Ambient Search: A Document Retrieval System 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

We present Ambient Search, an open source system for displaying and retrieving relevant documents in real time for speech input. The system works ambiently, that is, it unobstructively listens to speech streams in the background, identifies keywords and keyphrases for query construction and continuously serves relevant documents from its index. Query terms are ranked with Word2Vec and TF-IDF and are continuously updated to allow for ongoing querying of a document collection. The retrieved documents, in our case Wikipedia articles, are visualized in real time in a browser interface. Our evaluation shows that Ambient Search compares favorably to another implicit information retrieval system on speech streams. Furthermore, we extrinsically evaluate multiword keyphrase generation, showing positive impact for manual transcriptions.

1 Introduction

Recent advancements in Automated Speech Recognition (ASR) and Natural Language Understanding (NLU) have proliferated the use of personal assistants like Siri[1] or Google Now[2], with which people interact naturally with their voice. However, the activation of such systems has to be specifically triggered and they are targeted to an (ever–growing) set of anticipated question types and commands.

When taking part in a conversation or listening to a lecture, people may want to look up helpful information or check facts. Manually checking this information or interacting with a personal assistant would hamper the flow of the discussion, respectively distract from the lecture. In the following, we present *Ambient Search*, a system that ambiently researches relevant information, in the form of proposing relevant documents to users in conversations or users who passively listen to spoken language. In contrast to other personal assistants, our system is not specifically triggered, it unobtrusively listens to speech streams in the background and implicitly queries an index of documents. We see the following utility in our approach: The assistant stays in the background and does not disturb the user. Access to the displayed snippets is on demand and the user can access the information in context without the need to formulate a specific query.

On the other hand, these advantages are fundamentally based on how well the system is able to retrieve relevant documents, as the system's utility diminishes when proposing a lot of irrelevant documents. In this paper, we also evaluate how well the system is able to retrieve relevant Wikipedia articles in spite of average speech recognition word error rates (WER) of 15.6% on TED talks and show that it finds more relevant articles compared to another implicit information retrieval system on speech streams.

The next section discusses related research, while we give an overview and technical details of our approach in Section 3. We evaluate keyword recognition and retrieval relevance in Section 4, and conclude in Section 5.

[1] http://www.apple.com/ios/siri/

[2] https://www.google.com/landing/now/

2 Related Work

The Remembrance Agent (Rhodes and Starner, 1996) is an early prototype of a continuously running automated information retrieval system, which was implemented as a plugin for the text editor Emacs[3]. Given a collection of the user's accumulated email, Usenet news articles, papers, saved HTML files and other text notes, it attempts to find those documents that are most relevant to the user's current context. Rhodes and Maes (2000) 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-instrusive manner". A person's context can be a very broad term in this regard. E.g. Jimminy (Rhodes, 1997) represents a multimodal extension of the Remembrance Agent. 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.

Other systems cover a user's explicit information needs, e.g. Ada and Grace are virtual museum guides (Traum et al., 2012), that can suggest exhibitions and answer questions. The Mindmeld[4] commercial assistant can listen to conversations between people to improve the retrieval results by fusing the users location information with from transcripts extracted keywords. The FAME interactive space (Metze et al., 2005) is a multi-modal system that interacts with humans in multiple communication modes in order to suggest additional information to them. Although FAME supports speech recognition and voice commands, it only listens to conversations for a longer period of time when it guesses a conversation's topic and can suggest documents with explicit commands. Another class of systems try to record the content of a conversation or speech stream and visualize it using a network of terms: E.g. SemanticTalk (Biemann et al., 2004) iteratively builds a structure similar to a mind map and can also visualize conversation trails with respect to background documents.

The most similar approach to *Ambient Search* was presented by Habibi and Popescu-Belis (2015), extending earlier work of an Automatic Content Linking Device (ACLD) (Popescu-Belis et al., 2000). It uses an LDA topic model for the extraction of keywords and the formulation of topically separated search queries. The extracted set of keywords as well as the ultimately returned set of document recommendations fulfill a trade-off between topical coverage and topical diversity.

Because this system can be considered a state-of-the-art system of implicit information retrieval in speech streams, we compare our approach to this one in the evaluation in Section 4, alongside a TF-IDF-based baseline. A major difference to Habibi and Popescu-Belis (2015), that operates on complete speech transcriptions only, is that our implementation is also able to retrieve relevant documents in real time, e.g. process live speech input.

3 Our Approach to Ambient Search

Our approach is based on five major processing steps, as depicted in Figure 1. These steps are carried out in real-time and in a streaming fashion, i.e. we make use of a new transcription hypothesis as soon as it is available.

At first, the speech signal is streamed into an ASR system (1). It emits the partial sentence hypothesis and also predicts sentence boundaries. Once a full sentence has been hypothesized, new keywords and keyphrases are extracted in the current sentence, if available (2). These terms are then ranked (3) and merged with the ones from previous sentences. A query is then composed, which is submitted to a precomputed index of documents (4).

Eventually, the returned documents are also aggregated (5a), i.e. previously found documents decay their score over time 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 (5b) and updated as soon as changes become available. Alongside the n best documents, a time line of previously suggested articles is also maintained and displayed. The next subsections provide further details on the individual major processing steps.

[3] `https://www.gnu.org/software/emacs/`

[4] `http://www.mindmeld.com`

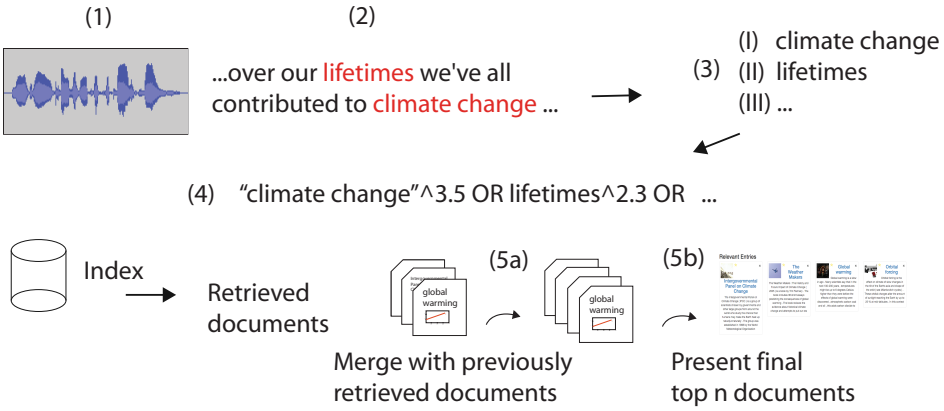

Figure 1: Processing steps of *Ambient search*

3.1 Speech Decoding

We use the popular Kaldi (Povey et al., 2011) open-source speech recognition framework and acoustic models based on the TED-LIUM corpus (Rousseau et al., 2014). We make use of online speech recognition, i.e. models that transcribe speech in real-time and emit partial transcription hypothesis, as opposed to offline models that operate on already recorded and complete utterances. The models were built using the standard recipe for online acoustic models based on a DNN-HMM acoustic model and i-vectors. We also make use of the TED-LIUM 4-gram language model (LM) from Cantab Research (Williams et al., 2015). The vocabulary of the speech recognizer is determined by its phoneme dictionary[5] and is confined to about 150k words. The online speech recognizer achieved an average WER of 15.6% on TED talks that we selected for the evaluation in Section 4.

We make use of kaldi-gstreamer-server[6], which wraps a Kaldi online 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 as JSON objects.

3.2 Keyphrase Extraction

A keyphrase, as opposed to a single keyword, can consist of one or more keywords that refer to one concept. We first precompute a DRUID (Riedl and Biemann, 2015) dictionary on a recent Wikipedia dump with scores for single adjectives or nouns and noun phrases. The restriction to only use adjectives and nouns is a common one in keyword detection, c.f. (Liu et al., 2010). DRUID is a state-of-the-art unsupervised measure for multiword expressions using distributional semantics. Intuitively, DRUID finds multiword expressions by combining an uniqueness measure for phrases with a measure for their incompleteness. Uniqueness in this context is based on the assumption that multiword expressions (MWEs) can often be substituted by a single word without considerably changing the meaning of a sentence.

The uniqueness measure uq(t) is computed with a distributional thesaurus, as the ratio of all similar unigrams of a term t divided by the number of n-grams similar to t. The incompleteness (ic) measure serves to punish incomplete terms in that it counts the number of times that the same words appear next to a term. The final DRUID measure for any term t is the subtraction of the incompleteness measure from the uniqueness measure: $DRUID(t) = uq(t) - ic(t)$. This helps to rank incomplete multiwords lower than their complete counterparts, e.g. 'red blood' is ranked lower than 'red blood cell'.

DRUID is implemented as a JoBimText (Biemann and Riedl, 2013) component, which can be down-

[5]The Kaldi TEDLIUM recipe uses CMUDICT (`http://www.speech.cs.cmu.edu/cgi-bin/cmudict`) plus a few automatically generated entries

[6]`https://github.com/alumae/kaldi-gstreamer-server`

loaded from the JoBimText project website[7] alongside precomputed dictionaries for English.

3.3 Term Ranking

We first precompute IDF and Word2Vec (Mikolov et al., 2013) 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 found with DRUID as opaque units. The final word embedding lookup table maps (in our case stemmed) word and phrase ids into a 100 dimensional continuous vector space. The model exploits the distributional properties of raw text for semantic similarity and the distance between embeddings can be used as a word and phrase similarity measure.

Using the lookup tables, we build a term ranking measure as follows. We extract all keyphrases from the last 10 sentences with a DRUID score of $\geq c$ and filter all stop words and any word that is not an adjective or noun, as determined by an off-the-shelf part of speech (POS)-tagger[8]. The cutoff constant c can be used to tune the amount of generated multiword candidates, with useful values ranging from 0.3 to 0.7 (see also Section 4). All multiwords and any single word remaining after filtering is proposed as a candidate. 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 term word vector of the last 10 sentences and multiply this with the TF-IDF score of the given term:

$$tr(term_k, trans) = d_{cos}\left(w2v(term_k), \frac{1}{|terms|}\sum_{i=1}^{|terms|} w2v(term_i)\right) \cdot TFIDF(term_k, trans)$$

where tr is the term ranking function that ranks a $term_k$ out of the set of all candidate $terms$ to a given transcript $trans$. $w2v$ yields the embedding of the given term, TFIDF yields the TFIDF score of the given term and transcript (IDF computed on the background Wikipedia corpus) and d_{cos} is the standard cosine similarity.

This ranking measure tr can be interpreted as a combination of the distance to the core topic (Word2Vec) and the general importance of the term for the text window (TF-IDF). We use the measure to extract up to 10 highest ranked candidate keywords and keyphrases in the text window. For the first third of Alice Bows Larkin's TED talk on climate change[9], the system would rank the terms as: "climate change", future, emissions, "negative impacts", potential, profound, workplaces, behaviors, gas.

3.4 Index Queries

We use Elastic Search[10] and stream2es[11] to build an index of the Simple English Wikipedia[12]. We index all articles, including special pages and disambiguation pages and use a query filter to obtain only regular articles when querying the index. We build an OR query where at least 25% of the query terms should match (by setting the "minimum_should_match" parameter), also assigning the scores obtained in the last section to the individual terms in the query.

With the example ranking from the previous section the query would be:

```
"climate change"^23.111 future^13.537 emissions^9.431 "negative impacts"^3.120
   potential^2.985 profound^2.679 workplaces^2.562 behaviors^2.368 gas^1.925
```

It would return the following Wikipedia articles (ranked by Elastic Search in that order):

[7] http://jobimtext.org/components/druid/

[8] The POS tagger we use is from the spacy library (http://spacy.io)

[9] https://www.ted.com/talks/alice_bows_larkin_we_re_too_late_to_prevent_climate_change_here_s_how_we_adapt

[10] https://www.elastic.co/

[11] https://github.com/elastic/stream2es

[12] https://simple.wikipedia.org

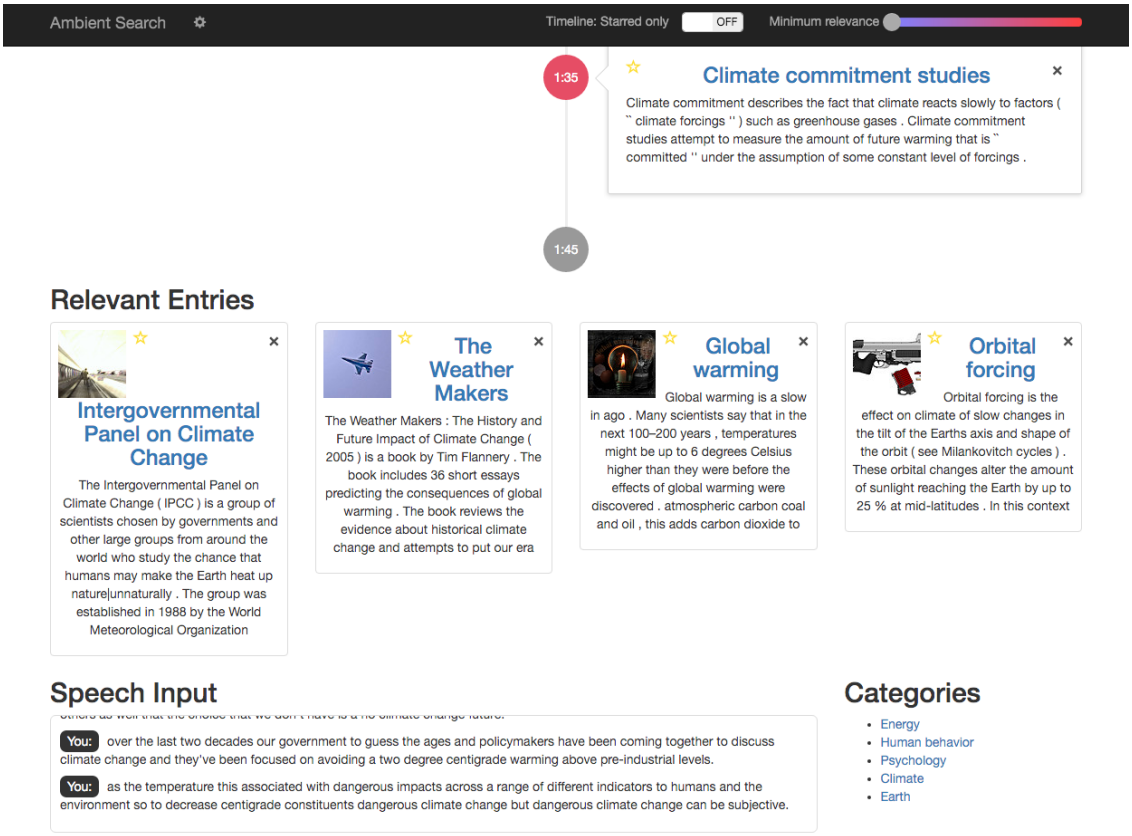

Figure 2: Screenshot of the system after listening to the first minutes of the TED talk "We're too late to prevent climate change - here is how we adapt" by Alice Bows Larkin [9]

```
"Intergovernmental Panel on Climate Change", "Global warming", "Climate change", "
    Global dimming", "The Weather Makers", "Greenhouse effect", "United Kingdom
    Climate Change Programme", "Ocean acidification", ...
```

This process is repeated for every new sentence and the scores of older retrieved documents decay (are multiplied with $d = 0.9$), to allow newer documents to rank higher.

3.5 Visual Presentation

Figure 2 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. We show excerpts of up to four relevant Wikipedia documents to the user. Clicking on such a document opens up a modal view to read the Wikipedia article. Articles are either retrieved online or using an offline version of the Simple English Wikipedia using XOWA[13]. Articles can be starred, to quickly retrieve them later and also removed, to signal the system that the article was irrelevant. When newer and more relevant articles are retrieved, older articles move into a timeline, which is constructed above the currently retrieved articles. The newest articles are at the bottom of the page and the page keeps automatically scrolling to the end, like a terminal, if the user does not scroll up. In the timeline, the relevance of a document is also visually displayed with different coloring of an element's circular anchor. The user can also regulate the threshold for minimum document relevance.

3.6 Implementation Details

We encapsulate the processing steps outlined in Section 3 into the following Python programs:

[13] https://gnosygnu.github.io/xowa/

(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 make use of message passing to communicate inputs and results of the individual programs using a redis-server[14]. Word2Vec and TF-IDF vectors are computed with the Gensim (Řehůřek and Sojka, 2010) package, while DRUID is precomputed as a list with JoBimText[15]. The Ambient Search web page is using HTML5/JS and Bootstrap[16] and connects to an ambient server instance running on the Python micro-framework Flask[17]. The web page is updated using Server Sent Events (SSE) or Long Polling as a browser fallback. This enables a reversed information channel, where the server pushes descriptions of new relevant documents to the browser client as it becomes available.

4 Evaluation

We base our evaluation on 30 fragments of 10 recent TED talks, which we downloaded as mp3 files from the TED.com website. These talks are not part of the TED-LIUM training dataset. In the following, we evaluate the proposed keywords and keyphrases, as well as the proposed documents from the in real-time transcribed audio file.

4.1 Keyphrase and Document Retrieval

We had two annotators manually pick terms (keywords and keyphrases) that are central to the topic of the talk and those that would cover a user's potential additional information needs. What should be included as a term can be very subjective, the inter-annotator agreement is $\kappa = 0.45$, with one annotator choosing 292 terms in total and the other 580. The overlapping set which we use in our evaluation consists of 206 terms and 460 other terms were chosen by only one of the annotators.

Finally, we also measure directly how relevant the retrieved documents are: We focus on an evaluation of the top-ranked documents returned by our ambient IR system for a particular TED talk fragment, since only top documents are suggested to the user of *Ambient Search*. 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. We had two annotators that used the standard relevance scale from 0-3, where 0 means irrelevant and 3 very relevant. NDCG directly measures how relevant the returned documents are. While the effort is considerably higher, since different system outputs have to be judged, NDCG measures the end-to-end performance of the system. 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). We use the standard NDCG measure with $k = 5$:

$$NDCG_k = \frac{DCG_k}{IDCG_k}$$

$$DCG_k = (rel_1 + \sum_{i=2}^{k} \frac{rel_i}{log_2 i})$$

where rel_i is a documents average relevance score in respect to the speech input. The Ideal Discounted Cumulative Gain (IDCG) assumes the best ranking of all possible relevant documents found in the set of all pooled judgements of a given transcript. The DCG on this optimal ranking, with respect to the set of documents retrieved by all systems for a particular transcript, is then used to compute IDCG.

[14] http://redis.io/
[15] http://jobimtext.org/components/druid
[16] http://getbootstrap.com/
[17] http://flask.pocoo.org/

4.2 Results

Keyword/keyphrase extraction method	Mean Recall (Std. Dev. in %)	Mean Precision (Std. Dev. in %)	Mean NDCG (Std. Dev. in %)
(1) TF-IDF baseline no MWEs, no filtering	26.97% (16.74%)	24.33% (15.42%)	0.188 (20.0%)
(2) TF-IDF baseline no MWEs, stopword filtering	39.24% (15.36%)	34.33% (10.86%)	0.387 (27.8%)
(3) TF-IDF baseline no MWEs, full filtering	40.91% (13.55%)	36.42% (10.99%)	0.426 (27.8%)
(4) TF-IDF baseline with MWEs (c=0.3), full filtering	43.22% (18.22%)	37.09% (16.61%)	0.392 (27.4%)
(5) Habibi and PB original implementation	36.68% (15.37%)	32.00% % (11.66%)	0.427 (28.0%)
(6) Habibi and PB our prep., without MWEs	43.76% (16.78%)	39.24% % (12.75%)	0.465 (24.1%)
(7) Our proposed method with MWEs (c=0.3)	**48.52% (21.55%)**	41.89 % (15.82%)	0.453 (25.7%)
(8) Our proposed method with MWEs (c=0.5)	48.08% (17.63%)	**42.42 % (13.20%)**	0.469 (26.9%)
(9) Our proposed method with MWEs (c=0.7)	48.48% (19.15%)	**42.42 % (13.45%)**	0.471 (26.1%)
(10) Our proposed method without MWEs	44.87% (17.24%)	40.08% (14.03%)	**0.481 (26.8%)**

Table 1: 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. The comparison is based on the same Kaldi transcriptions and the same training resources (Simple English Wikipedia from May 2016).

Keyword/keyphrase extraction method	Mean Recall (Std. Dev. in %)	Mean Precision (Std. Dev. in %)	Mean NDCG (Std. Dev. in %)
(11) Habibi and PB our prep., without MWEs	43.99% (15.26%)	39.33 % (12.63%)	0.476 (21.7%)
(12) Our proposed method with MWEs (c=0.3)	51.75% (20.43%)	45.67 % (16.47%)	0.518 (24.8%)
(13) Our proposed method with MWEs (c=0.5)	52.19% (19.09%)	46.19 % (15.27%)	0.574 (22.1%)
(14) Our proposed method with MWEs (c=0.7)	**52.68% (17.20%)**	**46.85 % (14.76%)**	**0.602 (22.1%)**
(15) Our proposed method without MWEs	47.81% (17.28%)	43.52 % (16.09%)	0.578 (25.2%)

Table 2: Comparison of the proposed LDA based keyword extraction method by Habibi and Popescu-Belis (2015) and our proposed method based on DRUID, Word2vec and TF-IDF on manual TED talk transcripts.

In Table 1, we show a comparison of different methods for automatic keyword extraction on TED talk transcriptions (as produced by kaldi-gstreamer-server / the Kaldi online model). All methods use the same resources, i.e. they are all pretrained on the same Simple English Wikipedia dump from May 2016. However, our proposed method and the TF-IDF baseline can also produce terms that are DRUID multiwords, whereas the original implementation of Habibi and Popescu-Belis (2015) can only produce single keywords. All methods where allowed to produce maximally 10 words in the keyword evaluation – partially covered keyphrases where also counted as a hit for the direct keyword evaluation and a multiword term was counted as multiple words. In the NDCG evaluation, we allow each system to produce an equal number of 10 terms.

For the TF-IDF baselines (1-4), preprocessing is the most important performance factor, with the best results obtained by filtering stop words and any words that are not adjectives and nouns. However, while DRUID multiwords help to gain much better keyword recognition scores, it did not achieve a better

NDCG score on speech transcripts. We also saw good results using the method proposed by Habibi and Popescu-Belis (2015), with the diversity constraint (λ) set to the default value of 0.75, which was the optimal value in the domain of meeting transcripts. However, we noticed that the publicly available Matlab implementation of this method[18] only removed stopwords as part of its preprocessing (5). When we use our preprocessing as input (6), we can improve both keyword and NDCG evaluation scores significantly.

Our proposed methods (7-9) with enabled multiword keyphrases seem to better represent the content of fragments, as shown by the keyword judgements. Again, DRUID further improved keyword recognition scores, but it did not achieve a better NDCG score on speech transcripts. The best NDCG score using speech transcripts was obtained with our proposed method *without* using multiwords (10). We experimented with different values of c: 0.3, 0.5 and 0.7, which all lowered NDCG scores. On average, this translates to 2.16, 1.56 and 0.53 multiword terms per query respectively. The numbers are slightly lower if we use manual transcripts (1.9, 1.4 , 0.5).

We also evaluated our methods on manual transcriptions (11-15), see Table 2. Here the picture is different, as using DRUID can improve NDCG scores. However, only the highest cutoff factor of 0.7 (producing the smallest number of multiword candidates) yielded the best performance, suggesting that the number of added multiword candidates is an important parameter in the query generation. The scores on manual transcriptions can also be understood as the theoretically best possible scores for each method, assuming a perfect speech transcription system. If we compare them, we find that imperfect transcriptions have a high impact on system performance for all methods, as NDCGs are considerably higher with manual transcripts. If we correlate WER with our method in (10), we only observe a weak negative correlation of -0.193. If we use multiword terms the negative correlation is higher with a coefficient of -0.293. The comparison system from Habibi and Popescu-Belis in (6) has the lowest negative correlation of -0.118 and it does not seem to gain as much in the NDCG evaluation on perfect transcriptions as our system.

4.3 Error Analysis

If we look at fragments individually and compare our method (10) to Habibi and Popescu-Belis (2015), we find that in 15 transcription fragments their system has a higher NDCG score and in 14 our system scores higher, with one equal score. On average, in cases where our system scored higher, WER was 14.8%, and where it scored lower WER was 16.8%. For example, for all 3 fragments of the talk "Kids Science" by Cesar Harada[19], where the accent of the speaker deteriorates WER to 33.4%, our system returns much more irrelevant documents. Our average NDCG for the talk is 0.180, while Habibi and Popescu-Belis' system scores 0.454. Among the articles found by our system are "National School Lunch Program", "Arctic Ocean", "Fresh water", "Water", "Coal preparation plant", "Coal mining", "Plant" with most of the articles being irrelevant to the talk.

Word errors and resulting erroneous search terms are responsible for most irrelevant documents, e.g. "in coal power plant" appears in the transcript instead of "nuclear power plant". On the other hand, in this example Habibi and Popescu-Belis' system finds better matching articles, like "Microscope", "Light Microscope", "Mangrove" , "Fresh water", "River delta", "Fishing" which can be attributed to finding the keyword "microscope" and otherwise picking simpler keywords like "ocean", "sea", "fishing" and "river", which our system entirely misses. This changes when we run the systems on the manually transcribed texts, as e.g. our system with enabled multiword terms (9) then finds "nuclear power plant", which helps to retrieve very relevant documents ("Nuclear reaction", "Nuclear chemistry", "Nuclear power plants").

Moreover, if we enable the use of multiword terms in our method with c=0.7, we observe that NDCG was improved by the keyphrase enabled method in 9 cases, but also decreased in 11, with the other 10 transcripts remaining unchanged. If WER is poor, the keyphrase enabled methods do not seem to contribute to improving NDCG performance and tend to lower it. E.g. in the 5 transcripts with the

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

[19] http://www.ted.com/talks/cesar_harada_how_i_teach_kids_to_love_science

highest WER (19.8-40.9%, average: 26.4%), 3 scores are lowered and 2 unchanged. If we group all cases where the NDCG performance drops, we observe an average WER of 16.9% vs. 12.3% for the cases where they help to improve the NDCG score (average of all transcripts is 15.6%). This further suggests that a query generation with multiword terms helps more in cases where word error rates are low.

Interestingly, in 5 out of 30 transcripts, no multiword terms are found with c=0.7 but NDCG values were still slightly lower in all cases compared to our single word method. While the set of terms in all queries were nearly unchanged, their ranking was affected. This might be attributed to how we build IDF and Word2Vec models: multiwords are opaque units in the models. This can change the dense vectors and IDF values for the constituents of multiwords compared to training on single words and thus affect ranking scores. However, in some of the automatic transcriptions, only constituents of the correct multiwords can be found because of transcription errors, so that our method has to rank the constituent instead of the full multiword.

5 Conclusion

We presented *Ambient Search*, an approach that can show and retrieve relevant documents for speech streams. Our current prototype uses Wikipedia pages, as this provides a large document collection for testing purposes with an universal coverage of different topics.

Our method compares favorably over previous methods of topic discovery and keyword extraction in speech transcriptions. We explored the use of multiword terms as keyphrases, alongside single word terms. Our proposed extraction method using Word2Vec (CBOW) embeddings and TF-IDF is fairly simple to implement and can also be adapted quickly to other languages as it does not need any labelled training data. The only barrier of entry can be the availability of a speech recognition system in the target language.

We have started first efforts to build open source speech recognition models for German in (Radeck-Arneth et al., 2015) and have plans to support this language in future prototypes of *Ambient Search*. These speech models target distant speech recognition and could help to apply Ambient Search to more challenging situations in the future, e.g. distant conversational speech.

We also plan to evaluate a more dynamic approach to query generation, where the number of terms is dynamically chosen and not simply capped at a maximum number of term candidates after ranking. As the proposed use of multiword terms seems to be somewhat dependent on the quality of the transcription, it might also make sense to include likelihood information of the speech recognition system. Our evaluation on manual transcriptions also suggests that there is quite a large headroom for our system to benefit from any future reductions in WER of the online speech recognition component.

For actual live deployment and usage in discussions, lectures or business meetings, confidential information can be present in the speech streams. A privacy aspect has already been addressed by *Ambient Search*: the speech recognition is not carried out "in the cloud" and can be deployed on one's own infrastructure. Similarly, an offline version of the Simple English Wikipedia and a corresponding search index is used to retrieve and find articles. It can be entirely circumvented that personal information is ever transmitted through the internet – a vital aspect for the acceptance of such an application.

We have published the source code of *Ambient Search* under a permissive license on Github[20], along with all pretrained models, a demonstration video, 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.

[20] https://github.com/bmilde/ambientsearch

References

C. Biemann and M. Riedl. 2013. Text: Now in 2D! A Framework for Lexical Expansion with Contextual Similarity. *Journal of Language Modelling*, 1(1):55–95.

C. Biemann, K. Böhm, G. Heyer, and R. Melz. 2004. SemanticTalk: Software for Visualizing Brainstorming Sessions and Thematic Concept Trails on Document Collections. In *Proc. ECML/PKDD*, pages 534–536, Pisa, Italy.

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.

Z. Liu, W. Huang, Y. Zheng, and M. Sun. 2010. Automatic Keyphrase Extraction via Topic Decomposition. In *Proc. EMNLP*, pages 366–376, Cambridge, MA, USA.

F. Metze, P. Gieselmann, H. Holzapfel, T. Kluge, I. Rogina, A. Waibel, M. Wölfel, J. Crowley, P. Reignier, D. Vaufreydaz, F. Bérard, B. Cohen, J. Coutaz, S. Rouillard, V. Arranz, M. Bertrán, and H. Rodriguez. 2005. The FAME Interactive Space. In *Proc. International Workshop on Machine Learning for Multimodal Interaction*, pages 126–137, Edinburgh, United Kingdom.

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.

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.

S. Radeck-Arneth, B. Milde, A. Lange, E. Gouvêa, S. Radomski, M. Mühlhäuser, and C. Biemann. 2015. Open Source German Distant Speech Recognition: Corpus and Acoustic Model. In *Proc. TSD*, pages 480–488, Pilsen, Czech Republic.

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.

B. Rhodes. 1997. The Wearable Remembrance Agent: A System for Augmented Memory. *Personal and Ubiquitous Computing*, 1(4):218–224.

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.

D. Traum, P. Aggarwal, R. Artstein, S. Foutz, J. Gerten, A. Katsamanis, A. Leuski, D. Noren, and W. Swartout. 2012. Ada and Grace: Direct Interaction with Museum Visitors. In *Proc. IVA*, pages 245–251, Santa Cruz, CA, USA.

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.