Implementing a speech-to-text pipeline on the MICO platform

Henrik Björklund, Johanna Björklund, Adam Dahlgren, Yonas Demeke

Dept. Computing Science, Umeå University, Sweden

henrikb@cs.umu.se, johanna@cs.umu.se, dali@cs.umu.se, yonas@cs.umu.se,

Abstract

MICO is an open-source platform for cross-media analysis, querying, and recommendation. It is the major outcome of the European research project Media in Context, and has been contributed to by academic and industrial partners from Germany, Austria, Sweden, Italy, and the UK. A central idea is to group sets of related media objects into multimodal content items, and to process and store these as logical units. The platform is designed to be easy to extend and adapt, and this makes it a useful building block for a diverse set of multimedia applications. To promote the platform and demonstrate its potential, we describe our work on a Kaldi-based speech-recognition pipeline.

Index Terms: cross-media analysis, technological platforms, speech recognition, diarization

1. Introduction

In the early days of the Internet, the content of a site was provided as text, possibly together with a few images. Today, the situation is different. Much of what would previously have been written is now communicated through images, audio, video, and other forms of rich media. Every minute of the day, Twitter users compose 350,000 tweets, Snapchat users capture 285,000 images, and Youtube users record 300 hours of video. This is a trend that continues to grow: By 2018, it is expected that 80% of all IP traffic will be made up of video [1]. This development poses new challenges for Internet search and recommendations. In particular, data indexing must be done with respect to a cross-media context, and combine information derived from different modalities.

A recent effort in this direction is the research project MICO – Media in Context.[1] The aim is to provide an efficient and affordable data-analysis platform for online media providers. A central idea is to view a related set of media assets as a single multimodal item. These composite objects are referred to as *content items* in MICO. This could for instance be a video with its audio track and subtitles, or a web page with its textual data, images, and style sheet. When a content item is analyzed, the process extracts and fuses information from the available modalities. By modeling the cross-media context in this way, there is more data to base decisions on, and a greater variety of features to use for classification.

As a simple example, consider the problem of using face recognition to index a digital movie archive. If the image analysis component estimates an equal probability of a subject being Judi Dench or Cecila Imre, the subtitles contains the words 'Dashwood' and 'Greenslade' (the latter is the name of Dench's character), and the speaker recognition component outputs Maggie Smith as the most likely speaker, followed in turn by Dench and Imre, then the integrated analysis would do well to tag the frame as featuring Judi Dench.

When a content item is ingested into the MICO platform for analysis, a semi-automated service orchestrator computes an appropriate processing pipeline, based on the modalities of the content item and user-provided configuration files. The media item is then passed through the pipeline, which is made up of *extractors*. These are analysis components such as face detection, face recognition, and mood detection. The extractors in a pipeline are applied in succession, so that the output of one extractor is the input of the next. Both intermediate and final results are stored as linked data or in a binary format, and can be used for querying and for generating recommendations.

The development of MICO has been oriented around a set of concrete usecases. Some of the usecases have been provided by the citizen-science portal Zooniverse, others by the media integrators InsideOut10 and concern their news application for user-contributed content.[2] The data sets are comprised of text, images, video, and audio, and language analysis is a recurring theme. The usecase partners wish, for instance, to recognise names of animal species or geographical locations, to transcribe video recordings, and to search online conversations based on the sentiments expressed.

The need to analyse and search spoken language is a common denominator for many of these usecases. For this reason, speech recognition is a key technological enabler for the MICO platform. It often appears together with noise reduction and diarization at the beginning of longer pipelines. The transcribed text is then processed further by passing it through, for example, parsers, named entity recognizers, and sentiment analyzers. When a page contains an uploaded video with little or no explanatory text surrounding it, this, together with image analysis, is the only way to get to the actual topic and thereby making the page searchable.

The MICO usecases have been such that precision is more important than recall. The goal is typically to find "relevant" content, and effectiveness is measured against manually searching the material. If the system only delivers a subset of the relevant matches, then the user can either be satisfied with what was retrieved, in which case everything is well, or he or she can fall back on manual search, and will eventually find the overseen content items. It is worse if the system returns a large number of false positives, in which case the user simply stops using the search functionality. In terms of speed, it is desirable that computations on streamed media should run in real-time, since it is required by many of the intended applications.

In this paper, we describe the extractor pipeline for speech recognition offered by the MICO platform. The pipeline inputs content items with audio or video parts, performs diarization, speech recognition, and finally produces metadata in the form

[1] See `www.mico-project.eu` for a more complete presentation.

[2] See `www.zooniverse.org` and `insideout.io`

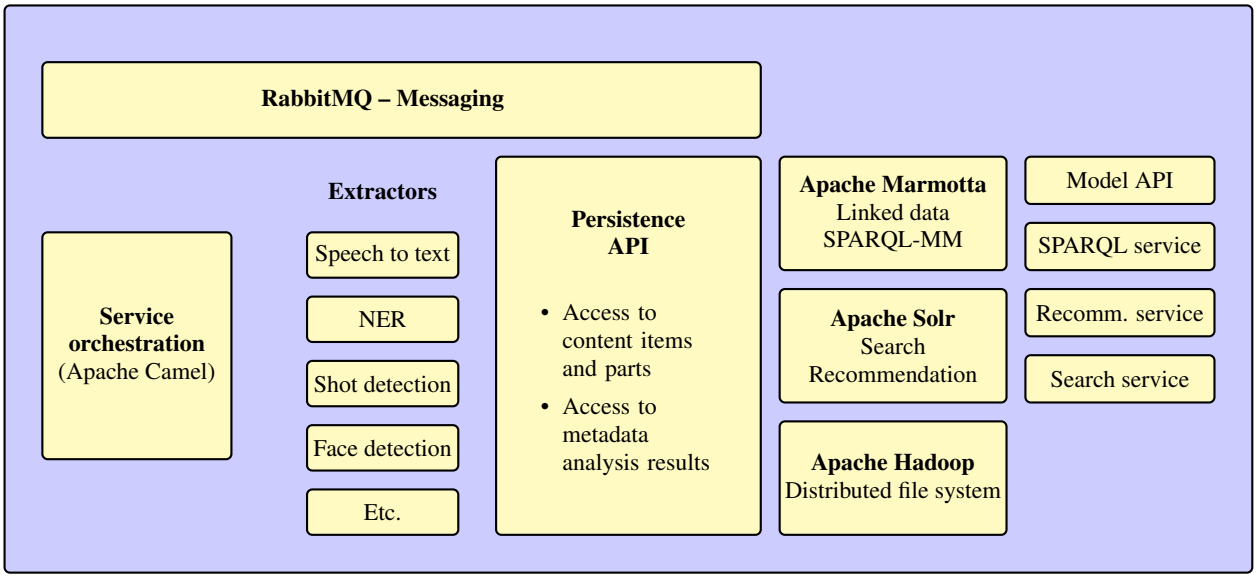

Figure 1: The MICO platform architecture.

of RDF annotations. We present the challenges involved in the creation of the pipeline, as well as the results and plans for further improvements. Our aim is to create interest for cross-media analysis within the speech processing community, and to promote the uptake of the MICO platform. The platform, as well as the majority of its associated metadata extractors, are publicly available as open source code.

1.1. Related work

There is a rich body of literature on cross-media analysis, including speech recognition. We do not attempt a complete survey, but rather reference some work that puts the MICO project and its speech-to-text pipeline into perspective.

A good place to start when it comes to the opportunities and challenges of cross-media analysis is the collection of papers from the 2008 AAAI fall session titled "Multimedia information extraction" [2]. Apart from contributions that present recent results in the area, it contains a number of "roadmap" articles, that take stock of the state of the art and contemplate the future.

There are also a large number of papers that deal with "multimodal fusion", that is, principles and techniques for combining the analysis results obtained from different media types. A survey of data fusion can be found in [3]. An example of a particular method at work is given in the article by Perperis et al. on violence detection in movies that uses cues from both the audio and the video track [4].

Many articles also describe concrete cross-media analysis tasks that include the use of speech recognition. Such efforts include (i) speaker identification in TV broadcasts based on face recognition, voice recognition, speech recognition, and available metadata [5], (ii) solving various video indexing tasks, such as story segmentation and concept detection, using speech recognition together with image analysis [6], and (iii) creating an on-demand system for video lectures, where user can search for keywords in the audio track [7].

A number of larger systems have also been built that do cross-media analysis for certain settings. Here, we mention the system presented by Mezaris et al. for multimodal semantic analysis of audio-visual news content [8].

What makes MICO stand out against previous work is that it is a general platform for cross-media analysis, where anyone can create their own analysis tool by providing the metadata extractors for the individual media types (or using those that have already been developed) and setting up the rules for how the metadata from the various parts should be combined and used in search and recommendation.

2. The MICO platform

The MICO platform follows the guiding principles of service oriented architectures. In particular, it allows independent components to communicate and work together without human intervention. An overview of the architecture is given in Figure 1.

The platform has three core components: (1) The metadata extractor pipelines and the service orchestration component, which contains a number of analysis services (referred to as 'extractors') to mine information from different types of media such as text, image, audio and video. Any number of extractors can be registered with the platform at a given time, and their interaction is coordinated by the service orchestration component. (2) The persistence API, which provides access to the binary data and the metadata storage back ends. (3) The querying and recommendation component, which provides full-text search and recommendation facilities.

The choices in the design of the platform were guided by the requirement analysis and the continous dialogue with the usecase partners. We discuss the main requirements briefly. The contents items, which are intended to be analyzed by the platform, are characterized by their large size and composite type. The extractors are required to run independently, but they need to be orchestrated during the analysis of the content items. The platform is also required to support metadata publishing and querying with RDF and SPARQL.

The platform recieves its input data in the form of content items. As previously mentioned, a content item can represent a web page and contain a number of parts, which can be texts, im-

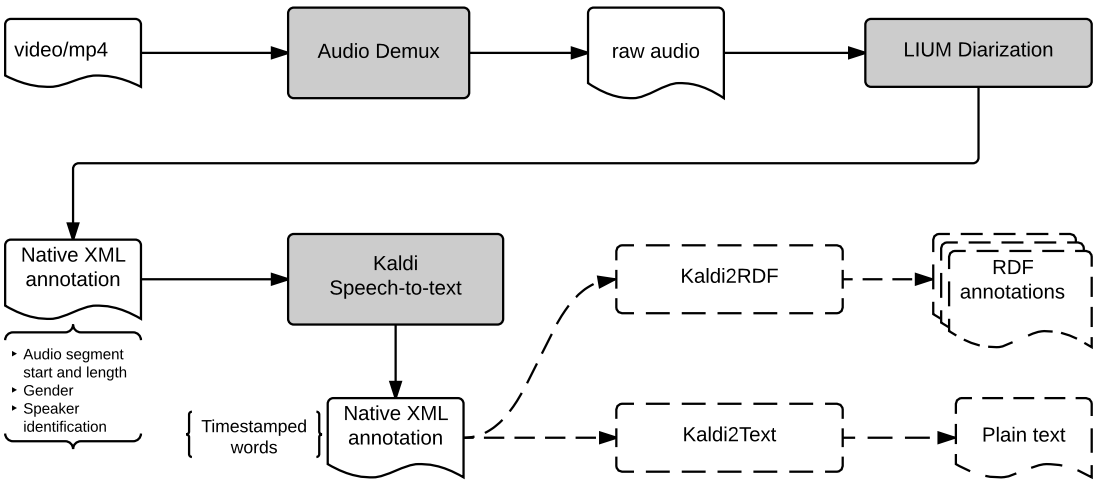

Figure 2: Speech-to-text pipeline within the MICO platform

ages, or video clips. Depending on the composition of the content item, the service orchestration component creates an execution plan and starts calling extractors to analyze the parts. The metadata thus obtained is either stored in binary format, using the Apache Hadoop distributed file system[3] or in RDF format, using Apache Marmotta[4] and a linked data platform. The latter option is used whenever possible, allowing the data to be searched using SPARQL-MM.[5]

Several open-source libraries have been used to build the platform. For example, the extractors working with text and speech recognition have been implemented using the Stanford CoreNLP[6] and the Kaldi library,[7] respectively. The inital version of the orchestration component was implemented in Java but the final version will be based on Apache Camel.[8] The Rabbit MQ messaging service has been used as a communication infrastructure to register extractors and support the overall interaction between them.[9] Apache Marmotta has been implemented as a Linked Data publishing component, which provides a metadata storage for the MICO platform. The analysis results for the input content item is accessed by using the URI. The metadata is represented by an RDF graph and stored with its URI. The SPARQL endpoint is used for querying and updating the content item metadata, which is available on the platform as a web service. Apache Solr[10] indexes the analysed content items for semantic and full-text search, and also provides content-based recommendations.

The MICO platform is developed iteratively. The most recent version, v.1.2.8, has just been released and is available in two ways: as a virtual machine image and as Debian packages. Its source code is also publicly available on Bitbucket.[11]

[3] http://hadoop.apache.org
[4] http://marmotta.apache.org
[5] http://marmotta.apache.org/kiwi/sparql-mm.html
[6] http://stanfordnlp.github.io/CoreNLP/
[7] http://kaldi-asr.org
[8] http://camel.apache.org
[9] https://www.rabbitmq.com
[10] http://lucene.apache.org/solr/
[11] http://code.mico-project.eu/platform

3. The speech-to-text pipeline

The Speech-to-text pipeline performs speech recognition on audio and video media, and outputs a text transcription. This analysis is composed of three steps; (1) audio demultiplexing, (2) speaker diarization, and (3) speech transcription. There is also a fourth optional step to produce metadata annotations in RDF. The complete pipeline is shown in Figure 2.

3.1. Audio demux

The reason for the first step, audio demultiplexing, is to facilitate video analysis and to downsample the audio signal to match the sample rate used in the transcription step. The sample rate used here is based on the settings used when training the model for 8 kHz 'telephone speech'. This model is provided together with the Kaldi software and the audio format is required by the usecases on user-contributed news content.

3.2. Diarization

The second step in the pipeline performs speaker diarization. The main body of work is done by LIUM, and results in segmentation information along with gender classification and speaker partitioning. The acoustic features needed by LIUM, such as MFCC parameters, are calculated by the open-source toolkit Sphinx4. LIUM uses techniques such as Cross-Likelihood Ratio and Hierarchical Agglomerative clustering together with Bayesian Information Criterions to detect and join segments of speech by the same speaker. Gaussian mixture models provide gender-identification and are trained beforehand. The segments given by combining these techniques are then cut so as not to exceed 20 seconds in length. Apart from providing speaker metadata, the transcription step can use this as segmentation data for more efficient analysis. In order to perform accurate transcriptions it is necessary to base the segmentation on speech patterns. Otherwise, the segmentation might make cuts in the middle of words, which would make the resulting snippets unrecognisable and decrease the overall accuracy of the results [9, 10, 11].

3.3. Transcription

The final extractor in the pipeline is responsible for transcribing the speech, and is based on the speech recognition toolkit Kaldi. It was chosen over alternative open source toolkits such as CMU Sphinx based on performance and accuracy [12]. Kaldi is also written in C++ which aligns well with the rest of the MICO platform. The language model for US English provided with the Kaldi toolkit has been the basis for experiments within the platform. Kaldi uses MFCC features to perform online decoding based on neural nets and Gaussian mixture models [13]. Online decoding was chosen although it implies a lower word accuracy, because it is more efficient and makes it possible to decode input audio in real time [14, 15, 16].

The extractor segments the incoming audio stream based on the diarization information and performs feature extraction and decoding. It then extracts words and timings to produce timestamped transcription results. The pipeline produces a transcription in XML format, but there are auxiliary components which translate the transcript to plain text or RDF, that can be used to simplify processing in downstream extractors.

4. Challenges and solutions

This section discusses the challenges encountered during the implementation work, and how they were met. Although some are specific to the speech-to-text pipeline, many apply to other processing pipelines as well.

The MICO platform currently only supports English and one challenge, which pertains to all language-technological extractors, is to incorporate dynamic multi-language support. Some of the usecases require speech recognition for Italian and Arabic, but finding good language models has proved hard. We tried to train our own models as well as convert existing models for CMU Sphinx, but the transcription accuracy was not satisfactory. In the first case, we believe that the data set used for training was simply too small, while in the latter case, the causes are less certain.

Another problem concerns the interfacing of speech recognition with different types of natural-language processing (NLP), for instance named entity recognition. The transcription provided by Kaldi includes metadata that is useful for NLP extractors, but the NLP libraries used in MICO require plain text as input. In the case of named entity recognition, indicators of context (e.g., subdivision into paragraphs) could be useful and can be inferred from timestamps. However, without modifying the NLP libraries we cannot convey this information. This touches on the larger problem of how to work with metadata annotation in such a way that the information is available and useful for downstream extractors. This problem is currently solved with RDF and SPARQL (outlined in Section 2) in combination with auxiliary on-demand extractors (described above). In the first year of the project, efforts were made towards a shared ontology, but the development team eventually decided to use standard MIME types, largely because of time constraints.

Many of the remaining problems with the speech-to-text pipeline concern performance. Early work aimed for a tight integration between the speech-recognition pipeline and the platform, but this led to issues with memory consumption and lengthy execution times. Speaker diarization was introduced to decrease memory consumption, and had the additional advantage of producing useful metadata. Our benchmarks suggest that diarization also led to an improvement in word error rates, but further evaluation is needed to confirm this.

The execution times were initially around 2.5 times the length of the audio for continuous speech. These have since been improved, but we continue to experiment with the internal settings of Kaldi, following the methodology of [16]. Another solution is to utilize parallelism and divide the audio into blocks that are analyzed side-by-side. This division can, for example, be based on the segmentation data given by the diarization. Parallelization is likely to be good for throughput, but may cause problems further down the pipeline. One example is the question of how to re-combine the produced metadata, and how to solve ambiguities and conflicts. A third option is to adapt the platform to streaming media, which could allow for greater use of the online decoding capabilities of the pipeline. As the platform currently saves files in Hadoop before further analysis, files will inherently take more time to process in proportion to the size of the file. Providing support for streamed media would allow extractors to begin analysis earlier, comparable to the benefits from parallelization.

Although the word error rate is important, relatively little effort has gone into this direction. The reason is simply that metadata modelling, platform integration, and performance have taken priority. For a commercial purposes, more mature and/or more specialized language and acoustic models should be used, as this alone can make a big difference [17, 18].

5. Conclusion and future work

The final version of the MICO platform and its extractors will be released during the spring of 2016. The platform itself and the majority of the extractors are licensed as open-source under the business-friendly Apache 2 license. There is also a smaller number of complementary, closed-source, extractors provided by Fraunhofer GmbH. The platform is likely to become part of the Apache Stanbol initiative for semantic content management, which has an active and engaged user community.

The development work on the platform will continue after the official close of the research project. There are several types of audio and speech analysis that would be useful to integrate. At the top of the list is noise reduction, which we would like to include as a preprocessing step for speech recognition since background noise is known to have a degrading effect on the transcription quality [19]. For the same reason, it would be beneficial to include speech/music discrimination [20]. Ideally this would simply cancel out any background music, but could in a less advanced form indicate what parts of the speech are audible enough to transcribe.

Other interesting additions are emotion recognition [21] and speaker recognition [22]. It would be immensely useful for publishers to be able to search their digital archives for, e.g., some particular presidential candidate when he or she sounds nervous. Another use for speaker recognition is to generate transcripts of, e.g., business meetings and public hearings.

6. Acknowledgements

We are thankful to the MICO team and its partner organisations. The project has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration grant agreement number 610480.

7. References

[1] Cisco Visual Networking Index, "The Zettabyte era —- trends and analysis," Cisco Systems, Inc., Tech. Rep., 2014.

[2] M. Maybury and S. Walter, "Multimedia information extraction. Papers from the 2008 AAAI fall session," AAAI Press, Tech. Rep. FS-08-05, 2008.

[3] P. K. Atrey, M. Anwar Hossain, A. El Saddik, and M. S. Kankanhalli, "Multimodal fusion for multimedia analysis: a survey," *Multimedia Systems*, vol. 2010, no. 16, pp. 345–379, 2010.

[4] T. Perperis, T. Giannakopoulos, A. Makris, D. I. Kosmopoulos, S. Tsekeridou, S. J. Perantonis, and S. Theodoridis, "Multimodal and ontology-based fusion approaches of audio and visual processing for violence detection in movies," *Expert Systems with Applications*, vol. 38, no. 11, pp. 14 102–14 116, 2011.

[5] H. Bredin, A. Roy, V.-B. Le, and C. Barras, "Person instance graphs for mono-, cross- and multi-modal person recognition in multimedia data: application to speaker identification in tv broadcast," *International Journal of Multimedia Information Retrieval*, vol. 2014, no. 3, pp. 161–175, 2014.

[6] S.-F. Chang, R. Manmatha, and T.-S. Chua, "Combining text and audio-visible features in video indexing," in *Acoustics, Speech, and Signal Processing*, 2005, pp. 1005–1008.

[7] A. Fujii, K. Itou, T. Akiba, and T. Ishikawa, "A cross-media retrieval system for lecture videos," *CoRR*, 2003.

[8] V. Mezaris, S. Gidaros, G. T. Papadopoulos, W. Kasper, J. Steffen, R. Ordelman, M. Huijbregts, F. de Jong, I. Kompatsiaris, and M. G. Strintzis, "A system for the semantic multimodal analysis of news audio-visual content," *EURASIP Journal on Advances in Signal Processing*, vol. 2010, 2010.

[9] S. Meignier and T. Merlin, "Lium spkdiarization: an open source toolkit for diarization," in *in CMU SPUD Workshop*, 2010.

[10] M. Rouvier, G. Dupuy, P. Gay, E. Khoury, T. Merlin, and S. Meignier, "An open-source state-of-the-art toolbox for broadcast news diarization," in *Interspeech*, Aug. 2013.

[11] X. Zhu, C. Barras, S. Meignier, and J.-L. Gauvain, "Improved speaker diarization using speaker identification," Online article, March 2016.

[12] C. Gaida, P. Lange, R. Petrick, P. Proba, A. Malatawy, and D. Suendermann-Oeft, "Comparing open-source speech recognition toolkits," 2014.

[13] 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, "The Kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*, 2011.

[14] C. Breslin, K. Chin, M. J. Gales, K. Knill *et al.*, "Integrated online speaker clustering and adaptation." in *Interspeech*, 2011, pp. 1085–1088.

[15] A. V. Ivanov, V. Ramanarayanan, D. Suendermann-Oeft, M. Lopez, K. Evanini, and J. Tau, "Automated speech recognition technology for dialogue interaction with non-native interlocutors," in *SIGDIAL 2015*, 2015, pp. 134–138.

[16] O. Plátek and Jurčíček, "Free on-line speech recogniser based on Kaldi asr toolkit producing word posterior lattices," in *SIGDIAL 2014*, 2014, p. 108112.

[17] C. Chelba, D. Bikel, M. Shugrina, P. Nguyen, and S. Kumar, "Large scale language modeling in automatic speech recognition," Google, Tech. Rep., 2012.

[18] F. Ehsani and E. Knodt, "Speech technology in computer-aided language learning: Strengths and limitations of a new call paradigm," *Language Learning & Technology*, vol. 2, no. 1, pp. 45–60, 1998.

[19] B. Juang, "Speech recognition in adverse environments," *Computer Speech & Language*, vol. 5, no. 3, pp. 275 – 294, 1991.

[20] A. Gallardo-Antolín and J. M. Montero, "Histogram equalization-based features for speech, music, and song discrimination," *Signal Processing Letters, IEEE*, vol. 17, no. 7, pp. 659–662, 2010.

[21] S. G. Koolagudi and K. S. Rao, "Emotion recognition from speech: a review," *International Journal of Speech Technology*, vol. 15, no. 2, pp. 99–117, 2012.

[22] T. Kinnunen and H. Li, "An overview of text-independent speaker recognition: From features to supervectors," *Speech Communication*, vol. 52, no. 1, pp. 12 – 40, 2010.