ScienceDirect

Speech Recognition System

Related terms:

Speech Recognition, Hidden Markov Models, Language Modeling, Continuous Speech, Facial Expression, Recognition Rate

Interface to the Virtual World — Input

William R. Sherman, Alan B. Craig, in Understanding Virtual Reality, 2003

Speaker-Dependent Versus Speaker-Independent Recognition.

General speech recognition systems are still largely speaker/situation dependent. Often seemingly subtle changes can affect the ability of the system to recognize commands. Differences, such as the choice of microphone or the number of people in a room, can interfere with some voice systems. It can be difficult to create or even predict the environment in which a VR experience will be run, making it difficult to train the voice system under the same circumstances in which it will need to perform.

Good speaker-independent recognition is achievable, however, if some restrictions are placed on the voice commands. Restrictions can be either a small vocabulary or a limited, well-defined grammar. The latter is often an option in applications designed to emulate military communications, which are often “by the book.”

Cognitive Computing: Theory and Applications

S. Jothilakshmi, V.N. Gudivada, in Handbook of Statistics, 2016

7.1.3 Developing Speech Recognition Systems

Development of speech recognition systems is a multistep process. First, relevant features are extracted from the speech signal. Second, reference models are developed using these features. Models are needed for each sound unit. Third, feature vectors are derived from speech utterances and are presented to all the reference models. The model which gives the highest confidence measure indicates the identity of the sound unit. The sequence of the identified sound units is validated using language models. In other words, language models are used to convert sequence of sound units into text.

Conceptually, approaches to developing speech recognition systems fall into two types: template and model based. In *template-based* approach, the system is initially trained using known speech patterns. Recognition is performed first by comparing the unknown speech signals with each possible pattern learned in the training

phase. Next, possible sequence of words that minimizes a distance function between the unknown patterns and the known pattern is computed. Well known template methods include the *DTW* and vector quantization (VQ). The DTW algorithm recognizes speech by measuring the similarity between two time series, which may vary in time or speed. In the VQ method, an ordered set of signal samples and parameters are coded by matching the input vector to a similar pattern or code vector (codeword) in a predefined codebook.

In the *model-based* systems, suitable features for each sound unit are extracted from the training data. Reference models need to be developed for each sound unit. Commonly used modeling techniques are GMM, HMM, NN, and SVM. Feature vectors are derived from the given test speech utterance and presented to all the reference models. The model which gives the highest confidence indicates the identity of the sound unit.

To achieve good results in speech recognition and SST, scalable language and acoustic modeling techniques are needed. Goodman (2001a) investigated the performance of various language modeling techniques on large data sets. As noted in Chelba et al. (2010), large training datasets improve the performance of statistical language and acoustic models (Chelba et al., 2013b). Mikolov et al. (2011) demonstrated that recurrent neural network models scale well to datasets that are hundreds of millions of words in size. However, it takes weeks to train such systems.

Robust Speech Recognition Under Noisy Ambient Conditions

Kuldip K. Paliwal, Kaisheng Yao, in Human-Centric Interfaces for Ambient Intelligence, 2010

6.2 Speech Recognition Overview

The objective of an automatic speech recognition system is to take the speech waveform of an unknown (input) utterance, and classify it as one of a set of spoken words, phrases, or sentences. Typically, this is done in two steps (as shown in Figure 6.2). In the first step, an acoustic front-end is used to perform feature analysis of the speech signal at the rate of about 100 frames per second to extract a set of features. This produces a sequence of feature vectors that characterizes the speech utterance sequentially in time.

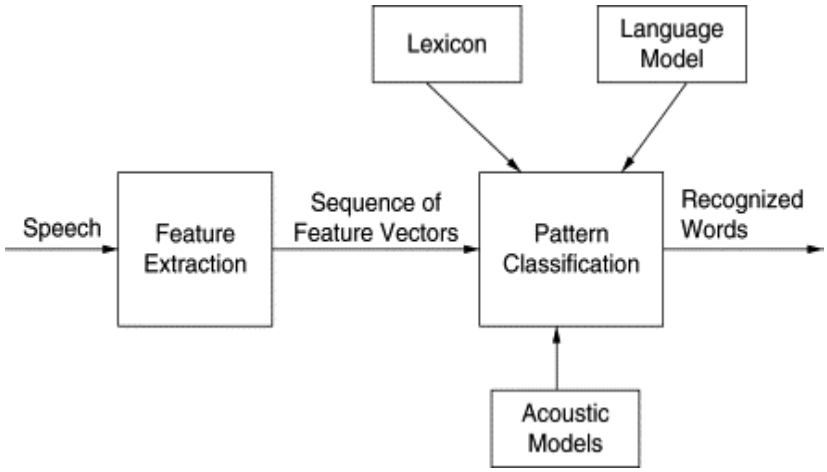

Figure 6.2. Block diagram of an automatic speech recognition system.

The second step deals with pattern classification, where the sequence of feature vectors is compared against the machine's knowledge of speech (in the form of acoustics, lexicon, syntax, semantics, etc.) to arrive at a transcription of the input utterance.

Currently, most speech recognition systems use a statistical framework to carry out the pattern classification task, and they generally recognize the input speech utterance as a sequence of words. Consider a sequence of feature vectors,

$$\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_T\}$$

representing the *T* frames of the input speech utterance. The task of the system is to find a word sequence,

$$W = \{w_1, w_2, ..., w_K\}$$

that maximizes the a posteriori probability of the observation sequence **Y**; that is, the recognized word sequence,

$$\widehat{W} = \{\widehat{w}_1, \widehat{w}_2, ..., \widehat{w}_{\widehat{K}}\}$$

is given by the following equation:

$$\widehat{W} = \underset{W}{\operatorname{argmax}} \ \Pr(W \mid \mathbf{Y}) \tag{6.1}$$

In this Equation (6.1), maximization of the a posteriori probability Pr(*W*|**Y**) is over all possible word sequences {w_1, w_2, ... , w_K} for all possible values of *K*. For a large-vocabulary continuous-speech system, this is a computationally exorbitant task. Fast search algorithms are available in the literature to carry it out [22–24].

Applying Bayes's rule and noting that Pr(**Y**) is independent of *W*, Equation (6.1) can be written as

$$\widehat{W} = \underset{W}{\operatorname{argmax}} \ \Pr(\mathbf{Y} \mid W) \cdot \Pr(W) \tag{6.2}$$

This is known as the maximum a posteriori probability (MAP) decision rule in the statistical pattern recognition literature [25].

Equation (6.2) indicates that we need two probabilities Pr(**Y**|*W*) and Pr(*W*) to carry

out the recognition task. These are computed through the acoustic and language models, respectively, which are briefly described as follows:

Acoustic models. The acoustic models are used to compute the probability Pr(**Y**|*W*). To do this, we need the probability of an observed sequence of feature vectors for each of the words in the vocabulary. This is done by representing each word by a hidden Markov model (HMM) [27] and estimating the HMM parameters from an independent (and preferably large) speech data set during the training phase. To capture the sequential nature of speech, the left-to-right HMMs are used to model individual words. For a large-vocabulary continuous-speech recognition system, it is not possible to have one HMM for each word, so we seek smaller units (subword units) to characterize these probabilities. Examples of subword units are phonemes, demisyllables, and syllables. If there are *M* phonemes in the (English) language, we can have *M* HMMs, each estimated from the training data belonging to a particular phoneme. These are called context-independent models. For a large-vocabulary speech recognition system, such models are not adequate, and one requires context-dependent modeling to get good recognition performance.

Current recognition systems use HMMs for all possible left and right contexts for each phoneme (triphone models). Once the acoustic models (in the form of HMMs) are available for individual subword units (e.g., triphones) from the training phase, the word models are constructed from the subword models according to the transcription of the words (in terms of subword units) contained in the lexicon.

Language model. The language model is used to compute the probability Pr(*W*). Note that Pr(*W*) is independent of the observed feature vector sequence **Y**. Like acoustic models, the language model is estimated from a large, independent corpus of training data. Among different language models proposed in the literature, the *N*-gram model (where *N* is typically 2, 3, or 4) is perhaps the most popular and simplest for representing the syntactic, semantic, and pragmatic sources of knowledge. In it, the probability of the current word depends on *N*–1 preceding words. Thus, it is very effective for capturing local dependencies between words. In an *N*-gram model, the probability $\Pr(w_k|w_1, w_2, \ldots, w_{k-1})$ is approximated by $\Pr(w_k|w_{k-1}, w_{k-2}, \ldots, w_{k-N+1})$. As an example, we show the procedure for calculating the probability Pr(*W*) using the trigram model (*N* = 3).

$$\begin{aligned}\Pr(W) &= \Pr(w_1, w_2, ..., w_K) \\ &= \prod_{k=1}^{K} \Pr(w_k \mid w_1, w_2, ..., w_{k-1}) \\ &= \prod_{k=1}^{K} \Pr(w_k \mid w_{k-1}, w_{k-2})\end{aligned} \tag{6.3}$$

Thus, we compute the acoustic and language models in the training phase from the data available for training the speech recognizer. Let us denote the set of acoustic models (i.e., subword HMMs) by $\Lambda_{\mathbf{X}}$ and the set of *N*-gram models by Υ_W. Then the MAP decision rule (Equation (6.2)) can be written in terms of these models as follows:

$$\widehat{W} = \underset{W}{\operatorname{argmax}} \ \Pr(\mathbf{Y} \mid W, \Lambda_{\mathbf{X}}) \cdot \Pr(W \mid \Upsilon_W) \tag{6.4}$$

During the test phase, the recognizer uses the acoustic and language models to compute the probabilities Pr(**Y**|*W*, $\Lambda_{\mathbf{X}}$) and Pr(*W*|Υ_W) and to carry out the

recognition of the input utterance according to the MAP decision rule given by Equation (6.4).

Voice Activity Detection-Based Home Automation System for People With Special Needs

Dharm Singh Jat, ... Charu Singh, in Intelligent Speech Signal Processing, 2019

6.3.1 Speech Recognition

The CMU's fastest PocketSphinx speech recognition system is able to translate speech in real-time. PocketSphinx is an open source speech recognition system, and it is a library written in C language. It is the most accurate engine for real-time application, and therefore it is a good choice for home Automation live applications [12] [13]. The chapter describes an application of controlling a home device from a research and commercial perspective.

PocketSphinx captures speech in a waveform, then splits it at utterances, maybe a cough, an "uh," or a breath in a speech. It then tries to recognize what is being said in each utterance. To achieve this, PocketSphinx takes all the possible combinations of words and tries to match them with the audio. Then, it chooses the best matching combination.

PocketSphinx is also capable of performing noise suppression. For this reason, the automation system employs PocketSphinx as both a noise-suppression model and a Speech-to-Text engine. This is done by using keyword spotting and by using an acoustic and language model. By using keyword spotting as a wake word, PocketSphinx only starts recording speech after the keyword is spoken or received. This reduces false positives and improves the accuracy of the system. CMUSphinx has several acoustic and language models such as U.S. English, German, Spanish, Indian English, etc. This allows PocketSphinx to adapt to the accent of the user. In this case, U.S. English was used in conjunction with a list from a dictionary file. The file had words representing electrical components that are to be switched on and off, and explains how the user should pronounce the words.

Fig. 6.3 shows the Raspberry Pi connected to a microphone using a USB webcam. Table 6.1 depicts the phonetic used by PocketSphinx for this research. The table contains sequences of phonemes and their respective vocabulary words that we are interested in recognizing. The dictionary file can be generated on the CMUSphinx website and be used when a narrow search of pertinent words is preferred instead of using an entire language model [14]. The dictionary file guides both PocketSphinx and the user on how words should be pronounced. That makes it easy for PocketSphinx to recognize spoken words, and reduce the words that it has to search. This improves both performance and rate of which spoken commands are converted into text. The words below represent operations the system is able to do. For example, if the user wants to switch on a light, the user would say "LIGHT" which is "L AY T" according to the PocketSphinx dictionary.

Table 6.1. Vocabulary Word and Respective Sequences of Phonemes

Vocabulary Word	Sequences of Phonemes

Vocabulary Word	Sequences of Phonemes
CLOSE	K L OW S
CLOSE(2)	K L OW Z
FAN	F AE N
GARAGE	G ER AA ZH
LIGHT	L AY T
OFF	AO F
ON	AA N
ON(2)	AO N
OPEN	OW P AH N
TIME	T AY M
TV	T IY V IY
TV(2)	T EH L AH V IH ZH AH N

Speech Recognition and Production by Machines

Chin-Hui Lee, in International Encyclopedia of the Social & Behavioral Sciences (Second Edition), 2015

Speech Recognition System Design

A block diagram of a typical speech recognition system is illustrated in the bottom panel of Figure 2, in which the feature extraction module on the left-hand side of the system converts the input signal into a sequence of feature vectors, X, which is then decoded by two modules, acoustic and language matchers, to come up with a recognized sentence. As in LPC-based speech analysis, discussed earlier, short-time Fourier transform (Rabiner and Schafer, 2010) is usually performed at every 10 ms with a 30-ms window to obtain the speech spectrum and then converted into short-time cepstrum (Rabiner and Schafer, 2010). The feature vector at every time instance, $x(t)$, is then formed by concatenating a 13-dimension cepstral vector with its first and second time derivatives to obtain a 39-dimension vector. The quantity X is simply represented by $\{x(t), t = 1, \ldots, T\}$, where T is the number of 10-ms frames in the speech utterance $\{s(t)\}$.

The two quantities needed to evaluate the product in the right-hand side of the above equation, $P(X|W)$ and $P(W)$, are obtained as follows: the acoustic matcher uses phoneme and word models to compute conditional acoustic probability, $P(X|W)$, for a given word sequence W, which can be represented by a sequence of phoneme sounds because each word w in W can also be characterized as a sequence of phonemes according to a word pronunciation dictionary. Furthermore, the language matcher uses task language models to compute word sequence probability, $P(W)$, which can be evaluated with word n-gram probabilities, trained from a large text corpus (Manning and Schutze, 1999). On the other hand the

acoustic probability is usually evaluated with hidden Markov model (HMM) (Rabiner, 1989) of phonemes. We will describe this model in detail in the next subsection.

Both models not only give good ASR performances but also fit into the framework of *finite state network* (FSN) representations of *knowledge sources* so that the speech recognition problem can be solved as a *network search* problem over a complex network representation of speech and language with each node in the network characterized by spectral variation of different sounds while each directed arc in the network representing permissible transitions between sounds following both phonotactic and syntactic models imposed by lexical and grammatical constraints shown in Figure 2. For a 20 000-word large *vocabulary* continuous speech recognition (LVCSR) task with each word on an average composed of 5 phonemes and 3 states per phoneme HMM, the FSN will have about 300 000 nodes and a few billion arcs for a trigram grammar. Moreover, the network search needs to be carried out at every 10-ms frame and this poses a tremendous computational challenge. *Dynamic programming* (DP) algorithms and *beam search* techniques have been adopted (Ney and Ortmanns, 2000) to efficiently search this type of huge networks in real time using modern hardwares.

A Real-Time Speech Emotion Recognition System and its Application in Online Learning

Ling Cen, ... Fengye Hu, in Emotions, Technology, Design, and Learning, 2016

Use of Emotion Detection in Online Learning

In the previous sections, we have shown a speech recognition system that can detect emotional states from continuous speech signal in real-time. As discussed in the Introduction section, this system, if combined together with traditional online learning systems, can be useful for making learning more effective and interesting. Specifically, it may be used as follows:

1. In the online learning environment, there lacks interaction between teachers and students. Detecting the emotional states from the verbal cues of the students in the online learning environment can help teachers understand their responses to corresponding learning content.
2. Sometimes, the emotional states of humans reflect their true response to the outside world more in facial expression than speech linguistic information, since the former is more difficult to keep hidden. Not all students wish to share their real thoughts in class. For example, when a teacher asked “have you got what I am teaching?,” some students may answer “yes” although they may not understand it well. The real response could be hiding and may not be consistent with the speech contents. Emotional states, however, can disclose their hidden thoughts. As a teacher, it is quite important to understand the true thoughts of students about the course.
3. According to the true response of students, the system can help teachers to assess student learning qualities and performance. For example, if a student is detected as being confused, he/she is more likely incapable of understanding what he/she is learning; if a student is detected with unchanged natural emotion, most of the time during class, it may be considered that he/she may

be unable to concentrate on the lesson; positive emotions can be considered as active and confident responses to learning.

4. Understanding students' true responses helps to customize online learning systems by responding to different users' emotions. This can then be used for course providers to adjust tutoring contents and the delivery speed of teaching to fit each student's learning ability. The courses can be delivered faster for the students with a higher learning ability, knowledge extension, and extra examples can be introduced for those who feel interested. Furthermore, question and answer sessions could be more specific or more detailed explanations could be properly added for the students who feel confused. Additionally, funny and interesting activities can be introduced for those students who feel bored.
5. In some cases, students themselves are not very sure how to describe their thoughts about the course. With the help of emotion detection, the learners can consciously be aware of their inner response, and accordingly adapt themselves to a better learning status during various learning stages.

Speech Summarization for Tamil Language

A. NithyaKalyani, S. Jothilakshmi, in Intelligent Speech Signal Processing, 2019

7.7.2.1 Related Work on Tamil Speech Recognition

Lakshmi et al. [41] proposed the syllable-based continuous speech recognition system. Here, the group delay-based segmentation algorithm is used to segment the speech signal in both training and the testing process and the syllable boundaries are identified. In the training process, a rule-based text segmentation method is used to divide the transcripts into syllables. The syllabified text and signal are used further to annotate the spoken data. In the testing phase, the syllable boundary information is collected and mapped with the trained features. The error rate is reduced by 20% while using the group delay based syllable segmentation approach, and so the recognition accuracy is improvised.

Radha et al. [42] proposes the automatic the Tamil speech recognition system by utilizing the multilayer feed-forward neural network to increase the recognition rate. The author-introduced system, eliminates the noise present in the input audio signal by applying the preemphasis, median, average, and Butterworth filter. Then the linear predictive cepstral coefficients features are identified and extracted from the preprocessed signal. The extracted features are classified by applying the multilayer feed-forward network, which classifies the Tamil language efficiently. The performance of the system is analyzed with the help of the experimental results, which reduces the error rate and increases the recognition accuracy.

Alex Graves et al. [43] recognize the speech features by applying the deep neural network because it works well for sequential data. The network works are based on the long- and short-term memory process to analyze the interconnection between the speech features. The extracted features are classified in terms of the connectionist temporal classification process. The implemented system reduces the error rate up to 17.7%, which is analyzed using the TIMIT phoneme recognition database.

Gales and Young [44] implement the large vocabulary continuous speech

recognition system for improving the recognition rate. The authors reduce the assumptions about the particular speech features, which are classified by applying the hidden Markov model. This model uses the various process like feature projection, discriminative parameter estimation, covariance modeling, adaption, normalization, multipass, and noise compensation process while detecting the speech feature. The proposed system reduces the error rate, and so the recognition rate is increased in an effective manner. Table 7.5 describes the performance of various speech recognition techniques and it demonstrates that the modified global delay function with the Gammatone wavelet coefficient approach yields the better recognition, comparatively.

Table 7.5. Comparison of Speech Recognition Techniques

Recognition Technique	Recognition Accuracy in Percentages
MFCC with HMM [45]	85
Mel-frequency cepstral coefficients (MFCC) with deep neural network (DNN) [46]	82.2
Gammatone cepstral coefficients (GTCC) with hidden Markov model (HMM) [47]	85.6
Gammatone cepstral coefficients (GTCC) with Deep Neural Network (DNN) [48]	88.32
Modified global delay function (MGDF) with Gammatone wavelet coefficient approach [49]	98.3
Syllable-based continuous approach [41]	80

Input

William R. Sherman, Alan B. Craig, in Understanding Virtual Reality (Second Edition), 2018

Speech Recognition (Audio Input)

Speech is another form of input available as an interface to a virtual world. As speech recognition systems become increasingly practical, they provide an excellent opportunity for natural communication with computer systems. This is especially true with VR applications, where a goal is to provide the most natural form of interface possible. The ultimate speech recognition system would understand context and use it to interpret speech, and it would be able to process a steady stream of speech from any speaker. Although many recognition systems have some of these features, this ultimate system has not quite yet been developed. Therefore, the application designer must choose which features are most important and make a choice that works within the constraints of the current technology.

In general, a speech recognition system works best when it is "trained" by the particular speaker who will be controlling the application and when it is interpreting each word as a discrete utterance rather than continuous speech. Obviously, the right type of voice input system must be combined with the goal of the application. If the application is meant for many users who will not have time to train the system to understand their voice, then it will be necessary to use a system that does not depend on training by the user. Deep learning algorithms that take advantage of cloud computing systems enable tools such as Siri to perform basic speech recognition tasks, especially when the context is known—such as setting an alarm or asking a question. Cloud-based applications such as Siri rely on access to significant computing and database retrieval operations to operate, all of which is masked from the user. If the application has a large, complex vocabulary but will be used by only a few specialists, then a speaker-trained system may work better, but also can be calculated locally.

Applications with a specific set of options can use speech recognition systems to map audio sounds to particular command strings. These strings are then matched with a set of preprogrammed possible responses.

Activation of speech recognition systems. One of the design considerations that must be addressed in any voice-controlled application is deciding when the recognition system should attend to what the user is saying. The simplest solution might be to always have the system listening to the user. However, a constantly listening system might cause problems when the user might also be speaking with people nearby. The system will continue to try to parse the conversation as commands, possibly resulting in unwanted operations. It is often wiser to implement selective attention to the user's voice.

Three methods of activating selective listening by the voice recognition system are (1) push to talk, (2) name to talk, and (3) look to talk.

Push to talk. The *push to talk* method is implemented by using a button on a handheld device or a microphone on/off switch to activate the speech software. There are many non-VR situations where push to talk is required (such as the LG "Magic Remote" device: Fig. 4-53). This method works well in scenarios that replicate traditional push to talk situations, for example, the scenario of an officer controlling a ship by verbal command to the pilot [Zeltzer and Pioch 1996]. The Siri smartphone tool is a push to talk system that handles dialing the phone or making appointments well.

Figure 4-53. This television remote enables voice commands to be recognized only when the push-to-talk button (upper right) is pressed.

Photograph by William Sherman.

Name to talk. In the *name to talk* method, the user says an activation word followed by an instruction. It is as if the user is addressing the computer (or an invisible, omnipresent agent in the virtual world) by name. An example might be: "Computer, exit application" or "Computer, please calculate a course to the Ford Galaxy." Because the VR system is always listening for the activation word, it is effectively omnipresent in the world. For example, with the Google Glass system, the wearer would indicate their desire to control the system by saying "Glass."

Look to talk. The *look to talk* method works by addressing a visible agent in the virtual world. This requires that there be an object in the virtual world that represents the agent (or multiple computer agents). The user signifies when they want an agent to perform an operation by looking at the visual representation of the agent, much like humans can tell who is being addressed based on direction of gaze ("I was looking at you, but talking to her"). This object is effectively the recognition system's avatar, and must be in the vicinity of the user for them to give commands (Fig. 4-54).

Figure 4-54. In this scenario, speech recognition is enabled when the user addresses a computer-generated avatar. Note the small, boom microphone near the participant's mouth. Other systems use a permanently mounted overhead microphone or a handheld microphone.

NICE application courtesy of Maria Roussos; photograph by William Sherman.

Speaker-dependent versus speaker-independent recognition. General speech recognition systems are still largely speaker/situation-dependent. Often seemingly subtle changes can affect the ability of the system to recognize commands. Differences, such as the choice of microphone or the number of people in a room, can interfere with some voice systems. It can be difficult to create or even predict the environment in which a VR experience will be run, making it difficult to train the voice system under the same circumstances in which it will need to perform.

Good speaker-independent recognition is achievable, however, if some restrictions are placed on the voice commands. Restrictions can be either a small vocabulary or a limited, well-defined grammar. The latter is often an option in applications designed to emulate military communications, which are often "by the book." Larger-scale speech recognition projects (like Siri), which use sizable speech databases located on remote computational servers are able to interpret speech better and allow the speaker to communicate in a more natural manner. Yet they too benefit from the context of how they will typically be used—for example, calling someone from a contact list or creating a reminder alarm.

Pros and cons of speech recognition. The advantage of voice communication is that it is a natural, unencumbering form of communication. Speech recognition technology continues to improve so that it is increasingly feasible in many more situations. However, due to the nature of speech, there are many tasks and situations in which a voice recognition system is not the best solution.

As an auditory communication channel, voice input exists over time. Speech input for control is not as instantaneous as using devices like buttons and valuators. Tasks that require precisely timed inputs, including those that are to be correlated to other physical movements, work best with physical control devices.

Another disadvantage of speech control is that speaking a command can interfere with the task being performed. This is especially true when that task requires listening or holding one's head absolutely still. On the other hand, in situations where hands need to remain still and subsecond timing is not essential, speech

recognition is advantageous because the user can trigger a command while holding their hands still.

Also, because people are accustomed to conversing with intelligent beings, they might then assume that they are doing so when communicating with artificial entities in a virtual world, even though it is unlikely that an underlying natural language understanding process is evaluating the incoming communication stream and parsing the semantics of the request. Or, they might believe that computers can easily understand our languages, as portrayed in popular science fiction or when processing actually takes place on larger, remotely located computers dedicated to this one task.

Speech can be a very beneficial form of interaction because in many VR systems there usually isn't a keyboard available to enter commands. It is generally impractical to use a keyboard when wearing a head-mounted display when the keyboard cannot be seen (although not impossible, and the ability to see a keyboard from within an HMD can be accomplished by capturing that portion of the real world using world-capturing technology such as the Leap Motion or Microsoft Kinect which can bring representations of the hands and real keyboard into the virtual).

Speech recognition summary. Overall, speech recognition systems can play an important role in making a VR experience more immersive and natural to use. However, until systems become capable of perfect recognition of continuous speech, the choice of system will need to be tailored to the particular task. Also, regardless of how good recognition systems become, there are interface tasks for which speech input is not appropriate. Indeed, there are times when we don't want others nearby to overhear our conversations with our virtual worlds.

Probabilistic methods

Ian H. Witten, ... Christopher J. Pal, in Data Mining (Fourth Edition), 2017

Hidden Markov Models

Hidden Markov models have been widely used for pattern recognition since at least the 1980s. Until recently most of the major speech recognition systems have consisted of large Gaussian mixture models combined with hidden Markov models. Many problems in biological sequence analysis can also be formulated in terms of hidden Markov models, with various extensions and generalizations.

A hidden Markov model is a joint probability model of a set of discrete observed variables O={O_1,...,O_T} and discrete hidden variables H={H_1,...,H_T} for T observations that factors the joint distribution as follows:

$$P(\mathrm{O},\mathrm{H}) = P(H_1)\prod_{t=1}^{T} P(H_{t+1}\,|\,H_t)\prod_{t=1}^{T} P(O_t\,|\,H_t),$$

Each O_t is a discrete random variable with N possible values, and each H_t is a discrete random variable with M possible values. Fig. 9.18C illustrates a hidden Markov model as a type of Bayesian network that is known as a "dynamic" Bayesian network because variables are replicated dynamically over the appropriate number of time steps. They are an obvious extension of first-order Markov models, and it is common to use "time-homogeneous" models where the transition matrix $P(H_{t+1}|H_t)$ is the same at each time step. Define **A** to be a transition matrix whose

elements encode $P(H_{t+1}=j|H_t=i)$, and **B** to be an emission matrix **B** whose elements b_{ij} correspond to $P(O_t=j|H_t=i)$. For the special $t=1$ the initial state probability distribution is encoded in a vector π with elements $\pi_i=P(H_t=i)$. The complete set of parameters is $\theta = \{\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}\}$, a set containing two matrices and one vector. We write a particular observation sequence as a set of observations $\widetilde{\mathrm{O}} = \{O_1 = o_1, ..., O_T = o_T\}$.

Hidden Markov models pose three key problems:

1. Compute $P(\widetilde{\mathrm{O}}; \theta)$, the probability of a sequence under the model with parameters θ.
2. Find the most probable explanation—the best sequence of states $H^* = \{H_1 = h_1, ..., H_T = h_T\}$ that explains an observation.
3. Find the best parameters θ for the model given a data set of observed sequences.

The first problem can be solved using the sum-product algorithm, the second using the max-product algorithm, and the third using the EM algorithm for which the required expectations are computed using the sum-product algorithm. If there is labeled data for the sequences of hidden variables that correspond to observed sequences, the required conditional probability distributions can be computed from the corresponding counts in the same way as we did with Bayesian networks.

The only difference between the parameter estimation task when using an hidden Markov model viewed as a dynamic Bayesian network and the updates used for learning the conditional probability tables in a Bayesian network is that one can average over the statistics obtained at each time step, because the same emission and transition matrices are used at each step. The basic hidden Markov model formulation serves as a point of reference when coupling more complex probabilistic models over time using more general dynamic Bayesian network models.

Breaking the Robustness Barrier: Recent Progress on the Design of Robust Multimodal Systems

Sharon Oviatt, in Advances in Computers, 2002

2.1 Recognition Errors in Unimodal Speech Systems

Spoken language systems involve recognition-based technology that by nature is probabilistic and therefore subject to misinterpretation. Benchmark error rates reported for speech recognition systems still are too high to support many applications [78], and the time that users spend resolving errors can be substantial and frustrating. Although speech technology often performs adequately for read speech, for adult native speakers of a language, or when speaking under idealized laboratory conditions, current estimates indicate a 20–50% decrease in recognition rates when speech is delivered during natural spontaneous interactions, by a realistic range of diverse speakers (e.g., accented, child), or in natural field environments.

Word error rates (WERs) are well known to vary directly with speaking style, such that the more natural the speech delivery the higher the recognition system's WER.

In a study by Weintraub *et al.* [79], speakers' WERs increased from 29% during carefully read dictation, to 38% during a more conversationally read delivery, to 53% during natural spontaneous interactive speech. During spontaneous interaction, speakers typically are engaged in real tasks, and this generates variability in their speech for several reasons. For example, frequent miscommunication during a difficult task can prompt a speaker to hyperarticulate during their repair attempts, which leads to durational and other signal adaptations [80]. Interpersonal tasks or stress also can be associated with fluctuating emotional states, giving rise to pitch adaptations [81].

Basically, the recognition rate degrades whenever a user's speech style departs in some way from the training data upon which a recognizer was developed. Some speech adaptations, like hyperarticulation, can be particularly difficult to process because the signal changes often begin and end very abruptly, and they may only affect part of a longer utterance [80]. In the case of speaker accents, a recognizer can be trained to recognize an individual accent, although it is far more difficult to recognize varied accents successfully (e.g., Asian, European, African, North American), as might be required for an automated public telephone service. In the case of heterogeneous accents, it can be infeasible to specifically tailor an application to minimize highly confusable error patterns in a way that would assist in supporting robust recognition [53].

The problem of supporting adequate recognition rates for diverse speaker groups is due partly to the need for corpus collection, language modeling, and tailored interface design with different user groups. For example, recent research has estimated that children's speech is subject to recognition error rates that are two-to-five times higher than adult speech [82–85]. The language development literature indicates that there are specific reasons why children's speech is harder to process than that of adults. Not only is it less mature, children's speech production is inherently more variable at any given stage, and it also is changing dynamically as they develop [86,87].

In addition to the many difficulties presented by spontaneous speech, speaker stylistic adaptations, and diverse speaker groups, it is widely recognized that laboratory assessments overestimate the recognition rates that can be supported in natural field settings [88–90]. Field environments typically involve variable noise levels, social interchange, multitasking and interruption of tasks, increased cognitive load and human performance errors, and other sources of stress, which collectively produce 20–50% drops in speech recognition accuracy. In fact, environmental noise currently is viewed as one of the primary obstacles to widespread commercialization of spoken language technology [89,91].

During field use and mobility, there actually are two main problems that contribute to degradation in system accuracy. The first is that noise itself contaminates the speech signal, making it harder to process. Stationary noise sources often can be modeled and processed successfully, when they can be predicted (e.g., road noise in a moving car). However, many noises in natural field environments are *nonstationary* ones that either change abruptly or involve variable phase-in/phase-out noise as the user moves. Natural field environments also present qualitatively different sources of noise that cannot always be anticipated and modeled. Speech technology has special difficulty handling abrupt onset and nonstationary sources of environmental noise.

The second key problem, which has been less well recognized and understood, is

that people speak differently under noisy conditions in order to make themselves understood. During noise, speakers have an automatic normalization response called the “Lombard effect” [92], which causes systematic speech modifications that include increased volume, reduced speaking rate, and changes in articulation and pitch [58,91,93–95]. The Lombard effect not only occurs in human adults, but also in young children, primates, quail, and essentially all animals [96–98]. From an interface design standpoint, it is important to realize that the Lombard effect essentially is reflexive. As a result, it has not been possible to eliminate it through instruction or training, or to suppress it selectively when noise is introduced [99].

Although speech originally produced in noise actually is *more* intelligible to a human listener, a system’s recognition accuracy instead degrades when it must process Lombard speech [91].

To summarize, current estimates indicate a 20–50% decrease in recognition rate performance when attempts are made to process natural spontaneous speech, or speech produced by a wider range of diverse speakers in real-world field environments. Unfortunately, this is precisely the kind of realistic speech that must be recognized successfully before widespread commercialization can occur. During the development of modern speech technology there generally has been an overreliance on hidden Markov modeling, and a relatively singular focus on recognizing the phonetic features of acoustic speech. Until very recently, the speech community also has focused quite narrowly on unimodal speech processing. Finally, speech recognition research has depended very heavily on the word error rate as a forcing function for advancing its technology. Alternative perspectives on the successful development of robust speech technology will be discussed throughout this chapter.