A Speech-based Just-in-Time Retrieval System using Semantic Search

Andrei Popescu-Belis, Majid Yazdani, Alexandre Nanchen, and Philip N. Garner
Idiap Research Institute
Rue Marconi 19, CP 592
1920 Martigny, Switzerland
{apbelis,myazdani,ananchen,pgarner}@idiap.ch

Abstract

The Automatic Content Linking Device is a just-in-time document retrieval system which monitors an ongoing conversation or a monologue and enriches it with potentially related documents, including multimedia ones, from local repositories or from the Internet. The documents are found using keyword-based search or using a semantic similarity measure between documents and the words obtained from automatic speech recognition. Results are displayed in real time to meeting participants, or to users watching a recorded lecture or conversation.

1 Introduction

Enriching a monologue or a conversation with related content, such as textual or audio-visual documents on the same topic, is a task with multiple applications in the field of computer-mediated human-human communication. In this paper, we describe the Automatic Content Linking Device (ACLD), a system that analyzes spoken input from one or more speakers using automatic speech recognition (ASR), in order to retrieve related content, in real-time, from a variety of repositories. These include local document databases or archives of multimedia recordings, as well as websites. Local repositories are queried using a keyword-based search engine, or using a semantic similarity measure, while websites are queried using commercial search engines.

We will first describe the scenarios of use of the ACLD in Section 2, and review previous systems for just-in-time retrieval in Section 3. The ACLD components will be outlined in Sections 4.1 to 4.5. Four types of evaluation results obtained with our system will finally be summarized in Sections 5.1 to 5.4.

2 Content Linking: Scenarios of Use

Just-in-time information retrieval, i.e. finding useful documents without the need for a user to initiate a direct search for them, is one of the ways in which the large quantity of knowledge that is available in networked environments can be efficiently put to use. To perform this task, a system must consider explicit and implicit input from users, mainly speech or typed input, and attempt to model their context, in order to provide recommendations, which users are free to consult if they feel the need for additional information.

One of the main scenarios of use for the ACLD involves people taking part in meetings, who often mention documents containing facts under discussion, but do not have the time to search for them without interrupting the discussion flow. The ACLD performs this search for them. Moreover, as the ACLD was developed on meetings from the AMI Corpus, it can also perform the same operations on a replayed meeting, as a complement to a meeting browser, for development or demonstration purposes.

In a second scenario, content linking is performed over live or recorded lectures, for instance in a computer-assisted learning environment for individual students. The ACLD enriches the lectures with related material drawn from various repositories, through a search process that can be guided in real

Proceedings of the ACL-HLT 2011 System Demonstrations, pages 80–85,
Portland, Oregon, USA, 21 June 2011.

time by its user. The advantage of real-time content linking over a more static enrichment, such as the Feynman lectures at Microsoft Research,[1] is that users can tune search parameters at will while viewing the lecture.

3 Just-in-Time Retrieval Systems

The first precursors to the ACLD were the Fixit query-free search system (Hart and Graham, 1997), the Remembrance Agent for just-in-time retrieval (Rhodes and Maes, 2000), and the Implicit Queries (IQ) system (Dumais et al., 2004). Fixit monitored the state of a user's interaction with a diagnostic system, and excerpts from maintenance manuals depending on the interaction state. The Remembrance Agent was integrated to the Emacs text editor, and ran searches over emails or notes at regular time intervals (every few seconds) using the latest 20–500 words typed by the user. The IQ system generated context-sensitive searches based on a user's ongoing activities on their computer, such as writing email. A version of the Remembrance Agent called Jimminy was conceived as a wearable assistant for taking notes, but ASR was only simulated for evaluation (Rhodes, 1997).

The Watson system (Budzik and Hammond, 2000) monitored the user's operations in a text editor, but proposed a more complex mechanism than the Remembrance Agent for selecting terms for queries, which were directed to a web search engine. Another assistant for an authoring environment was developed in the A-Propos project (Puerta Melguizo et al., 2008). A query-free system was designed for enriching television news with articles from the Web (Henziker et al., 2005).

The FAME interactive space (Metze and al., 2006), which provides multi-modal access to recordings of lectures via a table top interface, bears many similarities to the ACLD. However, it requires the use of specific voice commands by one user only, and does not spontaneously follow a conversation.

More recently, several speech-based search engines have become available, including as smart phone applications. Conversely, many systems allow searching of spoken document archives.[2] Inspiration from these approaches, which are not query-free, can nevertheless be useful to just-in-time retrieval. Other related systems are the Speech Spotter (Goto et al., 2004) and a personal assistant using dual-purpose speech (Lyons et al., 2004), which enable users to search for information using commands that are identified in the speech flow.

The ACLD improves over numerous past ones by giving access to indexed multimedia recordings as well as websites, with fully operational ASR and semantic search, as we now explain.

4 Description of the ACLD

The architecture of the ACLD comprises the following functions: document preparation, text extraction and indexing; input sensing and query preparation; search and integration of results; user interface to display the results.

4.1 Document Preparation and Indexing

The preparation of the local database of documents for content linking involves mainly the extraction of text, and then the indexing of the documents, which is done using Apache Lucene software. Text can be extracted from a large variety of formats (including MS Office, PDF, and HTML) and hierarchies of directories are recursively scanned. The document repository is generally prepared before using the ACLD, but users can also add files at will. Because past discussions are relevant to subsequent ones, they are passed through offline ASR and then chunked into smaller units (e.g. of fixed length, or based on a homogeneous topic). The resulting texts are indexed along with the other documents.

The ACLD uses external search engines to search in external repositories, for instance the Google Web search API or the Google Desktop application to search the user's local drives.

4.2 Sensing the User's Information Needs

We believe that the most useful cues about the information needs of participants in a conversation, or of people viewing a lecture, are the words that are spoken during the conversation or the lecture. For the ACLD, we use the AMI real-time ASR system (Garner et al., 2009). One of its main features is the use of a pre-compiled grammar, which allows it to retain accuracy even when running in real-

[1] See http://research.microsoft.com/apps/tools/tuva/.

[2] See workshops at http://www.searchingspeech.org.

time on a low resource machine. Of course, when content linking is done over past meetings, or for text extraction from past recordings, the ASR system runs slower than real-time to maximize accuracy of recognition. However, the accuracy of real-time ASR is only about 1% lower than the unconstrained mode which takes several times real-time.

For the RT07 meeting data, when using signals from individual headset microphones, the AMI ASR system reaches about 38% word error rate. With a microphone array, this increases to about 41%. These values indicate that enough correct words are sensed by the real-time ASR to make it applicable to the ACLD, and that a robust search mechanism could help avoiding retrieval errors due to spurious words.

The words obtained from the ASR are filtered for stopwords, so that only content words are used for search; our list has about 80 words. Furthermore, we believe that existing knowledge about the important terminology of a domain or project can be used to increase the impact of specific words on search. A list of pre-specified keywords can be defined based on such knowledge and can be modified while running the ACLD. For instance, for remote control design as in the AMI Corpus scenario, this list includes about 30 words such as 'chip', 'button', or 'material'. If any of them is detected in the ASR output, then their importance is increased for searching, but otherwise all the other words from the ASR (minus the stopwords) are used for constructing the query.

4.3 Querying the Document Database

The Query Aggregator (QA) uses the ASR words to retrieve the most relevant documents from one or more databases. The current version of the ACLD makes use of semantic search (see next subsection), while previous versions used word-based search from Apache Lucene for local documents, or from the Google Web or Google Desktop APIs. ASR words from the latest time frame are put together (minus the stopwords) to form queries, and recognized keywords are boosted in the Lucene query. Queries are formulated at regular time intervals, typically every 15-30 seconds, or on demand. This duration is a compromise between the need to gather enough words for search, and the need to refresh the search results reasonably often.

The results are integrated with those from the previous time frame, using a persistence model to smooth variations over time. The model keeps track of the salience of each result, initialized from their ranking among the search results, then decreasing in time unless the document is again retrieved. The rate of decrease (or its inverse, persistence) can be tuned by the user, but in any case, all past results are saved by the user interface and can be consulted at any time.

4.4 Semantic Search over Wikipedia

The goal of our method for semantic search is to improve the relevance of the retrieved documents, and to make the mechanism more robust to noise from the ASR. We have applied to document retrieval the graph-based model of semantic relatedness that we recently developed (Yazdani and Popescu-Belis, 2010), which is also related to other proposals (Strube and Ponzetto, 2006; Gabrilovich and Markovitch, 2007; Yeh et al., 2009).

The model is grounded in a measure of semantic relatedness between text fragments, which is computed using random walk over the network of Wikipedia articles – about 1.2 million articles from the WEX data set (Metaweb Technologies, 2010). The articles are linked through hyperlinks, and also through lexical similarity links that are constructed upon initialization. The random walk model allows the computation of a *visiting probability (VP)* from one article to another, and then a VP between sets of articles, which has been shown to function as a measure of semantic relatedness, and has been applied to various NLP problems. To compute relatedness between two text fragments, these are first projected represented into the network by the ten closest articles in terms of lexical similarity.

For the ACLD, the use of semantic relatedness for document retrieval amounts to searching, in a very large collection, the documents that are the most closely related to the words from the ASR in a given timeframe. Here, the document collection is (again) the set of Wikipedia articles from WEX, and the goal is to return the eight most related articles. Such a search is hard to perform in real time; hence, the solution that was found makes use of several approximations to compute average VP between the ASR fragment and all articles in the Wikipedia network.

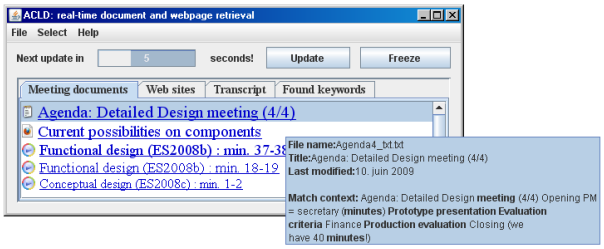

Figure 1: Unobtrusive UI displaying document results. Hovering the mouse over a result (here, the most relevant one) displays a pop-up window with more information about it.

4.5 The User Interface (UI)

The main goal of the UI is to make available all information produced by the system, in a configurable way, allowing users to see a larger or smaller amount of information according to their needs. A modular architecture with a flexible layout has been implemented, maximizing the accessibility but also the understandability of the results, and displaying also intermediary data such as ASR words and found keywords. The UI displays up to five widgets, which can be arranged at will:

1. ASR results with highlighted keywords.
2. Tag-cloud of keywords, coding for recency and frequency of keywords.
3. Names of documents and past meeting snippets found by the QA.
4. Names of web pages found via the Google API.
5. Names of local files found via the Google Desktop API.

Two main arrangements are intended, though many others are possible: an informative full-screen UI, shown in Figure 2 with widgets 1–4; and an unobtrusive widget UI, with superposed tabs, shown in Figure 1 with widget 3.

The document names displayed in widgets 3–5 function as hyperlinks to the documents, launching appropriate external viewers when the user clicks on them. Moreover, when hovering over a document name, a pop-up window displays metadata and document excerpts that match words from the query, as an explanation of why the document was retrieved.

5 Evaluation Experiments

Four types of evidence for the relevance and utility of the ACLD are summarized in this section.

5.1 Feedback from Potential Users

The ACLD was demonstrated to about 50 potential users (industrial partners, focus groups, etc.) in a series of sessions of about 30 minutes, starting with a presentation of the ACLD and continuing with a discussion and elicitation of feedback. The overall concept was generally found useful, with positive verbal evaluations. Feedback for smaller and larger improvements was collected: e.g. the importance of matching context, linking on demand, and the UI unobtrusive mode.

5.2 Pilot Task-based Experiments

A pilot experiment was conducted by a team at the University of Edinburgh with an earlier version of the unobtrusive UI. Four subjects had to complete a task that was started in previous meetings (ES2008a-b-c from the AMI Corpus). The goal was to compare two conditions, *with* vs. *without* the ACLD, in terms of satisfied constraints, overall efficiency, and satisfaction. Two pilot runs have shown that the ACLD was being consulted about five times per meeting. Therefore, many more runs are required to reach statistical significance of observations, and remain to be executed depending on future resources.

5.3 Usability Evaluation of the UI

The UI was submitted to a usability evaluation experiment with nine non-technical subjects. The subjects used the ACLD over a replayed meeting recording, and were asked to perform several tasks with it, such as adding a keyword to monitor, searching for a word, or changing the layout. The subjects then rated usability-related statements, leading to an assessment on the System Usability Scale (Brooke, 1996).

The overall usability score was 68% (SD: 10), which is considered as 'acceptable usability' for the SUS. The average task-completion time was 45–75 seconds. In free-form feedback, subjects found the system helpful to review meetings but also lectures, appreciated the availability of documents, but also noted that search results (with keyword-based

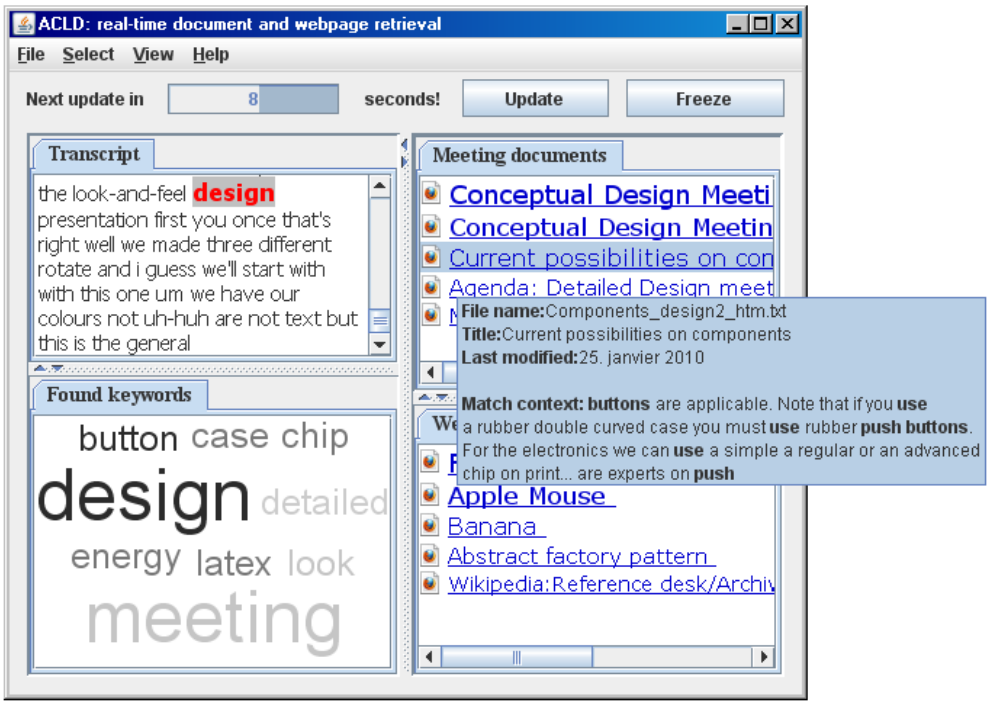

Figure 2: Full screen UI with four widgets: ASR, keywords, document and website results.

search) were often irrelevant. They also suggested simplifying the UI (menus, layout) and embedding a media player for use in the meeting or lecture replay scenario.

5.4 Comparing the Relevance of Keyword-based vs. Semantic Search

We compared the output of semantic search with that of keyword-based search. The ASR transcript of one AMI meeting (ES2008d) was passed to both search methods, and 'evaluation snippets' containing the manual transcript for one-minute excerpts, accompanied by the 8-best Wikipedia articles found by each method were produced. Overall, 36 snippets were generated. The manual transcript shown to subjects was enriched with punctuation and speakers' names, and the names of the Wikipedia pages were placed on each side of the transcript frame.

Subjects were then asked to read each snippet, and decide which of the two document sets was the most relevant to the discussion taking place, i.e. the most useful as a suggestion to the participants. They could also answer 'none', and could consult the result if necessary.

Results were obtained from 8 subjects, each seeing 9 snippets out of 36. Every snippet was thus seen by two subjects. The subjects agreed on 23 (64%) snippets and disagreed on 13 (36%). In fact, the number of true disagreements not including the answer 'none' was only 7 out of 36.

Over the 23 snippets on which subjects agreed, the result of semantic search was judged more relevant than that of keyword search for 19 snippets (53% of the total), and the reverse for 4 snippets only (11%). Alternatively, if one counts the votes cast by subjects in favor of each system, regardless of agreement, then semantic search received 72% of the votes and keyword-based only 28%. These numbers show that semantic search quite clearly improves relevance in comparison to keyword-based one, but there is still room for improvement.

6 Conclusion

The ACLD is, to the best of our knowledge, the first just-in-time retrieval system to use spontaneous speech and to support access to multimedia documents and web pages, using a robust semantic search method. Future work will aim at improving the relevance of semantic search, at modeling context to

improve timing of results, and at inferring relevance feedback from users. The ACLD should also be applied to specific use cases, and an experiment with group work in a learning environment is under way.

Acknowledgments

The authors gratefully acknowledge the support of the EU AMI and AMIDA Integrated Projects (http://www.amiproject.org) and of the Swiss IM2 NCCR on Interactive Multimodal Information Management (http://www.im2.ch).

References

John Brooke. 1996. SUS: A 'quick and dirty' usability scale. In Patrick W. Jordan, Bruce Thomas, Bernard A. Weerdmeester, and Ian L. McClelland, editors, *Usability evaluation in industry*, pages 189–194. Taylor and Francis, London, UK.

Jay Budzik and Kristian J. Hammond. 2000. User interactions with everyday applications as context for just-in-time information access. In *IUI 2000 (5th International Conference on Intelligent User Interfaces)*, New Orleans, LA.

Susan Dumais, Edward Cutrell, Raman Sarin, and Eric Horvitz. 2004. Implicit Queries (IQ) for contextualized search. In *SIGIR 2004 (27th ACM SIGIR Conference) Demonstrations*, page 534, Sheffield, UK.

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *IJCAI 2007 (20th International Joint Conference on Artificial Intelligence)*, pages 6–12, Hyderabad, India.

Philip N. Garner, John Dines, Thomas Hain, Asmaa El Hannani, Martin Karafiat, Danil Korchagin, Mike Lincoln, Vincent Wan, and Le Zhang. 2009. Real-time ASR from meetings. In *Interspeech 2009 (10th Annual Conference of the Intl. Speech Communication Association)*, pages 2119–2122, Brighton, UK.

Masataka Goto, Koji Kitayama, Katsunobu Itou, and Tetsunori Kobayashi. 2004. Speech Spotter: On-demand speech recognition in human-human conversation on the telephone or in face-to-face situations. In *ICSLP 2004 (8th International Conference on Spoken Language Processing)*, pages 1533–1536, Jeju Island.

Peter E. Hart and Jamey Graham. 1997. Query-free information retrieval. *IEEE Expert: Intelligent Systems and Their Applications*, 12(5):32–37.

Monika Henziker, Bay-Wei Chang, Brian Milch, and Sergey Brin. 2005. Query-free news search. *World Wide Web: Internet and Web Information Systems*, 8:101–126.

Kent Lyons, Christopher Skeels, Thad Starner, Cornelis M. Snoeck, Benjamin A. Wong, and Daniel Ashbrook. 2004. Augmenting conversations using dual-purpose speech. In *UIST 2004 (17th Annual ACM Symposium on User Interface Software and Technology)*, pages 237–246, Santa Fe, NM.

Metaweb Technologies. 2010. Freebase Wikipedia Extraction (WEX). http://download.freebase.com/wex/.

Florian Metze and al. 2006. The 'Fame' interactive space. In *Machine Learning for Multimodal Interaction II*, LNCS 3869, pages 126–137. Springer, Berlin.

Maria Carmen Puerta Melguizo, Olga Monoz Ramos, Lou Boves, Toine Bogers, and Antal van den Bosch. 2008. A personalized recommender system for writing in the Internet age. In *LREC 2008 Workshop on NLP Resources, Algorithms, and Tools for Authoring Aids*, pages 21–26, Marrakech, Morocco.

Bradley J. Rhodes and Pattie Maes. 2000. Just-in-time information retrieval agents. *IBM Systems Journal*, 39(3-4):685–704.

Bradley J. Rhodes. 1997. The Wearable Remembrance Agent: A system for augmented memory. *Personal Technologies: Special Issue on Wearable Computing*, 1:218–224.

Michael Strube and Simone Paolo Ponzetto. 2006. Wikirelate! Computing semantic relatedness using Wikipedia. In *AAAI 2006 (21st National Conference on Artificial Intelligence)*, pages 1419–1424, Boston, MA.

Majid Yazdani and Andrei Popescu-Belis. 2010. A random walk framework to compute textual semantic similarity: A unified model for three benchmark tasks. In *ICSC 2010 (4th IEEE International Conference on Semantic Computing)*, pages 424–429, Pittsburgh, PA.

Eric Yeh, Daniel Ramage, Christopher D. Manning, Eneko Agirre, and Aitor Soroa. 2009. WikiWalk: random walks on Wikipedia for semantic relatedness. In *TextGraphs-4 (4th Workshop on Graph-based Methods for NLP)*, pages 41–49, Singapore.