# Chapter 2
# Automatic Speech Recognition

Automatic speech recognition (ASR) systems convert speech from a recorded audio signal to text. Humans convert words to speech with their speech production mechanism. An ASR system aims to infer those original words given the observable signal. The most common and as of today best method is the probabilistic approach. A speech signal corresponds to any word (or sequence of words) in the vocabulary with a certain probability. Therefore, assuming a word $x$ or word sequence $X$ was spoken, we compute a score for matching these words with the speech signal. This score is calculated from the acoustic properties of speech sub-units (phonemes in the acoustic model), linguistic knowledge about which words can follow which other words. Including additional knowledge as the pronunciation score proposed in this work has also shown to be helpful. Finally, we sort the possible word sequence hypotheses by score, and pick the hypothesis with the highest score as recognition result.

The outline of a typical speech recognition system is shown in Fig. 2.1. The process of speech recognition can be divided into the following consecutive steps.

- pre-processing (which includes speech/non-speech segmentation,)
- feature extraction,
- decoding, the actual recognition employing an acoustic and language model as well as a dictionary,
- result post-processing.

In the following, these steps will be described in more detail.

## 2.1 Relevant Keywords from Probability Theory and Statistics

In order to understand automatic speech recognition, it is helpful to briefly review some key concepts from general probability theory. For further reading, we refer to [Huang 01, Bronstein 07].

R.E. Gruhn et al., *Statistical Pronunciation Modeling for Non-Native Speech Processing*, Signals and Communication Technology, DOI: 10.1007/978-3-642-19586-0_2,

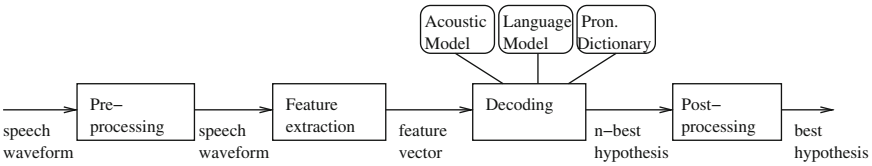


**Fig. 2.1** Schematic outline of a typical speech recognition system

**Table 2.1** Example for a discrete probability distribution

| | $X = a$ | $X = b$ | … | $X = x$ |
|---|---|---|---|---|
| P(X) | 0.1 | 0.07 | … | 0.09 |

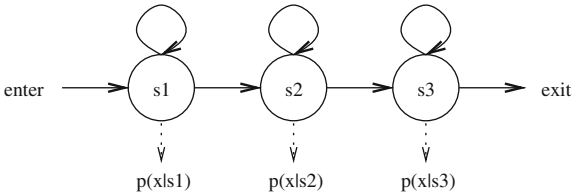


**Fig. 2.2** A three-state hidden Markov model

### 2.1.1 Discrete and Continuous Probability Distribution

A discrete random variable $X$ describes random experiments. An example is rolling a dice, where $X \in [1...6]$. The probability distribution $P(X = i)$ can be given as shown in Table 2.1. That means, p(x) is a non-parametric discrete distribution.

In the example shown in Table 2.1, X was a natural number. For the continuous case with $X \in \mathbb{R}$, we have a probability density function $p(X = x)$ with $\int_{-\infty}^{\infty} p(X)dX = 1$. The true data density function is estimated with a parametric curve. A mixture of N Gaussians is a good approximation to most densities:

$$p(X) = \sum_{i=1}^{N} \frac{1}{(\sqrt{2}\pi) \mid \sigma_i \mid} \exp\left[-\frac{(X - \mu_i)^2}{2\sigma_i^2}\right] \tag{2.1}$$

with $\mu_i$ the mean and $\sigma_i$ the variance of the $i$th Gaussian.

### 2.1.2 A Hidden Markov Models

A hidden Markov model (HMM) is a statistical model for a Markov process with hidden parameters. Figure 2.2 shows an example for a three-state HMM. Each state $s_i$ has a probability density $p_i, p(x|s_i)$ more precisely states the probability

density for the acoustic observation $x$ for the state $s_i$. The three states $s_1, s_2$ and $s_3$ together form the HMM for the word $S$. The observations $X$ could be acoustic observations or, in case of the pronunciation models, discrete phoneme symbols.

An HMM can be characterized by two sets of parameters: The transition matrix $A$ of the probabilities $a_{s_{t-1}s_t}$ to go from one state to another (including self-loops) and the output probabilities $B$ characterized by the densities.

### 2.1.3 Estimating HMM Parameters

HMMs are trained on data samples with the forward-backward or Baum-Welch algorithm [Baum 66], which is similar to the EM-algorithm [Schukat-Talamazzini 95]. There is no analytical method to determine the HMM parameters, the Baum-Welch algorithm performs an iterative estimation, which monotonously improves the HMM parameter set $\Phi = (A, B)$. [Huang 01] describes the algorithm in four steps as follows:

1. Initialization: Choose an initial estimate $\Phi$
2. E-Step: Compute the auxiliary function $Q(\Phi, \Phi')$ based on $\Phi$
3. M-step: Compute $\Phi'$ to maximize the auxiliary $Q$-function.
4. Iteration: Set $\Phi = \Phi'$, repeat from step 2

with $Q$ defined as

$$Q(\Phi, \Phi') = \sum_S \frac{P(X, S|\Phi}{P(X|\Phi)} \log P(X, S|\Phi') \tag{2.2}$$

where

$$P(X, S|\Phi) = \prod_{t=1}^{T} a_{s_{t-1}s_t} p_{s_t}(x_t) \tag{2.3}$$

The initial parameters play an important role for this algorithm. There is no guarantee that the algorithm will converge, the ideal number of iterations is typically determined heuristically.

HMM models are trained on data, for example an acoustic model on a speech database. If the corpus includes the data of a sufficient number of speakers, it can be assumed to be general, i.e. the model will acceptably represent the properties of speakers not observed in the training database. Such a model is called speaker-independent.

If the model is trained on a group of specific speakers, it is considered group-dependent; for example a model trained on speakers from an accent group is accent-dependent. It is theoretically possible to train a speaker-dependent model on the data of a single speaker. However, such models are less reliable as there is a

high chance that many relevant data items (such as phonemes in certain contexts) have not been uttered by that speaker. There are algorithms to adapt a general acoustic model to the properties of a specific speaker, such as the maximum a posteriori (MAP) adaptation. Such models will have increased recognition performance for the regarding subject but less match other speakers.

The MAP estimate to optimize the parameters $\Phi'$ of an HMM can be expressed as follows [Huang 01]:

$$\Phi' = \text{argmax}_{\Phi}[p(\Phi|X)] = \text{argmax}_{\Phi}[p(X|\Phi), p(\Phi)] \tag{2.4}$$

This equation can be solved with the EM algorithm, with the $Q$-function defined as:

$$Q_{MAP}(\Phi, \Phi') = \log p(\Phi') + Q(\Phi, \Phi') \tag{2.5}$$

## 2.2 Phonemes

Speech is composed of certain distinct sounds. A phoneme is defined as the smallest unit of speech that distinguishes a meaning. Phonemes are characterized by the way they are produced, especially:

- place of articulation,
- manner of articulation,
- voicing.

For each language, there is a specific set of phonemes. There are several notations how to transcribe these phonemes. The most notable is the International Phonetic Alphabet (IPA) [Association 99] which was developed by the International Phonetic Association beginning in 1888 with the goal of describing the sounds of all human languages. IPA consists of many symbols that are not normal Latin characters, i.e. not included in the standard ASCII codepage. This makes it inconvenient to use them on computers.

Several systems have been proposed to transcribe phonemes with ASCII symbols, with SAMPA [Wells 95] and ARPAbet [Shoup 80] being the most common. For these phonetic alphabets mapping tables exist that convert from one phonetic alphabet to the other, there is no fundamental difference between the alphabets that could influence experimental results. In this work phonemes are given in the ARPAbet notation, as the acoustic model is trained on the WSJ database and the pronunciation dictionary for that data is provided in ARPAbet notation.

For acoustic models in speech recognition, the units can be phonemes or units considering phonemes and their acoustic contexts. Units considering only the left or right context are called biphones, if they consider left and right context, they are called triphones.

## 2.3 Prosody

Besides the phonemes that carry the textual content of an utterance, prosodic information [Noeth 90] gives valuable support to understand a spoken utterance. In short, prosody is the rhythm, stress and intonation of continuos speech, and is expressed in pitch, loudness and formants. Prosody is an important mean of conveying non-verbal information.

Fujisaki [Fujisaki 96] considers two separate aspects of prosody, the "concrete aspect"—defining prosody in physical term, and the "abstract aspect' '—defining prosody as influence to linguistic structure.

concrete aspect: phenomena that involve the acoustic parameters of pitch, duration, and intensity

abstract aspect: phenomena that involve phonological organization at levels above the segment

Prosody in speech has both, measurable manifestations and underlying principles. Therefore the following definition is appropriate:

Prosody is a systematic organization of various linguistic units into an utterance or a coherent group of utterances in the process of speech production. Its realization involves both segmental features of speech, and serves to convey not only linguistic information, but also paralinguistic and non-linguistic information.

The individual characteristics of speech are generated in the process of speech sound production. These segmental and suprasegmental features arise from the influence of linguistic, paralinguistic, and nonlinguistic information. This explains the difficulty of finding clear and unique correspondence between physically observable characteristics of speech and the underlying prosodic organization of an utterance.

Linguistic information: symbolic information that is represented by a set of discrete symbols and rules for their combination i.e. it can be represented explicitly by written language, or can be easily and uniquely inferred from the context.

Paralinguistic information: information added to modify the linguistic information. A written sentence can be uttered in various ways to express different intentions, attitudes, and speaking styles which are under conscious control of the speaker.

Nonlinguistic information: physical and emotional factors, like gender, age, happiness, crying, … which cannot be directly controlled by the speaker. These factors are not directly related to (para-) linguistic contents, but influence the speech anyway.

Prosodic characteristics are typically expressed in several types of features, which can serve as basis for automatic recognition. The most prominent of those features are duration, loudness, pitch and glottal characteristics.

### *2.3.1 Duration*

Utterances can be lengthened or shortened; the relative length carries prosodic information. For example, [Umeno 03] shows that short non-verbal fillwords show affirmation, whereas lengthened fillwords express disagreement.

### 2.3.2 Power

The signal power or loudness of an utterance is another important prosodic feature. In German and English the intensity often marks or emphasizes the central information of a sentence. Without this information, spontaneous speech could be ambiguous and easily misunderstood. The loudness is measured by the intensity of the signal energy.

### 2.3.3 Pitch

At the bottom of the human vocal tract are the vocal cords, or glottis. For unvoiced speech, the glottis remains open, for voiced speech it opens and closes periodically. The frequency of the opening is called the fundamental frequency or pitch. It can be calculated from the spectrum [Kiessling 92] and its contour over the utterance reveals several information. E.g. in Mandarin Chinese, the F0 carries phonetic/lexical information, and in English or German, the pitch specifies a question by a final fall-rise pattern [Sun 06, Waibel 88].

### 2.3.4 Glottal Characteristics

Physiological voice characteristics also contribute to convey non-verbal information. The glottis is the vocal cord area of the human articulatory system and is most commonly known for creating voicing in pronunciation by opening and closing periodically. The interpretation and extraction of glottal characteristics directly from the waveform without the need of special recording equipment is described in literature [Hanson 97].

## 2.4 Speech to Text

In this section, we describe the steps from speech to text, beginning with recording and pre-processing, over feature calculation to decoding and finally rescoring.

### 2.4.1 Pre-processing

Speech is recorded with a microphone and the signal is discretized with a sampling frequency of e.g. 16 kHz. The Shannon sampling theorem states that a bandwidth limited signal can be perfectly reconstructed if the sampling frequency

is more than double of the maximum frequency. That means that in the sampled data, frequencies up to almost 8 kHz are constituted correctly. While this is not the total frequency range of human speech, it is more than double of what is transmitted over telephone networks. These are typically limited to the 5 Hz–3.7 kHz range, and has shown in research to be sufficient for speech recognition applications. It is possible to remove frequencies below 100 Hz with a high-pass filter as they tend to contain noise but can be considered of little relevance for speech recognition.

An important part of pre-processing is also speech/non-speech segmentation. As speech recognition systems will classify any sound to any phoneme with some (even if very low) probability, background noise can cause insertions of phonemes or words into the recognition result if the noise resembles the parameters of a phoneme model better than those of a silence model. Such insertions can be reduced by removing areas from the speech signal between the start of the recording and the point of time when the user starts to speak, and after the end of the utterance. This segmentation process is also called end point detection.

Signal energy based algorithms have been available for a long time [Rabiner 75, Reaves 93, Junqua 94]. When the signal energy exceeds a given threshold, the start of a speech segment is detected. When the signal drops below a certain threshold, the speech segment ends. As there are phonemes with low signal energy and short pauses between words, this algorithm must be enhanced by time windowing or additional prosodic features [Noeth 90, Gruhn 98] such as voicedness to be reliable. Another common approach is based on Gaussian mixture models [Binder 01]. Video signals can also be very helpful in detecting speaking activity [Murai 00, Nishiura 01a, Nishiura 01b, Nishiura 02a, Nishiura 02b].

### *2.4.2 Feature Extraction*

To calculate features, acoustic observations are extracted over time frames of uniform length. Within these frames, the speech signal is assumed to be stationary. The length of these frames is typically around 25 ms, for the acoustic samples in this window one multi-dimensional feature vector is calculated. The time frames are overlapping and shifted by typically 10 ms. On the time window, a fast Fourier transformation is performed, moving into the spectral domain.

Human ears do not perceive all frequency bands equally. This effect can be simulated with band-pass filters of non-uniform frequency band widths. Until 500 Hz, the width of the filters is 100 Hz, after that it increases logarithmically. The filter center frequencies are defined in the so called Mel scale. The spectrum is decorrelated with a discrete cosine transformation. Of the resulting coefficients, the first coefficients carry the most significance. Therefore only the first e.g. ten coefficients are selected as feature vector. The resulting features are called Mel cepstra, commonly abbreviated as MFCC. Usually the normalized energy is appended to the feature vector.

A common further feature processing step is cepstral mean subtraction (CMS). The goal of CMS is to remove the effects of a linear filter. For example the microphones for recording the training data and the microphone during testing are often different, CMS can contribute to recover from this effect. The average of the features is calculated for every utterance, and subtracted from each feature vector. As a result, the features during both test and training have a mean of zero.

Information lies not only in the feature vector itself, but also in the temporal change. There are two common approaches how to capture this information:

- create a supervector concatenating consecutive feature vectors,
- append the derivatives and second derivatives to the feature vector.

The first method will lead to a vector of very high dimensionality that must be projected down to a lower dimension with algorithms like principal component analysis or linear discriminant analysis [Vasquez 08]. Dimensionality reduction can also be applied to a feature vector with derivatives.

### *2.4.3 Decoding*

Decoding is the process to calculate which sequence of words is most likely to match to the acoustic signal represented by the feature vectors. For decoding three information sources must be available:

- an acoustic model with an HMM for each unit (phoneme or word,)
- a dictionary, typically a list of words and the phoneme sequences they consist of,
- a language model with word or word sequence likelihoods.

A prerequisite for decoding is to know which words can be spoken. Those words are listed in the dictionary, together with the according phoneme sequence. The acoustic model typically has a probability density function that is a mixture of Gaussians and gives a likelihood for each observed vector $p(x|w)$.

A language model is not an absolute requirement for decoding but increase word accuracy [Tanigaki 00]; in some cases like a credit card recognition system with a vocabulary consisting of the numbers 0–9, it can be acceptable to consider all words equally likely. Language models are typically fixed grammars or n-gram models. A 1-gram model lists words and their likelihoods, a 2-gram model lists words and their likelihood given a preceding word and so on, providing the word probability $p(w)$.

During decoding, we search for the word(s) $w^*$ that fits best to the observation $X$, as given in this fundamental equation:

$$w^* = \text{argmax}_w(p(X|w)p(w)) \tag{2.6}$$

with $p(w)$ Coming from the language model and $p(X|w)$ calculated from the sequence of phonemes in the word as defined by the dictionary:

$$p(X|w) = \text{argmax}_s(\prod_j (p(x|s_j)p(s_j))) \tag{2.7}$$

In theory, it is also necessary to consider $p(x)$, but as this term is the same for all competing hypotheses, it can be neglected.

As the space of possible state sequences is astronomically large, it is not possible to calculate the probabilities of all existing paths through the state network: for $T$ observations and $N$ states, the complexity is $O(N^T)$. To find the most likely sequence of hidden states, the Viterbi search algorithm [Viterbi 79] is employed. It can be summarized into four steps [Jelinek 97, Huang 01] as follows: To find the optimal state sequences, find the maximizing sequence $s = s_1, \ldots, s_j, \ldots, s_{i-1}, s_i$ whose probability is $V_t(i)$ to generate the observation $X_t$ at time $t$ and ends in state $i$.

- Initialization: Set $V_0(s_0) = 1$
  $V_1(s) = \max_{s'} p(x_1, s|s')V_0(s') = p(x_1, s|s_0)$
- Induction: $V_t(s) = \max_{s'} Vt - 1(s')p(x_{t-1}, s|s')$ i.e. for each state $s_j$ at time $t$ keep only one path that leads to this state and discard all paths with lower probability.
- Termination: Find the state $s^*$ at the end of the state sequence $s_i$ where $V_t(s)$ is maximal.
- Traceback: Trace back from this state $s^*$ to the initial state along the remaining transitions. The states along this path constitute the most likely state sequence $S^* = (s_1^*, s_2^*, \ldots, s_T^*)$.

The complexity of the Viterbi algorithm is only $O(N^2T)$. The list of all permitted paths in the state network is called the lattice.

The dictionary contains a list of all words defined in a recognition scenario and the phoneme sequence (and thereby HMM phoneme model sequence) of each word. If the number of words is very small and those words are acoustically different, very high speech recognition accuracies can be achieved. The larger the dictionary, the more confusions are possible, leading to decreasing recognition rates. The confusability depends not directly on the number of words, but on the number of entries, i.e. pronunciation alternatives. Words can be spoken differently even in native language, such as the digit 0 in English as /zero/ or /o/. Hitherto pronunciation dictionaries typically have more entries than words. Other than specialized recognition tasks such as digit recognition, most large vocabulary speech recognition systems have dictionary sizes of several 1,000 words, dictation systems can reach several 10,000 words. In a real-world system it is not always possible to predict which words a user might say. The larger the dictionary, the less words fail recognition as out of vocabulary, but adding many new words with similar phonemes leads to additional confusions and decreases recognition rates. The language model is also affected by vocabulary size: The smaller the number of words, the sharper a language model can describe the permitted word sequences, leading to a direct relation between number of words and recognizer performance.

### 2.4.4 Post-processing

The result of the Viterbi search is not a single sequence of words, but a list of all possible hypotheses sorted by total score. In practice, this number is usually limited to the five or ten best hypotheses, the so-called n-best list. Rescoring this list by employing additional sources of information is a common method to improve the recognition accuracy of the top-scoring result.

A possible source of information is a higher-order language model. As they require much more resources than unigram or bigram models, both in terms of memory and computation cost, combinations of bigram model for decoding followed by trigram model based rescoring are common.

In this work, we provide additional information about the pronunciation with pronunciation models and apply them with a rescoring algorithm.

## 2.5 Applying Speech Recognition

In order to apply a speech recognition system, it is necessary to judge the performance and to put it in a system environment with other related modules.

### 2.5.1 Evaluation Measures

The most common measures to evaluate the performance of a speech recognition system are correctness, accuracy and error. These measures can be calculated on phoneme and word level. There are three types of mistake a speech recognition system can make:

- Substitution: At the position of a unit (word or phoneme), a different unit has been recognized.
- Deletion: In the recognition result a unit is omitted.
- Insertion: The result contains additional units than were not spoken.

The type of error is determined by comparison with the correct transcription. Some transcriptions contain information about noises in the speech signal. Even though human noises can contain relevant information [Svojanovsky 04], omitting those non-verbal sounds is not counted as recognition mistake. Likewise, an acoustic model may contain special models to handle human noises such as breathing and lip smacks or environment noises; additional noises in the recognition result are not counted as insertion.

Correctness is defined as:

$$\text{Corr} = \frac{N - D - S}{N} \tag{2.8}$$

with $N$ being the number of units in the correct transcription, $D$ the number of deletions and $S$ the number of substitutions. The accuracy is similar to the correctness, but additionally takes the number of insertions $I$ into account:

$$Acc = \frac{N - D - S - I}{N} \tag{2.9}$$

Both measures are commonly given in percentage notation. Finally the error rate is the sum of all types of errors divided by number of words in the transcription. The error rate is closely related to the accuracy, usually only one of the two numbers is given.

$$Err = \frac{D + S + I}{N} = 100\% - Acc \tag{2.10}$$

In case the units are words, the error rate is commonly abbreviated WER, standing for word error rate.

To calculate correctness and accuracy, the transcription is compared to the highest scoring recognition result. In some cases it is necessary to determine the best matching path among all possible paths in the recognition network to measure the best theoretically achievable result. Such an extended evaluation yields the so-called network correctness or network accuracy.

### 2.5.2 Speech Dialog Systems

In real-world applications, speech recognition are not employed stand-alone, but embedded in an larger architecture that also involves other modules. The most perceivable module other than speech recognition is speech synthesis or text-to-speech (TTS). A TTS system receives a text string (and optionally phonetics) and generates an audio file, which is played to the user as feedback.

Recognition and TTS are the user communication interfaces of a speech dialog system (SDS). Depending on the application, an SDS also includes one or several backend modules. Most important is a dialog manager, which implements the dialog strategy [Minker 02b], i.e.:

- keep track of the information the user has already uttered,
- know which data must be available in order to fulfil a given task,
- trigger database queries,
- decide the next dialog step, especially feedback or question to the user.

Frequently employed other modules include a database access module or a natural language understanding module [Minker 98b] which extracts keywords and data from spontaneous utterances. Typical applications for SDSs are information retrieval services, such as travel assistance systems (e.g. [Seneff 98]) for hotel reservation, ticket booking etc., with a telephony or information kiosk interface.

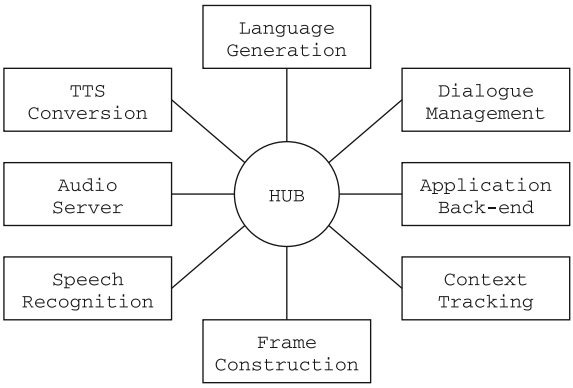


**Fig. 2.3** The DARPA Communicator [Seneff 98]

Also the control of a car navigation system can base on an SDS, keeping track of previously input information and guiding the user during navigation destination input.

There are several well-known systems to combine the modules to a SDS. The "DARPA Communicator", also known by the name of its reference implementation "Galaxy", is a modular architecture developed at MIT [Seneff 98]. A schematical layout is shown in Fig. 2.3. It was designed for speech recognition based information retrieval systems. It consists of a central hub which interfaces between servers like audio server, dialog manager, database etc. The hub behavior is defined through a set of rules implemented in a special scripting language.

A speech-to-speech translation system can be seen as a special case of an SDS, consisting of interacting modules, but without dialog manager or database backend [Gruhn 01b, Gruhn 01c]. As such systems are not necessarily applied in fixed places where computers with large processing power are available, but "in the field", access to translation servers from lightweight clients is a key feature. Popular approaches include cellular phones [Gruhn 00b, Gruhn 00a] or small-scale portable computers such as PDAs [Gruhn 99, Singer 99a].

User utterances are recognized, translated to a pre-selected target language and then played back with a TTS. To reduce the effect of recognition and translation errors, feedback such as display of the recognized result is crucial. SMILE is a translation system based on the architecture standard CORBA (Common Object Request Broker Architecture [OMG 90]), supporting various input types, including close-talking microphone and telephony hardware. As shown in Fig. 2.4, the modules are operating independently and event-driven without a managing module to steer the flow of information. The client interface is a reporting tool that provides final and intermediate results to the conversation partners. Each of the modules can be included several times on distributed servers for high-speed performance.

While evaluating a speech recognition system with correctness and accuracy is quite straightforward, evaluating a speech dialog system [Minker 98a, Minker 02a] or a speech-to-speech translation system [Sugaya 00] is still a open research topic.

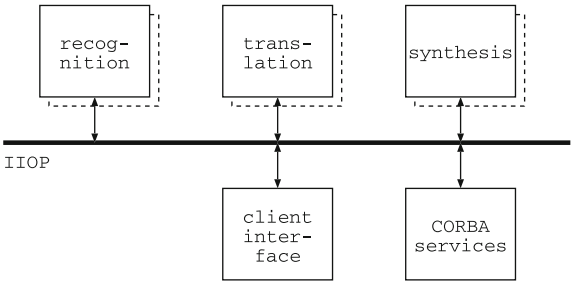


**Fig. 2.4** The speech and multimodal interface for multi-lingual exchange (SMILE) architecture

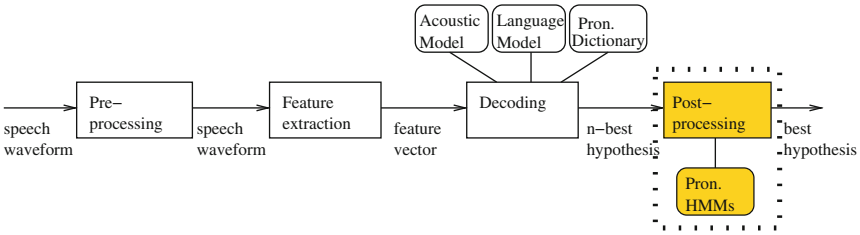


**Fig. 2.5** The focus in this work lies on post-processing

Additionally to the speech recognition evaluation, task completion rate is a popular measure: The share of dialogs in which the human customer was able to achieve his goal, such as getting a specific required information or booking a hotel room.

### *2.5.3 Focus of This Work*

This research bases on a standard automatic speech recognition system. The modules from speech recording to decoding follow the common practices as described in this chapter. Our novel approach described in Chap. 7 focusses on the post-processing part, as shown in Fig. 2.5. Here we apply discrete HMMs as statistical pronunciation models for rescoring an n-best list. These discrete models are trained as described in Sect. 2.1.3 with the result of a phoneme recognition as input data.

http://www.springer.com/978-3-642-19585-3

Statistical Pronunciation Modeling for Non-Native Speech Processing

Gruhn, R.E.; Minker, W.; Nakamura, S.

2011, X, 114 p., Hardcover
ISBN: 978-3-642-19585-3