January 2024

AUTOMATED MODEL-BASED REORGANIZATION OF SPEECH-TO-TEXT TRANSCRIPTS

Stephane Martin

Erwan Zerhouni

AUTOMATED MODEL-BASED REORGANIZATION OF SPEECH-TO-TEXT TRANSCRIPTS

AUTHORS:
Stephane Martin
Erwan Zerhouni

ABSTRACT

Techniques are presented herein that support the automated reorganization of a meeting transcript. According to the presented techniques, when a meeting is recorded it is first transcribed through an automated speech-to-text system. Then, the resulting raw document is decomposed into sections corresponding to different topics, the topics are reorganized into a more coherent and intelligible structure, and missing pieces of information may be identified and then added to the document. Such postprocessing increases the intelligibility and value of meeting transcriptions and ensures that they remain understandable and useful over the long term. Aspects of the presented techniques may leverage large language models (LLMs) and automatic speech recognition (ASR) capabilities.

DETAILED DESCRIPTION

The availability of powerful automatic speech recognition (ASR) systems makes it possible to efficiently and reliably transcribe professional meetings. However, because of the conversational nature of those meetings the resulting transcripts are often difficult to read and reutilize, especially when some time has passed following the transcription. For example, human discussions often jump from one topic to another, resulting in a transcript that is difficult to follow, that tends to be ill-organized and ill-ordered, and that is often repetitious and disjointed. Such content may also be lacking in those elements of contextual information that seem too obvious to the meeting participants to be worth mentioning. The absence of those elements will, however, make a resulting transcript less useful after some time, when the relevant information has been forgotten.

While text-searching tools can help a user find relevant parts in older transcripts, provided the user has an idea of what they are looking for, those tools cannot supply missing information. Moreover, even with the help of such tools the user still must invest some time and work to explore the documents on their own in order to identify interesting connections between the steps of a discussion.

An additional stage of editing, reorganization, annotation, and expansion could make the above-described transcripts truly valuable, but the effort that is required by those tasks is normally too time consuming and too costly to make it feasible in practice.

To address such a challenge, techniques are presented herein that support a system that leverages the recent development of large language models (LLMs) to perform the tasks of automated speech-to-text editing, post-processing, and structuring. While the models discussed in this document may sometimes make mistakes (which is especially possible in their early stages), and the quality of their output may not always compare with the quality of a human intervention, the gains that are afforded by the automation that is possible under the presented techniques will largely outweigh a few occasional errors.

Recordings that are intended to be released to a large public (for instance, podcasts and similar artifacts) may be edited manually, with the use of appropriate editing frameworks, because their intended use case justifies having a human editor expend the required amount of effort. But even in this case, automating the process (as is possible through the presented techniques), whether partially or in totality, will still translate into a considerable improvement in terms of speed and in the reduction of efforts required by human editors.

One use case, however, where automation (according to the presented techniques) will typically be advantageous comprises corporate and organization meetings, especially preparatory and informal meetings such as brainstorming sessions and the like, which are too frequent and too large in number for any organization to have the resources necessary to transcribe, reorganize, and manually edit even a small fraction of the resulting transcripts. The consequence is that the vast majority of those recordings, and their associated transcripts, quickly become useless and are effectively never employed, even though they contain valuable information.

As will be described and illustrated below, the presented techniques encompass seven interrelated components – an automated transcription module, a topic segmentation algorithm (TSegA), a segment structuring module, an audio structuring module, an information-supplementation module, a topic summarization module (TSumA), and a graphical user interface (GUI).

The first component of the presented techniques encompasses an automated speech-to-text transcription system. Such a system may transcribe oral meetings and discussions and produce written text. This component may rely on any of the existing technologies that have been devised for that purpose. Under one arrangement, the first component may take the form of an online system that provides a transcription while a discussion unfolds. Alternatively, this component may take the form of an offline system that may be applied to a recording sometime after a discussion has taken place.

The second component of the presented techniques encompasses a topic segmentation module (TSegA). This component proceeds by identifying the main articulations of a transcript (i.e., its main semantic inflections) and such a module may divide a transcript into a succession of segments (i.e., passages of texts discussing a single coherent topic or piece of subject matter). Among other things, the start and the end of each chapter may be marked with timestamps which refer to the corresponding moments of a recording. Additionally, the topic summarization module TSumA (see below) may provide, for each chapter, a header of its topic under the guise of a short phrase which summarizes the topic.

Topic segmentation (as described above) may be performed by any of the topic-segmentation or text-segmentation algorithms that are known in the literature. Alternatively, the second component may leverage the recent developments of large language models (LLMs). Those models have shown remarkable capabilities in a wide range of natural language processing (NLP) tasks, and they are especially suited to the above-described activity.

The third component of the presented techniques encompasses a segment (re-)structuring module. The function of this module is to the improve the raw, unordered sequence of segments by introducing a more meaningful structure. This can be done first by reordering the topics linearly into a new sequence that is more intelligible and useful than the original one, or, in a more sophisticated fashion, by discovering semantic connections between the segments which can then be presented to a user in the form of hyperlinks or any equivalent hypertextual devices. This second option allows a user to navigate through the enriched transcript in a non-linear, graph-like manner, skipping those segments that are deemed to be not relevant.

Under a first arrangement, after a transcript is segmented into chapters the segments may be reordered into a sequential, but more logical and intelligible, order by a topic-ordering algorithm. Such a reordering may be done by computing a measure of semantic distance between pairs of segments and by choosing the order that minimizes the sum of the distances between each pair of consecutive segments. Such a distance may be based on a semantic embedding of the content of the chapters.

Under a second arrangement, the reordering, also in a linear fashion, may be performed by providing a LLM with the list of chapter headers and an appropriate prompt. Such an arrangement reuses a frozen general-purpose LLM model and interrogates it with a specifically engineered prompt, which is associated with the list of chapters, to obtain a new list in a more appropriate order.

Under a third arrangement, the reordering may be obtained by training or fine-tuning a specific sequence-based model on correctly and incorrectly ordered lists of headers (i.e., a training set). Such a model should be able to output a probability of proper ordering such that lists of headers following a logical and intelligible order will have a higher probability score than that of a jumbled ordering. This model may then be applied to reorder the chapters that were produced by a TSegA. The above-described training can be done from scratch or by fine-tuning an LLM with an appropriate prompting strategy.

Under a fourth arrangement, instead of crudely reordering the segments into a new sequence, the segment (re-)structuring module may link every pair of segments sufficiently close to each other. In contrast to the linear reordering described above in the first, second, and third arrangements, such a graph need not link each segment to a single next one; nodes can have a higher degree as long as the linking criterion is met. The choice of which pairs of segments should be linked together may be performed as described above (i.e., by computing a metric of semantic proximity, by interrogating an LLM, or by fine-tuning an LLM specially for this task). In such a case, the restructuring component may define a graph over all of the segments (i.e., its nodes) through which the user will be able to navigate within a GUI (as will be described below) by going from one segment to the next through a relation of relevance instead of the chronological order.

In addition to the reordered and/or reorganized sequence of segments (as described above), the segment (re-)structuring module may output a mapping between the start- and end-timestamps of the segments in the original recordings and the timestamps of the segments they are linked with according to the transcript's new structure. Such a mapping may then be used by the audio structuring module that will be described below.

The fourth component of the presented techniques encompasses an audio structuring module. Once the sections of the original written transcripts have been restructured, the audio itself may be cut in accordance with the start- and end-timestamps of the segments, and the splits can be reorganized as the transcripts have been restructured, effectively reorganizing the original recording. The below-described GUI provides a user with the option to listen to each audio segment as they move through the written segments. An audio file can then be replayed along a new ordering at the same time and in the same way that the transcript can be explored by the user.

After the topics have been reordered into a logical sequence or into a logical graph (as described above), the fifth component of the presented techniques, an information supplementation module, can identify and fill any gaps (i.e., pairs of chapters immediately following each other with a semantic distance larger than a certain threshold). Those pairs represent breaks in the flow of information provided during the meeting. They signal aspects of the document that are in need of clarification or additional information. When the transcript has been restructured into a graph of segments, various graph algorithms may be applied to identify missing links between nodes (i.e., pairs of nodes which should be linked based on the structure of the graph, but which are not and which, consequently, signal a potentially missing piece).

Under a first arrangement, such gaps can be highlighted within the below-described GUI for a human user to add further segments. Under a second arrangement, a general purpose LLM may be leveraged to provide the missing information by inserting new paragraphs based on contextual and peritextual data, such as additional documents that are linked to the meeting or the transcripts of other meetings that were attended by the same participants.

Under a third arrangement, the system may be connected to a database of previous meetings and may leverage this source of information to fill the information gaps in the transcription of a new meeting. Over time, such a database of meetings (thus maintained by an organization) will grow and make this component increasingly useful. Under a fourth arrangement, a LLM can be fine-tuned to identify information gaps in a document and supply the necessary paragraphs to fill those gaps based on contextual information and a database of general knowledge.

The sixth component of the presented techniques encompasses a topic summarization module (TSumA). The restructured written transcripts (as described above) may be passed to an LLM in order to generate a summary of every section of the meeting as well as higher-level summaries of groups of sections. Under a first arrangement, such a summarization may be obtained by means of a pre-trained LLM with a reordered transcript and an appropriate prompt, thus reusing pre-existing models. Under an alternative arrangement, a pre-trained model may be reused and then fine-tuned with a dataset of topic-summary pairs.

The seventh, and final, component of the presented techniques encompasses a GUI whose purpose is to display the above-described outputs to a regular user and additionally allow an administrator and an editor to enter information during an editing process. The GUI may present the outputs of the system in such a way that a user can easily understand which elements have been reordered, linked together, or inserted and in which manner. Such a presentation allows the user to understand the history of the transcript and prevent misunderstandings.

The GUI can leverage the non-linear nature of hypertext to present the final transcripts in various orders, including according to the original order (while linking each segment to others that the user is likely to find most useful), according to a linear reordered sequence (while linking each segment with its original context should a user want to refer to it), and according to a graph-like display (letting the user understand the overall semantic structure of a discussion).

Hence, the first function of the above-described GUI is to provide some control over the system to a human editor and allow them to edit the various outputs. In a first arrangement, the reordering and contextual supplementations that were computed by the model are only presented to a human editor through the GUI, but not applied, in the form of suggestions to help the editor with their task. The editor will ultimately decide if they do or do not wish to apply each recommendation. This arrangement would significantly ease and accelerate the editing of any recording.

In a second arrangement, the reordering and contextual additions may be initially presented as recommendations to a human editor. The editor's reactions (e.g., their acceptance or rejection) may be logged and used to fine-tune the models in such a way that the models become progressively better at suggesting changes that match the editor's observed preferences. Over time, the rate of acceptance should rise, and the human editor will find it less and less necessary to intervene and apply their own changes. This arrangement thus allows for a progressive personalization of the models with respect to the requirements of a specific user or a specific organization.

In a third arrangement, the models may additionally compute a confidence score for each of the recommended changes. A user's reactions (e.g., their acceptance or rejection) may be logged and used for the constant fine-tuning and improvement of the models, as in the second arrangement that was described above, so that the models become able to output recommendations with better scores and either directly apply the changes to a transcript (when the confidence score of an editing change is high enough) or request the approval of a human editor (when a change is more dubious and associated with a lower score). As the models are trained on more and more such data, they become increasingly able to apply the changes directly, thus freeing the time of an editor and making the system more and more automated.

In a fourth arrangement, the system may apply any changes directly and the GUI may be used by the editors to only monitor the system and make occasional corrections after the fact.

The second function of the above-described GUI is to present the output of the whole process to a final user. This aspect of the GUI also displays a transcript according to the above-mentioned options, in a linear or hypertextual order, while letting the reader access the original context of each segment (for instance, through a hyperlink or a similar device) should they need that information. The consumer of a transcript can thus read the transcript in more meaningful orders, which is likely to be the simplest method of consumption and the easiest manner to understand the transcript while still referring to the original ordering when necessary.

Use of the presented techniques offers a number of advantages, including improving the transcription of meetings and increasing their value by dividing the raw documents that are output by ASRs into semantically cohesive sections; enriching such a list of sections by discovering semantic connections and reorganizing them into a more understandable structure, with a short summary for each section; filling information gaps to keep the transcripts understandable despite the passage of time; letting the whole process be automated either in part or in its entirety; and letting the system learn from the choices of the users and progressively personalize its own behavior.

Figure 1, below, presents elements of the initial processing steps (as described above) that may take place according to the presented techniques.

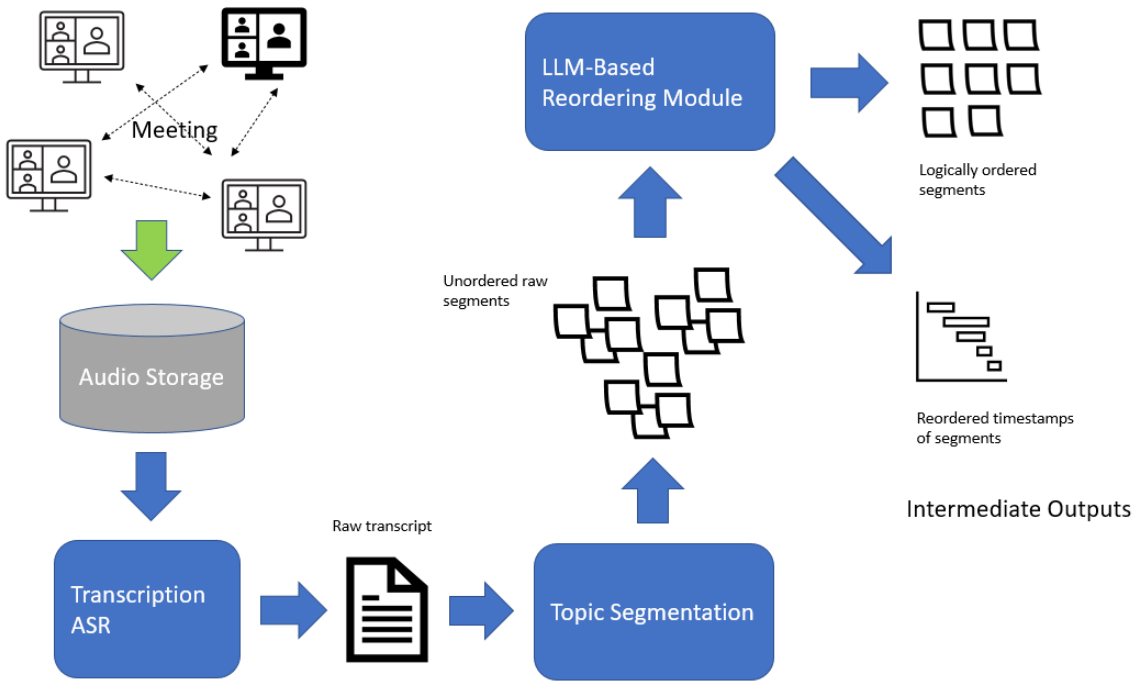

Figure 1: Initial Processing Steps

As depicted in Figure 1, above, the initial steps of the process encompass the transcription of a meeting, resulting in a draft transcript; the division of such a draft into smaller, semantically cohesive segments; and the extraction of the embedded structure by means of an LLM. This first sequence of steps produces two outputs – a list of logically restructured segments and a mapping of timestamps from the original audio to the new graph-like arrangement.

Figure 2, below, presents elements of the later processing steps (as described above) that may take place according to the presented techniques.

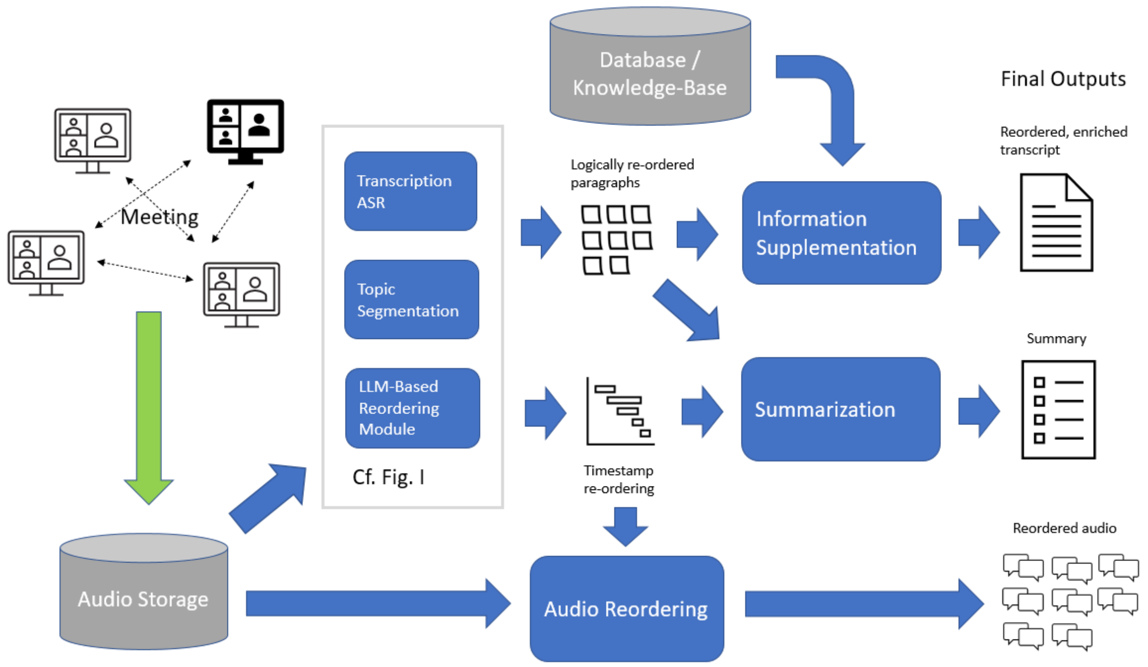

Figure 2: Later Processing Steps

As shown in Figure 2, above, the final processing steps (following the completion of the operations that were depicted in Figure 1, above) encompass the structuring of the audio recordings in accordance with the timestamps' mapping; the addition of supplementary information to the transcript; and the summarization of the segments. This yields the final outputs of the systems – a reorganized and expanded transcript, a summary of the meeting, and an associated graph of the audio recording.

As described and illustrated above, aspects of the presented techniques leverage LLMs. Compared with other uses of LLMs, it is important to note here that the tendency of such models to hallucinate (i.e., to output unrelated, meaningless, and potentially misleading elements) is less of a concern for the presented techniques since they will mostly reuse already existing passages either from a transcript under editing or from a database. Those passages which arise in a transcript from human discussions or other ground-truths, and are thus not the product of stochastic processes, can be compared with the original transcript to ensure that they do not diverge excessively. Some of the parts that may be added by the supplementation module (as described above) may be generated by an LLM, but they can be marked as such within the GUI in order to ensure that a user will not be fooled into believing that they were part of the original data.

In summary, techniques have been presented herein that support the automated reorganization of a meeting transcript. According to the presented techniques, when a meeting is recorded it is first transcribed through an automated speech-to-text system. Then, the resulting raw document is decomposed into sections corresponding to different topics, the topics are reorganized into a more coherent and intelligible order, and missing pieces of information may be identified and then added to the document. Such postprocessing increases the intelligibility and value of meeting transcriptions and ensures that they remain understandable and useful over the long term. Aspects of the presented techniques may leverage LLMs and ASR capabilities.