Speech Technology For Man-machine Communication

Aniruddha Sen
School of Technology and Computer Science
Tata Institute of Fundamental Research
Homi Bhabha Road
Mumbai 400 005 INDIA
asen@tifr.res.in

ABSTRACT

Communication with computers in Indian languages is a desired goal for bringing common Indians to computers and for adding value to the portable gadgets. Speech recognizer extracts text from the spoken message. It performs pattern matching at acoustic level, often by Hidden Markov Modeling (HMM). The result is refined by language model, usually bigram or trigram. Speech synthesizer converts text to speech. The text is first analyzed to get pronunciation and accent. The analyzed information is then synthesized: either by concatenation of pre-stored natural speech segments or by generating speech by a formant-based model. So far in Indian languages, only reliable isolated word recognizers were built. Continuous speech recognizers are emerging. Acceptable synthesizers in various languages were built and applied. But quality and accent need improvement. To streamline speech related work and projects, the activities should be categorized into areas e.g. research, development, data base building and manpower channeled accordingly. Speech I/O technology for computers, although crucial for India, is unlikely to be commercially viable soon. The dichotomy may be handled by small and focused projects, open source and resource sharing.

1. INTRODUCTION

We are witnessing *the boom* in IT and communication sectors in India. As a large part of our population is modest in literacy, speech is and will remain their preferred mode of communication for some time at least. And here steps in *speech technology*.

Role of speech technology for *man-to-man* communication needs no elaboration in this era of mobile phones. This paper aims to highlight a lesser-known facade of speech technology, the one related to *man-machine* communication.

Currently, our IT sector (including BPO) is mainly *outward looking*. To sustain growth, it must start *looking inward* too, better sooner than later. After all, true globalization is hardly possible if we leave one-fifth of its population alone! Increasing Indian customer base needs, among other things, Indian language supports in computers. Much has been achieved towards this end through government and private efforts, most notably under the *Technology Development for Indian Languages* (TDIL) initiatives.

Justifiably, such initiatives included the technology development for *written* as well as *spoken* languages. Consequently, along with Indian language scripts, word processors, spell checkers and OCRs, we see at least the first generation speech inputs and outputs for computers in Indian languages. Currently their utility is limited due to the limited quality of speech interfaces. But given that considerable awareness has been generated and also the technological foundation has been formed for speech I/Os in Indian languages, quality of products and consequently, their usefulness, are now bound to improve to desirable level soon.

To accept spoken input, computers use *speech recognizers* that extract and store the *linguistic contents* of spoken messages, often in *text* form. For spoken output, *speech synthesizers* usually convert text to speech.

We overview technical aspects of speech recognition, speech synthesis and some related issues in Section 2. We brief the situation in India in Section 3. In Section 4, we try to group the related activities to help streamline speech technology activities. We conclude in Section 5 with some suggestions.

2. SPEECH I/O TECHNOLOGY

2.1. Speech Recognition:

Speech recognition is extraction of underlying linguistic (or phonetic) transcription, given the speech signal. It is accomplished in two stages: *acoustic* and *linguistic* (Fig. 1).

In *acoustic* stage, the usual practice is to segment the utterance into tokens and then identify each token by some pattern matching method. Matching is done by comparing *selected features* of the spoken (test) sample with corresponding features of pre-stored *training samples*. Same utterance spoken by different speakers, or even by the same speaker at different times, have statistical variations and individual units of utterances (e.g. phonemes) have contextual variations in continuous speech. Pattern matching method has to filter out these variations as best as possible in order to identify the linguistic token. Feature selection is very important for this reason. Mel Cepstral Co-efficients are very popular. Alternate parameters have also been proposed [1].

Time alignment is an important pre-requisite for pattern matching. One way is *Dynamic Time Warping* (DTW) that tries to match two utterances of different duration by dynamic, best-fit algorithm. The resulting 'warping' is non-linear. But the most popular method is *Hidden Markov Modeling* (HMM) that models tokens by a series of states (roughly, speech units) and transitions between them. The model parameters are state transition probabilities and the observation sequences (roughly, mean speech parameters) expected from each state. Parameters for each model are estimated by training [2]. Recognition is performed by computing probability of each model generating the test sequence and the one with maximum probability is selected.

The acoustic front-end of the recognizer can be accurate only to some extent (say, up to 80%). For better accuracy, the *constraints of the languages* are to be used. The results from the front-end are often maintained as multiple hypotheses, with probabilities. A *language model* is then used to select the best one.

Language models can be grammar based. But grammars are not easy-to-formalize. The easy way out is to build a *statistical model* that predicts the apriori probability of the occurrence of a sequence of words. Bigram and trigram are models for two and three word sequences respectively. These are generated by collecting statistics from a huge body of electronically available text data.

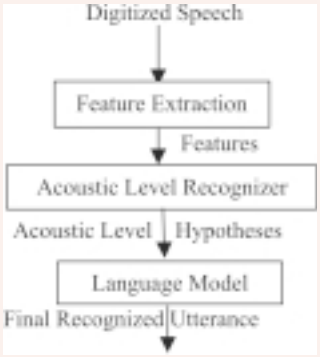

Fig. 1 Two Stages of Speech Recognition

In an HMM-based recognizer, the probability of any sequence of utterance is (roughly speaking) the product of its probability score from the acoustic model and the apriori probability of that sequence occurring in the language. The sequence with maximum (combined) probability is selected.

2.2. Speech Synthesis:

This, in most cases, is *text-to-speech* synthesis. The state-of-the-art supports synthesis of unlimited vocabulary continuous speech and we will imply that in subsequent discussions.

Speech is synthesized from text through two stages: *text analysis* and *synthesis*.

Text analysis establishes *pronunciation* and *prosody* (roughly, accent), given the text. Usually, a *text pre-processing* is done first, to separate out special-format parts of text (e.g. numbers, abbreviations, dates) from the main (grammatical) sentences. A standard way of generating pronunciation is to break the words into morphs (e.g. roots, prefixes and suffixes), obtain the pronunciation of morphs from phonetic dictionary and use letter-to-sound rules, if the dictionary look-up fails. Pronunciation is generated morph-wise. Adjacent words and morphs may modify each other's pronunciation and this is captured by phonological rules (e.g. 'sandhi' rules in Indian languages).

Text analysis is basically a *Natural language Processing* (NLP) job. It is quite elaborate for the languages with non-phonetic scripts, e.g. English. Even for most Indian languages (that have phonetic script), there is lack of correspondence between written and spoken versions and that leads to ambiguities. For example,

in most Indo-Aryan languages (e.g. Hindi, Marathi), schwa (roughly, the 1st vowel 'a') is selectively deleted, e.g. k*a*m*a*laa à k*a*mlaa. Here, the 2nd 'a' is deleted and the 1st one is retained. In languages such as Bangla, there are additional 'confusions' regarding pronunciation of sh, s, w, y etc. All these must be taken care of as much as possible by pronunciation rules.

Even more complex is to generate proper prosody, i.e. to vary intensity, pitch and duration in the way it is done in natural speech. Prosody for emotions being too complex to emulate, the current efforts are confined in incorporating a few stress nodes within the speech so that the synthetic speech doesn't sound machine-like. Fortunately, in Indian languages, stress node locations depend *more on position than on meaning*. Thus, simple prosody contours may be decided by: (a) determining the type of sentence, i.e. assertive, interrogative or exclamatory, (b) locating *keywords* for stress accordingly, e.g. query words for interrogative and adjectives for exclamatory sentences or (c) by putting stress on pre-determined default locations, e.g. on 1st vowel of 2nd word. This avoids intricate syntax and semantic analysis of the sentences and thus simplifies text analysis in Indian languages. However, it is necessary to break long sentences into parts by *phrase* or *clause* level parsers. Stress information may be passed on to synthesis stage by some pre-determined *stress markers*.

Synthesis stage generates *(digital) speech*, given the pronunciation and stress markers. The methods in vogue are: *concatenation* and *formant* syntheses.

In *concatenation synthesis*, continuous speech is generated by combining naturally spoken splices of pre-recorded speech. Acoustic realizations of *phonemes* (units of speech) are influenced very much by the context, and transition segments between phonemes carry vital perceptual information. Splices concatenated thereby are larger than phoneme, e.g. syllable, diphone. These are combined at relatively steady portions of speech, e.g. at the centre of vowels.

Concatenation synthesis has the advantages that the basic voice is natural, and complicated consonant-to-vowel transitions are encapsulated within splices. But the main problem is discontinuity at the splice joints and resulting noise. One way to reduce this is to do *spectral smoothing* at the joints [3], but this is never perfect. The current trend is to use large speech data bases with *redundancies*, so that the same utterance (e.g. sentence or paragraph) may be realized by alternative sequences of splices. Powerful search engines list alternatives and the best sequence is then selected on the basis of several criteria, e.g. spectral continuity.

Building concatenation synthesizer in Indian languages has got a boost with the availability of synthesizer-building tools. The most widely used one is *Festival* [4]. It is a multi-lingual concatenation synthesizer-building system, with some of the European language synthesizers in-built and with provisions to 'adapt' any other language by specifying the necessary rules, definitions, and data bases.

Formant synthesis is model-based, with *formants* (resonance frequencies of vocal tract) as the key parameters. Contextual variations and transitions can be implemented by appropriate rules (obtained by off-line analysis of speech data) to time-vary the parameters.

The advantage of formant synthesis is that here inter-segment transitions can be made very smooth by generating appropriate parameter track. Effect across far contexts can also be captured. At the negative side, approximation is inherent to modeling and that gives rise to mechanical voice. Consonant-vowel transitions are also too intricate to be captured by rules and are seldom perfected. Another problem is that the rules are obtained not by just objective analysis, but also by intuition. Thus, formant synthesis calls for *advanced expertise*.

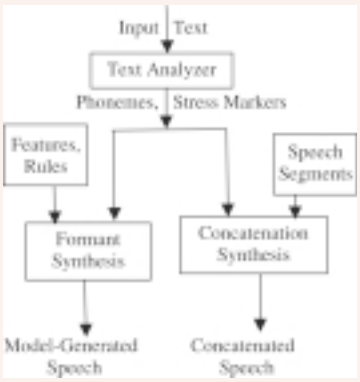

Fig. 2 Concatenation or Formant Synthesis

A formant synthesizer can be divided into two parts: (a) a rule module that generates model parameters, given the pronunciation and accent and (b) the model that generates speech, given the model parameters.

The rule module is language specific and must capture the static and dynamic behaviors of the speech sounds of the specific language. Indian phoneme set is markedly different from that of English. For example, Indian languages have separate aspirated and non-aspirated consonants (e.g. k and kh), retroflexed and dental t and d and also nasalized vowels (that is present in French). Substantial R & D is therefore needed to generate a separate rule set for Indian languages. However, phonetic differences *within Indian languages* are minimal. A rule set developed for any Indian language therefore can be (and was) used for many Indian languages, with only a limited number of changes.

The speech production model for formant synthesis used now is almost invariably some variant of Klatt's cascade synthesizer [5]. Experience shows that it can be adapted to any Indian language without much effort.

Fig. 2 shows the schematics of concatenation or formant synthesis.

3. SOME INDIAN SPEECH I/O SYSTEMS

3.1. Speech to Text:

Continuous speech recognition is the level that Indian language speech-to-text systems have to attain now. The reported systems are under development. IBM Research Laboratory's (IRL) Hindi speech recognition system [6] is based on HMM-based acoustic recognizer. It uses a trigram language model. The aim is to make desktop as well as telephone-based recognition systems, with preferably 90% plus accuracy. The systems are basically adaptation of IBM's ViaVoice recognition system for Indian languages. The biggest challenge here may be to map English phonemes into Indian languages, as the latter has many phonemes not known to the former. Another reported system is a research recognizer for Hindi at Tata Institute of Fundamental Research (TIFR) [7]. It is a small vocabulary system, used to demonstrate acceptance of spoken railway reservation queries. The system is also used in conjunction with a speaker authentication system for password verification.

But reliable isolated word, small or moderate vocabulary recognition systems were developed for Indian languages and were also used for limited applications with success. An early system built by the Central Electronic Engineering Research Institute (CEERI), is a 40 word system and was used to drive a wheelchair by oral commands (e.g. stop, left, right). TIFR has built a medium vocabulary, HMM-based system that was robust (with nearly 100% accuracy) and was used to provide speech input to a computer tutor and a travel guide demonstration system.

The current issues for transition to robust continuous speech recognition systems in Indian languages are mainly language specific, e.g. language modeling and acoustic modeling.

3.2. Text to Speech:

Continuous and unlimited vocabulary text-to-speech systems have been built in Indian languages and were also put to applications.

In concatenation synthesis, data driven approach was used by Indian Institute of Information Technology, (IIIT), Hyderabad [8]. It adapted Festival system to generate text-to-speech systems in Hindi and Telugu. Based on Festival, Indian Institute of Technology (IIT), Chennai [9] is attempting to build a multi-lingual corpora and thereby, TTS systems. Several TTS systems were developed under C-DAC, Pune and Kolkata units by concatenation synthesis, sans Festival. Those were used for Marathi, Bangla, Kannada and Indian English TTS.

In formant synthesis, TIFR has built a system that uses a Klatt-type speech production model [9]. Rules were developed for Indian phonemes, including the typical ones, e.g. retroflexed and dental t's, aspirated consonants (e.g. kh) and nasalized vowels. The basic phoneme-to-speech synthesizer was used to build text-to-speech systems in several languages, e.g. Hindi, Bangla, Marathi, Indian English. To adapt to a different language, the text analyzer of that language is to be developed. The change needed in phoneme-to-speech synthesizer is minimal for Indian languages. The synthesizer was used for aid to the blind in Bangla so far. CEERI, New Delhi also has developed TTS in Hindi and Bangla, based on formant synthesis. They have modified the basic Klatt's model, to accommodate aspirated consonants.

In Indian language TTS, the basic issue is now quality improvement. This can be gradually achieved and the areas that need focus are text analysis, prosody,

generation of big speech data base (for concatenation synthesis) and acoustic-phonetic research (for formant synthesis).

4. CATEGORIZATION OF ACTIVITIES

Speech I/O technology for computers in Indian languages has aroused government and private interests. It has also initiated developmental activities at various levels. All such efforts, however, are not matured enough. In view of the limited availability of expertise and resources in this area, there is a need to streamline the activities: in order to minimize repetitive or amateurish works.

An idealistic way may be to bring all efforts under a single command, analyze requirements and allocate scarce resources accordingly. But that is not absolutely possible nor is a high degree of centralization necessary. A good amount of harmony can be attained just by proper coordination, consultation, information exchange and resource sharing. As the government itself is sponsoring a large number of speech projects, it can take a leading role by first streamlining those.

Speech I/O technology is complex and multi-disciplinary. For streamlining, it is necessary to *categorize* the works and channel manpower and efforts accordingly. Here we suggest division of the entire speech I/O related work into some specific categories, with a mention of expertise needed in each. Such analysis may help to figure out what can be expected from a particular organization, as far as speech technology is concerned.

Application development: Speech expertise is high in demand and short in supply. It should therefore be utilized solely for development of speech-to-text and text-to-speech engines. Scope of various speech projects should be fixed accordingly. Once such engines, along with the data base and interface specifications are available, applications can be developed by any of the thousands of available software agencies.

Utilization of available technology: We need not re-invent wheel. Speech technology is language-specific. But whatever techniques, tools and resources are available should be fully utilized, at least to start with. Examples are HMM tool kits [10] and Sphinx software [11] for HMM-based recognition, Festival for concatenative synthesis and open source for Klatt's terminal (parameter-to-speech) formant synthesizer. Utilization of such tools or software makes it possible to fix some working system in few months with minimal speech expertise.

Reverse Engineering: As tempting as that may be, we cannot however stop here and should 'reverse engineer' the available tools/ software within a given time frame. For sustained quality improvement, we should ultimately have *what we need* rather than *what we have*. Such work needs guidance of speech experts, but may be implemented by qualified engineers.

Research: Speech recognition and synthesis need to formalize human behavior (that always remains an enigma), as well as languages (that are ever changing). This leads to imperfect understanding at any given moment that has to be gradually improved by open research. Current speech research issues are numerous. For recognition, handling noisy and spontaneous speech, auditory modeling, feature selection and language modeling are important research topics. For synthesis, pronunciation and prosody rules for Indian languages are vital. Some researches are pure and open ended, while some others (e.g. fact findings for some aspect of a language) may be undertaken as time bound projects.

If we want to hit the leading edge of speech technology, then research is a must. This can be done by academic or research institutions and we have many, with relevant expertise.

Development: As speech theory is constantly refined, at any given instance it should be put into practical use. Applications have to develop with the prevailing limitations and their scopes expand as quality improves with research.

In our narrower context, the research results (done in India or elsewhere) must be constantly absorbed to improve our speech-to-text and text-to-speech engines. This needs good liaison between research and development agencies and is a challenging task, considering that the worlds of the academics and the system developers differ. But experience has shown that the task can nevertheless be accomplished.

Resource building: As discussed in Sec. 2, speech I/O systems depend very much on speech data bases and language resources, particularly if data-centric

approaches are adopted. Speech data base needed for recognition, synthesis and acoustic-phonetic research are unfortunately different. Moreover, these are to be replicated for each language under consideration. Another problematic area is *labeling of speech data base*. Manual labeling needs considerable amount of manhours from acoustics-phonetics experts. As the size of data bases grow, that will become impossible. In line with the international trend, we will also have to settle for tools that automatically segment speech data and will have to live with the resulting inaccuracies.

Some language resources needed are huge bodies of text data (for building statistical language model) and phonetic dictionaries (for pronunciation in synthesis and lexical analysis in recognition).

In all, resource building is a huge task. But various government and private initiatives are already on. The aim should be to generate resources that can be accessed by developers, with nominal or no payment. That may not remove redundancy altogether, but will certainly reduce it.

5. CONCLUSIONS

In the coming days, speech I/O will have a great role to play in India, by enabling more people to interact with computers in their preferred way. Speech is also the suitable mode of interaction with computerized gadgets that cannot have attached keyboard and monitor and we will see such gadgets (starting with mobile phone) more and more in coming days. For the same reason, speech technology will have great significance for the armed forces that will use more and more computerized, portable gadgets.

So far, in speech I/O technology, we have tried to follow the developments of the English-centric west. We should soon find our own ways, as the needs will not totally overlap and we, in all likelihood, will have to extract additional mileages from speech technology.

Speech I/O for computers, although greatly in need now, may not be very much commercially viable at the moment. Even in international market, speech-to-text and text-to-speech products are not really making roaring business. They, however, 'silently' add value to the existing products and uplift the level. Many government and private initiatives to get back money from such products have failed. That need not give rise to despair and abandonment of initiatives. Rather, the efforts are to be considered as long term investments. As far as the government is concerned, they may fund small and focused projects to realize selective aspects of speech systems and pool all available outputs as resources: in public domain where feasible and for government sponsored projects otherwise. As for the private initiatives, the enthusiasm currently is all-pervasive. But it is likely that only the organizations big (or good) enough to sustain research will make much out of their efforts. The rest may do better to use the components available from market or public domain, at least for the time being.

6. REFERENCES

[1] K.K. Paliwal, "Sub-band spectral centroids for speech recognition", *Proc., ICASSP*, Vol.2, pp. 617-620, 1998

[2] L.R. Rabiner, "A tutorial on hidden Markov models and applications in speech recognition", *Proc. IEEE*, Vol. 77, No. 2, pp. 257-286, 1989

[3] D.T. Chappel and H.L. Hansen, "Spectral smoothing for concatenative speech synthesis", *Proc. ICSLP-98*, pp. 1935-1938, 1998

[4] http://www.festvox.org

[5] D.H. Klatt, "Software for a cascade/parallel formant synthesizer", *JASA*, 67, pp. 971-985, 1980

[6] http://www.research.ibm.com/irl/projects/speech.shtml

[7] K. Samudravijaya, "Computer recognition of spoken Hindi", *Proc., Intl. Conf. on speech, music and allied signal processing*, pp. 8-13, 2000

[8] S.P. Kishore, R. Sangal and M. Srinivas, "Building Hindi and Telugu voices using Festvox", *Proc. ICON-2002*, pp. 303-309, 2002

[9] N.S. Krishna, Hema A. Murthy and T.A. Gonsalves, "Text-to-speech (TTS) in Indian languages", *Proc. ICON-2002*, pp. 317-325, 2002

[10] X.A. Furtado and A. Sen, "Synthesis of unlimited speech in Indian languages using formant-based synthesis", *Sadhana*, June 1996, pp. 345-362

[11] http://htk.eng.cam.ac.uk

[12] http://www.speech.cmu.edu/sphinx