Automatic Speech Recognition

Notes on Speech and Audio Processing

Chia-Ping Chen

Department of Computer Science and Engineering
National Sun Yat-Sen University
Kaohsiung, Taiwan ROC

Feature Extraction

The goal of feature extraction is to find a representation for speech sound. Ideally, speech features are

- **discriminative**: The representations for different linguistic targets are distinct.
- **robust**: The representations are not easily and severely corrupted by environmental noise.
- **parsimonious**: Achieve the same performance with the number of features as small as possible.

Common Feature Vectors

Currently, most ASR systems uses a cepstral vector derived from a filter bank or linear prediction. The basic steps include

- estimate power spectrum of an analysis window
- integrate power spectrum over filter banks
- spectral adjustment by equal-loudness (or pre-emphasis)
- compress spectrum by cubic root or logarithm
- apply inverse DFT to get cepstrum
- further processing such as spectral smoothing, orthogonalization and liftering.

MFCC, LPC, and PLP

- The previous processings are used in MFCC (mel-frequency cepstral coefficients) and PLP (perceptual linear prediction).
- Both provide a representation of smoothed power spectrum that has been compressed.
- It is also interesting to compare LPC and PLP, as shown in Figure 22.4. PLP combines modules in LP and MFCC: it can be viewed as MFCC with LPC-like spectral smoothing, or as LPC analysis with MFCC-like auditory filters.

Dynamic Features

- MFCC or PLP represent smoothed estimates of local spectrum. They are called static features.
- However, it can be argued that a key characteristic of speech is its dynamic behavior. So it is desired to have local time derivative estimates in addition to the static features.
- The delta (a.k.a. velocity, from physics) features are the difference between current static feature and neighboring features.

Robustness

- Robustness for convolutional noise
  - The convolution of speech and noise becomes multiplication in the spectral domain. They are additive in the log spectral domain.
  - The cepstral mean subtraction (CMS) is quite effective in this case.
- Robustness for additive noise
  - The spectra of speech and noise are additive.
  - In this case, one can estimate the noise spectrum form non-speech segments and subtract it from speech segments. This is called spectral subtraction.

Temporal Processing, RASTA

- CMS can be viewed as a special case of a more general idea of filtering the time trajectory of speech features. In other words, we go through the feature sequences and change the feature values.
- Another example of temporal processing is the RASTA filtering, which is given in Figure 22.5.
- It suffices to say that temporal filtering is quite effective for noise robustness.

Linguistic Units

- The set of feature vectors extracted from speech signal is a representation of the linguistic categories (or sequence of categories) in the speech.
- These linguistic categories serve as the modeling units for ASR.
- During **training**, these feature vectors are associated with the known category. During **testing**, they are integrated over time to find the best sequence of linguistic categories.

Words

- Words appear to be a natural unit for ASR, both acoustically and linguistically.
- Using words allows the context of phones to be covered in the model.
- In small-vocabulary tasks, this is a good design as there are many instances for each word.
- However, word modeling ignores the commonality of phones in different words. In a large-vocabulary system, this will cause data sparsity.

Phones

- The issue of data sparsity leads to the modeling of sub-word units. The phones are a natural choice.
- Modeling phones is more flexible than modeling words. New word models can be constructed using basic phone models.
- Phone context can be modeled by context-dependent phone models.

Phones vs Phonemes

- The notions of phone and phoneme are often confused.
- Basically, phoneme is an abstraction and phone is an instantiation (of phoneme).
- Allophones: the different phones for a phoneme. For example, aspirated $[p^h]$ and unaspirated $[p]$ correspond to the same phoneme $/p/$.

Phonetic Alphabets

- A phonetic alphabet represents one sound with one symbol.
- A well-known example is the international phonetic alphabet (IPA).
- IPA has a base of about 75 consonants and 25 vowels, covering most languages.
- There is a large inventory (50 or so) of diacritics to modify base phones to achieve finer distinctions.
- For machine readability, an ASCII symbol set (alphabet) is used for sounds, such as the TIMIT alphabet.

Articulatory Features

- We know that some phones are close to one another while others are quite different.
- From a phonetic alphabet, it is not possible to tell which is close to which.
- The articulatory features describe how sounds are pronounced. They can be used to capture the similarity of sounds.

Consonants

- Consonants are made by constricting the tube of vocal tract in various ways, usually with the tongue.
- Two main features for consonants are the place and manner of articulation:
  - The place of articulation is the point of closest constriction in oral cavity.
  - The manner of articulation refers to the amount of constriction in a consonantal gesture.

Place of Articulation

The places of articulation found in English are as follows.

- Bilabial (or labial): [p] [b]
- Labiodental: [f] [v]
- Inter-dental (or dental): [θ]
- Alveolar: [t] [d] [s] [z]
- Palatal/Palatal-alveolar: [$\int$]
- Velar: [k] [g]
- Glottal: lo<u>t</u>us, ki<u>tt</u>en

Manner of Articulation

The manners of articulation seen in English are as follows.

- Stop (or plosive): airflow is completely stopped for a short period of time, e.g., [t] [d] [p] [b] [k] [g]
- Nasal: [m] [n]
- Fricative: airflow is constricted but not cut off, for example, [s] [z]
- Affricate: stop + fricative, [t$\int$]
- Liquids and glides: [l], [r], [y]

Vowels

Three parameters tend to be used for vowel articulation.

- Frontness (or backness): refers to the place of largest constriction. For example, [i] is a front vowel, “schwa” is central, and [u] is a back vowel.
- Height: refers to the distance between the jaws. For example, [i] is a high vowel, where the upper and lower jaws are close. [ϵ] is a low vowel.
- Roundness: indicates whether the lips have been rounded. For example, [u] is rounded and [i] is unrounded.

Why Use Features

- Phones of similar articulatory features are similar acoustically.
- We can group phones by the articulatory features. This allows generalization from single phones to phone classes.
- Phones regularly vary in different acoustic contexts. This can be captured by the so-called phonological (or pronunciation) rules.
- Pronunciation rules can be generalized with the articulatory features.

Subword Units

- Subword units refer to the basic units from which the pronunciation of words can be constructed.
- The TIMIT has 61 phones. The CMU dictionary only uses 40 phones. The trade-off is using fewer phones makes the discrimination easier, while using more phones allow finer distinction which may be required for contextual dependency.
- Some systems even use data-driven subword units. The IBM uses clustering to derive “fenones”.

Context-Dependent Phones

- The phonemic categorization leads to units that use less context information.
- The context of a phone can be explicitly modeled.
- Figure 23.2 shows an example of monophone and triphone models.
- Note that the number of models could be an issue for data sparsity. In practice, models can be clustered or model parameters can be tied to alleviate this problem.

Syllables

- In some language, including Mandarin, the syllable is a natural choice.
- Syllable is divided to [onset], nucleus and [coda].
- Simple syllabic types and timing patterns are shown to represent most fluent conversational speech.
- Word boundaries and syllabic boundaries do not have to coincide in fluent speech. Onsets are preferred over codas in English.

Issues in Phonological Modeling

- The pronunciation variability increases dramatically from read-speech (RM, WSJ) to spontaneous speech (switchboard, callhome) recognition tasks.
- The big challenge is to predict such variability.
- Specifically, variability is wildest for
  - speaking rates (correlate to WER)
  - function words (60 per word)
- Pronunciation modeling is important in spontaneous speech. It is a hot and difficult area.

Sequence Recognition

- We have now established feature representation for local short-term spectrum. This representation is associated with linguistic categories such as phones or other sub-word units.
- In ASR, we have a sequence of features associated with an unknown class.
- Therefore, we need a framework to handle recognition based on feature sequences.

General Setting

- Suppose we have K reference sequences, $X_k^{\text{ref}} = (x_{k,1}^{\text{ref}}, \ldots, x_{k,N_k}^{\text{ref}}), k = 1, \ldots, K$. Each reference sequence X^{ref} is called a template.
- We have a sequence of input feature vectors $X = (x_1, \ldots, x_N)$ and we wish to associate X with a second sequence $Q = (q_1, \ldots, q_N)$, where each q corresponds to a linguistic (or quasi-linguistic) unit. $q_i = j$ means x_i is aligned to x_j^{ref}.
- Q is often chosen such that some error function is minimized.

Linear Time Warp

- The simplest case is that the template has the same length as the input sequence. Then we can use

$$D(X^{\text{ref}}, X) = \sum_{i=1}^{N} d(x_i^{\text{ref}}, x_i),$$

  where N is the length of sequence and the global distance is the sum of local distances.
- To choose the best template, simply choose the one with the least global distance.
- If the lengths are different, we can downsample or interpolate the template sequence.

Dynamic Time Warping

- Linear time warping does not properly compensate for the speaking rate: often the vowels are elongated while the consonants are roughly constant.
- It is thus desired to deal with this variety by dynamic time warping, where we allow for different warping factor at different segments of speech.
- An alignment is a correspondence between the vectors in two sequences.
- We want to find the alignment between input and template sequences such that the total distortion is minimized.

Distortion for a Reference Template

- Define the distortion between the input and reference template to be the minimum over all possible alignments.
- Let the smallest cumulative distortion between $x_{1:i}$ and $x^{\text{ref}}_{1:j}$ is stored in D_{ij}. then

$$D(i,j) = d(i,j) + \min_{p(i,j)} [D(p(i,j)) + T((i,j), p(i,j))],$$

  where T is a transition cost.
- The minimum in the last column of D is the distortion

$$D = \min_j D(N,j).$$

Further Comments for DTW

- The optimal path yields the template distortion. The optimum distortion yields the optimal template.
- One may apply global and local constraints to reduce the search space.
- One may use clustering to reduce the number of templates, e.g. for speaker-independent systems.
- One may use probabilistic distance (or other distance measures) rather than the Euclidean distance.
- End-pointing is often mandatory for template-based system.

Connected Word Recognition

- Effects of allowing connected words
  - pronunciation may be altered due to context
  - number of hypotheses increases drastically
  - need to deal with both segmentation and recognition
- One can still use DTW for this problem!
  - The basic idea is to use a large matrix consisting of all the word templates.
  - Backtrack information can be stored by two lists per frame: $T(i)$, the lowest-cost template ending at frame i, and $F(i)$, the end frame of the previous template.

Segmental Approaches

- We mainly illustrate dynamic-time warping using words as templates. But the same idea can be applied to subword units as well.
- Subword units have been used in statistical systems, with essentially the same dynamic-programming search algorithm for best hypothesis.

Statistical Sequence Recognition

- With DTW, local distortions (distance) between acoustic frames are integrated temporally to compute the global distance between two templates.
- For fast computation, DP can be used for DTW.
- The notion of distance can be generalized to a statistical framework: a large distance signals a small probability.

Statistical Methodology

- We assume that the speech features are generated according to the probability models of the underlying linguistic units.
- During the *training* phase, the model parameters are learned from labelled data (features).
- During the *testing* phase, the learned models are used to find the hypothesis with the maximum a posteriori (MAP) probability given speech features.

Bayes Rule and MAP

- The fundamental equation for pattern recognition is the MAP criterion and the Bayes rule:

$$
\begin{aligned}
M^* &= \arg\max_M P(M|X) \\
&= \arg\max_M P(X|M)P(M)
\end{aligned}
$$

- The problem is solved in principle if
  - we have accurate models for $P(X|M)$ ad $P(M)$.
  - we have a way to search M^*.

Markov Models

- A Markov model consists of a set of states, an initial distribution, and a transition probability matrix.
- Two model assumptions
  - The probability of state q_t given q_{t-1} is independent of any state prior to $t-1$.

$$p(q_t|q_{t-1}, q_{t-2}, \ldots, q_1) = p(q_t|q_{t-1}).$$

  - The transition probability does not vary with t,

$$p(q_t = j|q_{t-1} = i) = a_{ij} \quad \forall\, t.$$

An Example of Markov Model

- A starter W of the New York Yankees
- State space: $\mathcal{X} = \{1 = \mathrm{A}, 2 = \mathrm{B}, 3 = \mathrm{T}\}$.
- Transition probability

$$A = \begin{bmatrix} 0.6 & 0.1 & 0.3 \\ 0.4 & 0.3 & 0.3 \\ 0.5 & 0.3 & 0.2 \end{bmatrix}$$

- Initial probability is uniform.
- $p(W \text{ leaves with leads in the first } 10 \text{ games}) = \ ?$

Hidden Markov Models

- We may want to say something about the pitcher's performance (state) given the team's record for T consecutive games (observation).
- In an HMM, the state identities are *hidden* and the *observed* sequence depends probabilistically on the state sequence.
- In addition to the components required in a Markov model, in HMM there are the observation likelihoods, denoted by $b_i(o_t)$, representing the probability of observing o_t when the state $q_t = i$.

Coin-toss Models

- Suppose there are a number of coins, each with its own bias.
- One of the coins, coin q_t, is randomly selected. The selection probability is dependent on the identity of the previous coin, q_{t-1}.
- Coin q_t is tossed and the outcome (head or tail) o_t is recorded, *but not the coin*.
- The probability is

$$p(q_1^T, o_1^T) = p(q_1)p(o_1|q_1)\prod_{t=2}^{T} p(q_t|q_{t-1})p(o_t|q_t).$$

Urn-and-ball Models

- Suppose there are N urns, each consisting of a distinct composition of colored balls.
- One of the urns, urn q_t, is randomly selected. The selection probability is dependent on the identity of the previous urn, q_{t-1}.
- A ball is picked from the selected urn q_t and the color of the ball is recorded as o_t.
- As we only observe the colors of balls, do we really know how many urns there are?

Basic Problems in HMM

- Probability evaluation: Given the observations O and the model parameters λ, compute the data likelihood $p(O|\lambda)$.
- Optimal state sequence: Given the observations O and λ, determine the optimal state sequence Q^*

$$Q^* = \arg\max_{Q} p(O, Q|\lambda).$$

- Parameter estimation: Given the observations O, choose the model parameters λ to maximize the data-likelihood

$$\lambda^* = \arg\max_{\lambda} p(O|\lambda).$$

Forward-Backward Algorithm

- We use special states, state 1 and state N, for non-emitting entering and exiting state.
- Denote the parameters in HMM by
  - the initial probability $\pi_i = a_{1i}$
  - the transition probability a_{ij}
  - the observation likelihood $b_i(o_t)$
- Given these parameters, the data likelihood can be computed via the forward-backward algorithm.
- With data likelihood, many conditional probabilities can be computed.

Forward Probability

Define the forward probability α as

$$\alpha_i(t) = p(o_1, \ldots, o_t, q_t = i).$$

Then

$$\alpha_j(1) = a_{1j} b_j(o_1),$$

$$\alpha_j(t) = \sum_{i=2}^{N-1} \alpha_i(t-1) a_{ij} b_j(o_t),$$

$$\alpha_N(T) = \sum_{i=2}^{N-1} \alpha_i(T) a_{iN}.$$

Backward Probability

Similarly, define the backward probability β as

$$\beta_i(t) = p(o_{t+1}, \ldots, o_T | q_t = i).$$

Then

$$\beta_i(T) = a_{iN},$$

$$\beta_i(t) = \sum_{j=2}^{N-1} a_{ij} b_j(o_{t+1}) \beta_j(t+1),$$

$$\beta_1(1) = \sum_{j=2}^{N-1} a_{1j} b_j(o_1) \beta_j(1).$$

Data Likelihood

- The joint probability of $q_t = j$ and O is

$$p(O, q_t = j) = \alpha_j(t)\beta_j(t).$$

- The data likelihood is

$$p(O) = \sum_j p(O, q_t = j) = \sum_j \alpha_j(t)\beta_j(t).$$

- Alternatively,

$$p(O) = \alpha_N(T) = \beta_1(1).$$

Viterbi Approximation

- Best-path approximation to $p(O)$ is

$$p(O) \triangleq \sum_Q p(Q,O) \sim \max_Q p(Q,O) \triangleq \bar{p}(O).$$

- Define $\delta_j(t) \triangleq \max_{q_{1:t-1}} p(q_{1:t-1}, q_t = j, o_{1:t})$. Then

$$\begin{aligned}\delta_j(t) &= \max_i \max_{q_{1:t-2}} p(q_{1:t-2}, q_{t-1} = i, o_{1:t-1})\; a_{ij}\; p(o_t | q_t = j) \\ &= \max_i \delta_i(t-1)\; a_{ij}\; b_j(o_t).\end{aligned}$$

- Taking logarithm, this is similar to DTW.

Model Training

- We have seen how one can compute data-likelihood and posterior probability with HMM through the forward-backward algorithm.
- There is one problem left: In order to compute the likelihood, the parameters in the model must be known. How do we know their values?
- This is not like throwing dice or flipping coin that we can reasonably assign probabilities. In this case, the parameters must be learned from data.
- We have seen that maximum-likelihood criterion can be used in model training and the EM algorithm is one way to do it. Here we apply EM to the HMMs.

The $\mathcal{Q}$ Function for HMM

- When applying to HMM, the hidden variables are the sequence of states. Let Q denote a state sequence. Define the $\mathcal{Q}$ function as

$$\mathcal{Q} \triangleq \sum_{Q} p(Q|O, \Theta_o) \log p(Q, O|\Theta).$$

- We will show how to simply the $\mathcal{Q}$ function and relate it to quantities computable from the forward-backward algorithm.

Simplifying $\mathcal{Q}$ Function

- From the independence assumption of HMM,

$$p(Q,O) = p(Q)p(O|Q)$$
$$= p(q_1)\prod_{t=2}^{T} p(q_t|q_{t-1}) \prod_{t=1}^{T} p(o_t|q_t).$$

- Taking the logarithm, we have

$$\log p(Q,O)$$
$$= \log p(q_1) + \sum_{t=2}^{T} \log p(q_t|q_{t-1}) + \sum_{t=1}^{T} \log p(o_t|q_t).$$

Simplifying $\mathcal{Q}$ Function

Putting it together,

$$\begin{aligned}
\mathcal{Q} &\triangleq \sum_{Q} p(Q|O,\Theta_o)\log p(Q,O|\Theta) \\
&= \sum_{Q} p(Q|O,\Theta_o)\log p(q_1|\Theta) + \sum_{Q} p(Q|O,\Theta_o)\sum_{t=1}^{T}\log p(o_t|q_t,\Theta) \\
&\quad + \sum_{Q} p(Q|O,\Theta_o)\sum_{t=2}^{T}\log p(q_t|q_{t-1},\Theta) \\
&= \sum_{i=2}^{N-1} p(q_1 = i|O)\log \pi_i + \sum_{t=1}^{T}\sum_{i=2}^{N-1} p(q_t = i|O)\log b_i(o_t) \\
&\quad + \sum_{t=2}^{T}\sum_{i=2}^{N-1}\sum_{j=2}^{N-1} p(q_{t-1} = i, q_t = j|O)\log a_{ij}
\end{aligned}$$

Posterior Probabilities

Certain posterior probabilities can be computed through forward-backward algorithm. Specifically

$$\gamma_i(t) = p(q_t = i|O) = \frac{\alpha_i(t)\beta_i(t)}{\sum_j \alpha_j(t)\beta_j(t)}$$

$$\xi_{ij}(t) = p(q_t = i, q_{t+1} = j|O) = \frac{p(q_t = i, q_{t+1} = j, O)}{p(O)}$$

where the joint probability of $p(q_t = i, q_{t+1} = j, O)$ is given by

$$p(q_t = i, q_{t+1} = j, O) = \alpha_i(t)a_{ij}b_j(o_{t+1})\beta_j(t+1).$$

Occupancy Numbers

- The expected number of transitions from state i to state j at time t is $\xi_{ij}(t)$. The expected number of transitions from state i to state j is

$$\sum_{t=1}^{T-1} \xi_{ij}(t).$$

- The occupancy number for state i is the expected number of times that $q_t = i$, and is given by

$$\sum_{t=1}^{T-1} \gamma_i(t).$$

Parameter Update Equations

The parameters are uncoupled in the $\mathcal{Q}$ function so the maximization can be carried out independently. The new set of parameters are

$$\begin{cases} \pi_i^* = \gamma_i(1) \\ a_{ij}^* = \frac{\sum_t \xi_{ij}(t)}{\sum_t \gamma_i(t)} \\ \mu_i^* = \frac{\sum_t \gamma_i(t) o_t}{\sum_t \gamma_i(t)} \\ {\sigma^2}_i^* = \frac{\sum_t \gamma_i(t)(o_t - \mu_i)(o_t - \mu_i)'}{\sum_t \gamma_i(t)} \end{cases}$$

One epoch of training finishes here and another starts. It continues until some stopping criterion is met.

Viterbi Training

- The likelihood of a model can be approximated by the Viterbi (best-path) approximation.

$$p(O|M) = \sum_Q p(Q, O|M) \sim \max_Q p(Q, O|M).$$

- In Viterbi training, we update the parameters to maximize the likelihood of the best path in the correct model.
- Each frame is associated with a state, or equivalently the posterior probability of state is 0 or 1. Dynamic programming is used for this (Viterbi) alignment of time and states.

Examples

- Viterbi training makes update equations significantly easier. For Gaussian emission density,

$$\mu_j = \frac{\sum_{\text{frames s.t. } q_t = j} x_t}{\text{no. frames s.t. } q_t = j}$$

$$\sigma_j^2 = \frac{\sum_{\text{frames s.t. } q_t = j} (x_n - \mu_j)^2}{\text{no. frames s.t. } q_t = j}$$

- For discrete emission probability (such as in VQ),

$$p(x_t = y_k | q_t = j) = \frac{\text{no. frames s.t. } q_t = j \text{ and } x_t = y_k}{\text{no. frames s.t. } q_t = j}$$

Tied Mixture of Gaussians

- An emission density for state j can be assumed to be a weighted sum of a pool of Gaussian densities,

$$p(x_t|q_t = j) = \sum_{k=1}^{K} c_{jk} N(x_t; \mu_k, \sigma_k).$$

- This is often referred to as soft VQ. The HMMs are often referred to as semi-continuous HMMs.

Independent Mixture of Gaussians

- The Gaussians for different states are not required to be tied in general. They can be independent.
- This provides a more detailed estimate of densities, at the cost of requiring more data.
- Parameter-tying is a method to balance the data amount and the number of parameters.
- For data sparsity, smoothing methods such as backoff and interpolation are often used for reliable parameter estimation.

Issues with MLE

- The posterior probability for model M is

$$P(M|X) = \frac{P(X|M)P(M)}{\sum_{M'} P(X|M')P(M')}$$
$$= \frac{1}{1 + \sum_{M' \neq M} \frac{P(X|M')P(M')}{P(X|M)P(M)}}$$

- Changing parameters to increase $P(X|M)$ does not necessarily increase $P(M|X)$.

Discriminative Training

- With the maximum-likelihood criterion, we train the model parameters so that the data likelihood for the correct model is non-decreasing.
- However, the increase of likelihood of a wrong model can be more than that of the correct model.
- A training method aiming to increase the difference of the likelihoods between the correct and incorrect models is called discriminative training.

Maximum Mutual Information

- The mutual information between M and X is

$$I(M,X|\lambda) = \log \frac{P(M,X|\lambda)}{P(M|\lambda)P(X|\lambda)}$$
$$= \log \frac{P(X|M,\lambda)}{\sum_{M'} P(X|M',\lambda)P(M'|\lambda)}$$

- This is quite similar to the posterior probability (except for $\log$ and $P(M|\lambda)$).
- The denominator can consist an infinite number of terms. There are feasible approximations to the denominator such as lattice or N-best.

Discriminant Functions

- A framework for classification using discriminant functions is as follows. We define a discriminant function for each class,

$$g_j(X; \lambda), \quad j = 1, \dots, K.$$

- The classification rule is simply

$$j^* = \arg\max_j \; g_j(X; \lambda).$$

- A sample X of class j will be classified correctly if

$$g_j(X; \lambda) > g_i(X; \lambda) \;\; \forall i \neq j.$$

Misclassification Measure

- We can define a misclassification measure based on the values of discriminant functions. Specifically, for data X of class j, we can define

$$d_j(X;\lambda) = \log\left\{\frac{1}{K-1}\sum_{k\neq j} e^{\eta g_k(X;\lambda)}\right\}^{\frac{1}{\eta}} - g_j(X;\lambda).$$

- If X is classified correctly, then $d_j(X) < 0$. Put differently, if $d_j(X) > 0$, then a classification error occurs.

Minimum Classification Error

- The number of errors in the training data is lower bounded by the number of data where $d_j(X) > 0$.
- For the entire data set, this amounts to

$$E(\lambda) = \sum_j \sum_{X \in M_j} F(d_j(X;\lambda)),$$

where $F(x)$ is approximately a step function. $E(\lambda)$ is minimized for the optimal model parameters λ^*.

Recognition and Understanding

- We have described ASR as a pattern recognition problem requiring signal processing, probability estimation and temporal integration.
- To have the probability of a hypothesized sequence of words in a speech, we need the language models.
- We will show how these aspects are integrated in the decoding process for recognition.
- In addition, we discuss a speech-understanding system based on speech recognition and further language processing, as illustrated in Figure 28.1.

Word Pronunciations

- An ASR system needs to know the pronunciation(s) of each word. Specifically, these pronunciations are expressed as the modeling units such as phones.
- A simple way to this is to use a lexicon consisting of the pronunciation for every word in the vocabulary.
- For a refined system, one may want to learn the pronunciation from data. The results are often stored in the form of pronunciation models (Figure 28.2) or phonological rules (Table 28.1).

Language Models

- MAP decoding requires the model probability in addition to data likelihood. This is given by a language model, which assigns a probability for every sentence.
- How do we assign such probability?
  - There are infinitely many sentences.
  - The probabilities of these sentences sum to 1.

Entropy

- The entropy rate of a stationary stochastic process is defined by

$$H = \lim_{n\to\infty} \frac{-1}{n} \log p(w_1, \ldots, w_n).$$

- If the probability is estimated to be q, then

$$\begin{aligned}\lim_{n\to\infty} \frac{-1}{n} \log q(w_1, \ldots, w_n) &= \lim_{n\to\infty} \frac{-1}{n} \log p(w_1, \ldots, w_n) + \lim_{n\to\infty} \frac{1}{n} \log \frac{p(w_1, \ldots, w_n)}{q(w_1, \ldots, w_n)} \\ &\geq \lim_{n\to\infty} \frac{-1}{n} \log p(w_1, \ldots, w_n) = H.\end{aligned}$$

- The left-hand side is called the cross entropy. It is an upper bound for H.

Perplexity

- The perplexity is the base-2 exponential of H. It is the average number of candidates for next word,

$$p(w_1, \ldots, w_n) \sim 2^{-nH} = (\frac{1}{2^H})^n.$$

- The perplexity of a language model q is the base-2 exponential of the cross-entropy,

$$\mathrm{PPL} = q(w_1, \ldots, w_n)^{\frac{-1}{n}}.$$

It is an upper bound for the true perplexity.

n-Gram Models

- n-gram LM is an $(n-1)$th order Markov model. I.e.

$$p(w_i|w_{1:i-1}) = p(w_i|w_{i-n+1:i-1}).$$

- In general, the probability of a sentence is

$$p(w_{1:N}) = p(w_1| < s >) \prod_{i=2}^{N} p(w_i|w_{1:i-1})\, p(< /s > |w_{1:N}).$$

- With n-gram, this becomes

$$p(w_{1:N}) = p(w_1| < s >) \prod_{i=2}^{N} p(w_i|w_{i-n+1:i-1})\, p(< /s > |w_{N-n+2:N}).$$

Issues of n-Gram Language Models

- The long-range word dependency is not directly modeled in n-gram unless n is large.
- Syntactic and semantic rules are not explicitly implemented.
- The data sparsity problem: in n-gram, there are V^n parameters to be learned from data. This is a huge number when $n \geq 4$. Normally we use $n = 3$. Even in this case there are many unseen trigrams and we need smoothing schemes.

Smoothing

- The maximum likelihood estimate of n-gram is

$$p(w_i|w_{i-n+1:i-1}) = \frac{c(w_{i-n+1:i})}{c(w_{i-n+1:i-1})}$$

- Some n-grams may not appear in the corpus used for counting, resulting in 0 probability.
- There are so many n-grams (say $n = 3$) that a corpus can not cover all of them. Therefore we need to deal with the 0-occurrence problem.

Add-One Smoothing

- The count of each n-gram is increased by 1. The total count is increased by V, so

$$p_i^* = \frac{c_i + 1}{N + V}.$$

- Equivalent to modify the counts of the ith n-gram by

$$c_i^* = (c_i + 1)\frac{N}{N + V} \quad (\text{and } p_i^* = \frac{c_i^*}{N})$$

- Every occurrence of the ith n-gram is discounted to

$$d_i = \frac{c_i^*}{c_i}.$$

Backoff

- If we have no occurrence for an n-gram, we use the occurrence count of $(n-1)$-gram. For trigram,

$$\hat{p}(w_i|w_{i-1}w_{i-2}) = \begin{cases} \tilde{p}(w_i|w_{i-1}w_{i-2}), & c(w_{i-2}w_{i-1}w_i) > 0 \\ \alpha(w_{i-1}w_{i-2})\hat{p}(w_i|w_{i-1}), & \text{otherwise} \end{cases}$$

- The α's are used to make the total probability 1,

$$\alpha(w_{i-1}w_{i-2}) = \frac{1 - \sum_{w\in A} \tilde{p}(w|w_{i-1}w_{i-2})}{1 - \sum_{w\in A} \hat{p}(w|w_{i-1})},$$

where $A = \{w|c(w_{i-2}w_{i-1}w) > 0\}$.

Decoding

- With HMM and bigram language model, a time synchronous Viterbi search algorithm can be used.
- For higher order language models or more refined acoustical models, one can
  - use depth-first decoding. A priority queue holds the best hypotheses, which are popped, extended, rescored and inserted. Long-range models can be used as the entire history is evaluated.
  - use multiple-pass decoding. The first pass generates good candidates with fast and simple model. Later passes re-score these hypotheses with more refined models.

Evaluation of Recognition

- Word error rates is the standard metric for evaluation. It is defined by

$$\text{WER} = \frac{I + S + D}{N} * 100\%,$$

  - I is the number of insertions,
  - D is the number of deletions,
  - S is the number of substitutions,
  - N is the number of words in the reference.
- These numbers are based on the optimal alignment between the output and the reference transcription.

A Complete System: BERP

- Berkeley Restaurant Project
  - medium vocabulary (1200 words)
  - speaker-independent
  - spoken dialog system
- Scenario
  - A user calls the system to inquire about restaurants around Berkeley.
  - The system answers the query by having a dialog with the user.

Understanding Speech in BERP

- The system is a knowledge consultant whose domain is restaurants in Berkeley.
- The system knows something about the world with a knowledge base. The query is mapped against the knowledge base.
- User's speech is recognized and processed with stochastic context-free grammar (SCFG) so the fields in a database query can be filled in.
- The SCFG consists of 1389 rules trained by 4786 sentences.
- Initiating a sentence with "PLEASE" is 0.1 for German but 0.00057 for American English speakers.

Practical Concerns

- We want high confidence on recognition results. Low-confidence result is likely to be wrong.
  - can use a garbage model to measure confidence.
- Rejection policy: recognition result is rejected if the confidence is lower than a threshold.
- For spontaneous speech, a speech understanding system needs to deal with disfluency, non-speech sounds, and speech fragments.