CSME 206A Natural Language & Speech Processing **Spring Semester**

Lecture 2: Speech Recognition Models and Applications.

Lecturer: K.R. Chowdhary *: Professor of CS*

2.1 Speech Recognition

Speech has long been viewed as the future of computer interfaces, promising significant improvements in ease of use over the traditional keyboard and mouse. In the past few decades, dramatic improvements in speech recognition technology have made high-performance algorithms and systems available. These advancements have been supported by government-sponsored scientific research in industrial research labs and in Universities.

Despite this long history of research progress, speech technology was, at least initially, slow to find commercial application. However, in the past couple of years the market has begun developing rapidly, and several interesting and useful speech applications are now feasible.

2.1.1 Two Application modes

We can consider applying speech recognition in either of two primary modes: using speech as spoken input or as data / knowledge source. Spoken input addresses applications like dictation systems and navigation or transactional systems. With dictation applications, the system transcribes spoken words verbatim into written text. These applications can create text such as personal letters, business correspondence, or even e-mail messages. In transactional applications, users can navigate around the application or use speech to conduct a transaction. For example, speech can be used to purchase stock, reserve an airline itinerary, or transfer bank account balances.

In other type of application, i.e., using speech as a data / knowledge source paves the way for applications such as meeting capture and knowledge management. These applications begin modestly as multimedia indexing systems that use speech recognition to transcribe words verbatim from an audio file into text. Subsequently, information retrieval techniques applied to the transcript create an index with time offsets into the audio. Users can access the index using text keywords to search a collection of audio or video documents.

2.1.2 Speech application basics

At the simplest level, speech-driven programs are characterized by the words or phrases you can say to a given application and how that application interprets them. An application's active vocabulary-what it listens for-determines what it understands. A speech recognition system requires a *speech engine*, which is is language-independent where the data it recognizes can include several domains. A domain consists of a vocabulary set, pronunciation models, and word usage models associated with a specific speech application.

It also has an acoustic component reflected in the voice models, which, the speech engine uses during recognition. These voice models can be either standard (speaker independent) or unique per speaker.

Figure 2.1 illustrates the resources for typical speech engine using during the recognition process. The domain-specific resources (for example medical domain), such as the vocabulary, can vary dynamically during a given recognition session. A dictation application can transcribe spoken input directly into the document's text content, a transaction application can facilitate a dialog leading to a transaction, and a multimedia indexing application can generate words as index terms.

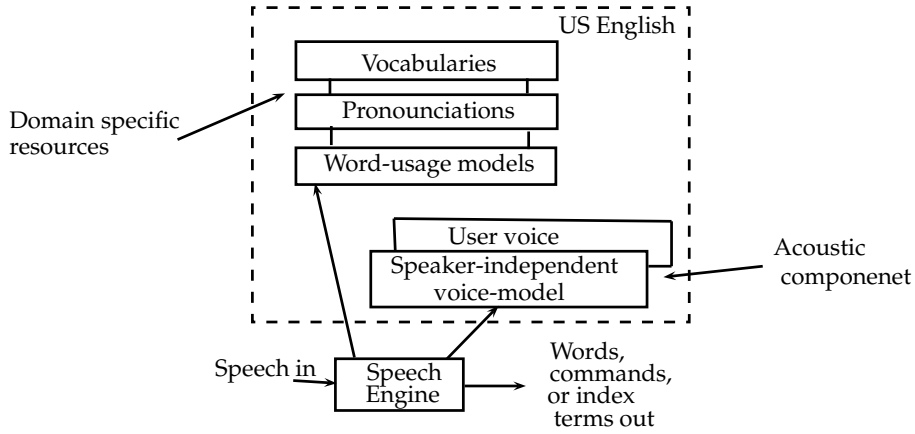

Figure 2.1: Phonetic Process.

In terms of application development, speech engines typically offer a combination of programmable APIs and tools to create and define vocabularies and pronunciations for the words they contain. A dictation or multimedia indexing application may use a predefined large vocabulary of 100,000 words or so, while a transactional application may use a smaller, task-specific vocabulary of a few hundred words.

Although adequate for some applications, smaller vocabularies pose usability limitations by requiring strict enumeration of the phrases the system can recognize at any given state in the application. For example, pronounce the PIN number after having inserted the ATM card, in four separate words. To overcome this limitation, transactional applications define speech grammars for specific tasks. These grammars provide an extension of the single words or simple phrases a vocabulary supports. They form a structured collection of words and phrases bound together by rules that define the set of speech streams the speech engine can recognize at a given time. For example, developers can define a grammar that permits flexible ways of speaking a date, a dollar amount, or a number. Prompts that cue users on what they can say next are an important aspect of defining and using grammars. It turns out that speech grammars are a critical component of enabling the Voice Web.

2.2 Speech Recognition Algorithms

Providing the computer with a natural interface, including the ability to understand human speech, has been a research goal for almost 40 years. Speech recognition research started with an attempt to decode isolated words from a small vocabulary. As time progressed, the research community began working on large-vocabulary and continuous speech tasks. However, the practical versions of such systems have become moderately usable and commercially successful only in last few years. Even now, these commercial applications either restrict the vocabulary to a few thousand words, in the case of banking or airline reservation

systems, or require high-bandwidth, high-feedback situations such as dictation, which requires modifying the user's speech to minimize recognition errors. For example, when some one pronounces "iland", the feedback system may suggest "Are you telling Island?" Based on your input, it will understand current word as well the future words, spoken similarly.

Early attempts at speech recognition tried to apply expert knowledge about speech production and perception processes, but researchers found that such knowledge was inadequate for capturing the complexities of continuous speech. To date, statistical modeling techniques trained from hundreds of hours of speech have provided most speech recognition advancements. Speech researchers have combined these modeling techniques with the massive increase in available computing power over the past several years to explore complex models with hundreds of thousands of parameters.

Example 2.1 *Research Problem: Like Google provided free service of Google maps to public, and through feedback, it collected important information about geometric position of places and roads, it is possible to provide free service of speech translation to people on-line, and in turn improve on the speech system model as well as the corpus of speech data-base.*

Many organizations' multi-site speech recognition research cooperation and competition, supported through government agencies such as DARPA, have also fueled advancements in this field. In addition to participating in government-sponsored competitions, industrial labs, universities, and other companies have fostered rapid advances in speech recognition technology by sharing data and algorithms

A by-product of these cooperative efforts has been that most successful systems share roughly the same architecture and algorithms because each site immediately copies other sites successful algorithms. To enable next-generation applications such as speech recognition over cellular phones, transcription of call center interactions, and recognition of broadcast news, researchers continue to work on the automatic speech recognition (ASR) system. An understanding of today's ASR systems architecture provides a basis for exploring the recent advances motivated by next-generation applications.

2.2.1 Basic Architecture

The process of speech recognition starts with a sampled speech signal. This signal has a good deal of redundancy because the physical constraints on the articulators that produce speech, i.e., the glottis, tongue, lips, and so on, prevent them from moving quickly. Consequently, the ASR system can compress information by extracting a sequence of acoustic feature vectors from the signal.

Typically, the system extracts a single multidimensional feature vector every 10 ms that consists of 39 parameters. Researchers refer to these feature vectors, which contain information about the local frequency content in the speech signal, as *acoustic observations* because they represent the quantities the ASR system actually observes. The system seeks to infer the spoken word sequence that could have produced the observed acoustic sequence. That means, you need to find the *cause* on observing the acoustic sequence, i.e., *effect*, see figure 2.2.

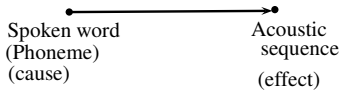

Figure 2.2: Cause-effect relation.

To simplify the design, researchers assume that the ASR system knows the speaker's vocabulary. This approach restricts the search for possible word sequences with in the words listed in the *lexicon*, which lists

the vocabulary and provides *phonemes* - a set of basic units, which are usually individual speech word sounds for the pronunciation of each word.

Commercial lexicons (database of collection of words) typically include tens of thousands of words, however, and the length of the word sequence uttered by the speaker is unknown. Let us assume that the length of the word sequence is N. If V represents the size of the lexicon, the ASR system can hypothesize V^N possible word sequences. Language constraints dictate that these word sequences are not equally likely to occur. For example, the word sequence "give me a call" is much more likely than "give call a me." Further, the acoustic feature vectors extracted from the speech signal provide significant clues about the phoneme that produced them.

The sequence of phonemes that corresponds to the acoustic observations implies the word sequence that could have produced the sequence of sounds. Consequently, the acoustic observations provide an important source of information that can help further narrow the space of possible word sequences.

The ASR system uses this information to assign a probability that the observed acoustic feature vectors were produced when the speaker uttered a particular word sequence. Essentially, the system efficiently computes these probabilities and outputs the most probable sequence as the decoded hypothesis.

Theorem 2.2 *Bayes Theorem: Given that an even B exists, what is probability that the event A_i has caused it, can be represented by Bayes conditional probability, expressed as,*

$$p(A_i|B) = \frac{P(B|A_i)p(A_i)}{P(B} \tag{2.1}$$

where, $p(A_i)$ is called prior probability.

In fact B might have been caused by any of the A_i (A_1, A_2, ..., A_i, ..., A_n). In that we need to choose the correct A_i. Obviously, the correct one is that which has got maximum probability. Thus,

$$maxarg_{A_i}\ p(A_i|B) = maxarg_{A_i}\ \frac{P(B|A_i)p(A_i)}{P(B} \tag{2.2}$$

Since the denominator $p(B)$ is common for all the expressions, it would not effect the order of probability of $p(A_i|B)$, hence the denominator can be dropped, without effecting the result. Thus, we get,

$$maxarg_{A_i}\ p(A_i|B) = maxarg_{A_i}\ P(B|A_i)p(A_i) \tag{2.3}$$

This new formula is called *Naive Bayes*.

All of today's most successful speech recognition systems use a "generative probabilistic model" that encapsulates this sequence of steps. The equation 2.4 for this model shows that the recognizer seeks to find the word sequence $\hat{w}_1^N$ that maximizes the word sequence's probability, given some observed acoustic sequence y_1^T (called feature vector of size T). This approach applies the Bayes' law and ignores the denominator term to maximize the product of two terms: 1) the probability of the acoustic observations given the word sequence (first part in equation 2.4) and the probability of the word sequence itself (II part in equation 2.4).

$$\begin{aligned}\hat{w}_1^N &= argmax_{w_1^N}\ p(w_1^N|y_1^T) \\ &\equiv argmax_{w_1^N}\ p(y_1^T|w_1^N)p(w_1^N)\end{aligned} \tag{2.4}$$

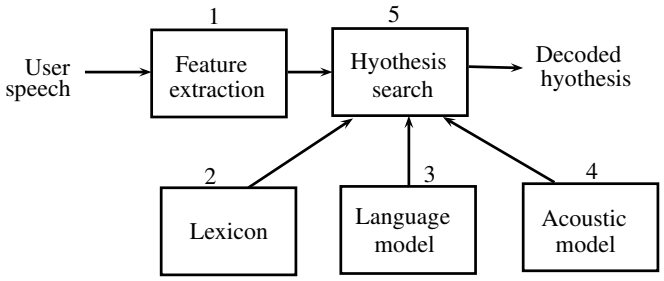

Figure 2.3: Speech recognition system.

Figure 2.3 shows the process described by Equation (2.4). In Block 1, the search extracts multidimensional features from the sampled speech signal. In Block 5, the search hypothesizes a probable word sequence. Several components drive this step:

- the lexicon in Block 2 defines the possible words that the search can hypothesize, representing each word as a linear sequence of phonemes;
- the language model in Block 3 models the linguistic structure but does not contain any knowledge about the relationship between the feature vectors (y_1^T) and the words (w_1^N), that is, the probability ($p(w_1^N|y_1^T)$) ; and
- the acoustic model in Block 4 models the relationship between the feature vectors and the phonemes, ($p(y_1^T|w_1^N)$), which means, given phonemes, you can fid out the probability of feature vectors, and then maximize it. The probability of feature vector for a given phoneme is available in the acoustic model database.

2.2.2 Feature vectors

When working with feature vectors, the most commonly used schemes extract a multidimensional vector from the sampled speech signal at a uniform frame rate, typically every 10 ms. These feature vectors provide clues about the phonemes that produced them. Consequently, the procedure for extracting them is modeled on the workings of the human auditory system.

The human auditory system simulates a constant Q-filter bank, in which the sensitivity to the energy in each channel follows a logarithmic relationship.

In a constant Q-filter bank, the ratio of the center frequencies of adjacent filters is constant, and the filter bandwidth is proportional to the center frequencies. Most feature-extraction schemes mimic these steps, with some variations in modeling perceptual aspects, such as mel-frequency versus perceptual linear prediction cepstral parameters. Because the temporal variation of the speech signal's local spectral content also contains information that helps infer spoken phonemes, the system often computes and appends the temporal derivatives and second derivatives of these features to form the final feature vector.

Table 2.1: Typical lexicon, with the word *the* with two pronunciations.

Word	Phonetic representation
The	Dhah
The	Thiy
Cat	Kaet
Pig	Pihg
Two	Tuw

2.2.3 Hypothesis Search

Three basic components comprise the hypothesis search are: a lexicon, a language model, and an acoustic model.

Lexicon. The typical lexicon shown in Table 2.1 lists each word's possible pronunciations, constructed from phonemes, of which English uses approximately 50 pronunciations per word. An individual word can have multiple pronunciations, which, complicates recognition tasks. The system chooses the lexicon on a task-dependent basis, trading off vocabulary size with word coverage. Although a search can easily find phonetic representations for commonly used words in various sources, task-dependent jargon often requires writing out pronunciations by hand.

Language model. The search for the most likely word sequence in Equation (2.4) requires the computation of two terms, $p(y_1^T|w_1^N)$ and $p(w_1^N)$. The second of these computations is called the *language model*. Its function is to assign a probability to a sequence of words w_1^N.

The simplest way to determine such a probability would be to compute the relative frequencies of different word sequences. However, the number of different sequences grows exponentially with the length of the sequence, making this approach infeasible.

A typical approximation assumes that the probability of the current word depends on the previous two words only, so that the computation can approximate the probability of the word sequence as:

$$p(w_1^T) \approx p(w_1)p(w_2)|w_1) \prod_{i=3}^{i=N} p(w_i|w_{i-1}, w_{i-2}) \tag{2.5}$$

The computation can estimate $p(w_i|w_{i-1}, w_{i-2})$ by counting the relative frequencies of word trigrams, or triplets:

$$p(w_i|w_{i-1}, w_{i-2}) \approx N(w_i, w_{i-1}, w_{i-2})/N(w_{i-1}, w_{i-2}) \tag{2.6}$$

where N refers to the associated event's relative frequency. Typically, training such a language model requires using hundreds of millions of words to estimate $p(w_i w_{i-1}, w_{i-2})$. Even then, many trigrams do not occur in the training text, so the computation must smooth the probability estimates to avoid zeros in the probability assignment.

Acoustic models. An acoustic model computes the probability of feature vector sequences under the assumption that a particular word sequence produced the vectors. Given speech's inherently stochastic nature, speakers usually do not utter a word the same way twice. The variation in a word's or phoneme's

pronunciation manifests itself in two ways: duration and spectral content, also known as acoustic observations. Further, phonemes in the surrounding context can cause variations in a particular phonemes spectral content, a phenomenon called co-articulation.

2.3 Hidden Markov models

A hidden Markov model offers a natural choice for modeling speech's stochastic aspects. HMMs function as probabilistic finite state machines. The model consists of a set of states, and its topology specifies the allowed transitions between them. At every time frame, an HMM makes a probabilistic transition from one state to another and emits a feature vector with each transition.

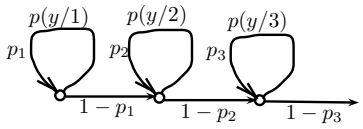

Figure 2.4: Hidden Markov model for a phoneme.

Figure 2.4 shows an HMM for a phoneme. A set of state transition probabilities–p_1, p_2, and p_3 – governs the possible transitions between states. They specify the probability of going from one state at time t to another state at time $t + 1$. The feature vectors emitted while making a particular transition represent the spectral characteristics of the speech at that point, which vary corresponding to different pronunciations of the phoneme. A probability distribution or probability density function models this variation. The functions –$p(y|1)$, $p(y|2)$, and $p(y|3)$ – could be different for different transitions. Typically, these distributions are modeled as parametric distributions–a mixture of multidimensional Gaussian, for example.

The HMM shown in Figure 2.4 consists of three states. The phoneme's pronunciation corresponds to starting from the first state and making a sequence of transitions to eventually arrive at the third state. The duration of the phoneme equals the number of time frames required to complete the transition sequence. The three transition probabilities implicitly specify a probability distribution that governs this duration. If any of these transitions exhibits high self-loop probabilities, the model spends more time in the same state, consequently taking longer to go from the first to the third state. The probability density functions associated with the three transitions govern the sequence of output feature vectors.

A fundamental operation is the computation of the likelihood that an HMM produces a given sequence of acoustic feature vectors. For example, assume that the system extracted T feature vectors from speech corresponding to the pronunciation of a single phoneme, and that the system seeks to infer which phoneme from a set of 50 was spoken. The procedure for inferring the phoneme assumes that the ith phoneme was spoken and finds the likelihood that the HMM for this phoneme produced the observed feature vectors. The system then hypothesizes that the spoken phoneme model is the one with the highest likelihood of matching the observed sequence of feature vectors.

If we know the sequence of HMM states, we can easily compute the probability of a sequence of feature vectors. In this case, the system computes the likelihood of the t-th feature vector, y_t, using the probability density function for the HMM state at time t. The likelihood of the complete set of T feature vectors is the product of all these individual likelihoods. However, because we generally do not know the actual sequence of transitions, the likelihood computation process sums all possible state sequences. Given that all HMM dependencies are local, we can derive efficient formulas for performing these calculations recursively.

2.4 Conclusion

Apart from the the methods discussed above, the other methods used are HMM (Hidden Markov Models), and ANN (Artificial Neural networks). There are benchmarking systems, for performance evaluation of different types of speech recognition systems.

2.5 Exercises

1. What are the various challenges of speech recognition?
2. What are the two application modes where automatic speech recognition can be useful? Explain.
3. Give your own version of algorithm to produce indexing of a large data-base of sound collection, e.g., of in music or speech?
4. Explain the working of speech engine? What are its inputs and outputs, as well the other resources required?
5. How the modern speech recognition takes place? Explain the formal approach used for generating phonemes, having observed the features of the sound signal.
6. What you mean by feature vectors? Give some examples.
7. Explain the Bayes theorem for modeling of conditional probability. How the Naive bayes models differs from this? Explain the naive bayes model also.

References

[1] D. Jurafsky and J. Martin, "Speech and Language Processing," *Pearson India*, 2002, Chapter 1.

[2] S. Srinivasan and E. Brown, "Is Speech Recognition Becoming Mainstream?", *Computer*, April 2002, pp. 38-41.

[3] M. Padmanabham and M. Picheny, "Large-Vocabulary Speech Recognition Algorithms", *Computer*, April 2002, pp. 43-50.