# Demonstrating Ambient Search: Implicit Document Retrieval for Speech Streams

**Benjamin Milde**[1,2], **Jonas Wacker**[1], **Stefan Radomski**[2], **Max Mühlhäuser**[2], and **Chris Biemann**[1]

[1] Language Technology Group / [2] Telecooperation Group
Computer Science Department
Technische Universität Darmstadt, Germany

## Abstract

In this demonstration paper we describe Ambient Search, a system that displays and retrieves documents in real time based on speech input. The system operates continuously in ambient mode, i.e. it generates speech transcriptions and identifies main keywords and keyphrases, while also querying its index to display relevant documents without explicit query. Without user intervention, the results are dynamically updated; users can choose to interact with the system at any time, employing a conversation protocol that is enriched with the ambient information gathered continuously. Our evaluation shows that Ambient Search outperforms another implicit speech-based information retrieval system. Ambient search is available as open source software.

## 1 Introduction

In the recent past, personal assistants like Siri or Google Now emerged, providing a natural voice-based interface for querying and finding information. These developments have been made possible by recent advances in Automated Speech Recognition (ASR) and Natural Language Understanding (NLU). Typically, these systems are actively triggered by users and are constrained to an ever–growing, yet finite set of hard-wired commands and question types. However, people may want to look up helpful information or check facts during a conversation or while listening to a lecture. In these situations, the interaction with a personal assistant or searching the web for the information hampers the flow of the discussion, respectively leads to distraction. In this work, we demonstrate Ambient Search. The system displays relevant information on-the-fly, solely based on speech input streams. In contrast to the above-mentioned personal assistants, our system does not require explicit interaction, as it unobtrusively listens to speech streams in the background, updating relevant results as time progresses. As a proof of concept, we make use of TED talks to demonstrate our system.

## 2 Related Work

The Remembrance Agent (Rhodes and Starner, 1996) is an early prototype of a continuously running automated information retrieval system, implemented as a plugin for the text editor Emacs. Given a collection of user-accumulated email and personal files, it attempts to find those documents that are most relevant to the user's current context. Rhodes and Maes (2000) also defined the term *just-in-time information retrieval agents* as "a class of software agents that proactively present information based on a person's context in an easily accessible and non-intrusive manner". Dumais et al. (2004) introduced an implicit query (IQ) system, which serves as a background system when writing emails. It also uses Term Frequency – Inverse Document Frequency (TF-IDF) scores for keyword extraction, like the Remembrance Agent. The most similar system to *Ambient Search* was presented by Habibi and Popescu-Belis (2015), extending earlier work on an Automatic Content Linking Device (Popescu-Belis et al., 2000).

## 3 System Description

Our system is based on the following processing steps. They are carried out in real-time:





*Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, pages 233–237, Osaka, Japan, December 11-17 2016.

**Speech decoding.** We stream the speech signal into an ASR system, emitting partial sentence hypotheses and predicting sentence boundaries. We use Kaldi (Povey et al., 2011), an open-source speech recognition framework and acoustic models based on the TED-LIUM corpus (Rousseau et al., 2014) and the TEDLIUM 4-gram language model (LM) from Cantab Research (Williams et al., 2015). We make use of kaldi-gstreamer-server[1], which wraps a Kaldi model into a streaming server that can be accessed with websockets. This provides a bi-directional communication channel, where audio is streamed to the server application and partial and full sentence hypothesis and boundaries are simultaneously returned.

**Keyword and keyphrase extraction.** Once a full sentence has been hypothesized, new keywords and keyphrases are extracted in the current sentence, if available. A keyphrase, as opposed to a single keyword, can consist of serveral words that refer to one concept. We first precompute a DRUID (Riedl and Biemann, 2015) dictionary on a recent Wikipedia dump with scores for noun phrases. DRUID is a state-of-the-art unsupervised measure for multiword expressions (MWEs) using distributional semantics and precomputed dictionaries for English can be downloaded from the JoBimText project website[2]. All keyphrases with a DRUID score over a certain threshold (e.g. 0.7, see also Section 4) and all remaining words that are adjectives and nouns, as determined by an off-the-shelf part of speech (POS)-tagger[3], are used as candidate terms.

**Candidate term ranking.** We score candidate terms according to a Word2Vec (Mikolov et al., 2013) and TF-IDF ranking measure. We first precompute IDF and lookup tables for all unique words in the Simple English Wikipedia and for all multiword terms in our DRUID dictionary. Word2Vec (CBOW) is our source of semantic similarity. We train it on stemmed text and treat multiwords as opaque units. We then compute the average Word2Vec vector over all candidate terms. Finally, we score each candidate term according to the cosine distance of each term word vector to the average word vector of the last 10 sentences and multiply this with the TF-IDF score of the given term.

**Index queries.** We use Elastic Search[4] and stream2es[5] to build an index of the Simple English Wikipedia[6]. We build an OR query of all top query terms (e.g. up to the top 10 query words), assigning the computed scores to the individual terms in the query. Eventually, the returned documents are also aggregated, i.e. older documents found with previous sentences decay their score over time (multiplied with $d = 0.9$) and newer documents are sorted into a list of $n$ best documents. This list is thus sorted by topical relevance of the documents and by time, with newer documents having precedence. Finally, the $n$ best relevant documents are presented to the user and updated as soon as changes become available.

**Implementation Details.** We encapsulate the processing steps into the following Python programs: (1) A Kaldi client program, that either uses the system's microphone or an audio file, streaming it in real time to obtain partial and full transcription hypothesis. (2) A relevant event generator program, that searches for new keywords and keyphrases and queries the elastic search index to obtain relevant documents. (3) The Ambient Search server, which sends out appropriate events to the browser view, to display the current top $n$ relevant documents and to move older documents into a timeline. We connect the individual modules with message passing through a common channel on a redis-server[7]. Through it, modules can send and receive events and act accordingly, e.g. to the availability of a new utterance hypothesis from the recognition module. Word2Vec and TF-IDF vectors are computed with the Gensim (Řehůřek and Sojka, 2010) package. The Ambient Search web page is implemented in HTML5/JS and connects to a server instance running on the Python micro-framework Flask[8], making use of Server Sent Events (SSE) to push new information from the server to the web browser. This enables a reversed information channel, where the server pushes descriptions of new relevant documents to the browser client as it becomes available.

### 3.1 Visual Presentation

Figure 1 gives a visual impression of our system, after it had been listening for a few minutes to Alice Bows Larkin's TED talk on climate change[9]. On the bottom of the page, we show excerpts of up to four

[1] https://github.com/alumae/kaldi-gstreamer-server [2] jobimtext.org/components/druid/
[3] http://spacy.io [4] https://elastic.co/ [5] https://github.com/elastic/stream2es
[6] https://simple.wikipedia.org [7] http://redis.io/ [8] http://flask.pocoo.org/
[9] http://ted.com/speakers/alice_bows_larkin

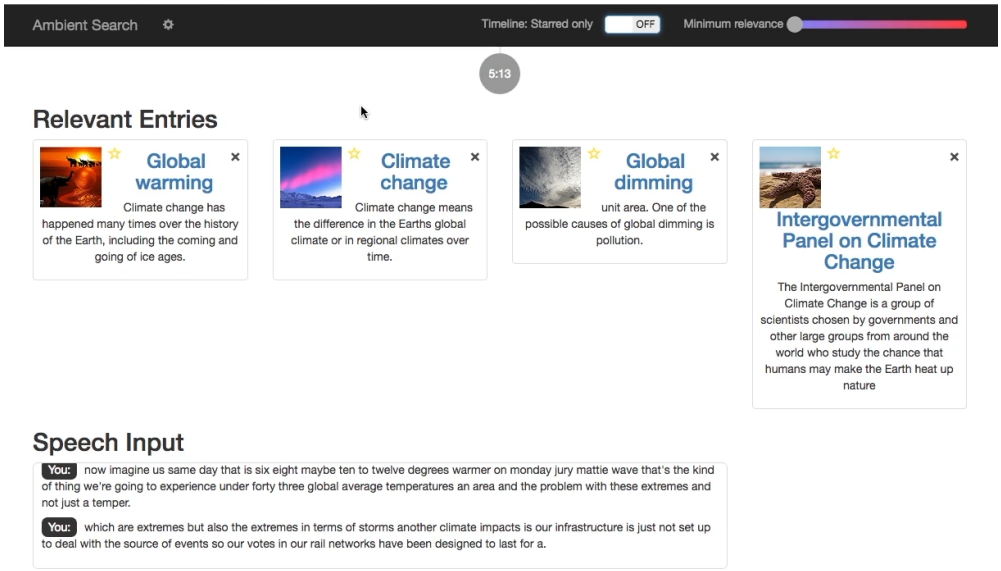


Figure 1: Screenshot of the system after listening to five minutes of the TED talk "We're too late to prevent climate change - here is how we adapt" by Alice Bows Larkin [9]

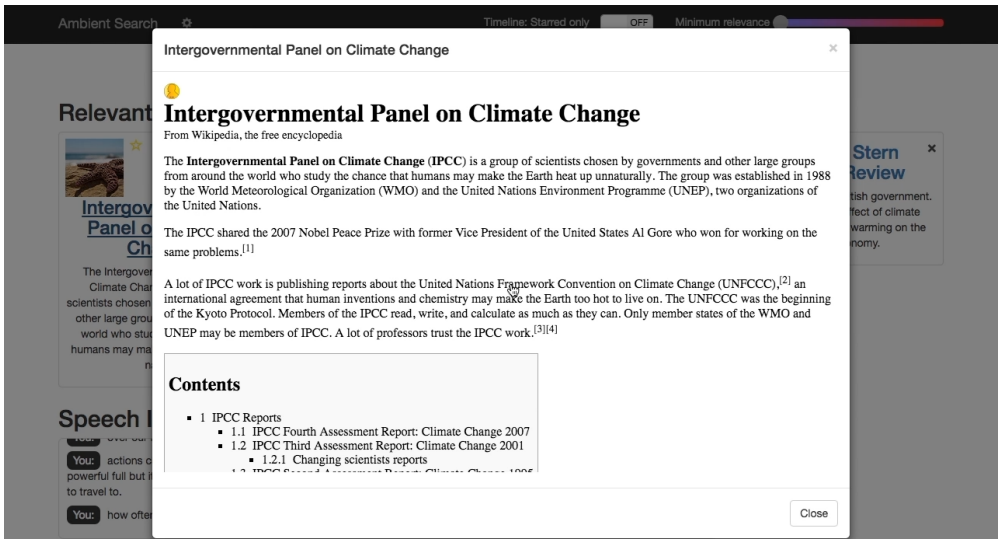


Figure 2: Screenshot of the modal dialog that opens after clicking on a relevant item, that can be used to read one of the proposed Wikipedia articles.

relevant Wikipedia documents to the user. Clicking on such a document opens up a modal view to read the Wikipedia article, as depicted in Figure 2. When newer and more relevant articles are retrieved, older articles move into a timeline, which is constructed above the currently retrieved articles. While the user is at the bottom of the page, the page keeps automatically scrolling to the end, like a terminal.

In the timeline (see also Figure 3), users can go back to previously displayed relevant entries and can adjust the minimal relevance of the shown elements, to filter entries by the systems confidence. Elements can also be bookmarked, to quickly retrieve them later. This can be used to quickly mark interesting articles to be read later, e.g. while Ambient Search is being used listening to a talk and the user only briefly interacts with it while listening. The displayed entries can also be removed from the web page at any time by clicking on the X in the boxes.

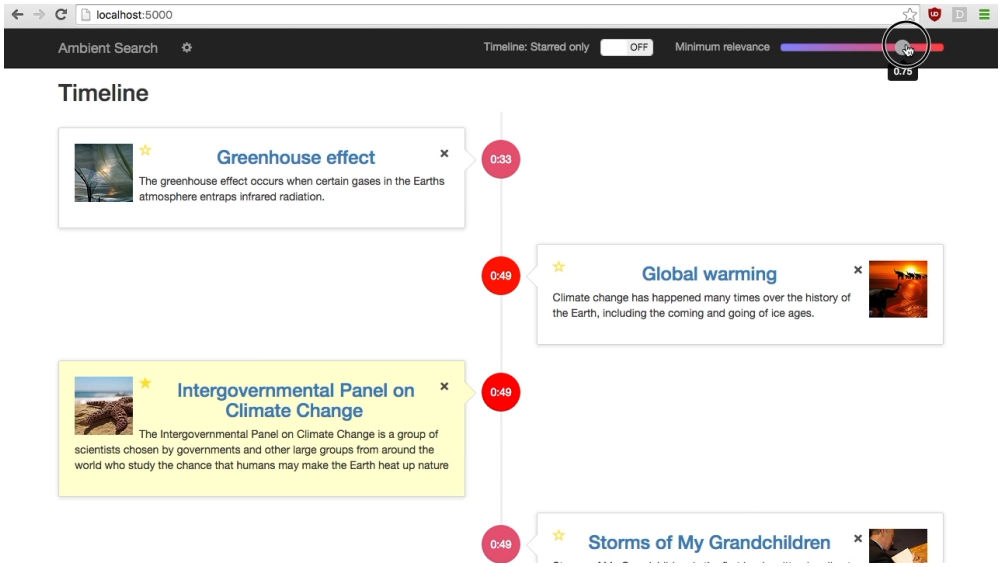


Figure 3: Screenshot of the timeline that constructs above the relevant entries, showing previously displayed relevant entries. Users can also adjust the minimal relevance bar at the top right of the screen, to filter entries by the systems confidence.

## 4 Evaluation

We directly measure how relevant the retrieved documents are: We focus on an evaluation of the top-ranked documents returned by the IR system for a particular TED talk fragment transcription, since only top documents are suggested to the user. The Normalized Discounted Cumulative Gain (NDCG) measure (Järvelin and Kekäläinen, 2002) is a popular choice to evaluate search engines and also takes into account the ranking of the proposed documents. We evaluate on the top-5 returned documents of the complete system, with two annotators and a standard relevance scale from 0-3. For computing NDCG, we pool all judgments across systems, obtaining an average of 27.7 relevance judgments per fragment, following standard practices for IR evaluations (Clarke et al., 2012).

| Method | NDCG (std. dev.) |
|---|---|
| (1) TF-IDF baseline no MWEs | 0.426 (27.8%) |
| (2) Habibi and PB original implementation | 0.427 (28.0%) |
| (3) Habibi and PB our prep. | 0.465 (24.1%) |
| (4) Habibi and PB our prep., **gold trans.** | 0.476 (21.7%) |

| Method | NDCG (std. dev.) |
|---|---|
| (5) Our proposed method with MWEs (c=0.7) | 0.471 (26.1%) |
| (6) Our proposed method without MWEs | **0.481 (26.8%)** |
| (7) Our proposed method without MWEs, **gold trans.** | 0.578 (25.2%) |
| (8) Our proposed method with MWEs (c=0.7), **gold trans.** | 0.602 (22.1%) |

Table 1: NDCG comparison of TF-IDF baseline keyword and keyphrase extraction methods, the proposed LDA based keyword extraction method by Habibi and Popescu-Belis (2015) and our proposed method based on DRUID, Word2vec and TF-IDF.

In Table 1, we show a comparison of different methods for automatic keyword extraction on TED talk transcriptions (as produced by kaldi-gstreamer-server). All methods use the same resources, i.e. they are all pretrained on the same Simple English Wikipedia dump from May 2016. We allow each system to produce an equal number of 10 terms per query. We did see good results using the method proposed by Habibi and Popescu-Belis (2015), beating our TF-IDF baseline (1). However, we noticed that the publicly available Matlab implementation of this method[10] did only remove stopwords as part of its preprocessing (2). When we use our preprocessing as input (3), we can improve both keyword and NDCG evaluation scores significantly. The best NDCG score using speech transcripts was obtained with our proposed

[10] https://github.com/idiap/DocRec

method *without* using multiwords (6). We experimented with different values of c: 0.3, 0.5 and 0.7, which all lowered NDCG scores. However, making our pipeline multiword-aware raised our NDCG score on manual (gold) transcriptions, cf. experiments (7 vs. 8). We have done further experiments and an in-depth error analysis in Milde et al. (2016), including further experiments on manual transcriptions.

## 5 Conclusion

We demonstrated *Ambient Search*, a system that can show and retrieve relevant documents for speech streams. As a proof-of-concept we indexed Wikipedia pages, as this provides an universal coverage of different topics and a large document collection for testing purposes. Our approach compares favorably over previous methods of topic discovery and keyword extraction in speech transcriptions. Our proposed term extraction and ranking method using Word2Vec (CBOW) embeddings and TF-IDF is simple to implement. It can also be adapted quickly to other languages without the need for any labeled training data. The only barrier of entry can be the availability of a speech recognition system in the target language.

As the proposed use of multiword terms seems to be somewhat dependent on the quality of the transcription, we consider including likelihood information of the speech recognition system in the future. Ambient search is published on Github [11] as an open source software licensed under the Apache 2.0 license. A demonstration video is also available, along all pretrained models, evaluation files and scripts that are necessary to repeat and reproduce the results presented in this paper.

**Acknowledgments.** This work was partly supported by the Bundesministerium für Bildung und Forschung (BMBF), Germany, within the Dialog+ project within the program KMU-innovativ. We also want to thank Alexander Hendrich for contributing to improve the HTML5/JS display client and Michelle Sandbrink for helping out with the relevance judgements of the retrieved documents.

## References

C. Clarke, N. Craswell, and E. Voorhees. 2012. Overview of the TREC 2012 Web Track. In *Proc. TREC*, Gaithersburg, MD, USA.

S. Dumais, E. Cutrell, R. Sarin, and E. Horvitz. 2004. Implicit Queries (IQ) for Contextualized Search. In *Proc. SIGIR*, page 594, Sheffield, UK.

M. Habibi and A. Popescu-Belis. 2015. Keyword Extraction and Clustering for Document Recommendation in Conversations. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(4):746–759.

K. Järvelin and J. Kekäläinen. 2002. Cumulated Gain-based Evaluation of IR Techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446.

T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Proc. NIPS*, pages 3111–3119, Lake Tahoe, NV, USA.

B. Milde, J. Wacker, S. Radomski, M. Mühlhäuser, and C. Biemann. 2016. Ambient Search: A Document Retrieval System for Speech Streams. In *Proc. COLING*, Osaka, Japan.

A. Popescu-Belis, J. Kilgour, P. Poller, A. Nanchen, E. Boertjes, and J. De Wit. 2000. Automatic Content Linking: Speech-based Just-in-time Retrieval for Multimedia Archives. In *Proc. SIGIR*, page 703, Athens, Greece.

D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely. 2011. The KALDI Speech Recognition Toolkit. In *Proc. IEEE ASRU*, Waikoloa, HI, USA.

R. Řehůřek and P. Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proc. LREC*, pages 45–50, Valletta, Malta.

B. Rhodes and P. Maes. 2000. Just-in-time Information Retrieval Agents. *IBM Systems Journal*, 39(3):686.

B. Rhodes and T. Starner. 1996. Remembrance Agent: A Continuously Running Automated Information Retrieval System. In *Proc. Practical Application Of Intelligent Agents and Multi Agent Technology*, pages 487–495.

M. Riedl and C. Biemann. 2015. A Single Word is not Enough: Ranking Multiword Expressions Using Distributional Semantics. In *Proc. EMNLP*, pages 2430–2440, Lisbon, Portugal.

A. Rousseau, P. Deléglise, and Y. Estève. 2014. Enhancing the TED-LIUM Corpus with Selected Data for Language Modeling and More TED Talks. In *Proc. LREC*, pages 3935–3939, Reykjavik, Iceland.

W. Williams, N. Prasad, D. Mrva, T. Ash, and T. Robinson. 2015. Scaling Recurrent Neural Network Language Models. In *Proc. ICASSP*, pages 5391–5395, Brisbane, Australia.

[11] `https://github.com/bmilde/ambientsearch`