Adaptive Speech Analytics: System, Infrastructure, and Behavior

Upendra V. Chaudhari, Ganesh N. Ramaswamy, Eddie Epstein
Sasha P. Caskey, Mohamed K. Omar

IBM T.J. Watson Research Center
Rt. 134, Yorktown Heights, NY 10598
uvc@us.ibm.com

10.21437/Interspeech.2005-91

Abstract

This paper describes an adaptive system and infrastructure for Speech Analytics, based on the UIMA framework and consisting of a set of analysis engines (analytics) and control units, whose input is an unspecified and ever changing number of continuous streams of audio data and whose output is the detection of events consistent with a focus of analysis and/or the discovery of relationships among the outputs of the constituent analytics in the system. The central theme presented concerns the ability of the system to use the meta-data generated during the analysis to adapt both the behavior of the underlying analytics engines and the overall data flow to adjust the granularity and accuracy of the analysis in order to allow processing of increasing amounts of data with limited resources.

1. Introduction

The view of speech analytics (mining) presented combines the operations of transduction and annotation to convert a stream of bits into usable information within the context of the analysis. Generally, usable information will support decision making with respect to a focus of analysis, described in terms of a data model characterizing the meta-data produced within the analysis. Contemporary systems must be able to process potentially unbounded amounts of data with a bounded set of resources and there are a number of general observations that can be made. First, the amount of data containing relevant information is orders of magnitude smaller than the amount of data available for analysis. Second, the amount of data that can be analyzed is orders of magnitude smaller than the amount of data available for analysis. Thus, a critical component in contemporary analytics systems is the proper filtering of large volumes of data down to manageable amounts. It is apparent that the intermediate results of analysis that are generated, the meta-data, can favorably affect the operation of constituent analysis components. The ability to feedback information is a prime example. However, the relationship between the amount of data containing relevant information and the amount of data the can be analyzed is unknown. It is to cope with this uncertainty, in the context of bounded resources, that the analytics system and architecture is developed.

That is, presented herein is the design and implementation of a speech analytics architecture, based on UIMA (Unstructured Information Management Architecture) [1], and constituent elements capable of dynamic reconfiguration in response to resource availability, throughput requirements, accuracy requirements, or a change in the focus of analysis. This is the speech component of a general cross-modal, cross-lingual system [2]. Here, the focus is on adaptation, which is accomplished by careful filtering of data and sequencing of light and heavy weight analytics based on an evaluation of the intermediate meta-data generated in the course of analysis. This information is also exploited to adapt the analysis engines for better accuracy. In the realm of audio processing, light weight, or low complexity, analytics might include functions such as speaker identification or channel identification, whereas more heavy weight, or high complexity, analytics might include speech recognition or topic detection. In actuality, there is a gradation of complexity not only among the different analytics, but also within the implementation of a single analytic. This allows for a tradeoff between accuracy and resource usage in a given component function or in the interaction of the components.

The main point is that the results that are generated in the course of the analysis, the meta-data, can be used to adapt the ongoing analysis both to improve accuracy and to improve data throughput. Toward this end, an analysis scenario will be described along with the constituent components. Then, the cost of analysis will be addressed. Adaptability will be introduced, both to respond to cost fluctuations and to facilitate a more accurate analysis.

2. UIMA

UIMA[1][1] offers a set of interfaces and components that facilitate the extraction of information from "unstructured data," defined as that for which the form of the data, e.g. natural language text, audio, video, etc., does not give significant cues as to the meaning of the data. The significance in the context of the paper is that it provides a standardized framework, isolating system considerations from algorithmic components, which allows the combination and coordination of a set of **Analysis Engines** via a component called the Collection Processing Manager. This facilitates a rich analysis of the input data (or Collection; for example a sequence of documents in the text domain) in pursuit of relevant information. Also of significance, the analysis engines may be developed independently of one another, since by conforming to the UIMA specification, they would become pluggable components.

2.1. CAS

Central to the UIMA specification, is the use of the the CAS (Common Analysis Structure) which serves as both the input and output of an analysis engine. The CAS, which can be naturally rendered as an XML structure, has four primary components. First, it contains the data (or a URI referring to the data) that is to be analyzed. This is typically referred to as the artifact or "subject of analysis." Second, the data model characteriz-

[1]The UIMA SDK is available for download at http://www.alphaworks.ibm.com/tech/uima

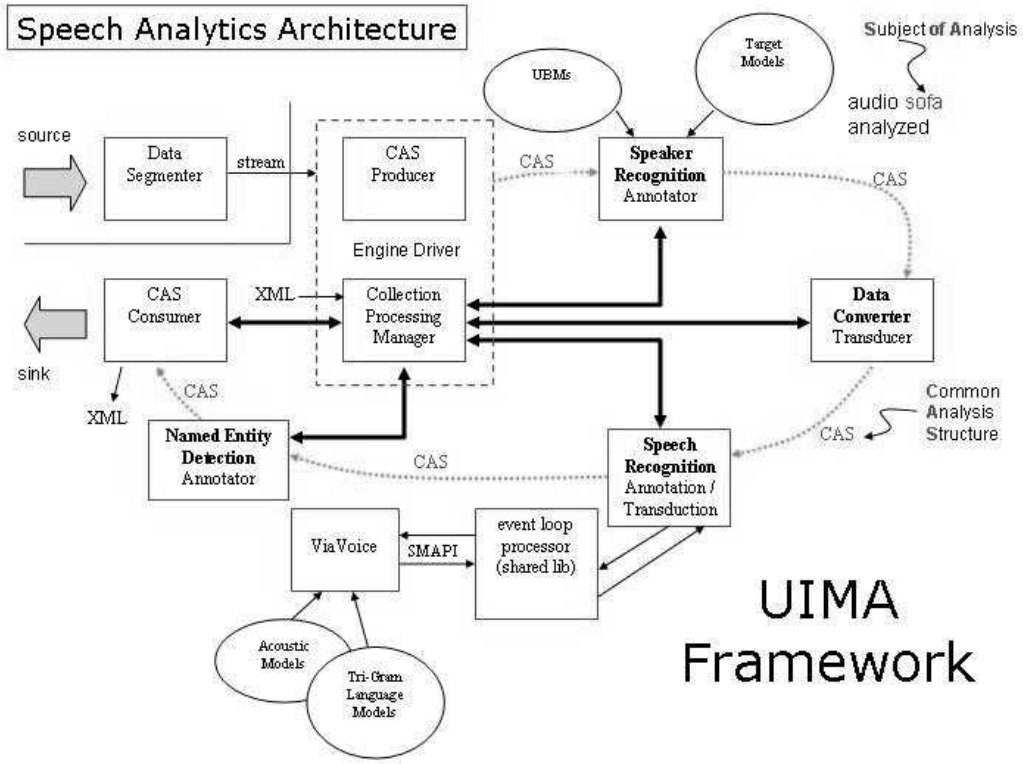

Figure 1: Speech Analytics Scenario

ing any meta-data that will be stored in the CAS is described by a Type system (see 2.2). Third, the CAS stores the actual meta-data, also referred to as feature structures, which are instantiations of Type system elements generated by the operation of the analysis engines on the subject of analysis. And finally, it contains a repository of associated indices for iterating over the feature structures. Given raw data, a component called the CAS Producer, creates a corresponding CAS. After the analysis, a CAS Consumer collects the meta-data generated for all CAS structures, arising, for example, from a sequence of raw data units.

2.2. Type System

An important consideration is nature of the meta-data that is generated in the system. It is not arbitrary. The possible values of the meta-data are described by a hierarchical type system consisting of both primitive and complex (derived) types. A feature structure is an instance of a type from the type system. This system is global and accessible to all analytic components, allowing them to know what forms of meta-data might be available in the CAS. This is significant because one engine may be able to use the results of another to improve its analysis, i.e. each engine can view the meta-data generated by previous analyses and then add to or update this information.

3. System Description

3.1. Scenario

Consider the scenario where a stream of audio data is to be analyzed with respect to speech and text analytics, including data segmentation/classification, speaker detection, speech recognition, and named entity detection. In general, there are many other analytics available, such as mood analysis, language detection, topic detection, etc. These analysis engines embody two fundamental processes, that of annotation and that of transduction, although in the text they are sometimes all referred to as analysis engines. An architecture for sequencing these operations for efficient data analysis using elements of the UIMA framework, heretofore used primarily for text processing, is shown in Figure 1, where now the input is an audio stream which is algorithmically converted into a sequence of artifacts via segmentation.

That is, in practice, the system operates on streams of audio data which are first routed from the source to the Data Segmenter, whose output is streamed to the Engine Driver where a sequence of CAS structures, one for each segment, is created. Multiple streams can be multiplexed in this unit if desired. Subsequently, the sequence is routed through an analysis pipeline, an example of which is indicated by arrows in the diagram. At every point in the analysis, any component, whether it be the Driver or Analysis Engine, can examine the contents of the CAS to search for useful meta-data. When a CAS is no longer useful, it is sent to a sink which may either store or destroy the data. One analogy for the operation is to view it as a sequence of Unix pipes where the downstream operations are a function of the results of the upstream operations. Various operating modes are discussed below, after a description of the components.

3.2. Component Analytics and Transducers

The following gives a brief description of the analysis engines in the scenario shown in Figure 1.

3.2.1. Stream Segmentation

The Data Segmenter is the first component to see the input data streams. Its function is to partition the input into segments, which can be uniform, as with a fixed window size, or non-uniform and based on a variety of algorithms that decide whether or not a segment boundary occurs at any given point in time. The algorithms cover a range of complexity/accuracy tradeoffs [3]. The fixed partitioning essentially has no time cost associated and no attempt to accurately find segmentation points. The more complex algorithms, with high accuracy on a number of test sets, still operate an order of magnitude faster than real time.

3.2.2. Speaker Recognition

The speaker recognition engine [4] [5] produces a top-N list of hypothesized speakers, drawn from a speaker library, ranked according to a measure indicating the degree of confidence that the speaker is detected in the data. It updates the CAS by entering the hypothesized speakers and associated confidence scores for the subject of analysis and operates at .1xRT to 0.5xRT. Complexity/accuracy tradeoffs can be made by altering the model size and/or data sub-sampling.

3.2.3. Data Conversion

This element is general and serves as a transducer from available data types to required data types. For example, input data to the system will typically be in one format, whereas the Speech and Speaker recognition engines might require different formats. One function of the data converter is to effect the required transductions.

3.2.4. Speech Recognition

The speech recognition engine supports real-time LVCSR (Large Vocabulary Continuous Speech Recognition) based on a statistical language model over a wide range of environmental conditions [6]. It can act both as an annotator and a transducer. For example, the engine enters the decoded sequence of words along with start and end times, but also creates a stream of text that can be processed by a downstream component such as the Named Entity (Mention) Detector. The engine has the capability to adapt recognition models to individual speakers. Furthermore complexity/accuracy tradeoffs can be made by altering search and model parameters.

3.2.5. Named Entity (Mention) Detection

The named entity detection engine evaluates the text, derived from the ASR engine in the scenario. Maximum Entropy modeling techniques are used to identify mentions of entities (Person, Organization, Location, etc.) [7].

3.2.6. Engine Driver

This unit contains the logic to control the flow of data segments through the available analysis engines. It also includes the functions of the CAS Producer and Collection Processing Manager. It has the ability to examine the contents of each CAS. This enables filtering of the data and dynamic sequencing of the engines. In the analogy with Unix pipes, this would be the entity that decides the order of programs in the pipeline. See section 4 for a detailed description of its behavior.

4. System Adaptation

4.1. The Cost of Analysis

In general, the accuracy of analysis of a given sample of data is a function of three parameters: computational power, storage/memory, and time. The cost of the analysis is a complex function of the three parameters. However, a number of observations can be made and exploited. In particular, every analysis engine need not be exposed to every data element and only certain interesting elements that relate to the focus of analysis should be fully analyzed. Thus the analysis is dynamic and accuracy is dependent on the nature of the data set (stream) in addition to the other parameters. Within this context there are decisions that could be made to intelligently trade off granularity and accuracy of analysis for a reduction in resource usage.

4.2. Adaptation Capabilities

A key property of the analytics system and architecture presented is the ability to benefit from meta-data generated in the course of analysis, both in the form of increasing the accuracy of the component analytics, via the CAS infrastructure, and in the ability to easily and quickly respond, via the analysis driver, to a shift in the operating point, as determined by an internally or externally driven shift in the parameters. Some examples follow:

Engine Adaptation to Increase Accuracy:

- The Speech Recognition engine can examine the CAS to see if a speaker hypothesis is present, and if so, use speaker specific recognitions models.
- The Speaker Recognition engine could benefit from a knowledge of the classification of the audio channel as this would allow better score normalization[4], an important factor in handling mismatched data.
- In the Speaker Recognition analysis, the detection of a particular speaker in a particular environment could alter the search space of target speakers for data segments that are near in time to the current segment. Here, multiple forms of meta-data from multiple engines are used to effect a change.

Engine Adaptation for Increased Throughput:

- The evaluation metric for the Speaker Recognition engine allows for sub-sampling of the input data resulting in a gradual degradation in accuracy while increasing the speed by the sub-sampling rate.
- The Speech Recognition engine's active search space for decoding could be reduced gradually to increase the speed of transcription gradually, with an accompanying loss in accuracy.

As the information in the CAS is dynamic, the analysis sequence need not be pre-defined. Such considerations are necessary since the amount of data available to be analyzed is many orders of magnitude greater than the amount containing relevant information and/or the amount that it is possible to analyze.

Analysis Control for Increased Throughput:

- The Engine Driver could pass on for analysis only

those segments with a specific type of audio, e.g. speech, as determined by the data segmenter.
- If after analysis by the Speaker Recognition Engine, the CAS contains a speaker hypothesis indicating an uninteresting speaker, analysis can be curtailed.

Another adaptation results from the fact that there may be multiple analysis engines that could perform the same task with differing accuracies, differing complexities, and requiring different input formats. In this case, a transduction engine could be used to convert data from its native format into one where the analysis has a different complexity or accuracy. However, the cost of transduction, the reduction/increase in complexity, and the reduction/increase in accuracy must all be evaluated to determine if this conversion should be pursued.

Ultimately, the flexibility of the architecture and the adaptive ability of the component engines yield an analytics infrastructure with many interesting properties:

- The ability of any engine (most likely in the low complexity mode) to act as a filter to downstream processing.
- The ability of any engine to adapt its analysis based on the results of upstream processing.
- The sequence of engines through which the data flows can depend on external cues.
- The analysis result of an engine at any point in the sequence can alter the remaining sequence of engines. Thus, an analysis tree is traversed with branches taken based on the data.

5. Operating Modes

The basic operation of the system can be described as either goal driven or discovery driven.

5.1. Discovery Driven Control

For the case where discovery is desired, the task is to analyze the input data with respect to all of the analytics and examine the resulting meta-data to find relationships among the variables. An example might be the discovery of a correlation between the identity or characteristics of a channel and the speakers who normally use it. Another could be a relationship of speakers and words used, etc. The Engine Driver could make these observations, as could another analysis component. In this case, the control preference would be to rout the data to all analysis components.

5.2. Goal Driven Control

Consider the data route shown in Figure 2 which depicts a scenario where the data flows to specific engines only when the initial analysis yields results consistent with a global analysis focus.

In Figure 2, the goal is to find specific mentions of events spoken by a limited set of target speakers on the broadcast channel. A determination is made to ignore data at various points in the flow when the meta-data becomes inconsistent with the focus.

As more engines conforming to the UIMA specification and with a variety of capabilities are added to the system, richer data flows can be generated.

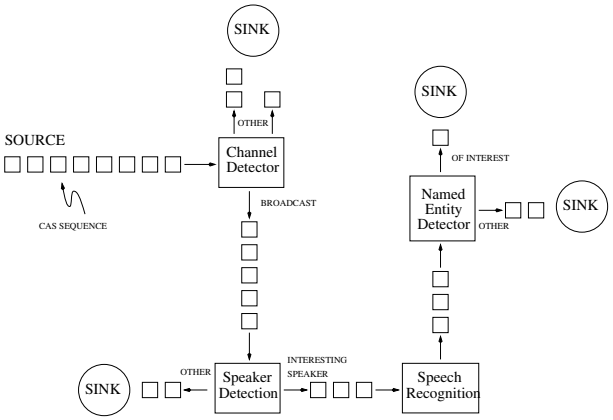

Figure 2: Dynamic Analytics Route

6. Conclusion

This paper presented a system and infrastructure for speech analytics based on UIMA with a variety of adaptive capabilities which can be triggered externally or internally, via a continuous evaluation of the meta-data generated in the course of a goal driven analysis. Methods to trade off elements in the cost of analysis were described in general and within the context of a functional scenario.

7. References

[1] D. Ferrucci and Adam Lally, "UIMA: An Architectural Approach to Unstructured Information Processing in the Corporate Research Environment", Journal of Natural Language Engineering, vol. 10, Issue 3-4, pp. 327-348, September, 2004.

[2] S. Caskey, U. Chaudhari, E. Epstein, R. Florian, J. S. McCarley, M. Omar, G. Ramaswamy, S. Roukos and T. Ward, "Infrastructure and Systems for Adaptive Speech and Text Analytics", First International Conference on Intelligence Analysis Methods and Tools, MITRE, McLean, VA, May, 2005.

[3] M.K. Omar, U.V. Chaudhari, G.N. Ramaswamy, "Blind Change Detection for Audio Segmentation", ICASSP-2005 (to appear), Philadelphia, March, 2005.

[4] G.N. Ramaswamy, J. Navratil, U.V. Chaudhari, R.D. Zilca, "The IBM system for the NIST 2002 cellular speaker verification evaluation", ICASSP-2003, Hong Kong, April, 2003.

[5] U.V. Chaudhari, J. Navratil, G.N. Ramaswamy, S. Maes, "Very Large Population Text-Independent Speaker Identification Using Transformation Enhanced Multi-Grained Models", ICASSP-2001, Salt Lake City, May, 2001.

[6] S.S. Chen, E. Eide, M.J.F. Gales, R.A. Gopinath, D. Kanevsky, and P. Olsen, "Automatic Transcription of Broadcast News,", Speech Communication, vol. 37, pp. 69-87, 2002.

[7] R. Florian, H. Hassan, A. Ittycheriah, H. Jing, N. Kambhatla, X. Luo, N. Nicolov and S. Roukos, "A Statistical Model for Multilingual Entity Detection and Tracking", HLT/NAACL 2004, Boston, pp. 1-8, May, 2004.