GRESHAM COLLEGE

27 NOVEMBER 2018

SPEECH PROCESSING:
HOW TO WRECK A NICE PEACH

PROFESSOR RICHARD HARVEY FBCS

As I write this, we are in the run-up to Christmas and the "must have" Christmas present is technology that has some sort of voice capability. Amazon Alexa, Apple Siri and Google's, as yet unnamed, voice assistant but probably your car, your oven and your vacuum cleaner, also understand your voice commands. The term speech technology covers everything from creating realistic computer voices, speech synthesis, the removal of noise and corruptions, speech enhancement and the original "grand challenge" problem of how to recognise speech or "how to wreck a nice peach" as the title has it. In this lecture I want to focus on speech recognition, but I hope you will be able to draw parallels with the other problems.

Fortunately, there has been an enormous and sustained interest in *signal processing* which allows us to be confident in handling real, continuously varying, analogue waveforms, digitally. On the face of it, it may seem surprising that we can replace a continuous audio signal such as speech, with a sequence of numbers at, say 8 kHz (8000 samples per second) but this is handled by a nice theorem known at the Nyquist criterion:

If the maximum bandwidth of a signal is B Hz, then the signal can be perfectly reconstructed, without error, provided the sampling rate is higher than 2B Hz.

For speech, one can truncate the bandwidth to 4kHz and still it remains comprehensible, so the minimum sampling frequency for speech is 8 kHz. The theory relating to the replacement of continuous waveforms with a fixed number of discrete[1] levels is not so tidy, but human listening experiments tell us that almost all people report no loss in quality[2] with around 8 bits (256 levels) and the 8-bit, 8kHz signal is still quite commonplace in telephony.

Signal processing theory also provides with a very useful way of working with waveforms that is known as the *Fourier transform.* The Fourier transform comes in several varieties depending on whether the waveform is periodic or not. If the waveform is periodic then you will usually see it referred to as a *Fourier series* but, provided one gets one's mathematical definitions right[3] then they are all the same idea which is that any signal can be represented as a sum of sine and cosine waves. This *frequency domain* representation is completely equivalent to the time domain but mathematically more convenient. For example, in the frequency domain, a signal passing through a linear system such as a filter, is modelled by a multiplication whereas in the time domain it is a complicated integral known as a convolution. This is quite useful for modelling speech as the speech waveform is often thought of a power driving and excitation which is then filtered by the vocal tract. In the frequency domain this is the multiplication of these three quantities – a observation we shall use when deriving features for the pattern recognition sub-system.

The basic idea behind conventional pattern recognition, of which speech processing is a form, is to transform the input signal into a form that makes it easier to determine the *classes*. That intermediate form is usually called a

[1] "discrete" meaning separate is almost always the word one needs when speaking of computers. Unfortunately, it is very commonly confused with the word "discreet". Computers may or may not be discreet depending on their security settings.

[2] Actually the measurement of quality is a rather interesting subject. Of course many people will complain about the quality of something but here I mean whether their performance on a task reduces by a significant margin.

[3] Technically one needs to define a "generalised function" to be able to consider Fourier series and Fourier transforms in a coherent way.

feature. Good features are invariant to all sorts of change that do not matter but vary considerably for the class. For example, if we were building a "yes"/"no" recogniser (or classifier) then we would want a feature that does not change however quickly or slowly I say the word "yes", it does not change as I vary the pitch, or accent, or stress or … And this is the problem of speech recognition, the list of things that a feature needs to be invariant to, is a very long list indeed. Fortunately, there has been much work on what makes a good feature for speech recognition and, so far, there are two features that dominate: linear predictive coefficients (LPCs) and Mel Frequency Cepstral Coefficients (MFCCs)[4].

MFCCs[5] have quite a long chain of processing. In the first step there is some simple compensation for the fact that the low-frequency sounds tend to be louder than the high frequency ones. This is known as *pre-emphasis* and usually takes the form a simple high-pass filter. The signal is then split up into blocks usually known as *frames*. Motivated by some evidence that the ear decomposes sounds into its Fourier components (sine and cosine waves), we then compute a Fourier transform on these blocks using a Fast Fourier Transform (FFT) algorithm. We ignore the phase, because the ear is known to be insensitive to phase, and we then sum adjacent components into filter channels. The width of these channels varies as a *Mel-scale* to mimic the way the cochlea filters sounds. The next step is to apply a logarithm to the numbers. The log converts multiplication into addition which is handy for separating out the effects of, say, the excitation signal (generated by the vocal chords) from the filtering (due to the vocal tract, mouth position and so on). The final step is to apply another Fourier transform to compute the spectrum of the spectrum (or *cepstrum*). This makes sense for the spectrum is highly repetitive, because the voice generates lots of harmonics. The cepstrum is compressive and produces more compact features. An illustration of the process is shown in Figure 1.

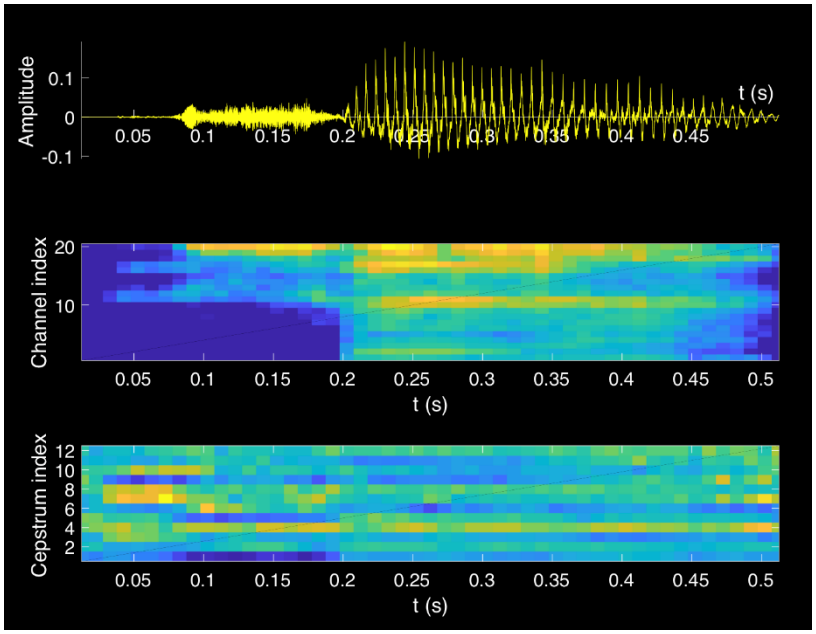

Figure 1: *Waveform of the utterance "Sir" (top); log of filter-bank energies (middle) and Mel-Frequency Cepstral Coefficients or MFCCs (bottom). Note how the "S" at the start of the utterance is quite distinctive in the low-order cepstral coefficients.*

All of this can be done quite quickly as the only computationally intensive algorithm is the Fourier transform and there is a very quick algorithm called the *Fast Fourier Transform* (FFT)[6]. All of this can take place on your mobile phone or car radio in real time. At this point we have converted our time waveform into a set of numbers called features. But, because, people can say words at different rates, there is still the problem that the signal, and hence the features, have a variable duration.

[4] Although an honourable mention should be made of very recent systems that attempt "end-to-end" recognition. Such systems eschew features and attempt to recognise letters or words directly from the waveform. Needless to say, such systems require vast amounts of training data.

[5] See, for example, S.B. Davis and P.Mermelstein, "*Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences*," IEEE Trans. Acoustics, Speech, and Signal Processing, vol. 28, no. 4, pp. 357–366, 1980.

[6] Many people speculate that the FFT is the algorithm that uses the most CPU cycles in any one day. In that sense it is thought to be the world's most popular algorithm.

To handle the variable duration, one needs a classification algorithm that can be programmed to handle time-varying waveforms. In the slides accompanying this lecture we consider three utterances of the word "Sir". I plot the trajectories of the MFCCs and one can see that each utterance transits through several regions but not necessarily at the same time. As an analogy consider the commuter on the train from Norwich to London which travels via Diss, Ipswich, Manningtree, and Stratford. This railway line is not very reliable, and most commuters are familiar with "points at Whitham," "bridge strikes" again usually at "Whitham", "slow running goods trains" and any number of other excuses. Despite these problems, no commuter has ever experienced a different order of stations. If we are at Ipswich, then the next station will always be Manningtree. We need a model that preserves the order but allows some flexibility in the duration spent at each section of track. Such a model is called a Markov Model.

Markov models, or probabilistic finite state machines, are general models that allow one to assign probabilities to the transition between events. For speech we will insist that all the transitions are in one direction across the page, the "left-to-right" model. A *hidden* Markov Model or HMM has one further refinement which is that each state of the model has another probabilistic machine which emits features according to some probability distribution. In our example of the word "Sir", the first state might be associate with the "S" sound of "Sir", the next the "uh" sound and the next the "ur" sound. Each one of the sounds has a characteristic set of feature vectors and the model has to emit them according to some learn probability distribution. So, in that case, we have decided to model the word "Sir" with a three-state left-to-right HMM. We have not stated how we will model the hidden distribution, but a mixture of Gaussian distributions is common (a HMM-GMM) or nowadays a Deep Neural Network (a HMM-DNN). The beauty of the HMM is that, given some labelled, or partially labelled, *training* data it is possible to learn all the parameters of the model. If the model fails to learn then there are well known fall backs, such as reducing the complexity of the model or using prior knowledge about similar sounds to "tie" the various states together.

So far, we have been assuming that we should build a HMM for each word that we wish to classify. Such systems are possible and might be used on very small vocabularies such as though found in cars "Dial 0783987566 now please" or certain specialised applications "Hey Siri!" but, general speech recognition is about generating a transcript from speech and it is infeasible to build models of every single word. The solution is to break the words down into sub-units called *phonemes* which are the atomic sounds of speech[7]. In a simple classifier we would build a model for each phoneme and silence. We would then be able to classify an acoustic waveform as a sequence of phonemes. Given that this list of phonemes is almost always errorful there are a number of well-developed techniques to make this useable.

The first technique is to not only record the most probable sequence of phonemes but also to back-back track to produce the N-best sequences of phonemes. Secondly it would be unusual to work with just mono-phones, many practical recognisers word with combinations of phonemes (sometimes called n-grams) such as biphones (pairs), triphones (triples) and so on. Of course, there are more biphones than phones so more training data is required. n-grams also help a little the increase in variability due to *coarticulation*[8]- the hope is that the n-gram might be able to model some of the coarticulation variability. The third method is to build very comprehensive *language models*. Language models range from simple homophone resolution – is it the "rite stuff" or the "right stuff"? -- to anaphora and cataphora[9] resolution which are close to artificial intelligence since they may require an understanding of the world.

Putting these technologies together gives a modern speech recogniser. Human transcription error is around 4% Word Error Rate (WER) and modern speech recognition systems are operating around 5% to 6% error rate which is very satisfactory. However modern systems are built around absolutely enormous corpuses of training data. Even when trained the recognition cannot be handled on your portable device. One deployment is to perform some feature extraction on your mobile device as features are smaller than the waveform so use less of your mobile

[7] Most people would have that British English contains 45 phonemes including silence but, as is often the case in linguistics, opinions differ.

[8] Try saying "Get some of that soap" at normal conversational speed. How close were you to saying "Guess some ov vat soap"?

[9] Anaphora in a linguistic context are backward references such as "Sam dropped his musket. It clattered to the ground." The "it" here refers to the musket. Cataphora are forward references: "It lay on the ground because Sam would not pick up his musket".

data allowance. The recognition is done on servers placed around the globe and the synthesized voice is sent back to your device. This also means that the manufacturers can deploy continuous updates to the technology without having to inform their users. These systems are complicated so very substantial teams of people are involved. I believe it is fair to say that, at the time of writing, that all the teams of scientists working the public domain are totally dwarfed by the private sector. Effective speech technology has some considerable commercial value, so the race is on to produce highly effective transcription and that race discourages the sharing of information between the competitors. Indeed, it is probably fair to say that no-one now knows how the language models of more than one competitor work. This is a slightly unusual position – patented technology is at least in the public domain because the patents have to be published. Speech technology is only in the public domain to the extent that various commercial organisations wish to be so (and there is considerable variability on that). For the time being therefore, academics are leaving speech technology to retire, to work on something else, or go to work for a major manufacturer.

That does not mean that speech technology research is dead. Far from it, not only is the commercial sector very vibrant and competitive, there are also a number of areas where the gains are promising but not yet commercialised. I have picked one that I am intimately involved with, lip-reading, but I could have picked accent-identification, noisy speech recognition, multi-lingual recognition, recognition in "minority" languages and so-on.

Everyone lip-reads to some extent. Benjamin Franklin having invented the bifocal spectacles realised that his French improved when he could see the lips of the people speaking to him and the well-known, McGurk[10] effect is a very natty illusion that demonstrates unequivocally that all people lip-read. That said, most people lip-read very badly which is because lip-readers cannot see the whole of the vocal instrument[11]. Given all we have said about speech recognition one would hope very much that lip-reading ought to be able to use the same classifier technology (HMMs), same language models and similar architecture. This turns out to be true, at least in our systems developed at UEA. The main difference is that now we need visual features. Obviously, those features, which in our case capture the shape and appearance of the lips, are quite different from MFCCs or LPCs. The extraction of visual features from scenes is a whole new field of study called "Computer Vision" and will be the subject of a later lecture in this series. Our current system performs rather well which is very gratifying but, compared to acoustic recognition it is poor (as with human lip-readers).

The lipreading system is a good illustration of how one tends to solve classification problems – one picks a classification architecture that looks well matched to the problem and then a large amount of effort is devoted to the engineering of appropriate features. For the purposes of science this is quite satisfactory since good features usually allow the construction of new insights into the signal. However, it is block on engineering new systems since every system needs a "domain expert" who can help design good features. Furthermore, as the speech pioneer Fred Jelinek remarked, domain experts quite often get it wrong[12]. More recently there has been a revolution in artificial intelligence called "deep learning" which, provided there is available huge amounts of training data, allows one to dispense with features. From an engineering perspective such systems are highly attractive since they allow one to get the job done merely by collecting huge amounts of data. From a scientific perspective they are more troubling since the classifier is now a black box and it may not provide much scientific insight. And from a governance perspective the result is yet more troubling since the artificial intelligence has little intuition or descriptive power. These new deep learning machines will be the subject of later lecture in this series.

[10] McGurk H., MacDonald J. (1976). "Hearing lips and seeing voices". *Nature*. **264** (5588): 746–8

[11] It is commonplace to be told that lip-readers cannot tell the difference between, for example, "bill", "pill" and "mill" since they all start with bilabial phonemes. In practice I have found that skilled lip-readers can separate these via different cheek roundings. However, the words "loch", "lock" and "log" are very different to distinguish as the difference between them is the final tongue position.

[12] His quote is often repeated as "Every time I fire a linguist, my speech recogniser improves." While it is possible he meant it as casual abuse of linguists, it is more likely that he was observing that elaborate theories are best tested in a classifier if they are to be practically useful