MoM: Minutes of Meeting Bot

Benjamin Milde[1,2], *Tim Fischer*[1,2], *Steffen Remus*[1], *Chris Biemann*[1]

[1] Language Technology Group, Dept. of Informatics, Universität Hamburg
[2] Hamburger Informatik Technologie-Center e.V. (HITeC e.V.)
{lastname,5fischer}@informatik.uni-hamburg.de

Abstract

We present MoM (Minutes of Meeting) bot, an automatic meeting transcription system with real-time recognition, summarization and visualization capabilities. MoM works without any cloud processing and does not require a network connection. Every processing step is local, even its speech recognition component, to address privacy concerns of meetings. MoM can be used to assisted writing a (summarized) protocol of a meeting, but may also help the hearing-impaired to follow a discussion. We address meeting-related issues, e.g. local vocabulary of an organization or company with active learning of G2P models and custom vocabulary extensions.

Index Terms: speech recognition, meeting, speech, human-computer interaction, summarization

1. Introduction

Meeting assistants offering automatic transcriptions and summarization can improve productivity and meeting participation. Real-time subtitles can also make a meeting accessible to persons with hearing limitations.

In [1] a meeting transcription system was developed for in-house meetings with out-of-vocabulary-words (OOVs) being the major issue. In [2], a dynamic dictionary adaption approach was proposed to address this issue. The CALO Meeting Assistant was designed for distributed meeting capture, e.g. a multiparty telephone conference [3]. A system for live subtitling, and real-time translation of the transcriptions for conference presentations was presented in [4].

In MoM, we generate automatic transcriptions with off-the-shelf models for the speech recognition toolkit Kaldi [5]. The German model is based on the kaldi-tuda-de project [6], while the English one was trained on TED-LIUM v3 [7]. We implemented a custom decoding server called kaldi-model-server using the PyKaldi wrapper library [8]. We integrated support for two languages (English and German), but extending MoM to other languages is straightforward. We interface the audio hardware directly and do online decoding in real time, our software can be installed locally. The custom decoding software can record multiple audio streams in parallel, one for each speaker. We compute volume for each channel and select the channel with the currently highest volume for decoding.

2. Architecture

Figure 1 illustrates our architecture, consisting of several microservices that communicate through events (messages in a Redis server channel) or API calls. We created kaldi-model-server using PyKaldi [8]. Our custom nnet3 decoding server is meant for live decoding with Kaldi directly from multiple input microphones. Our server records the input channel from multi channel audio input and decodes the most active audio channel. In our hardware setup, each speaker is given a (wireless) clip-on microphone, which also identifies the speaker. Using the online end pointing module in Kaldi, utterances are segmented on the fly with a rule-based system that takes speech pauses into account. At a hypothesized end point, we compute word confidences using Minimum Bayes Risk (MBR) decoding [9] given the decoding lattice of the current utterance. Kaldi model server then broadcasts partial and complete utterances with speaker information to the event server. The event server directly communicates with our VUE browser app, the current hypothesis is constantly updated through server side events that are send to the browser app that displays the current hypothesis. The browser app uses a keyword microservice to extract relevant keywords on the fly. Once the meeting is finished, users can generate a PDF with a short summary and full transcription of the meeting, where the browser apps interacts with the summarization server. The resulting PDF is generated directly in the browser with javascript.

3. Vocabulary

We created the speech-lex-edit project[1], to extend phonetic lexicon entries semi-automatically using an active learning approach. Multiple pronunciation variants are suggested by a pretrained Sequitur [10] grapheme-to-phoneme (G2P) model and can then be manually approved and edited. Audio feedback is provided by synthesizing the phonetic entries with a TTS engine[2]. We added 14,268 additional lexicon entries for German.

4. Interface

Figure 2 shows an example screenshot of our system. A timeline with separated boxes displays the transcribed text and conversation history in real time (1). We are showing the current hypothesis of the online model here as well. New boxes appear for each speaker change. Keywords are highlighted after each completed utterance. The text shown in the timeline can be shown in full, or in a reduced form (2), then only displaying keywords and some words of context around them. The system can be paused, resumed or stopped at any time (3).

In (4), we show a word cloud generated from discovered keywords. Similar words are clustered and displayed in the same color. Newer keywords are weighted higher than old ones, so that the word cloud can gradually visualize changes in topics. This visualization is fluid, i.e. if one scrolls back in the conversation, then the word cloud displays keywords that focus on the displayed time interval. The interface also offers several additional features: An agenda can be imported from emails, i.e. by adding and parsing standard ICS calendar files. There is also an integrated transcript editor, so that transcription errors

[1] https://github.com/uhh-lt/speech-lex-edit
[2] http://mary.dfki.de/

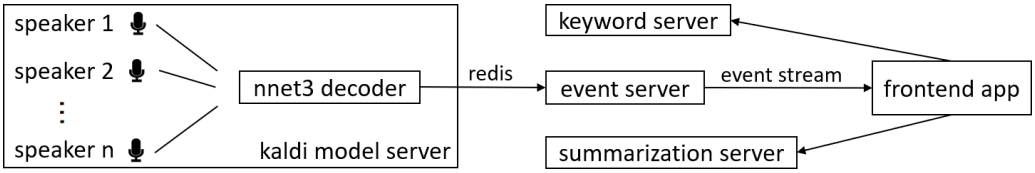

Figure 1: *Architecture of MoM bot. The backend consists of a model server and various microservices, while the frontend is a* VUE *browser app.*

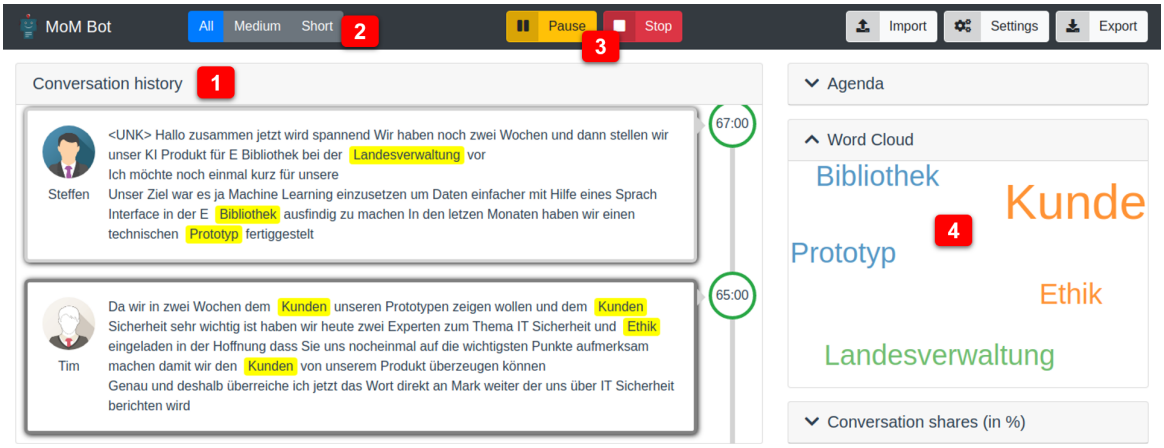

Figure 2: *Example screenshot of the meeting bot system.*

can be manually fixed. This editor also has visual aids showing confidence scores from the Kaldi model, as computed by MBR. It is also possible to directly export the conversation history to a PDF in the browser frontend, which additionally contains an automatic summarization of the spoken content per agenda point.

5. Summarization

We apply TextRank [11] on the sentence-level to generate an extractive summarization. First, a fully-connected graph is build where the vertices represent the sentences. Then, the edges are weighted based on their similarity which is computed using cosine similarity of sentence embeddings. Finally, the top N sentences after running the PageRank algorithm on this graph are selected as the summary. Additionally, we integrated abstractive summarization models with BertSumAbs by [12] for English and German (using SwissText [13] data).

6. Conclusions

We implemented a fully functioning open source meeting transcription and summarization app. This application works in real-time and has interactive elements, that visualize a meeting. Everything can be run locally on a single computer, ensuring that there are no privacy issues regarding the unwanted sharing of recordings. MoM bot is open-source and is freely available[3]. Since the Kaldi model server can be accessed via a micro service, future plans include to build a centralized version for online meetings. **Acknowledgments.** We thank Deutsche Telekom AG for supporting the development of MoM bot.

7. References

[1] N. Morgan, D. Baron, J. Edwards, D. Ellis, D. Gelbart, A. Janin, T. Pfau, E. Shriberg, and A. Stolcke, "The meeting project at ICSI," in *HLT*, 2001.

[2] H. Yu, T. Tomokiyo, Z. Wang, and A. Waibel, "New developments in automatic meeting transcription," in *ICSLP*, 2000, pp. 310–313.

[3] G. Tur, A. Stolcke, L. Voss, J. Dowding, B. Favre, R. Fernández, M. Frampton, M. Frandsen, C. Frederickson, M. Graciarena *et al.*, "The CALO meeting speech recognition and understanding system," in *IEEE-SLT Workshop*, 2008, pp. 69–72.

[4] O. Bojar, D. Macháček, S. Sagar, O. Smrz, J. Kratochvíl, P. Polák, E. Ansari, M. Mahmoudi, R. Kumar *et al.*, "ELITR multilingual live subtitling: Demo and strategy," in *EACL*, 2021, pp. 271–277.

[5] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The Kaldi speech recognition toolkit," in *ASRU*, 2011.

[6] B. Milde and A. Köhn, "Open source automatic speech recognition for German," in *ITG*, 2018, pp. 251–255.

[7] F. Hernandez, V. Nguyen, S. Ghannay, N. Tomashenko, and Y. Estève, "TED-LIUM 3: twice as much data and corpus repartition for experiments on speaker adaptation," in *Proc. SPECOM*, 2018, pp. 198–208.

[8] D. Can, V. R. Martinez, P. Papadopoulos, and S. S. Narayanan, "PyKaldi: A python wrapper for Kaldi," in *ICASSP*, 2018, pp. 5889–5893.

[9] H. Xu, D. Povey, L. Mangu, and J. Zhu, "Minimum Bayes risk decoding and system combination based on a recursion for edit distance," *CS&L*, vol. 25, no. 4, pp. 802–828, 2011.

[10] M. Bisani and H. Ney, "Joint-sequence Models for Grapheme-to-phoneme Conversion," *Speech Comm.*, vol. 50, no. 5, pp. 434–451, 2008.

[11] R. Mihalcea and P. Tarau, "TextRank: Bringing order into text," in *EMNLP*, 2004, pp. 404–411.

[12] Y. Liu and M. Lapata, "Text summarization with pretrained encoders," in *EMNLP-IJCNLP*, 2019, pp. 3730–3740.

[13] D. Frefel, M. Vogel, and F. Märki, "2nd German text summarization challenge," in *Proc. of SwissText/KONVENS*, ser. CEUR Workshop Proceedings, S. Ebling, D. Tuggener, M. Hürlimann, M. Cieliebak, and M. Volk, Eds., 2020.

[3] https://github.com/uhh-lt/MeetingBot