41. Automatic Language Recognition Via Spectral and Token Based Approaches

D. A. Reynolds, W. M. Campbell, W. Shen, E. Singer

Automatic language recognition from speech consists of algorithms and techniques that model and classify the language being spoken. Current state-of-the-art language recognition systems fall into two broad categories: spectral- and token-sequence-based approaches. In this chapter, we describe algorithms for extracting features and models representing these types of language cues and systems for making recognition decisions using one or more of these language cues. A performance assessment of these systems is also provided, in terms of both accuracy and computation considerations, using the National Institute of Science and Technology (NIST) language recognition evaluation benchmarks.

41.1 **Automatic Language Recognition** 811
41.2 **Spectral Based Methods** 812
41.2.1 Shifted Delta Cepstral Features 812
41.2.2 Classifiers 813
41.3 **Token-Based Methods** 815
41.3.1 Tokens 815
41.3.2 Classifiers 816
41.4 **System Fusion** 818
41.4.1 Methods 819
41.4.2 Output Scores 820
41.5 **Performance Assessment** 820
41.5.1 Task and Corpus 820
41.5.2 Systems 821
41.5.3 Results 822
41.5.4 Computational Considerations 822
41.6 **Summary** 823
References 823

41.1 Automatic Language Recognition

Automatic language recognition from speech consists of algorithms and techniques that model and classify the language being spoken. The term recognition can refer to either identification of the spoken language from a set of languages known to the system, or detection of the presence of a particular language out of a larger set of languages unknown to the system. As with other information conveyed by the speech signal, cues about the language being spoken occur at several levels. At the top level the words being spoken to communicate a coherent message are the basis of what defines a language. However, relying on automatic extraction of the words presupposes the availability of a reliable speech recognizer in the languages of interest, which is often not the case in many practical applications requiring language recognition. Further, language recognition is often used as a low computation pre-processor to determine which higher computation speech recognizer should be run, if any. Fortunately, cues about the language are also transmitted at lower levels in the speech signal, such as in the phoneme inventory and co-occurrences (phonotactics), the rhythm and timing (prosodics) and the acoustic sounds (spectral characteristics), that are more amenable to automatic extraction and modeling.

Current state-of-the-art language recognition systems are based on processing these low-level cues and fall into two broad categories: spectral based and token sequence based. This chapter describes algorithms for extracting features and models representing these types of language cues and systems for making recognition decisions using one or more of these language cues. Additionally, a performance assessment of these systems is provided, in terms of both accuracy and computation considerations, using the National Institute of Science and Technology (NIST) language recognition evaluation benchmarks.

41.2 Spectral Based Methods

Spectral-based methods for language recognition operate by extracting measurements of the short-term speech spectrum over fixed analysis frames and then modeling characteristics of these measurements, or features, for each language to be recognized. Classification techniques that have proved successful for language recognition are generative, via Gaussian mixture models (GMMs), and discriminative, via support vector machines (SVMs). This section first describes the spectral feature extraction process and then the GMM and SVM classifiers used for recognition.

41.2.1 Shifted Delta Cepstral Features

The features used for language recognition systems are based on the cepstral coefficients derived from a mel-scale filterbank analysis typically used in other speech processing tasks such as speech and speaker recognition. A block diagram of the filterbank feature extraction system is shown in Fig. 41.1. The feature extraction consists of the following steps. Every 10 ms the speech signal is multiplied by a Hamming window with a duration of 20 ms to produce a short-time speech segment for analysis. The discrete Fourier spectrum is obtained via a fast Fourier transform (FFT) from which the magnitude squared spectrum is computed. The magnitude spectrum is multiplied by a pre-emphasis filter to emphasize the high-frequency portion of the spectrum and the result is put through a bank of triangular filters. The filterbank used is similar to that in [41.1] and simulates critical band filtering with a set of triangular bandpass filters that operate directly on the magnitude spectrum. The critical band warping is done by approximating the mel-frequency scale, which is linear up to 1000 Hz and logarithmic above 1000 Hz. The center frequencies of the triangular filters follow a uniform 100 Hz mel-scale spacing, and the bandwidths are set so the lower and upper passband frequencies of a filter lie on the center frequencies of the adjacent filters, giving equal bandwidths on the mel-scale but increasing bandwidths on the linear frequency scale. The number of filters is selected to cover the signal bandwidth $[0, f_s/2]$ Hz, where f_s is the sampling frequency. For 8 kHz sampled telephone speech, there are 24 filters.

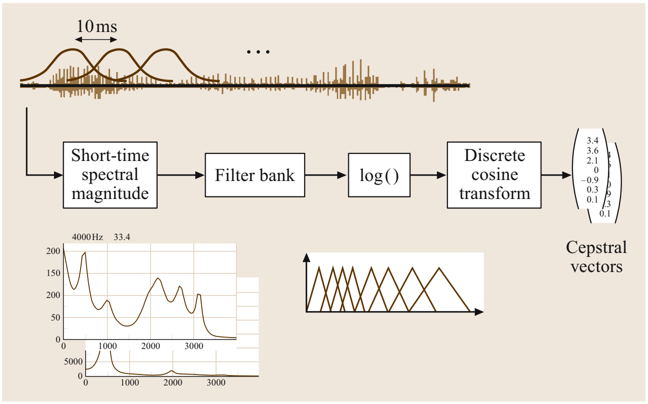

Fig. 41.1 Mel-scale filterbank cepstral feature extraction

The log energy output of each filter is then used as an element of a filterbank feature vector and the short-term mel-scale cepstral feature vector is obtained from the inverse cosine transform of the filterbank vector. To model short-term speech dynamics, the static cepstral vector is augmented by difference (delta) and acceleration (delta–delta) cepstra computed across several frames.

Since the speech used for training and testing language recognition systems can come from a variety of sources (involving different microphones and channels), it is important to apply some form of channel compensation to the features. Typical channel compensation techniques include blind deconvolution via RASTA (relative spectral) filtering [41.2] and per-utterance feature normalization to zero mean and unit variance. More-sophisticated compensation techniques, such as feature mapping [41.3], that explicitly model channel effects are also successfully used.

One of the significant advances in performing language recognition using GMMs was the discovery of a better feature set for language recognition [41.4]. The improved feature set, the shifted delta cepstral (SDC) coefficients, are an extension of delta-cepstral coefficients.

SDC coefficients are calculated as shown in Fig. 41.2. SDC coefficients are specified by four parameters, conventionally written as N-d-P-k. For a frame of data at time t, a set of MFCCs are calculated, i. e.,

$$c_0(t), c_1(t), \dots, c_{N-1}(t) . \qquad (41.1)$$

Note that the coefficient $c_0(t)$ is used. The parameter d determines the spread across which deltas are calculated, and the parameter P determines the gaps between successive delta computations. For a given time t,

$$\Delta c(t, i) = c(t + iP + d) - c(t + iP - d) \qquad (41.2)$$

represents an intermediate calculation, where $i = 0, 1, \dots, k$. The SDC coefficients are then k stacked

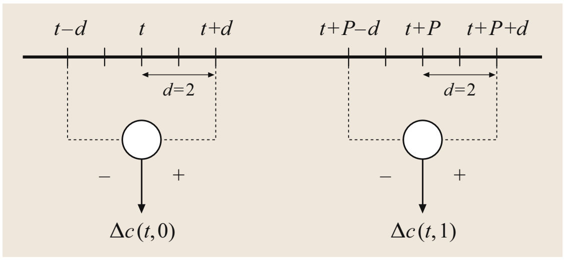

Fig. 41.2 Shifted delta cepstral coefficients

versions of (41.2),

$$\mathrm{SDC}(t)=\begin{pmatrix}\Delta c(t,0)\\ \Delta c(t,1)\\ \vdots\\ \Delta c(t,k-1)\end{pmatrix}. \tag{41.3}$$

Prior to the use of SDC coefficients, GMM-based language recognition was less accurate than alternate approaches [41.5]. SDC coefficients capture variation over many frames of data; e.g., the systems described in this chapter use 21 consecutive frames of cepstral coefficients. This long-term analysis might explain the effectiveness of the SDC features in capturing language-specific information.

41.2.2 Classifiers

Current state-of-the-art language recognition systems use either or both generative GMM-based classifiers and discriminative SVM-based classifiers.

Gaussian Mixture Models

The approach for a GMM classifier is to model the distribution of cepstral features for each language to be recognized. A language-specific GMM is given by

$$p(\boldsymbol{x}|\lambda)=\sum_{i=1}^{M}\pi_i\mathcal{N}_i(\boldsymbol{x}), \tag{41.4}$$

where $\boldsymbol{x}$ is a D-dimensional feature vector, $\mathcal{N}_i(\boldsymbol{x})$ are the component densities, and π_i are the mixture weights. Each component density is a D-variate Gaussian function of the form

$$\mathcal{N}_i(\boldsymbol{x})=\frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}_i|^{1/2}}\times\exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^{\mathrm{T}}\,\boldsymbol{\Sigma}_i^{-1}\,(\boldsymbol{x}-\boldsymbol{\mu}_i)\right\}, \tag{41.5}$$

with mean vector $\boldsymbol{\mu}_i$ and covariance matrix $\boldsymbol{\Sigma}_i$. The mixture weights satisfy the constraint

$$\sum_{i=1}^{M}\pi_i=1, \tag{41.6}$$

which ensures that the mixture is a true probability density function. The complete language-specific Gaussian mixture density is parameterized by the mean vectors, covariance matrices, and mixture weights from all component densities and is represented by the notation

$$\lambda=\{\pi_i,\,\boldsymbol{\mu}_i,\,\boldsymbol{\Sigma}_i\}\quad i=1,\dots,M. \tag{41.7}$$

For language recognition using SDC features, diagonal covariances and a mixture order of $M=2048$ are used. It was empirically determined [41.4] that higher-order GMMs provided substantial performance improvements when using SDC features.

The GMM parameters are estimated via the expectation-maximization (EM) algorithm [41.6] or by Bayesian adaptation from a universal background model (UBM) [41.7], as is done in speaker recognition (see Part F of this Handbook). Adapting from a background model has computational advantages in that it speeds up both model training and likelihood computations. For language recognition, the UBM is typically constructed by training a GMM using an aggregation of the training data from all available languages. Recently, discriminative training of GMM parameters based on maximum mutual information optimization has yielded very promising results [41.8].

For a general language recognition task, a GMM, λ_l, is trained for each of the desired languages, $l=1,\dots,L$, and the likelihood of a sequence of feature vectors, $\boldsymbol{X}=(\boldsymbol{x}_1,\dots,\boldsymbol{x}_T)$, extracted from a test utterance is computed as

$$p(\boldsymbol{X}|\lambda_l)=\prod_{t=1}^{T}p(\boldsymbol{x}_t|\lambda_l). \tag{41.8}$$

The model likelihood is computed assuming independence between the feature vectors which means the temporal ordering of the feature vectors is unimportant. However, the SDC feature vectors $\boldsymbol{x}_t$ were themselves extracted over a multiframe time span and thereby encode local temporal sequence information.

The likelihood scores from the set of language models are then used by the back-end fusion/decision system to compute likelihood ratios or to fuse with other system scores (Sect. 41.4).

Support Vector Machines

Support vector machines (SVMs) are flexible classifiers that have recently shown promise in language recognition. SVMs rely on separating data in high-dimensional spaces using the maximum margin concept [41.9]. An SVM, $f(\boldsymbol{Z})$, is constructed from sums of a kernel function $K(\cdot,\cdot)$,

$$f(\boldsymbol{Z}) = \sum_i \alpha_i K(\boldsymbol{Z}, \boldsymbol{Z}_i) + d\,, \tag{41.9}$$

where $\boldsymbol{Z}$ is the input feature vector, $\sum_i \alpha_i = 0$, and $\alpha_i \neq 0$. The $\boldsymbol{Z}_i$ are support vectors and are obtained from the training set by an optimization process [41.10]. The ideal classifier values are either 1 or -1, depending on whether the corresponding support vector is in class 0 or class 1. For classification, a class decision is based on whether the value $f(\boldsymbol{Z})$ is above or below a threshold.

SVM kernels provide a method for comparing sequences of feature vectors (e.g., shifted delta cepstral coefficients). Given sequences of feature vectors from two utterances, $\boldsymbol{X} = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_{T_x})$ and $\boldsymbol{Y} = (\boldsymbol{y}_1, \cdots, \boldsymbol{y}_{T_y})$, the kernel produces a comparison that provides discrimination between the languages of the utterances. If the utterances are from the same language, a large positive value is desired; otherwise, the kernel should produce a large negative value.

The general setup for training a language recognition system using support vector machines is a *one-versus-all* strategy as shown in Fig. 41.3. For the example of English in the figure, class 1 contains only target (English) language data and class 0 contains all of the nontarget (non-English) language data pooled together. Training proceeds using a standard SVM training tool with a sequence kernel module (e.g., [41.10]). The resulting process produces a model for recognizing the target language.

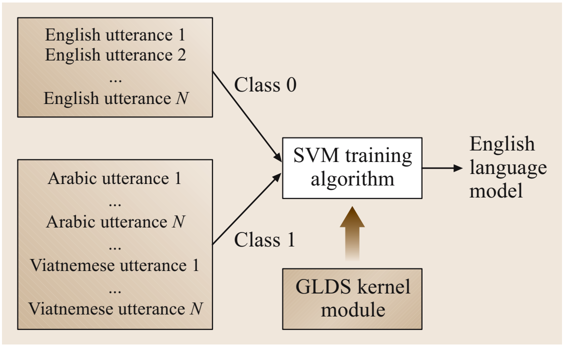

Fig. 41.3 Training a support vector machine for language recognition

A useful kernel for language recognition is the generalized linear discriminant sequence (GLDS) kernel [41.11]. The GLDS kernel is simply an inner product between average high-dimension expansions of a set of basis functions $b_j(\cdot)$,

$$\boldsymbol{b}(\boldsymbol{x}) = \left(b_1(\boldsymbol{x}) \;\cdots\; b_K(\boldsymbol{x})\right)^{\mathrm{T}}\,, \tag{41.10}$$

where K is the number of basis functions. Typically, monomials up to a given degree are used as basis functions. An example basis is

$$\boldsymbol{b}(\boldsymbol{x}) = \begin{pmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_n \\ x_1^2 \\ \vdots \\ x_1^{i_1} x_2^{i_2} \ldots x_n^{i_n} \\ \vdots \end{pmatrix}. \tag{41.11}$$

The formula for the kernel is

$$K_{\mathrm{GLDS}}(\boldsymbol{X}, \boldsymbol{Y}) = \bar{\boldsymbol{b}}_x^{\mathrm{T}} \bar{\boldsymbol{R}}^{-1} \bar{\boldsymbol{b}}_y\,. \tag{41.12}$$

The mapping $\boldsymbol{X} \to \bar{\boldsymbol{b}}_x$ is defined as

$$\boldsymbol{X} \to \frac{1}{T_x} \sum_{t=1}^{T_x} \boldsymbol{b}(\boldsymbol{x}_t)\,. \tag{41.13}$$

The vector $\bar{\boldsymbol{b}}_y$ is defined in an analogous manner to (41.13). $\bar{\boldsymbol{R}}$ is a correlation matrix derived from a large set of data from multiple languages and many speakers and is usually diagonal.

In a general language recognition task, a set of SVM language models $\{f_l\}$ are trained to represent the target languages $l = 1, \cdots, L$. The sequence of vectors $\boldsymbol{X}$ is extracted from a test utterance, mapped to an average expansion using (41.13), and converted to a set of scores $s_l = f_l(\bar{\boldsymbol{b}}_x)$. The SVM scores s_l are usually transformed to s'_l using a likelihood-ratio-type calculation

$$s'_l = s_l - \log\left(\frac{1}{M-1} \sum_{j \neq l} e^{-s_j}\right). \tag{41.14}$$

The transformation (41.14) assumes that SVM scores behave like log-likelihood ratios (see [41.12, 13] for related discussions).

Several computational simplifications are available for implementing an SVM system for language recognition. For instance, language recognition scoring can be reduced to a single inner product. Also, language model training can be simplified to a linear kernel in Fig. 41.3. The reader is referred to [41.13] for additional details.

41.3 Token-Based Methods

Unlike acoustic approaches in which fixed-duration windows of the speech signal are used to extract features for classification, token-based approaches segment the input speech signal into logical units or tokens. These units could be based on a phonetic segmentation (e.g., phones), or a data-driven segmentation (e.g., GMM mixture component tokens [41.14]). Features are then extracted from the token stream, and classification is performed using a language model. Figure 41.4 illustrates the process.

This basic architecture allows for a variety of token-based approaches, but the phoneme recognition followed by language modeling (PRLM) approach originally described in [41.15, 16] has been shown to work well for language recognition. The remainder of this section will focus on PRLM and related techniques that model the phonotactic properties (phone stream dependencies) of languages. Other token-based techniques that should be noted include the work of [41.14] in which abstract tokens were used in place of phonetic units to achieve near state-of-the-art language recognition performance.

41.3.1 Tokens

In the PRLM framework, the tokenizer is a phone recognizer (PR) that converts an input speech signal into a sequence of phonetic units (phones). Each phone and its surrounding contexts are then combined to form phone n-grams, the features used for scoring. These features are scored against n-gram language models that correspond to specific target languages to make language decisions. Intuitively, these features represent the phonotactic dependencies (the rules governing the distribution of phonetic sounds within a language) between individual phonetic units. In an early work [41.17], *House* and *Neuberg* used transcribed data to demonstrate the feasibility of exploiting phonotactics to perform automatic language identification. Figure 41.5 illustrates the process.

Phone Recognition

Phone recognition is typically accomplished using hidden Markov models (HMMs) to represent phone units. Figure 41.6 shows a standard topology for different phonetic units. During recognition, these models are combined to allow any phone to transition to any other phone with equal probability. This configuration is often referred to as an *open phone loop* or *null* grammar, as shown in Fig. 41.7.

The phone decoding process is based on a sequence of observations $\boldsymbol{X} = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_T)$, where T is the number of speech frames. The observation $\boldsymbol{x}_t$ is a vector of acoustic features, usually mel-frequency cepstrum coefficients (MFCC) [41.1] or perceptual linear predic-

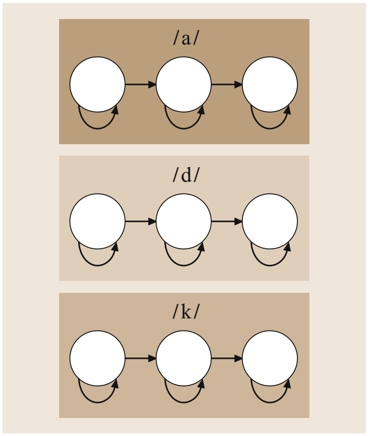

Fig. 41.6 HMM: typical HMM topology for phonetic units

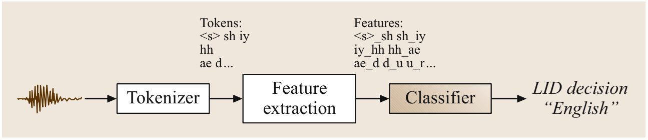

Fig. 41.4 Token-based language recognition

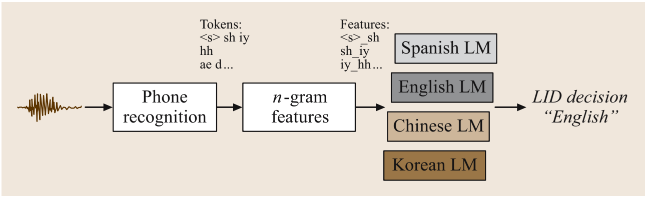

Fig. 41.5 PRLM: phone recognition followed by language modeling

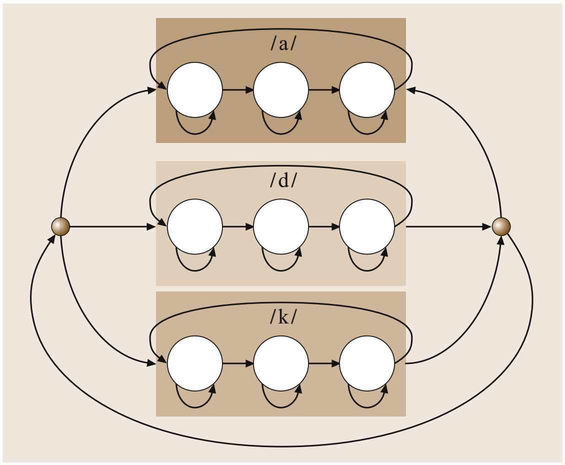

Fig. 41.7 Open phone loop (*null* grammar) with phone HMMs

tion (PLP) coefficients [41.18]). These vectors may also include dynamic features (i. e., first and second differences). Using standard assumptions (see Part E of this Handbook), the probability of this sequence with respect to the language model λ is

$$p(\boldsymbol{X}|\lambda) = \sum_{\forall s} \prod_{t=1}^{T} p(\boldsymbol{x}_t|s_t,\lambda) \cdot P(s_t|s_{t-1},\lambda) \,, \quad (41.15)$$

where $\boldsymbol{s} = s_1, \cdots, s_t, \cdots, s_T$ is a hypothesized sequence of states at times $t = 1, \cdots, T$. Each state s_t of the model has an observation probability $p(\boldsymbol{x}_t|s_t,\lambda)$ and a series of transition probabilities between all states and s_t. For simplicity, a bigram model $P(s_t|s_{t-1},\lambda)$ has been selected for the state transition model. With a null grammar, the probabilities $P(s_t|s_{t-1},\lambda)$ are uniform for fixed s_{t-1}. Typically, the observation probability $p(\boldsymbol{x}_t|s_t,\lambda)$ is modeled as a mixture of Gaussians resulting in a continuous-density HMM.

Given a sequence of observation vectors $\boldsymbol{X}$, the phone recognizer attempts to decode the maximum likelihood sequence of HMM states corresponding to $\boldsymbol{X}$:

$$\underset{\boldsymbol{s}}{\operatorname{argmax}}\ P(\boldsymbol{s}|\boldsymbol{X}) \,. \quad (41.16)$$

Applying Bayes' rule and recognizing that the maximization is independent of $p(\boldsymbol{X})$:

$$\underset{\boldsymbol{s}}{\operatorname{argmax}}\ \frac{p(\boldsymbol{X}|\boldsymbol{s})p(\boldsymbol{s})}{p(\boldsymbol{X})} = \underset{\boldsymbol{s}}{\operatorname{argmax}}\ p(\boldsymbol{X}|\boldsymbol{s})P(\boldsymbol{s}) \,. \quad (41.17)$$

Equation (41.17) is the Viterbi approximation of (41.15); the goal is to maximize the following probability

$$p(\boldsymbol{X}|\boldsymbol{s},\lambda)P(\boldsymbol{s}|\lambda) = \prod_{t=1}^{T} p(\boldsymbol{x}_t|s_t,\lambda) \cdot P(s_t|s_{t-1},\lambda) \quad (41.18)$$

by selection of the state sequence $\boldsymbol{s}$.

The Viterbi algorithm is generally used to search the space of possible state sequences and a variety of pruning strategies can be applied to keep the search problem tractable. For language recognition, context independent phonetic units are typically used for decoding (see [41.19] as an exception). For a more-detailed description of HMM models of speech refer to Part E of this Handbook.

Phone Recognizer Training

A vital consideration in the training of phone recognizers is the availability of phonetically or orthographically labeled speech for multiple languages. These phone transcripts can be obtained from manual labeling by a trained listener or automatic labeling from a word transcript and pronunciation dictionary. As discussed later, the languages of the phone recognizers need not be those of the target languages. Phonetically transcribed corpora are relatively uncommon, and for a number of years were available only as part of the Oregon Graduate Institute multilanguage telephone speech (OGI-TS) corpus [41.20], which contained phonetically hand-labeled speech for six languages (English, German, Hindi, Japanese, Mandarin, and Spanish) collected over telephone channels. More recently, investigators have employed other corpora for training phone recognizers, including Switchboard (English) [41.21] (e.g., [41.22, 23]), CallHome (Arabic, Japanese, Mandarin, and Spanish) [41.21] (e.g., [41.22, 23]), and SpeechDat-East (Czech, Hungarian, Polish, Russian, and Slovak)d [41.24] (e.g., [41.8]).

41.3.2 Classifiers

Using the sequence of phone tokens, a number of different classification techniques can be applied to make a language recognition decision. Two popular techniques are described here in detail: PRLM, a generative model of phonetic n-gram sequences, and PR-SVM-LATT (phone recognition followed by lattice-based support vector machines), a discriminative approach to language recognition. Both approaches assume that languages differ in their phonotactic characteristics and that these differences will cause a phonetic recognition system to produce different distributions of token sequences.

PRLM

The basic PRLM decision rule is described by

$$L^* = \underset{L}{\operatorname{argmax}}\, P_L(W|\boldsymbol{X})\,, \tag{41.19}$$

where $\boldsymbol{X}$ are the acoustic observations, W is the hypothesized token sequence $W = w_1, \ldots, w_N$, and L^* is the optimal language decision. The term $P_L(W|\boldsymbol{X})$ can be decomposed using Bayes' rule:

$$P_L(W|\boldsymbol{X}) = \frac{\overbrace{p(\boldsymbol{X}|W)}^{\text{same } \forall L} \cdot P_L(W)}{\underbrace{p(\boldsymbol{X})}_{\text{same } \forall L}}\,. \tag{41.20}$$

For a given tokenizer, the terms $p(\boldsymbol{X}|W)$ and $p(\boldsymbol{X})$ are the same across language models, since $\boldsymbol{X}$ is known and W is determined from the phone recognition decoding process described above. As such, the following simplified PRLM decision rule can be applied:

$$L^* = \underset{L}{\operatorname{argmax}}\, P_L(W)\,. \tag{41.21}$$

As (41.21) suggests, only language model scores are used to make a language recognition decision. Phone recognition acts solely to discretize the input speech signal.

In real implementations $P_L(W)$ is approximated by an n-gram language model of fixed order,

$$P_L(W) \approx \prod_{i=1}^{N} \overbrace{P_L(w_i|w_{i-1} \ldots w_{i-(n-1)})}^{\text{language model}}\,. \tag{41.22}$$

Here, the probability $P_L(w_i|w_{i-1} \ldots w_{i-(n-1)}) = P_L(\hat{w}_i)$, is a look up of the frequency of occurrence of n-gram $\hat{w}_i = w_{i-n+1}w_{i-n+2} \cdots w_i$ in language L's training data. n-grams are a simple yet effective way of capturing the context preceding a token. For the language recognition problem, n-grams of order two and three (i. e., bigrams and trigrams) are commonly employed. Increasing the n-gram order should theoretically increase the ability of language recognition modeling techniques to incorporate longer term dependencies, but in practice these models are difficult to estimate given the paucity of training data. As the order n increases, the number of possible unique n-grams increases exponentially as $|\mathcal{W}|^n$, where $|\mathcal{W}|$ is the size of the tokenizer's phone inventory. Standard smoothing techniques have been shown to help mitigate the n-gram estimation problem for language recognition tasks [41.25]. Other language modeling techniques, such as binary decision trees, have also been used successfully to model n-gram sequences with $n > 3$ [41.26].

Lattice-Based PRLM. As stated above, the standard 1-best PRLM model relies solely on the probability of the phone token sequence in its decision rule. This acts as an approximation of the acoustic hypothesis space generated by the underlying HMM phonetic models for a given input.

In [41.22], a better approximation using an extension of the 1-best PRLM model is proposed in which posterior probabilities of phone tokens are incorporated into the estimation of the language models and the observation likelihoods. In this model, estimates of expected counts derived from phone lattices are used in place of 1-best counts during LM training and scoring.

To derive this extension, the standard 1-best language modeling equation can be reformulated in log form as

$$\log P_L(W) = \sum_{\forall \hat{w}} \mathrm{C}(\hat{w}) \log P_L(w_i|w_{i-1}, \ldots, w_{i-n+1})\,, \tag{41.23}$$

where

$$\hat{w} = w_i w_{i-1} \ldots w_{i-n+1} \tag{41.24}$$

and $C(\hat{w})$ is the count of the n-gram $\hat{w}$ in the sequence W. In (41.23), the sum is performed over the *unique* n-grams $\hat{w}$ in W.

In the lattice formulation, n-gram counts from the 1-best hypothesis are replaced with expected counts from a phone lattice generated by the decoding of an input message:

$$E_{\mathcal{L}}[\log P_L(W)] = \sum_{\forall \hat{w}} E_{\mathcal{L}}\left[\mathrm{C}(\hat{w})\right] \log P_L(w_i|w_{i-1} \ldots, w_{i-n+1})\,.$$

Like the 1-best hypothesis, the phone lattice is only an approximation of the acoustic hypothesis space for a given speech utterance. There is evidence that both the quality of the phonetic models and the phone lattices used for language recognition are important [41.19,27]. Many methods of generating lattices have been proposed in the automatic speech recognition (ASR) literature [41.28–30]. The Viterbi n-best technique with a fixed number of trace-backs per state is often used [41.31].

Parallel PRLM (PPRLM)

An important aspect of the PRLM method, implied by (41.21), is that the operations of tokenization (by the phone recognizer) and language modeling are decoupled. As a consequence, it is not necessary for the phone

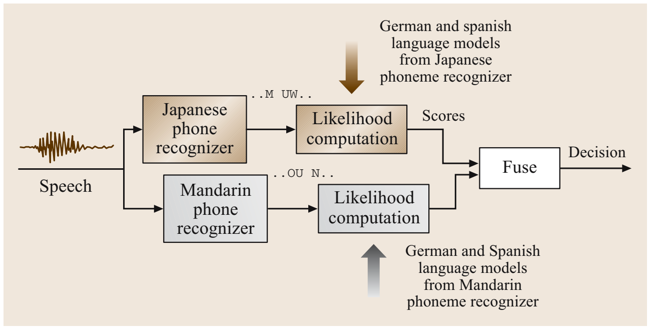
German and spanish language models from Japanese phoneme recognizer
Japanese phone recognizer
..M UW..
Likelihood computation
Scores
Fuse
Decision
Speech
Mandarin phone recognizer
..OU N..
Likelihood computation
German and Spanish language models from Mandarin phoneme recognizer

Fig. 41.8 PPRLM: *parallel* phone recognition followed by language modeling

recognizer to be trained in any of the target languages as long as the tokenization of the target languages is sufficiently distinct. In a natural extension of this observation, *Zissman* [41.15] proposed running phone recognizers for multiple languages in parallel in order to provide more-diversified coverage of the phones encountered in target languages. The resulting system configuration, commonly known as parallel PRLM (PPRLM), employs a bank of phone recognizers, each of which is followed by a set of language models trained from the corresponding token sequences that it generates. As with PRLM, the multiple phone recognizers are trained using phonetically or orthographically labeled speech, but need not be of any of the target languages. An example is shown in Fig. 41.8 where parallel Japanese and Mandarin phone recognizers are used to tokenize speech in a German versus Spanish language recognition task. The next section will describe methods by which the individual outputs are fused to produce a final score.

Language recognition systems based on PPRLM were found to outperform their competitors [41.5] and dominated the field of language recognition for a number of years. More recently, employing banks of lattice-based phone recognizers has led to further language recognition performance improvement [41.22]. Efforts have also been made to reduce the computation load of a PPRLM system by employing a single multilingual phone recognizer but to date these approaches have not proved to be more effective than PPRLM [41.32].

PPR-SVM-LATT

The PPR-SVM-LATT approach is an application of SVM classification for token streams [41.33]. For the PPR-SVM-LATT classifier, the phone token sequence for each utterance $\boldsymbol{X}$ in a language corpus is used to compute a vector of n-gram probabilities,

$$\boldsymbol{b} = \begin{pmatrix} c_1 p(\hat{w}_1|\boldsymbol{X}) \\ c_2 p(\hat{w}_2|\boldsymbol{X}) \\ \vdots \\ c_N p(\hat{w}_N|\boldsymbol{X}) \end{pmatrix} . \tag{41.25}$$

The probability $p(\hat{w}_i|\boldsymbol{X})$ represents the frequency of occurrence of n-gram $\hat{w}_i$ in the utterance $\boldsymbol{X}$, and $N = |\mathcal{W}|$ is the number of possible n-grams. The n-gram probabilities can be computed from counts from the 1-best phone sequence or from expected counts from phone lattices. Experiments have determined that expected counts from lattices produce better performance.

The values c_i in (41.25) are weights that normalize the components of the vector to a common scale. Typically, a weighting such as the term-frequency log likelihood ratio (TFLLR) [41.33] can be applied. For TFLLR,

$$c_i = \frac{1}{\sqrt{p(\hat{w}_i|\{\boldsymbol{X}_j\})}} , \tag{41.26}$$

the probability $p(\hat{w}_i|\{\boldsymbol{X}_j\})$ is calculated across all utterances of all languages in the training corpus.

After converting utterances to vectors, a simple linear kernel, $K(\boldsymbol{b}_1, \boldsymbol{b}_2) = \boldsymbol{b}_1^{\mathrm{T}}\boldsymbol{b}_2$, is used, where the $\boldsymbol{b}_i$ are the expansion of two utterances as in (41.25). Training and scoring are done in a manner analogous to the cepstral SVM.

41.4 System Fusion

The previous sections of this chapter described a variety of methods available for implementing a language recognition system. In practice, additional performance improvements can be obtained by running multiple systems in parallel and fusing the individual scores using a back-end. If the recognizers do not behave uniformly (i. e., their errors are not completely correlated) then combining the scores may make it possible to exploit complementary information that exists in the scores. Score fusion was first adopted as a means of combining the scores produced by a PPRLM language recognizer and has since been applied to fusing scores of different types of recognizers, such as those described in Sects. 41.2 and 41.3.

41.4.1 Methods

For purpose of this discussion it is convenient to divide approaches to fusion into two types, rule based and classifier based. Rule-based fusion applies a fixed formula to the scores, such as a weighted-sum rule or product rule, where weights are either uniform or are chosen empirically using a set of development data. One example of a rule-based method for combining the scores in a PPRLM system will be discussed below. The second approach to fusion utilizes a development data set to train a classifier from a feature vector constructed from the individual system scores. In theory, the joint statistics among the scores (elements of the feature vector) can be exploited by the classifier to produce overall performance that is superior to that achieved with a rule-based back.end.

Product-Rule Fusion

A method for fusing the scores of a PPRLM language recognition system using the product rule was proposed by *Zissman* [41.5]. Assume a PPRLM language recognizer that uses K phone recognizers to detect L languages. For each test input, a set of KL (linear) scores is generated, with each phone recognizer producing L scores. For an input utterance $\boldsymbol{X}$, single per-language scores $y(\boldsymbol{X}|l)$ are computed by first normalizing the likelihoods $s_{k,l}$ produced by phone recognizer k and then multiplying the normalized per-language values across all phone recognizers:

$$y(\boldsymbol{X}|l) = \prod_{k}^{K} \frac{s_{k,l}}{\sum_{l}^{L} s_{k,l}} \,. \qquad (41.27)$$

The normalization step puts scores across phone recognizers into a common range, and the multiplication step creates a final score. Probabilistically, the first step turns the raw phone recognizer scores into posteriors (with an assumption of equal priors) and the second step computes a joint probability of the observed data under the assumption that the information sources (individual PRLM systems) are independent. A potential drawback of product-rule fusion is that a single score close to zero may have an undesirably large impact on the final result. Despite the apparent inaccuracy of the independence assumption, the product-rule method proved to be an effective way of combining PPRLM scores [41.5].

Classifier Fusion

Classifier-based fusion for PPRLM language recognition systems was originally proposed by *Yan* and *Bernard* [41.34] using neural networks and by *Zissman* [41.35] using Gaussian classifiers, and these authors reported that performance was superior to that obtained with the product-rule approach. A block diagram of a Gaussian-based fusion approach for language recognition is shown in Fig. 41.9. The method consists of linear discriminant analysis (LDA) normalization followed by Gaussian classification. Although there can be no guarantee that this particular classifier-based fusion architecture produces optimum results in all language recognition applications, extensive development testing has shown that the technique produces reliably superior performance. Other investigators have reported results using neural network-based back-end fusion [41.22].

The generic block diagram in Fig. 41.9 shows a set of core language recognizers (e.g., GMM, SVM, PRLM, PPRLM), each of which produces a score or score vector for each language hypothesis given an input utterance. These scores are used to form the elements of a feature vector that is transformed via linear discriminant analysis, a method by which the feature vector is projected into a reduced dimensional space in such a way as to maximize interclass separability and minimize intraclass variability. In the figure, the transformed space is of dimension $L-1$, where L is the number of classes (languages) represented in the back-end training data. The transformed vector also has the desirable property that its features are decorrelated. It is important to recognize that the individual recognizers may produce different sized score vectors and that the scores may represent likelihoods of any languages, not just the target languages.

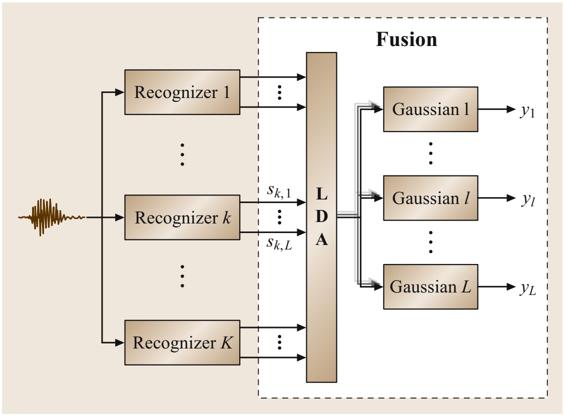

Fig. 41.9 Gaussian-based fusion for language recognition

The transformed feature vectors are then processed by a parallel bank of Gaussian classifiers. Each Gaussian has input dimension $L-1$, a diagonal grand covariance (i. e., pooled across all the back-end training data), and produces an output $y(\boldsymbol{X}|l)$.

41.4.2 Output Scores

The form of the final system scores depends on the nature of the language recognition application. For closed-set identification, the index of the maximum score, $\max_l y(\boldsymbol{X}|l)$, produced using either the rule- or classifier-based method is sufficient to identify the winning language hypothesis. For a detection task, where decisions are made using a fixed threshold, the output score must be formed in such a way as to be consistently meaningful across utterances. Typically, this is accomplished by interpreting the (linear) scores $y(\boldsymbol{X}|l)$ as likelihoods and forming an estimated likelihood ratio $r(\boldsymbol{X}|l)$ between the target language score and the sum (or average) of the others:

$$r(\boldsymbol{X}|l) = \frac{y(\boldsymbol{X}|l)}{\frac{1}{L-1}\sum_{j\neq l}^{L} y(\boldsymbol{X}|j)} . \qquad (41.28)$$

The estimated likelihood ratio can then be used to set a threshold to achieve a specific application dependent operating point.

Language recognition applications often require that scores be reported as posterior probabilities rather than as likelihoods or likelihood ratios. This may be the case when the downstream consumer (human or machine) requires confidence scores or scores that have been mapped from unconstrained likelihoods or likelihood ratios to a normalized range (0–1 or 0–100). In principle, the scores $y(\boldsymbol{X}|l)$ produced by the back-end fusion for input utterance $\boldsymbol{X}$ can be used directly to generate posterior probabilities $P(l)$ for each target language l via Bayes' rule:

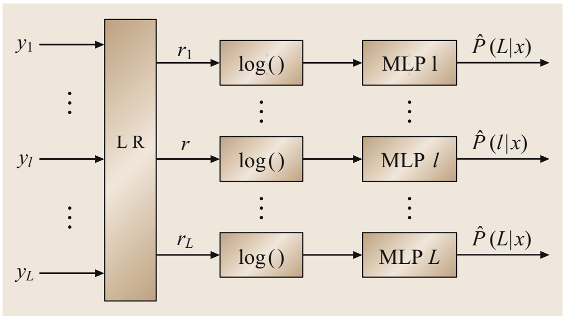

Fig. 41.10 Posterior probability estimation

$$P(l|\boldsymbol{X}) = \frac{y(\boldsymbol{X}|l)P_l}{\sum_l^L y(\boldsymbol{X}|l)P_l} , \qquad (41.29)$$

where P_l represents the prior probability of target language l. In practice, posteriors estimated this way may be unreliable due to the inaccurate assumptions made in constructing the overall system. Rather than treating the scores as true likelihoods, it is preferable to view them as raw scores and to estimate posteriors using another classifier.

A block diagram of a method that has been found useful for posterior probability estimation is shown in Fig. 41.10. The fusion back-end log-likelihood ratio estimates $\log r(\boldsymbol{X}|l)$ are treated as raw inputs to a bank of language-dependent single-input logistic regression classifiers (implemented using LNKnet [41.36] as multilayer perceptrons) that produce estimates $\hat{P}(l|\boldsymbol{X})$ of the posterior probabilities. These estimates can then be interpreted by humans as meaningful measures of confidence.

41.5 Performance Assessment

The language recognition approaches described in this chapter have been subject to extensive development, and this section describes their evaluation under conditions that follow protocols established by the US National Institute of Standards and Technology (NIST). Since 1996, NIST has coordinated several evaluations with the goal of assessing the existing state-of-the-art in language recognition technology. The most recent language recognition evaluation (LRE 2005) protocol will form the basis of the performance results presented in this section. A description of the 2005 NIST language recognition evaluation plan can be found in [41.37].

41.5.1 Task and Corpus

The task for LRE 2005 was the detection of the presence of one of seven target languages (English, Hindi, Japanese, Korean, Mandarin, Spanish, Tamil) in telephone speech utterances with nominal durations of 30, 10, and 3 seconds selected from unspecified languages and sources. The evaluation material contained 3,662 segments for each nominal duration. Although NIST evaluated and reported performance results under several conditions, its primary evaluation condition consisted of trials for which the segments had 30 second durations, were from one of the seven target languages, and were from material collected by the Oregon Health Sciences University (OHSU). Unless otherwise stated, the results presented here will be for the NIST primary evaluation condition. For a complete description of LRE 2005, see [41.38].

Training data for the systems described below was drawn from four telephone speech corpora: CallFriend, CallHome, Mixer, and Fisher, and cover 13 languages (for more information on these corpora, see [41.21]). CallFriend and CallHome contain conversations between acquaintances, while Mixer and Fisher contain conversations between strangers. For a detailed description of the use of this material for training the recognizers and the fusion back-end, see [41.23].

41.5.2 Systems

The fully fused system contained five core language recognizer components. For each recognizer, nonspeech frames were removed from the train and test utterances using a GMM-based speech activity detector.

GMM

The Gaussian mixture model (GMM)-based language recognizer used 2048 mixture components, shifted delta cepstral (SDC) coefficients, feature normalization across utterances to zero mean and unit variance on a per-feature basis, and feature mapping [41.3]. The SDC parameters N-d-P-k were set to 7-1-3-7 (Sect. 41.2.1), resulting in a 49-dimension feature vector computed over a 21-frame (210 ms) time span. In addition, gender-dependent language models were trained for each of the 13 languages in the training material. A background model was trained by pooling all the training material, and gender-dependent target language models were trained by adapting from the background model. This method of training permits fast scoring during recognition and was proposed by [41.39] for language recognition. The GMM system produced a total of 26 scores per test utterance.

SVM

The support vector machine (SVM)-based language recognizer used a generalized linear discriminant sequence (GLDS) kernel, expansion into feature space using monomials up to degree three, and SDC coefficients derived as for the GMM recognizer. The features were normalized across utterances to zero mean and unit variance on a per-feature basis. Thirteen language models were constructed from the training material and 13 scores were produced per test utterance.

PPRLM

The PPRLM-based language recognizer used six OGI-trained phone recognizers, a 39-dimensional MFCC feature vector containing cepstra, delta, and double delta coefficients, and trigram language modeling. Each phone recognizer's token sequences were used to train 13 language models, resulting in an output feature vector of dimension 78. The phone recognizers and language models were trained using Cambridge University engineering department's hidden Markov model toolkit (HTK) [41.40].

PPRLM-LATT

A PPRLM based language recognizer using lattices to generate probabilistic counts of phone frequencies for bigram and trigram language modeling. Phone recognizers were trained using material from CallHome (Arabic, Japanese, Mandarin) and Switchboard English (landline and cellular). A total of seven phone recognizers were trained, resulting in 91 (7×13) scores per input utterance. The tokenizers were selected based on results of development testing conducted in conjunction with LRE 2005 ([41.41]).

PPR-SVM-LATT

The PPR-SVM-LATT-based language recognizer used lattices to generate probabilistic counts of phone frequencies that are then classified with support vector machines. Probability vectors were constructed from unigrams and bigrams, and a linear kernel with TFLLR weighting was applied. Lattices of phones were produced using HMM models trained on six OGI languages, as in PPRLM. A total of 78 (6×13) scores were produced per input utterance.

The scores of the core language recognizers were stacked into a 286-dimension feature vector and fed to the fusion back-end for dimension reduction (linear discriminant analysis) and classification (seven Gaussians with pooled diagonal covariances). The outputs of the back-end were treated as estimated target language likelihoods and log likelihood ratios were formed by taking the log of the quantity in (41.28). These scores were treated as the final outputs of the system.

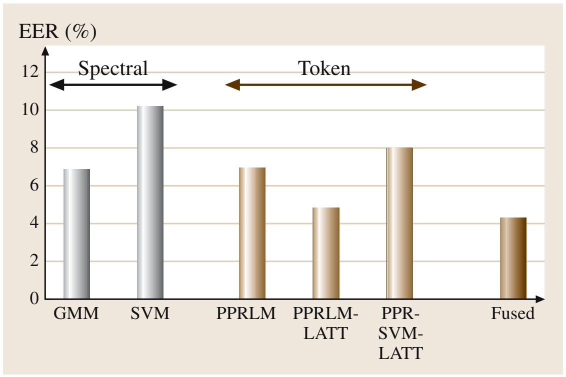

Fig. 41.11 Performance (percentage EER) of spectral, token, and fused language recognition systems on the 2005 NIST LRE primary condition test

41.5.3 Results

NIST language recognition evaluations require participants to provide two outputs for each test utterance, a hard decision and an arbitrarily scaled likelihood score. The hard decisions are used by NIST to evaluate a cost function while the scores are used to sweep out a detection error trade-off (DET) curve [41.42], a variant of the familiar receiver operating characteristic (ROC) curve. To compare detection performance across competing systems, it is convenient to use a summary statistic such as the equal error rate (EER), and this statistic has been adopted for comparison of the language recognition systems described in this section. Results for the NIST primary condition described above, given in percentage EER, are shown in Fig. 41.11. The recognizers have been categorized according to whether they are spectral (GMM or SVM) or token based (PPRLM, PPRLM-LATT, PPR-SVM-LATT). Individually, the best performing recognizer is PPRLM-LATT, while the SVM system produces the most errors. The remaining systems (GMM, PPRLM, and PPR-SVM-LATT) perform comparably. It is worth noting that fusing the scores of the spectral systems (SVM and GMM) results in a large performance gain [41.23]. Fusing the scores of all five recognizers produces the best overall result.

Figure 41.12 shows the DET plots for five-recognizer-system fusion for the LRE 2005 OHSU target language test segments with durations 30, 10, and 3 seconds. Table Table 41.1 gives the corresponding EERs for these DET curves. It is apparent that language recognition performance becomes increasingly challenging as the test segment duration decreases, although recent work has demonstrated that performance at shorter durations can be substantially improved [41.8].

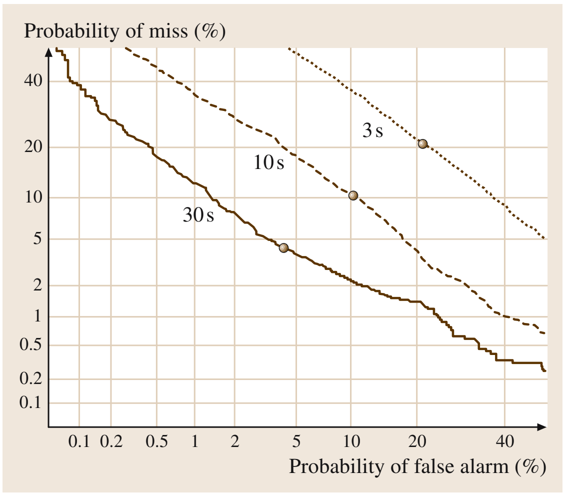

Fig. 41.12 Performance of five-system fusion on the LRE 2005 OHSU target language test segments for durations 30 s (*solid line*), 10 s (*dashed line*), and 3 s (*dotted line*). Filled circles indicate equal error points

41.5.4 Computational Considerations

Although the focus of the discussion of language recognition systems has been performance, any practical implementation of such a system needs to account for computational complexity. A comparison of process-

Table 41.1 Performance, measured as percentage equal error rate (EER) of full-fusion system for 30 s, 10 s, and 3 s test utterances

Duration	EER(%)
30	4.3
10	10.3
3	21.2

Table 41.2 Processing speed of selected language recognizers as measured with a 3 GHz Xeon processor with 2 GB RAM running Linux. The recognition task involved 12 target languages and five minute audio file. The PPRLM system contained six OGI-trained phone recognizers (no lattices). All factors refer to speeds faster than real time (RT)

Reognizer	Speed
GMM	17 RT
SVM	52 RT
PPRLM	2 RT

ing speed for three types of recognizers is shown in Table 41.2. Note that all factors indicate the degree to which the algorithms are *faster* than real time. All measurements were made using 12 language models and a single 5 minute audio file. The platform was a 3 GHz Xeon processor with 2 GB RAM and Linux OS. The GMM, SVM, and PPRLM systems were described in previous sections of this chapter. The PPRLM language recognizer used six OGI tokenizers and no lattices.

41.6 Summary

This chapter has presented a variety of methods for implementing automatic language recognition systems. Two fundamental approaches, based on either the spectral or phonotactic properties of speech signals, were successfully exploited for the language recognition task. For the spectral systems, these included GMM and SVM modeling using frame-based shifted delta cepstral coefficients. For phonotactic systems, variations of the classic parallel phone recognition followed by language modeling were described. Performance gains can be achieved by fusing the scores of multiple systems operating simultaneously, and a framework for implementing classifier based fusion was described. Finally, performance results were presented for the spectral, token, and fused systems using the standardized protocols developed by NIST. For the latest advances in the field of automatic language recognition, the reader is referred to the most recent proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), the Odyssey Speaker and Language Recognition Workshop, and the International Conference on Spoken Language Processing (Interspeech – ICSLP).

References

41.1 S. Davis, P. Mermelstein: Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences, IEEE Trans. Acoust. Speech Signal Process. (1980) pp. 357–366

41.2 H. Hermansky, N. Morgan, A. Bayya, P. Kohn: Compensation for the Effect of the Communication Channel in Auditory-Like Analysis of Speech, Proc. Eurospeech (1991) pp. 1367–1371

41.3 D.A. Reynolds: Channel robust speaker verification via feature mapping, Proceedings of the, International Conference on Acoustics Speech and Signal Processing (2003) pp. II–53, –56

41.4 P.A. Torres-Carrasquillo, E. Singer, M.A. Kohler, R.J. Greene, D.A. Reynolds, J.R. Deller Jr.: Approaches to language identification using Gaussian mixture models and shifted delta cepstral features, International Conference on Spoken Language Processing (2002) pp. 89–92

41.5 M. Zissman: Comparison of Four Approaches to Automatic Language Identification of Telephone Speech, IEEE Trans. Speech Audio Process. **4**(1), 31–44 (1996)

41.6 A. Dempster, N. Laird, D. Rubin: Maximum likelihood from incomplete data via the EM algorithm, J. R. Stat. Soc. **39**, 1–38 (1977)

41.7 D.A. Reynolds, T.F. Quatieri, R. Dunn: Speaker Verification Using Adapted Gaussian Mixture Models, Digital Signal Process. **10**(1-3), 19–41 (2000)

41.8 P. Matějka, L. Burget, P. Schwarz, J. Černocký: Brno University of Technology System for NIST 2005 Language Recognition Evaluation, Proc. IEEE 2006 Odyssey: The Speaker and Language Recognition Workshop (2006)

41.9 V.N. Vapnik: *Statistical Learning Theory* (Wiley, New York 1998)

41.10 R. Collobert, S. Bengio: SVMTorch: Support Vector Machines for Large-Scale Regression Problems, J. Mach. Learn. Res. **1**, 143–160 (2001)

41.11 W.M. Campbell: Generalized Linear Discriminant Sequence Kernels for Speaker Recognition, Proceedings of the International Conference on Acoustics Speech and Signal Processing (2002) pp. 161–164

41.12 J.C. Platt: Probabilities for SV Machines. In: *Advances in Large Margin Classifiers*, ed. by A.J. Smola, P.L. Bartlett, B. Schölkopf, D. Schuurmans (MIT, Cambridge 2000) pp. 61–74

41.13 W.M. Campbell, J.P. Campbell, D.A. Reynolds, E. Singer, P.A. Torres-Carasquillo: Support vector machines for speaker and language recognition, Comput. Speech. Lang. **20**(2-3), 210–229 (2006)

41.14 P.A. Torres-Carrasquillo, D.A. Reynolds, J.R. Deller Jr.: Language Identification Using Gaussian Mixture Model Tokenization, Proceedings of the International Conference on Acoustics Speech and Signal Processing (2002) pp. 757–760

41.15 M. Zissman, E. Singer: Automatic Language Identification of Telephone Speech Messages Using Phoneme Recognition and n -Gram Modeling, Proceedings of the International Conference on

Acoustics Speech and Signal Processing (1994) pp. 305–308

41.16 Y. Yan, E. Barnard: An Approach to Automatic Language Identification Based on Language-Dependent Phone Recognition, Proceedings of the International Conference on Acoustics Speech and Signal Processing (1995) pp. 3511–3514

41.17 A. House, E. Neuberg: Toward automatic identification of the language of an utterance. I, Preliminary methodological considerations, J. Acoust. Soc. Am. **62**, 708–713 (1977)

41.18 H. Hermansky: Perceptual linear predictive (PLP) analysis for speech, J. Acoust Soc. Am. **87**, 1738–1752 (1990)

41.19 D. Zhu, M. Adda-Decker, F. Antoine: Different Size Multilingual Phone Inventories and Context-Dependent Acoustic Models for Language Identification, Proc. Interspeech (2005) pp. 2833–2836

41.20 Y.K. Muthusamy, R.A. Cole, B.T. Oshika: The OGI Multi-language Telephone Speech Corpus, International Conference on Spoken Language Processing (1992) pp. 895–898

41.21 Linguistic Data Consortium, http://www.ldc.upenn.edu/ (2007)

41.22 J.L. Gauvain, A. Messaoudi, H. Schwenk: Language Recognition Using Phoneme Lattices, International Conference on Spoken Language Processing (2004) pp. 2833–2836

41.23 W. Campbell, T. Gleason, J. Navritil, D. Reynolds, W. Shen, E. Singer, P. Torres-Carrasquillo: Advanced Language Recognition using Cepstra and Phonotactics: MITLL System Performance on the NIST 2005 Language Recognition Evaluation, Proc. IEEE 2006 Odyssey: The Speaker and Language Recognition Workshop (2006)

41.24 Eastern European Speech Databases for Creation of Voice Driven Teleservices, http://www.fee.vutbr.cz/SPEECHDAT-E/ (2007)

41.25 E. Singer, P.A. Torres-Carrasquillo, T.P. Gleason, W.M. Campbell, D.A. Reynolds: Acoustic, Phonetic, and Discriminative Approaches to Automatic Language Identification, Proceedings of Eurospeech (2003) pp. 1345–1348

41.26 J. Navratil: Recent advances in phonotactic language recognition using binary-decision trees, International Conference on Spoken Language Processing, Vol. 2 (2006)

41.27 P. Matějka, P. Schwarz, J. Černocký, P. Chytil: Phonotactic Language Identification using High Quality Phoneme Recognition, Proceedings of Interspeech (2005) pp. 2833–2836

41.28 F. Weng, A. Stolcke, A. Sankar: New Developments in Lattice-based Search Strategies in SRI's H4 system, Proceedings of DARPA Speech Recognition Workshop (1998) p. 100

41.29 H. Ney, X. Aubert: A word graph algorithm for large vocabulary, continuous speech recognition, International Conference on Spoken Language Processing (1994) pp. 1355–1358

41.30 F. Weng, A. Stolcke, A. Sankar: Efficient Lattice Representation and Generation, International Conference on Spoken Language Processing (1998) pp. 100–100

41.31 R. Schwartz, S. Austin: A Comparison of several approximate algorithms for finding multiple (n-best) sentence hypotheses, Proceedings of the International Conference on Acoustics Speech and Signal Processing (1991) pp. 701–704

41.32 J. Navratil: Automatic Language Identification. In: *Multilingual Speech Processing*, ed. by T.S. Pittsburgh, K. Kirchhoff (Academic, New York 2006)

41.33 W.M. Campbell, J.P. Campbell, D.A. Reynolds, D.A. Jones, T.R. Leek: High-Level Speaker Verification with Support Vector Machines, Proceedings of the International Conference on Acoustics Speech and Signal Processing (2004) pp. I-73–I-76

41.34 Y. Yan, E. Barnard: Experiments for an Approach to Language Identification with Conversational Telephone Speech, Proceedings of the International Conference on Acoustics Speech and Signal Processing (1996) pp. 789–792

41.35 M.A. Zissman: Predicting, Diagnosing and Improving Automatic Language Identification Performance, Proceedings of Eurospeech (1997) pp. 51–54

41.36 R. Lippman, L. Kukolich: LNKnet Pattern Recognition Software Package, http://www.ll.mit.edu/IST/lnknet/ (2007)

41.37 A. Martin, G. Doddington: The 2005 NIST Language Recognition Evaluation Plan, http://www.nist.gov/speech/tests/lang/2005/LRE05EvalPlan-v5-2.pdf (2005)

41.38 A.F. Martin, A.N. Le: The Current State of Language Recognition: NIST 2005 Evaluation Results, Proc. IEEE 2006 Odyssey: The Speaker and Language Recognition Workshop (2006)

41.39 E. Wong, S. Sridharan: Methods to improve Gaussian mixture model based language identification system, International Conference on Spoken Language Processing (2002) pp. 93–96

41.40 Hidden Markov Model Toolkit, http://htk.eng.cam.ac.uk/ (2007)

41.41 W. Shen, W. Campbell, T. Gleason, D. Reynolds, E. Singer: Experiments with Lattice-based PPRLM Language Identification, Proc. IEEE 2006 Odyssey: The Speaker and Language Recognition Workshop (2006)

41.42 A. Martin, G. Doddington, T. Kamm, M. Ordowski, M. Przybocki: The DET Curve in Assessment of Detection Task Performance, Proceedings of Eurospeech (1997) pp. 1895–1898