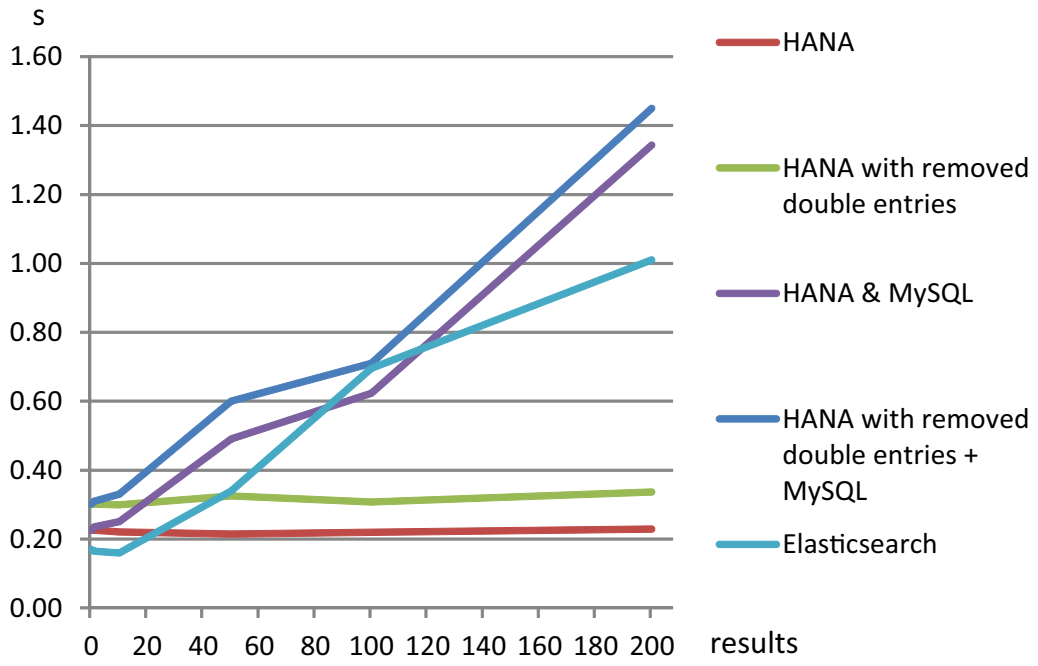

Fig. 3. Request time for HANA index search with and without sorting out double lectures from different years and MySQL search time for lecture information

out of both search result fits best by their personal feeling without knowledge about the used search algorithm. The result is shown in Figure 4. Furthermore, all 5 search results with lecture name and semester recording information are available in Tables 2 to 6.

Users answering question 1 were not sure if the new in-memory approach with OCR analysis is better or the old weighted search. Reasons for this undecided behavior of the user group could be the lecture name of the result visible in Table 2. All lecture names are "TCP / UDP" which seems to be a very good result at the weighted search. But, taking into account that these lectures are the same which is indicated by the same name, same lecturer and the semester when they were recorded. Viewing all this lectures would not teach students a variety of information. Typically only minor improvements are made in lectures over the years. Hence, the HANA search result is more appropriate in fact of the variety of found lectures. This overlaps with the survey result even when it is not so clear for this question.

In the following question 2 "Mobile IP" was searched with both search algorithms shown in Table 3. Users attending the survey decided clearly for the HANA result. Properly, in fact no title of the weighted search contains the words "Mobile IP" and in the HANA search nearly every match includes the search term.

"Linux Scheduling" was the search term in question 3. The results are shown in Table 4. Obvioulsy there seems to be no lecture describing Linux scheduling exclusively. Nevertheless, the weighted search is finding Scheduling lectures with Windows connection only. Whereas, the HANA search approach is

TABLE II. Q1: SEARCH TERM: UDP (TRANSLATED)

HANA Search	
Found Lecture Names	*Semester*
TCP / UDP	Summer 2014
Internetworking with TCP / UDP	Summer 2011
Internet Applications	Summer 2014
Weaknesses of Internet Protocols	Winter 2013

Weighted MySQL Search	
Found Lecture Names	*Semester*
TCP / UDP	Summer 2014
TCP / UDP	Summer 2013
TCP / UDP	Summer 2012
TCP / UDP	Summer 2010

TABLE III. Q2: SEARCH TERM: MOBILE IP (TRANSLATED)

HANA Search	
Found Lecture Names	*Semester*
Mobile IP	Summer 2009
Internetworking with IPV6 and Mobile IPV6	Summer 2012
Expanding the Telco Model to Support IP Smart Objects	Summer 2010
Taking Service Providers to IPv6: Mobile Networks	Summer 2010

Weighted MySQL Search	
Found Lecture Names	*Semester*
GDI - Definition - Styled Layer Descriptor	Summer 2013
The Principle of SOA	Summer 2013
Internetworking with IPV6	Summer 2013
IPv6 - Still in a fledgling stage or already at the top	Winter 2012

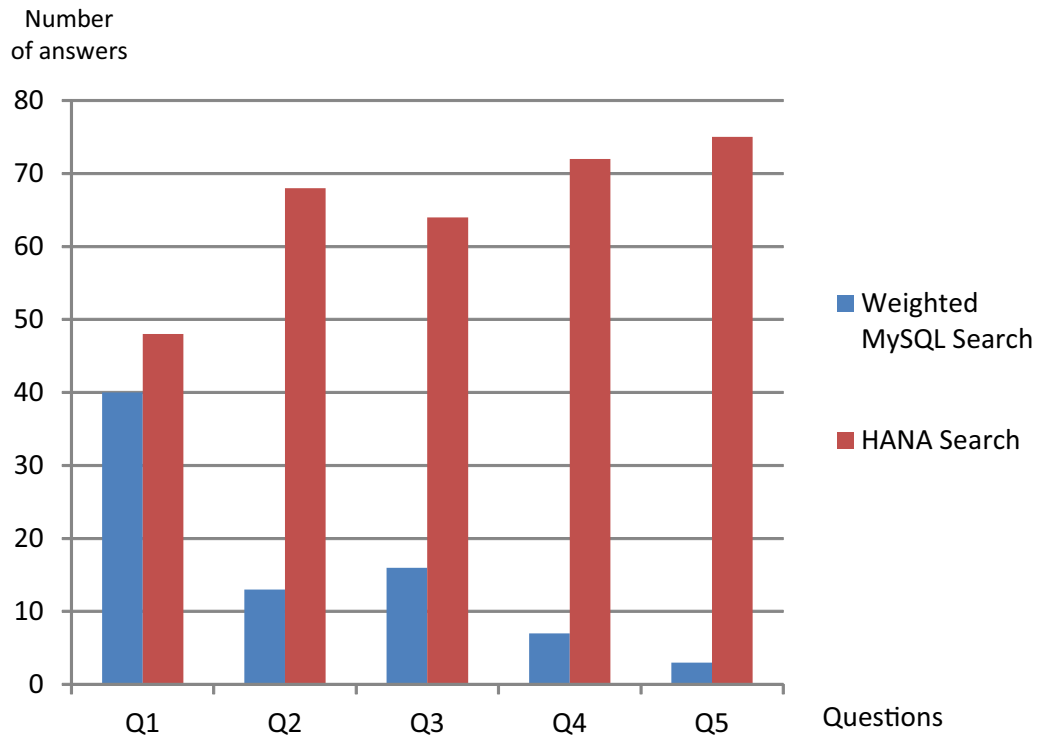

Fig. 4. Prefered search result of users. Red new HANA OCR approach against blue old MySQL weighted search approach

TABLE IV. Q3: SEARCH TERM: LINUX SCHEDULING (TRANSLATED)

HANA Search	
Found Lecture Names	*Semester*
Scheduling (3)	Winter 2012
Scheduling (2)	Winter 2012
Advanced Windows Thread Scheduling	Winter 2013
Scheduling Opportunities on Manycore Architectures	Winter 2010

Weighted MySQL Search	
Found Lecture Names	*Semester*
Windows Process and Threads & Windows Thread Scheduling	Winter 2014
Windows Thread Scheduling & Memory Management	Winter 2014
Windows Thread Scheduling	Winter 2013
Advanced Windows Thread Scheduling	Winter 2013

going to find general scheduling lectures which probably will describe Linux scheduling as well. That could be a reason most users decided for the HANA search approach at this question.

Like in the previous examples it was already visible searching phrases mostly does not lead to satisfiable results with the weighted search. This is similar for the search result of question 4 visible in Table 5. In the weighted search the lecture names do not mention any marketing aspect. In contrast, the HANA search has marketing relations in the lecture names of the search result. A similar behavior is also noticeable in the last question 5. The search results are shown in Table 6. Likely, these vague search results are the reason that users decide in question 4 and 5 in the survey for the HANA search approach as well.

TABLE V. Q4: SEARCH TERM: MARKETING INTERNET (TRANSLATED)

HANA Search	
Found Lecture Names	*Semester*
On the Effects of Reputation in the Internet of Service	Winter 2012
Internet Security	Winter 2010
Information Management in the Internet	Summer 2010
eBusiness Marketing	Summer 2010

Weighted MySQL Search	
Found Lecture Names	*Semester*
Internetworking with IPV4	Summer 2013
Weaknesses of Internet Protocols	Winter 2013
Internet Security	Summer 2014
Internet Security - Law and Ethics	Winter 2013

TABLE VI. Q5: MEMORY MANAGEMENT (TRANSLATED)

HANA Search	
Found Lecture Names	*Semester*
Memory Management	Winter 2012
Memory Management - Concepts (1)	Winter 2012
Memory Management - Implementation (2)	Winter 2012
Memory Management - Concepts (3)	Winter 2012

Weighted MySQL Search	
Found Lecture Names	*Semester*
Process Management in Companies	Winter 2013
Virtualized Resource Management	Summer 2014
Stages of Operational Process Management	Winter 2014
Sustainable Innovation Management	Winter 2013

V. Results and Future Work

This paper deals with the approach to use an in-memory database and the extracted OCR data to extend the search capabilities of lecture archives. Additionally, it has to be taken into account that the search processing time should be as short as possible. Furthermore, the search results should offer better matching results for the user and "search engine like"-functionality like excluding terms or search for a special phrase. As seen in the evaluation the OCR data search of lecture slides for big amounts of lectures can be done in a reasonable time of approximately 300ms with removing repeated lectures from old years. Moreover, the results out of 78 completed surveys show the obvious confirmation that the OCR analysis approach with support by an in-memory database detects more clear results under tested circumstances. This result shows that the approach has proven to be used in lecture archives to enhance the user experience by finding lectures efficiently.

Even when the search results are very reasonable, there is still a lot of future work to do. After analyzing "Optical Character Recognition" (OCR) data also "Automatic Speech Recognition" (ASR) data which we have already can be included for the search. This approach would make the search even more precise. Another idea is to integrate semantic web technologies into the search to find lectures with synonyms automatically and add the option to search for semantically connected words. Furthermore, lecturers should be able to upload PDF files containing additional lecture material. This material can be additionally analyzed by the database and shown in the search result together with the lecture. Moreover, the search can be optimized in different areas. The database give suggestions based on commonly used search terms. These commonly used search terms can be cached additionally to remove the database workload. Finally, all data which is search related should be stored in the in-memory database. This will decrease execution time for collecting additional lecture data from the MySQL database. This would remove the current bottleneck shown in Figure 3.

References

[1] L. A. Rowe, D. Harley, P. Pletcher, and S. Lawrence, "Bibs: A lecture webcasting system," *Center for Studies in Higher Education*, 2001.

[2] V. Schillings and C. Meinel, "tele-task: Teleteaching anywhere solution kit," in *Proceedings of the 30th Annual ACM SIGUCCS Conference on User Services*, ser. SIGUCCS '02. New York, NY, USA: ACM, 2002, pp. 130–133. [Online]. Available: http://doi.acm.org/10.1145/588646.588673

[3] R. B. Miller, "Response time in man-computer conversational transactions," in *Proceedings of the December 9-11, 1968, Fall Joint Computer Conference, Part I*, ser. AFIPS '68 (Fall, part I). New York, NY, USA: ACM, 1968, pp. 267–277. [Online]. Available: http://doi.acm.org/10.1145/1476589.1476628

[4] Cisco, *15-Minute Guide to Pulse Video Analytics*, 2012. [Online]. Available: http://www.cisco.com/c/dam/en/us/products/collateral/conferencing/show-share/at_a_glance_c45-708001.pdf

[5] Panopto, *Smart Search by Panopto*, 2014.

[6] H. Yang, M. Siebert, P. Luhne, H. Sack, and C. Meinel, "Lecture video indexing and analysis using video ocr technology," in *Signal-Image Technology and Internet-Based Systems (SITIS), 2011 Seventh International Conference on*, Nov 2011, pp. 54–61.

[7] R. Vernica and C. Li, "Efficient top-k algorithms for fuzzy search in string collections," in *Proceedings of the First International Workshop on Keyword Search on Structured Data*, ser. KEYS '09. New York, NY, USA: ACM, 2009, pp. 9–14. [Online]. Available: http://doi.acm.org/10.1145/1557670.1557677

[8] C. Gormley and Z. Tong, *Elasticsearch: The Definitive Guide*. " O'Reilly Media, Inc.", 2015.

[9] J. Adcock, M. Cooper, L. Denoue, H. Pirsiavash, and L. A. Rowe, "Talkminer: a lecture webcast search engine," in *Proceedings of the international conference on Multimedia*. ACM, 2010, pp. 241–250.

[10] X. Wu, A. G. Hauptmann, and C.-W. Ngo, "Practical elimination of near-duplicates from web video search," in *Proceedings of the 15th International Conference on Multimedia*, ser. MULTIMEDIA '07. New York, NY, USA: ACM, 2007, pp. 218–227. [Online]. Available: http://doi.acm.org/10.1145/1291233.1291280

[11] F. Färber, S. K. Cha, J. Primsch, C. Bornhövd, S. Sigg, and W. Lehner, "Sap hana database: Data management for modern business applications," *SIGMOD Rec.*, vol. 40, no. 4, pp. 45–51, Jan. 2012. [Online]. Available: http://doi.acm.org/10.1145/2094114.2094126

[12] R. A. Wagner and M. J. Fischer, "The string-to-string correction problem," *J. ACM*, vol. 21, no. 1, pp. 168–173, Jan. 1974. [Online]. Available: http://doi.acm.org/10.1145/321796.321811

[13] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," in *Soviet physics doklady*, vol. 10, 1966, p. 707.

[14] S. Robertson, "Understanding inverse document frequency: on theoretical arguments for idf," *Journal of Documentation*, vol. 60, no. 5, pp. 503–520, 2004. [Online]. Available: http://dx.doi.org/10.1108/00220410410560582