Chapter 9. Feature Extraction for ASR

Table of Contents

9.1. Sources of Variability in Speech

9.2. Linear Predictive Coding

9.3. Signal Modelling Techniques

In this section we revisit the subject of the last part of SLP801 to look at how the speech signal is processed prior to matching against stored patterns. To provide an overview of the goals and methods of this stage we will look at a paper by Joe Picone (Signal Modeling Techniques in Speech Recognition from the Proceedings of the IEEE, 1993). This paper describes this stage as Signal Modeling to emphasis that we are attempting to derive a parameterised version of the input speech signal which captures it's important qualities while discarding unimportant and distracting features. In addition, we will refer to the last part of Chapter 6 (6.6 onwards) and Chapter 8 from Harrington and Cassidy for details on one signal modelling technique.

9.1. Sources of Variability in Speech

No two utterances of the same word or sentence are likely to give rise to the same digitial signal; this obvious point underlies the difficulty in speech recognition but also means that we may be able to extract more than just a sequence of words from the signal. If we can understand the different sources of variability in the signal then we can begin to approach the problem of either separating them out for subsequent analysis stages.

Let's consider the factors which could cause two random speech samples to differ from one another:

\*

Phonetic identity: the two samples might correspond to different phonetic segments, eg. a vowel and a fricative. Another source of variability is coarticulation between phonemes.

\*

Pitch: pitch and other source features such as breathiness and amplitude can be varied independently.

\*

Speaker: different speakers have different vocal tracts and source physiology. Speakers also get colds, get emotional and do other things to modify their voice properties.

\*

Microphone: and other properties of the transmission channel (eg. fixed vs. mobile telephone).

\*

Environment: background noise, room acoustics, distance from microphone.

Clearly the kind of variability we want to preserve in our signal model for speech recognition is that due to phonetic identity, which is largely due to the vocal tract configuration. All of the other sources might be considered noise to be removed from the signal as far as possible; however, there might be situations when some of these could provide useful information. We are helped by the observation that these sources of variability are largely independant from each other and so might be amenable to some kind of de-convolution operation; remember that this is what we did to separate the source and filter components in cepstral analysis.

The requirements of the signal modelling stage will differ depending on the application environment: telephone speech recognition applications may need to use techniques to compensate for line noise and changes in the properties of the telephone line during each utterance; an in-car system will need to be robust to significant amounts of background noise. These may call for different signal modelling techniques in addition to different approaches to pattern matching.

9.1.1. Separating Phoneme Classes

The aim of signal modelling is to derive a feature vector such that the vectors for the same phoneme are as close to each other as possible while the vectors for different phonemes are maximally different to each other. In other words, the feature vectors we derive for all /a/ vowels should be very similar to each other (by some metric that we have yet to define) while being different to those for any other phoneme class.

This goal can best be achieved by finding a signal transform that isolates the variability due to phonetic class from the other sources of variability in the speech signal.

9.2. Linear Predictive Coding

In order to understand LPC and many other signal processing techniques it is necessary to look at the mathematics behind the z-transform. Unfortunately this is quite a hairy topic for a non-mathematical audience and so can be a hurdle in properly understanding these techniques. In our book we try to give a simple step-by-step coverage of the z-transform and how it relates to digital filtering and I would like to walk through the major points in these notes. The aim of this exercise is so that you can look at, for example, equation 18 in Picone's paper and not only stay upright but maybe even say, "hmm, I see".

The story begins back in Chapter 6 on page 178. If you remember back to 801, we talked about frequency domain filtering as a mulitplication operation between the spectra of the source and filter. In this section we develop a new way of talking about the frequency domain, the z-transform, such that we can characterise a signal and a filter as a polynomial, and perform the filtering operation by multiplying these together.

9.2.1. The Spectrum and the Z-transform

Section 6.6.1 explains how the amplitude and phase of a digital signal can be expressed as a vector of complex numbers and in particular that the spectrum of an impulse [1 0 0 0 ...] can be expressed as the vector X[k] = 1 + 0i for all k. This corresponds to saying that the unit impulse is the sum of sinusoids at all (digital) frequencies with amplitude 1 and phase 0 radians. Following this we see that the spectrum of a time shifted version of this impulse signal differs only in the phase component and can be neatly expressed via Euler's relation as X[k] = Aexp(-iWkp) (where exp denotes the exponential and the signal has an amplitude of A and has been shifted by P points relative to the original unit impulse). We then define a new variable z = exp(iWk) so that the shifted spectrum can be written as X(z) = Az-p -- a polynomial in z or the z-transform of the shifted impulse signal.

We next note that any signal can be seen as the sum of weighted, shifted, impulse signals ([2 3 4 5] = 2\*[1 0 0 0] + 3\*[0 1 0 0 ] etc). Hence (since we are dealing with linear time-invariant signals) the z-transform of an arbitary signal can be derived from the z-transforms of the shifted impulses.

x[n] = [4 2 1 3 0 0 0 0]

X(z) = 4 + 2z-1 + z-2 + 3z-3

(note that there's a typo in this example in the book (top of p185) where the exponent of the third term is -3 instead of -2.)

So, now we know what a z-transform is (a way of writing the spectrum of a signal) and we can derive the z-transform for any digital signal we come across. Note that by substituting z = exp(iWk), where k is a vector of digital frequency values, into any z-transform we can get back to the DFT of a signal.

9.2.2. Convolution as Multiplication

We know that we can apply a digital filter either by convolving a time domain signal with the impulse response of the filter or by multiplying the fourier spectra of the signal and the filter. Section 6.6.2 describes how we can also apply a filter by multiplying the z-transforms of the signal and the filter. In 6.6.3 we see that writing the source filter equation in terms of z-transforms:

Y(z)A(z) = B(z)X(z)

allows us to combine the recursive (A(z)) and non-recursive (B(z)) parts of the filter into a single transfer function H(z) which is the z-transform of the impulse response of the filter -- that is, the z-transform of the signal you would get if you passed a unit impulse [1 0 0 0 ...] through the filter. H(z) is a complicated polynomial in z but knowing that we can characterise any filter by a polynomial H(z) is the important point to take away.

Figure 6.16 summarises the relationship between signals, spectra and z-transforms. The impulse response and the spectrum are two ways of characterising the effect of filter; either can be derived from the convolution equation and we can transform between them using the DFT and IDFT.

9.2.3. LPC analysis

Moving now to Chapter 8 we will begin to look at Linear Predictive Coding. LPC is another method of separating out the effects of source and filter from a speech signal; similar in intention to cepstral analysis but using quite different methods. One way of thinking about LPC is as a coding method -- a way of encoding the information in a speech signal into a smaller space for transmission over a restricted channell. LPC encodes a signal by finding a set of weights on earlier signal values that can predict the next signal value:

y[n] = a[1]y[n-1] + a[2]y[n-1] + a[3]y[n-3] + e[n]

If values for a[1..3] can be found such that e[n] is very small for a stretch of speech (say one analysis window), then we can transmit only a[1..3] instead of the signal values in the window. The speech frame can be reconstructed at the other end by using a default e[n] signal and predicting subsequent values from earlier ones. Clearly this relies on being able to find these values of a[1..k] but there are a couple of algorithms which can do this (one is covered in the book). The result of LPC analysis then is a set of coefficients a[1..k] and an error signal e[n], the error signal will be as small as possible and represents the difference between the predicted signal and the original.

There is an obvious parallel between the LPC equation and that of a recursive filter (y\*a = x):

y[n] = -a[1]y[n-1] - a[2]y[n-1] - a[3]y[n-3] + ... + x[n]

where we have rearranged the terms as in Equation 8.9 in the text. The LPC coefficients correspond to those of a recursive filter and the error signal corresonds to a source signal. Moreover, the conditions under which the error signal is minimised in LPC analysis mean that the error signal will have a flat spectrum and hence that the error signal will approximate either an impulse train or a white noise signal. This is a very close match to our source filter model of speech production where we excite a vocal tract filter with either a voiced signal (which looks like a series of impulses) or a noise source. So, LPC analysis has the wonderful property of finding the coefficients of a filter which will convert either noise or an impulse train into the original frame of speech.

The result isn't quite perfect; as pointed out on page 214 the filter coefficients derived by LPC analysis contain information about the glottal source filter, the lip radiation/preemphasis filter and the vocal tract itself. However since these are much less variable than the vocal tract filter we can factor them out in practice (eg. by preemphasis before LPC analysis).

9.2.4. Formants and Smooth Spectra

Why did we need to know about z-transforms to cover LPC analysis? Well, if this were as far as we were going then we didn't need z, but LPC is really just a way in to some more interesting signal analysis techniques.

The LPC coefficients make up a model of the vocal tract shape that produced the original speech signal. A spectrum generated from these coefficients would show us the properties of the vocal tract shape without the interference of the source spectrum. From our earlier discussion we know that we can take the spectrum of the filter in various ways, for example by passing an impulse through the filter and taking it's DFT, or by substituting for z=exp(iWk) in the z transform of the signal. Either way, the result can be quite useful in signal analysis.

Looking at an LPC smoothed spectrum of voiced speech we can clearly see the formant peaks; they tend to be much more well defined than in a cepstrally smoothed spectrum. As discussed on p223, we can use the z-transform notation to find the locations of these formant peaks for a given set of LPC coefficients, corresponding to the points at which A(z) is zero. This is the key to automatic formant tracking of speech signals -- derive the LPC coefficients, solve the z-transform equation and record the resulting formant positions. Unfortunately since the LPC model isn't a perfect fit to real speech production (it assumes a lossless, all pole model, for example) this method will derive spurious formants; most of the work in a good formant tracking program is working out which of the candidiate formants is the real thing.

LPC coefficients can also be used to derive cepstral coefficients and area functions as described in the remainder of Chapter 6. LPC is a powerful signal modelling technique and is very important in speech recognition and speech analysis.

9.3. Signal Modelling Techniques

The paper by Picone gives a nice framework for thinking about the signal modelling/feature extraction process, I'll briefly talk through his categories here. In reading the paper you will find a lot of technical detail since his audience are electrical engineers who are very comfortable with this. It is not necessary for you to digest all of the equations etc but you should at least get the general idea in most cases.

I will only cover the first three stages outlined by Picone, leaving the topics covered in Statistical Modelling until the next session.

9.3.1. Spectral Shaping

Before any kind of features are derived from the signal it must first be digitised and prepared for analysis. As we discussed in SLP801 the digitisation process requires that all frequencies above the Nyquist frequency be removed before digitisation to avoid aliasing. After digitisation a digital filter is applied to pre-emphasise the signal; this boosts the high frequencies and, as Picone explains, can be said to be compensating for the -6db/octave rolloff of the voiced source or to simulate our more sensitive hearing above 1kHz. Either way this has been shown to be a useful preparatory operation on speech signals.

9.3.2. Spectral Analysis

In this section Picone discusses a number of analysis methods which derive frequency domain parameters from the speech signal. This is the core of signal modelling where we are finding numerical vectors which encode the kind of variability we are most interested in whilst discarding spurious information. In common with all of these procedures is the use of a time windows which is stepped along the input speech signal selecting frames for analysis; each frame is then turned into one feature vector.

9.3.2.1. Fundamental Frequency and Power

While fundamental frequency isn't generally a useful feature in English speech recognition it can be useful in deriving prosodic information and is becoming useful in tone languages such as Mandarin where different pitch contours can cue lexical differences. Picone describes the families of algorithms that are used to derive pitch contours from a speech signal.

The power of a signal is easily measured and is useful as an adjunct to other feature sets.

9.3.2.2. Filter banks

The filter bank approach is equivalent to passing the signal through a set of band-pass filters of varying widths and centre frequencies and using the mean amplitude of the output signal from each filter as the feature vector. A linear filter bank would consist of a set of equal width filters spaced at, say, 100Hz intervals (100, 200, 300 etc); a non-linear filter bank uses different width filters at non-linear spacings. A common approach is to try to model the response of the human ear and have many narrow filter banks below 1000Hz and gradually wider bands above; the widths and placements of these can be defined by the mel, Bark or Erb scales.

While filter banks can be implemented as filters it is more common to use a fourier spectrum and derive the filter bank values from that. Picone calls this the Digital Filter Bank.

9.3.2.3. Cepstral Analysis

We've covered cepstral analysis already and as Picone says this is one of the most important analysis methods in ASR. It is common to use a frequency warped version of the cepstrum, typically the mel scale is used. These are calculated by frequency warping the log spectrum prior to taking its DFT (remember that a cepstrum is the DFT of a log spectrum).

9.3.2.4. LPC Analysis

As we saw earlier, LPC analysis produces a set of coefficients which model the vocal tract filter as a lossless all pole filter. The LPC coefficients can be used by themselves or as the basis for calculating cepstral or filterbank features.

9.3.3. Parameter Transforms

The only point I want to take from this section is the use of differencing to generate additional feature sets for input to the recognition stage. It is commonly believed that in addition to the static per frame features some information about phonetic identity is encoded in the way that the speech signal changes through time. One simple way of encoding this information is to take the difference between successive feature vectors and use that as an additional input to the pattern matching stage. The difference operation approximates taking the gradient or first derivative of the feature vectors with respect to time. In some cases it might be useful to include the second derivative or acceleration which can be approximated by differencing the first derivative vectors. It is now common practice to include at least the first difference as input to a speech recognition system.

<http://web.science.mq.edu.au/~cassidy/comp449/html/ch09.html>

Cepstrum

From Wikipedia, the free encyclopedia

Jump to: navigation, search

A cepstrum (pronounced /ˈkɛpstrəm/) is the result of taking the Fourier transform (FT) of the decibel spectrum as if it were a signal.[citation needed] Its name was derived by reversing the first four letters of "spectrum". There is a complex cepstrum, a real cepstrum, a power cepstrum, and phase cepstrum.

There are many ways to calculate the cepstrum. Some of them need a phase-wrapping algorithm; others do not.

Operations on cepstra are labelled quefrency analysis, quefrency alanysis, liftering, or cepstral analysis.

Contents

[hide]

\* 1 Origin and definition

\* 2 Applications

\* 3 Cepstral concepts

\* 4 Liftering

\* 5 Convolution

\* 6 References

\* 7 Further reading

[edit] Origin and definition

The power cepstrum was defined in a 1963 paper by Bogert et al.[1] It may be defined

\* verbally: the power cepstrum (of a signal) is the square of the Fourier transform of the logarithm of the squared magnitude of the Fourier transform of a signal[2]

\* mathematically: power cepstrum of signal =\left|\mathbf{F}\left\{\mbox{log}(\left|\mathbf{F}\left\{\mbox{the signal}\right\}\right|^2)\right\}\right|^2

\* algorithmically: signal → FT → abs() → square → log → FT → abs() → square → power cepstrum

The complex cepstrum was defined by Oppenheim in his development of homomorphic system theory.[3] It may be defined

\* verbally: the complex cepstrum (of a signal) is the Fourier transform of the logarithm (with unwrapped phase) of the Fourier transform (of a signal). Sometimes called the spectrum of a spectrum.

\* mathematically: complex cepstrum of signal = FT(log(|FT(the signal)|)+j2πm) (where m is the integer required to properly unwrap the angle or imaginary part of the complex log function)

\* algorithmically: signal → FT → abs() → log → phase unwrapping → FT → cepstrum

The real cepstrum uses the logarithm function defined for real values. The real cepstrum is related to the power via the relationship (4 \* real cepstrum)^2 = power cepstrum, and is related to the complex cepstrum as real cepstrum = 0.5\*(complex cepstrum + time reversal of complex cepstrum).

The complex cepstrum uses the complex logarithm function defined for complex values. The phase cepstrum is related to the complex cepstrum as phase spectrum = (complex cepstrum - time reversal of complex cepstrum).^2

The complex cepstrum holds information about magnitude and phase of the initial spectrum, allowing the reconstruction of the signal. The real cepstrum uses only the information of the magnitude of the spectrum.

Many texts state that the process is FT → abs() → log → IFT, i.e., that the cepstrum is the "inverse Fourier transform of the log of the spectrum". [4] This is not the definition given in the original paper, but it is widespread.[citation needed] Note that the Fourier inversion theorem inherently relates the two processes.[dubious – discuss]

[edit] Applications

The cepstrum can be seen as information about rate of change in the different spectrum bands. It was originally invented for characterizing the seismic echoes resulting from earthquakes and bomb explosions. It has also been used to analyze radar signal returns.

The autocepstrum is defined as the cepstrum of the autocorrelation. The autocepstrum is more accurate than the cepstrum in the analysis of data with echoes.

The cepstrum is a representation used in homomorphic signal processing, to convert signals (such as a source and filter) combined by convolution into sums of their cepstra, for linear separation. In particular, the power cepstrum is often used as a feature vector for representing the human voice and musical signals. For these applications, the spectrum is usually first transformed using the mel scale. The result is called the mel-frequency cepstrum or MFC (its coefficients are called mel-frequency cepstral coefficients, or MFCCs). It is used for voice identification, pitch detection and much more. The cepstrum is useful in these applications because the low-frequency periodic excitation from the vocal cords and the formant filtering of the vocal tract, which convolve in the time domain and multiply in the frequency domain, are additive and in different regions in the quefrency domain.

[edit] Cepstral concepts

The independent variable of a cepstral graph is called the quefrency. The quefrency is a measure of time, though not in the sense of a signal in the time domain. For example, if the sampling rate of an audio signal is 44100 Hz and there is a large peak in the cepstrum whose quefrency is 100 samples, the peak indicates the presence of a pitch that is 44100/100 = 441 Hz. This peak occurs in the cepstrum because the harmonics in the spectrum are periodic, and the period corresponds to the pitch.

[edit] Liftering

Playing further on the anagram theme, a filter that operates on a cepstrum might be called a lifter. A low pass lifter is similar to a low pass filter in the frequency domain. It can be implemented by multiplying by a window in the cepstral domain and when converted back to the time domain, resulting in a smoother signal.

[edit] Convolution

A very important property of the cepstral domain is that the convolution of two signals can be expressed as the addition of their cepstra:

x\_1 \star x\_2 \rightarrow x'\_1 + x'\_2

where \star = convolution operator.

[edit] References

1. ^ B. P. Bogert, M. J. R. Healy, and J. W. Tukey: "The Quefrency Alanysis of Time Series for Echoes: Cepstrum, Pseudo Autocovariance, Cross-Cepstrum and Saphe Cracking". Proceedings of the Symposium on Time Series Analysis (M. Rosenblatt, Ed) Chapter 15, 209-243. New York: Wiley, 1963.

2. ^ Norton, Michael; Karczub, Denis (2003). Fundamentals of Noise and Vibration Analysis for Engineers. Cambridge University Press. ISBN 0521499135. http://books.google.com/books?id=jDeRCSqtev4C&pg=PA353&dq=power-cepstrum+log+square+fourier+transform&lr=&as\_brr=3&ei=fZT8SMWmCJ-aMu2tzMsL.

3. ^ A. V. Oppenheim, "Superposition in a class of nonlinear systems" (Ph.D. dissertation), Res. Lab. Electronics, Massachusetts Institute of Technology, Cambridge, MA, 1965.

4. ^ Curtis Roads, "The Computer Music Tutorial", Massachusetts Institute of Technology Press, Cambridge, MA, 1996.

[edit] Further reading

\* D. G. Childers, D. P. Skinner, R. C. Kemerait, "The Cepstrum: A Guide to Processing," Proceedings of the IEEE, Vol. 65, No. 10, October 1977, pp. 1428-1443.

\* "Speech Signal Analysis"