The Role of Phoneticians in Speech Technology

Alejandro Acero
Microsoft Corporation
Redmond, WA, USA

In this essay I will present my view, from an industry perspective, on careers in speech technology for graduates in phonetics and linguistics. I will start by introducing the career prospects in speech technology. I will then describe typical tasks that a phonetician working in speech technology would perform. Finally I will discuss the requirements that employers would like to see in their applicants.

Speech technology: an exciting career with a promising future

Recent advances in computer technology now make it feasible to commercialize products that perform man-machine communication by voice in real time. This has fueled many companies to invest in speech technology, creating many jobs during the last few years. Academic research has also benefited from this growth because companies are conducting joint projects with universities. Many of these projects are funded by the European Commission.

Nowadays, there are software packages for personal computers that can perform limited Automatic Speech Recognition (from here on abbreviated to ASR). After the system has adapted to the user's voice, it is able to recognize words separated by pauses with error rates below 5%. Likewise, there are software-only Text-To-Speech (from here on abbreviated to TTS) systems that can generate intelligible speech. Modern microprocessors are powerful enough as to perform both TTS and limited ASR in real time, without the need for additional hardware. While acknowledging the many accomplishments, we also have to accept the many limitations of current systems. While intelligibility of the best TTS systems is high enough to be useful in certain applications, speech quality is still low enough that the technology will not be ubiquitous until a major breakthrough appears. The limitations of ASR systems are even greater: the word error rate for continuous speech is still too high to be useful except for some special applications. Even the best systems are too fragile to the presence of new words and moderately noisy environments. The technology is still in its infancy and the challenges are large indeed, but momentum is clearly growing and commercially viable spoken language interfaces will emerge before the year 2000.

A solution of the ultimate problem in speech technology, the development of a conversational computer, is an extremely difficult task that has eluded researchers for the last 30 years. While a great deal of progress has been achieved, it could easily be another 30 years until we have a machine that can pass the so-called Turing test (under this test, a blind-folded human cannot distinguish whether he or she is talking to another human or to a computer). This means that while both industry and academia are creating many job opportunities today, they will likely create many more in the years to come. A market research study conducted in 1992 (Meisel, 1992) forecast that world-wide revenues from speech technology products in 1995 will approach $2.5 billion, reaching $26 billion in

the year 2000. A total of 137 organizations were listed in this study as suppliers of speech technology products in 1992, 22 of those being European.

The existence of many different languages in Europe makes it difficult for a speech product to easily reach a broad coverage. Unlike other computer products such as word processors, spreadsheets and databases, which are relatively easy to translate from one language to another, localization of speech technology products is a very labor intensive process. This barrier will inevitably slow down the introduction of speech products in some countries with smaller markets. Nevertheless, it also implies that a number of specific jobs will be created to generate a version of the product for each language. Nevertheless, it is important to note that advances in speech technology are reducing the dissimilarities of speech systems in different languages by defining more general frameworks under which to share more components. The possibility of contributing to change the way we communicate with machines is a very exciting proposition. Building a system like HAL (the human-like robot in "2001: A Space Odyssey") promises to be a very challenging task, and the road to these systems will be filled with excitement.

The position of the phonetician in speech technology: a job description

Speech technology is a multi-disciplinary field that occupies engineers, computer scientists and linguists. Unlike the speech sciences, whose main goal is to gain a better understanding of the speech production and generation process, speech technology's main goal is to build systems. Therefore, a linguist willing to pursue a career path in speech technology has to be a practical person and a team player. In this section I will describe some of the tasks required from a phonetician working in a speech technology group. A spoken language system consists of a computer hardware running a software program. This program contains computer code and data tables. It is the mission of engineers to write and maintain this code, written mostly in assembly language and C and C++ programming languages. Some of the data tables required are the result of running another computer program on some labeled speech data. Other data tables contain acoustic and linguistic knowledge about a language. A linguist can play a major role in developing these data tables, and also in architecting the algorithms.

In particular, a phonetician/linguist's role in a speech technology team is to build primary data tables, secondary data tables and rules and algorithms:

1. Primary data tables

They constitute the basic infrastructure for the system and it is essential that they be carefully designed. We can mention the following components:

- *Phoneme alphabet for a language*. This is probably one of the first tasks required for both TTS and ASR. This is normally a phoneme set instead of an allophone set. TTS uses multi-phone units to capture the particular realization of that phoneme in context, whereas ASR usually has different models for different contexts of a phoneme. With the advent of international TTS and ASR, the need for the use of an international phoneme set will increase.

- *Dictionary for a language*. Since in many languages it is not feasible to store all inflected forms, it is imperative that the dictionary contains all baseforms, affixes, as well as procedures to construct a word from these smaller segments for ASR, and to decompose a word into those segments for TI'S.
- *Mapping between the dictionary and the phoneme alphabet*. This is carried out by establishing pronunciation baseforms in the dictionary, and/or letter-to-sound rules. Unfortunately, in many languages such as English, this process is not a one-to-one mapping. A word has sometimes several possible pronunciations (i.e. homographs for read in "I will read the book" and "I have read the book" are pronounced differently) and a given pronunciation can correspond to different words (i.e. homophones for "R EH D" can be red or read). Therefore, it is important to have linguistic knowledge to disambiguate these cases. Both TTS and ASR systems require this component. It is also important to capture in the pronunciation dictionary the variability caused by dialects, speaking rate amid different speakers.
- *Mapping between symbols and words in the dictionary*. Since typical text contains numbers, abbreviations and acronyms, a set of rules needs to be defined to convert them to characters. These text normalization rules will expand things like "$1234.59" into "twelve hundred thirty four dollars and fifty nine cents". Several possibilities appear on how to expand "Dr.", which can be "doctor" or "drive" depending on the context. These rules have to be bidirectional as both ASR and TTS systems need them.
- *Language models*. Speech systems need to know how words in the dictionary can be combined together. In ASR, Finite-State Grammars (FSG) are widely used for command and control, and statistical n-grams are used for dictation applications. In TTS, syntactic parsers are used as pre-processors for prosody generation. A shallow analysis has typically been sufficient, as it is not critical to have a deep understanding of the sentence to be able to transcribe it (ASR) or to speak it (TTS). Nevertheless, we could safely assume that a deeper understanding of the text will help to improve prosody in a TTS system and increase robustness of an ASR system.

2. Secondary data tables

Typically these tables are the result of processing some annotated speech corpora with a computer program. Examples of the tasks involved to obtain these tagged corpora are the following:

- *Creation of lists of phonetically balanced sentences*. Most ASR and TTS systems decompose words into smaller sub-word units. Many modern TTS systems synthesize speech by concatenating sub-word units, typically diphones, from an inventory obtained from a set of speech recordings. In order to recognize speech, ASR systems create a statistical model of a word by concatenating triphone models, where each triphone is defined as the realization of a phoneme preceded and followed by other phonemes. These triphone models are obtained from a speech database. In order to have TTS and ASR systems that exhibit high quality across many different word sequences, the phonetic coverage of the speech database has to be sufficiently large. A large number of different phonetic

contexts need to be present in the recordings. In the case of TTS, this is generally done by generating a list of phonetically balanced sentences that the speaker will read. While for ASR systems the same technique can be applied, sometimes large amounts of text from different sources are used, while controlling their phonetic coverage.

- *Data collection for TTS*. The design of any TTS requires some speech, typically coming from a single speaker. With the current technology, it is important to select a speaker not only with a pleasant voice, but also wit h a regular voice that is well suited for the current algorithms. The speaker has to read a list of phonetically balanced sentences. A large phonetic coverage is necessary because at least one instance of each sub-word unit in the voice inventory has to be present in these recordings. The prosody module of a TTS also requires speech waveforms with a broad coverage of intonation patterns.
- *Data annotation for TTS*. For concatenative synthesizers in TTS, an inventory of sub-word units is extracted from speech recordings. In order to do this, the speech waveforms need to be annotated with phoneme boundary information. The development of a prosodic module for a TTS often requires that the speech waveforms be annotated with pitch contours. These tasks can sometimes be aided by automatic program.
- *Data collection for ASR*. The recognition error rate of current ASR algorithms is heavily dependent on having large amounts of speech data. Since the majority of the systems are speaker-independent, it is important to have samples of speech from a large representative fraction of the population that will use it. If users with different dialects are to use the system, speech samples are required from speakers from that population group. Since ASR systems typically model every phoneme in every possible phonetic context, large phonetic coverage is required to properly train these models. Large corpora of text are also required to build the statistical language models. Finally, to design an ASR system for a specific application, sometimes a data collection named "Wizard of Oz" is used to simulate the existence of an ASR system. In this paradigm, the subject is led to believe that he or she is talking with a machine, while in reality there is a person in another room transcribing the subject's speech into text in a computer terminal.
- *Data annotation for ASR*. Modern ASR systems only need to have the word transcription for each sentence and no further segmentation is required. Nevertheless, it is important to detect noises in the waveform and incorporate them into the transcription.

3. Rules and algorithms

A lot of the computer code needed in a spoken language system involves the coding of a procedure or algorithm. A phonetician can participate in this task by, for instance, suggesting mappings between all possible intonation contours that can occur in a language and their dependencies with type of clause. The phonetician can participate in brainstorming sessions with the engineers on how to implement in a computer program some of that expertise. The better the rules and algorithms are, the better the performance of the system is. While some of the traditional expert systems were exclusively rule-based, robust speech systems should not be totally dependent on these rules, but rather

use them in a fuzzy manner, that is, complemented with statistical knowledge.

The position of the phonetician in speech technology: job requirements

In this section, I will describe some of the desired qualifications for a phonetician working in a speech technology group. In addition to a good knowledge of linguistics, with breadth being more important than depth, there are other sets of qualifications that could make the linguist a much better fit in the team.

One of the characteristics of speech technology is that it is multi-disciplinary, and therefore cross-training is extremely important. Given the complexity of spoken language systems, they cannot be built by single individuals, hut rather require a team of several people. In order for this team to be effective, every member has to have a basic understanding of the system. While a programmer will need to know something about Phonetics, a successful phonetician working on a spoken language system will need to have some knowledge of computers, algorithms, statistics and signal processing. One of the reasons why there are not more linguists working in building Spoken Language Systems is that in many cases, lack of training in these other disciplines prevent them to be as effective in the team as an engineer or a programmer.

It would be very desirable that, in addition to the traditional course work in Linguistics, a phonetician working in a speech technology team would have taken a course on the fundamentals of computer systems and basic programming. Even though the linguist would not have to do any programming, it is important that all the members in the team can speak the same language to some extent, with that language being in our case a programming language. The majority of the difficulties I have had in the past when cooperating with linguists stemmed from the fact that they gave me suggestions that were either very hard to incorporate in a computer program, or would probably not make any impact in overall system's performance.

Many of the algorithms used in ASR and ITS make use of advances in signal processing and statistics. Again, while the phonetician does not have to be an expert in these fields, it would be extremely helpful if he or she has some basic understanding of signal processing and statistics. The contributions of a team member that understands the big picture can be greatly increased.

Also desired is proficiency with common computing environments such as Windows, UNIX and Macintosh, text editors, and speech analysis packages. Finally, the most important aspect I would mention is the need to focus on system performance. In an ASR system the main objective is to improve the recognition accuracy and usability. In a TTS system, the major objective is to improve speech quality. Any work that is not headed along those lines will not benefit system's performance. For example, system's performance is likely to be affected more by a wrong pronunciation entry in the dictionary for a common word, than by an error in classifying a rare word as a name instead of as an adjective. It is essential for every team member not to lose focus on the big picture.

References

Meisel W.S. (1992). *Trends in Speech Recognition: a Technology and Market Analysis*. TMA Associates. 92. Encino, CA (USA).