CHAPTER 33

MACHINE-BASED DECODING OF PARALINGUISTIC VOCAL FEATURES

MAXIMILIAN SCHMITT AND BJÖRN SCHULLER

33.1 INTRODUCTION

Machine learning is a major discipline in computer science, dealing with the question of how to imitate biological learning for technical purposes. Within the audio domain, a field of growing interest is *computational paralinguistics* (i.e. the machine analysis of 'how' *and* 'what' a speaker says aiming at the automatic recognition of speaker *states* and *traits*). This has numerous possible applications, such as, for example, automated market research or a more natural human–computer interaction through a virtual agent. Speaker states such as emotion evolve quickly over time. Speaker traits, in contrast, are describing permanent characteristics of a person such as gender, personality, or mother tongue, and alter—if at all—only in the long run. Somewhere in between are longer-term speaker states such as being intoxicated, having flu, or age.

The task of the machine is to find the optimum mapping between a speech signal and, for example, the current emotional state of the speaker. One of the first questions when building an intelligent machine is how the different possible states, the so-called *targets*, are modelled. There are two generally different approaches: speaker states and traits can be expressed either by a finite set of categories or as a value on a continuous scale. One example for categories is the 'Big Six' emotions by Ekman (Ekman, 1999)—*anger, disgust, fear, happiness, sadness,* and *surprise.* For the continuously-valued representation, the most common way is to use the two dimensions *arousal* and *valence* when dealing with emotion. As not all emotions can be described on this two-dimensional plane, further dimensions, such as *dominance* (Burkhardt et al., 2005), are sometimes added. As another example, one typical approach to model speaker personality-related traits is the Big-Five ('OCEAN') model of personality (Wiggins, 1996), consisting of five dimensions—*openness, conscientiousness, extraversion, agreeableness,* and *neuroticism.*

When the targets are defined, a classifier or, in the case of continuous targets, a regressor must usually be learned. In machine learning or, to be precise, in *supervised learning*, this is accomplished by giving the machine a large number of examples of speech signals, together with the corresponding targets. These examples are the so-called *training data*; their number and selection has a crucial influence on the performance of the final system. If the training database is too small, the examples are not (sufficiently) representative, or, if the target labels are not reliable, the classifier (or regressor) will not generalize well enough, meaning that the accuracy of its predictions is insufficient for the application.

For supervised learning, many different approaches exist, and some of them are introduced in Section 33.3. Most of those methods, however, do not work on the raw audio signal but on *features*, which are extracted from the signal in a first step. A typical chain for the analysis of human affect (or other states and traits) is displayed in Figure 33.1. As described, the first step is usually the extraction of acoustic low-level descriptors (LLDs) on *frame level* (i.e. the computation of feature vectors that describe short-term properties of the speech signal) (Schuller, 2012). As the frame-level audio features (i.e. the LLDs) do not contain enough information to recognize the state of the speaker, several successive frames must be observed and summarized. The duration of those *segments* depends on several aspects (e.g. the target labels and the employed machine-learning scheme). Very often, the speech is segmented on *utterance level* (i.e. a spoken sentence or statement of the speaker). In a fully automatic approach, this segmentation can be the result of *voice-activity detection* (Eyben, Weninger, Squartini, & Schuller, 2013).

There are different ways to summarize the information of the frame-level features from one segment. Two common approaches to get such *suprasegmental* features are described in Section 33.2—*functionals* and the *bag-of-words method*.

As an alternative to or complement for the acoustic feature space, *linguistic* features can be considered. For the recognition of emotion and further states and traits in speech, the exploitation of emotional keywords or phrases can be of benefit, especially for the prediction of valence. To obtain a textual representation of the spoken words (i.e. the transcription of what is said in written language), *automatic speech recognition* (ASR) (Schuller et al., 2009; Wöllmer, Weninger, et al., 2013) is usually applied. The resulting text document is then further processed using techniques from *natural language processing* (NLP), such as a *bag-of-words* representation (Schuller, Mousa, & Vasileios, 2015).

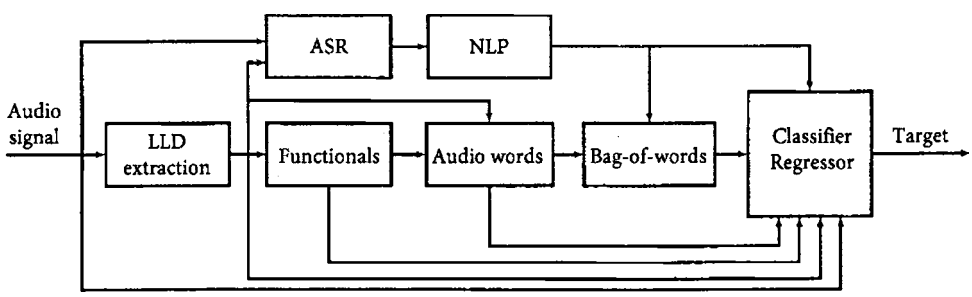

FIGURE 33.1 Exemplary speech-processing chain for the analysis of human affect and further states and traits.

The features are finally fed into a machine-learning scheme. As described in Section 33.3, methods may work on different representations, on segment level, frame level, or even on the raw audio signal.

33.2 Vocal Features

In this section, the most common acoustic low-level descriptors are first introduced; then, two techniques are explained that can be used to obtain a feature-space representation of the whole audio segment—functionals and bags-of-words.

33.2.1 Low-Level Descriptors

LLDs are extracted from the audio waveform in order to obtain the short-term acoustic properties of the human voice signal. The computation is based on frames of approximately 25–60 ms in length. During these short periods of time, the audio signal is assumed to be quasi-stationary (i.e. its properties are not supposed to vary during these time intervals). Successive frames usually have an overlap of between 50 % and 75 % (i.e. the distance between the centres of adjacent frames, the so-called *hopsize*, is between 6 ms and 30 ms).

Most LLDs are based on the *magnitude spectrum* of the signal frames, which is mostly obtained by the *discrete Fourier transform* (DFT) (Schuller, 2013). For the computation of the DFT, a fast algorithm exists with the *fast Fourier transform* (FFT). In general, the Fourier transform is based on the fact that a signal can be represented by the sum of a limited number of weighted sinusoids. Prior to the DFT, the samples within each frame are weighted with a window function, such as the *Hamming*, the *von Hann*, or a *Gaussian* window. Each window has specific properties which make it more or less suitable with regard to the application (Schuller, 2013).

A commonly used group of features are the prosodic features, which are related to:

- *Intensity*: can be approximated by the signal energy of the frame. The more sophisticated approach of (perceived) *loudness* modelling also takes into account psychoacoustic effects, such as *masking* or frequency dependencies (Zwicker & Fastl, 2013).
- *Intonation*: is usually derived from the *fundamental frequency* of the human voice ('Fo'). Fo or (perceived) 'pitch' tracking is a quite challenging task, where many different approaches have been proposed (Schuller, 2013).
- *Speed*: describes the rhythm of the speaker. One simple method to obtain a rhythmic feature is to exploit the rate of changes between *voiced* and *unvoiced* frames and the duration of pauses.

Note that in voiced frames, a fundamental frequency can be detected, whereas this is not the case for unvoiced frames.

Another relevant group are the voice-quality features. Very well-known voice-quality features are the *harmonic-to-noise ratio* and the *microprosodic* features, *jitter* and *shimmer*.

Jitter is a measurement of the slight variations in successive period lengths of the fundamental frequency; shimmer describes the variations of the loudness of the voice between those periods.

Further feature types comprise the spectral and cepstral features. The spectral features are computed directly from the spectrum. Typical examples are the *spectral centroid, spectral flux*, or *spectral roll-off*. Cepstral features, on the other hand, are computed from the so-called *cepstrum*, which can be computed by the inverse Fourier transform of the logarithm of the spectrum. This procedure is motivated by the concept of the *source-filter model*, in which a voice or similar signal is modelled in the frequency domain as the product (multiplication) of a harmonic source signal (usually an 'excitation' pulse train in the time domain) and the frequency response of a filter that shapes the (excited) sound. The filter represents the transmission path of the source signal, which is, in the case of a speech signal, predominantly the vocal tract. Taking the logarithm in the frequency domain converts the product of the source and filter spectra into a sum. This sum is preserved during the inverse Fourier transform due to its linear property. As a result, harmonic content with low amplitudes can be recognized and separated easily from the filter part. The probably most-spread feature in speech analysis is derived from cepstral analysis—the Mel-frequency cepstral coefficients (MFCCs). Compared to the coefficients of the cepstrum, two modifications are applied to the algorithm in order to obtain MFCCs. First, the bands of the power spectrum are summarized according to the *Mel scale* (i.e. a non-linear frequency scale aiming to model the human auditory perception of frequencies). The Mel scale can be approximated by (O'Shaughnessy, 1987)

$$f_{\text{mel}} = 2595 \cdot \log_{10}\left(1 + \frac{f}{700}\right) \tag{1}$$

The filters have a triangular shape in the frequency domain where their centre frequencies are distributed according to the Mel scale. With this mapping step, a dimensionality reduction is performed. After taking the logarithm, an inverse transform is applied in order to decorrelate the Mel-spectrum coefficients. Instead of the inverse discrete Fourier transform (IDFT), the *discrete cosine transform* (DCT) is applied more often, and further alternatives are used. The whole processing chain is shown in Figure 33.2. From the resulting coefficients, those with the indexes 0 to 12 are mostly used in speech analysis. In ASR (speech-to-text), Mel-frequency cepstral coefficients are the most widely spread feature; pitch, which is mainly a speaker-dependent feature in non-tonal languages, is attenuated during the computation process, whereas the filter responses of the vocal tract, which contain the information on the articulated vowels, are preserved.

An overview of further relevant acoustic features can be found in Schuller (2013). From all described LLDs, their differences in evolution over time—the so-called *delta coefficients*—can also be computed, to augment the feature space, since, after all, the speech signal is represented as a contour of acoustic frame-level features over time. In order to reduce the impact of noise on the speech-analysis process, *smoothing* can be applied (e.g. using a *moving average filter*).

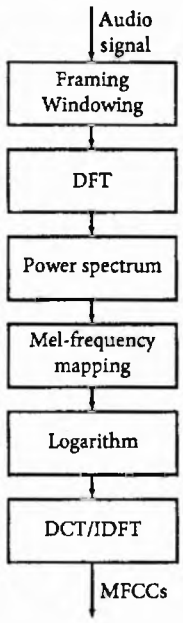

FIGURE 33.2 Computation of Mel-frequency cepstral coefficients.

33.2.2 Functionals

Once the contour of all LLDs over time has been extracted, the typical (yet optional) next step is to derive some statistics of them over a longer segment, to reach a *suprasegmental* feature vector representation. The most popular functionals used are the *mean value* and the *standard deviation* of the LLDs over each segment. Other commonly used functionals comprise *higher-order moments* (e.g. variance, skewness, kurtosis), *extrema* (maximum and minimum), *quartiles, percentiles*, and *regression coefficients*, as well as spectral LLD analysis. Note that hierarchical processing can be chosen, just as in the case of LLDs, where derived LLDs are frequently found. Examples include means of extrema or spectral coefficients of the number of segments, etc.

Naturally, the optimum choice of LLDs and corresponding functionals highly depends on the task at hand, just like the employed machine-learning model does. Nevertheless, some standardized feature sets have broadly established themselves in the community of computational paralinguistics. A recent dominant example is the *ComParE* feature set, which was introduced with the INTERSPEECH 2013 Computational Paralinguistics Challenge (Schuller et al., 2013) and has been used since in follow-up challenges. This set is a large-scale 'brute-forced' feature set, which consists of functionals from sixty-five LLDs and their delta coefficients—in total, 6,373 features per segment. It has been the result of steady refinement, and proven to be suitable for a variety of speech-recognition tasks including personality (Schuller et al., 2012), pathology (Schuller et al., 2013), cognitive and physical load (Schuller et al., 2014), and eating condition (Schuller et al., 2015). One drawback of such large-scale

feature sets is, however, that there is a risk of *over-fitting* (i.e. the model adapts too much to the given training examples and does not generalize in a way that provides the same performance on unknown data).

In contrast to the ComParE feature set, the Geneva Minimalistic Acoustic Parameter Set (GeMAPS) is a small, expert-knowledge-based feature set recommendation made by a larger group of experts. It comprises only eighteen LLDs with selected functionals, resulting in a feature vector of size 62 (Eyben et al., 2016). An extended version of this set (eGeMAPS) has been introduced, comprising eighty-eight features in total. GeMAPS has proven to be suitable for the prediction of short-term speaker states (Ringeval et al., 2014, 2015). The criteria for the selection of the acoustic features were mainly their potential capabilities to model the physiological changes in voice production related to emotion and affect.

Compared to this expert-based, handcrafted selection of features, methods of *automatic feature selection* exist (Eyben et al., 2013). Applied to large-scale acoustic feature sets, such as ComParE, an improvement in performance can possibly be achieved.

Various tools have been introduced to extract acoustic LLDs and corresponding functionals from audio data. One popular example is *openSMILE* (Eyben, Weninger, Groß, & Schuller, 2013), which provides a large number of features relevant for the analysis of human speech, general sound, and also music. It provides the aforementioned standard feature sets and several further feature sets used in challenges in the field.

33.2.3 Audio Words

Instead of using functionals, LLD contours can be further processed in other ways. The LLDs of one frame can be considered as a vector of 'continuous numbers', which may then be subject to *vector quantization*. This means that the whole vector is assigned to a template vector from a *codebook* of audio words, sometimes also referred to as a *dictionary*. It is usually learned from the LLDs computed from training data by applying a clustering algorithm, such as *k-means* (Pancoast & Akbacak, 2012) or *expectation maximization* (EM) (Grzeszick et al., 2015) clustering. An even simpler way to generate a codebook is to select the required number of feature vectors randomly or distributed equidistantly from the training data (Rawat et al., 2013).

The assignment step is based on the distance between the LLD vector and the audio words from the codebook. Typically, the audio word w with the smallest *Euclidean distance* or further-suited distance measure is then assigned to each LLD vector F:

$$w = \arg\min_{w'} \sqrt{\sum_{m=1}^{M} \left(F_m - C_{w',m}\right)^2}, \tag{2}$$

with the w'-th audio word $C_{w'}$, the index of the LLD m, and the number of LLDs M. In Figure 33.3, the process of vector quantization is exemplified by a short segment of audio consisting of fifteen frames of Mel-frequency cepstral coefficients with indexes 1–12 and logarithmized energy as LLDs. A codebook of five audio words has been learned beforehand. For each LLD vector, the index of the audio word which is closest in terms of the Euclidean distance is specified.

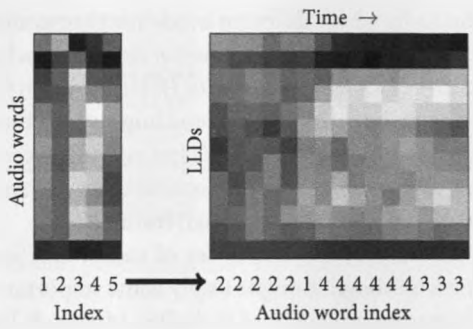

FIGURE 33.3 Example for vector quantization. The codebook consists of five audio words.

The motivation behind vector quantization is to obtain a compact representation of each frame in the audio signal. Given the fact that a whole segment can now be described by a sequence of integers (specifying the audio word indexes in the codebook), this results in a very compressed representation of the speech signal. This can be compared to the most frequent type of linguistic features (i.e. the transcription of the spoken words in a speech segment). Thus, the 'audio words' are suitable for further processing with methods inspired by *natural language processing*. The first of these methods is the bag-of-words method, which is then called *bag-of-audio-words* and will be described in Section 33.2.4. Other potential ideas would include stemming or retagging, which is similar to part-of-speech tagging (i.e. assigning broader word classes such as noun, verb, and adjective), by, for example, a hierarchical clustering of audio words. Obviously, also 'stopping' by selection of audio words is feasible. Additionally, more refined versions of audio words could be reached (e.g. with variable length) by means of, for example, dynamic warping-enabled clustering. Finally, audio words can be built from functional-based vectors rather than LLDs.

33.2.4 Bag-of-Words

In the bag-of-words (BoW) method, the words (or other linguistic cues, such as syllables or characters) within a document are represented as a *histogram* of their frequencies of occurrence. This means that, for each term in a dictionary, it is counted how often it is present in the text at hand. The histogram vector of *term frequencies*, which is of the same size as the dictionary, is then the feature vector to be fed into the classifier (Schuller, Mousa, & Vasileios, 2015).

In the audio domain, once the frame-level features have been assigned to audio words, the same procedure can be performed as for linguistic units. This bag-of-audio-words (BoAW) method has received increasing attention in various audio-recognition tasks, but mostly in *acoustic event detection* (Grzeszick et al., 2015; Lim et al., 2015; Plinge et al., 2014) and *multimedia event detection* (Liu et al., 2010; Pancoast & Akbacak, 2012; Rawat et al., 2013), although also in *music information retrieval* (Riley et al., 2008), *speech-based emotion recognition* (Pokorny et al., 2015; Schmitt, Ringeval, et al., 2016), and healthcare (Schmitt, Janott, et al., 2016). Also, in the field of image and video content analysis, the corresponding *bag-of-visual-words* is now a well-established approach (Fei-Fei & Perona, 2005; Sivic et al., 2005).

Moreover, in recognition tasks where different modalities are available, the bags from different domains can be easily fused in an *early fusion* approach, where the bags are simply concatenated to form a single, larger-feature vector (Joshi et al., 2013). This applies especially for speech-recognition tasks as both the acoustic and linguistic domains are always available.

The open-source toolkit, termed *openXBOW*, has recently been published (Schmitt & Schuller, 2016), providing bag-of-words-based 'crossmodal' representations of arbitrary numerical (acoustic and video) and symbolic (textual) features.

In the basic bag-of-words approach, the order of each word, and hence its context, is not taken into account, even though it might imply some important information. To overcome this problem, *n-grams* can be exploited (Schuller, Mousa, & Vasileios, 2015). Using n-grams, the dictionary does not only consist of single words, but also of sequences of up to n words. The grams that are present in the dictionary are again learned from the training data, so that only relevant sequences are considered. The approach can easily be adopted for bag-of-audio-words, by adding sequences of audio words to the framework (Pancoast & Akbacak, 2013).

The resulting bags (i.e. frequency histograms) can further be processed in order to make them more robust, with the ultimate goal to improve the recognition performance on unseen data samples. The necessity to apply the methods introduced in the following depends on both the data and the used machine-learning scheme, as different methods are able to cope, more or less, with differing histogram-based feature representations.

In order to compress the range of the (words') term frequency (TF) i, the logarithm is very often applied, as expressed in the equation

$$\mathrm{TF}_{\log,i} = \log(\mathrm{TF}_i + c), \tag{3}$$

where c is a constant, mostly chosen as 1 in order to avoid negative values. Another very common modification is to choose the *term frequency inverse document frequency* (TF-IDF), in which the TF is multiplied with the *inverse document frequency* (IDF) measure. The IDF for each term i is computed as

$$\mathrm{IDF}_i = \log\left(\frac{N}{\mathrm{DF}_i}\right), \tag{4}$$

where N is the number of instances in the training data, and DF_i (document frequency) is the number of instances therein where the term (e.g. word) i is present. Instance here refers to one sample in the available data (i.e. one speech segment in the case of voice analysis). The motivation behind IDF is to increase the weight of rare words in the histogram, which are assumed to be more meaningful than words occurring very often.

In a further step, the bag-of-word histograms can be *normalized*. This might be beneficial if the given segments vary in length, as the ranges of the TFs are then larger for segments of longer duration. Normalization can be considered by, for example, dividing each TF by the *absolute* or the *Euclidean length* of the histogram, or even by the number of frames in the segment.

Finally, it must be stated that the described extensions of bag-of-words are not always beneficial and that their usefulness depends on many parameters (Schmitt, Janott, et al.,

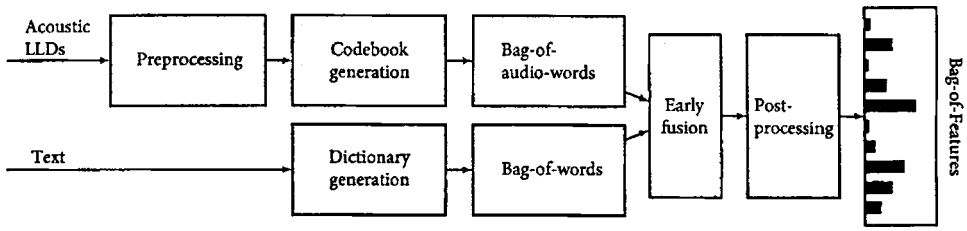

FIGURE 33.4 Bag-of-features processing chain for multimodal input.

2016). Thus, the optimum configuration is ideally evaluated each time a term frequency histogram-based representation is used. The whole process of multimodal bag-of-words' generation with early fusion of acoustic and linguistic (and potentially further) features is displayed in Figure 33.4. In general, one can also speak of *bag-of-features* implicating linguistic, acoustic, and/or visual words. As an initial step, a *preprocessing* of the LLDs (in case of linguistics, these are usually strings) might be necessary if the ranges of their values differ. If we consider, for instance, Mel-frequency cepstral coefficients and pitch, the variance of the pitch contour is usually larger, so that pitch would have a higher impact on the Euclidean distance in the word-assignment step. Thus, either a *standardization, unity-based normalization*, or further suited representation of the LLDs is mostly beneficial. The difference between both techniques is that in unity-based normalization (also known as *feature scaling*), all values are mapped onto a scale of, for example, −1 to +1 (by normalization to the minimum/maximum occurring value), whereas in standardization, after subtraction of the mean value, each LLD is divided by its standard deviation. Therefore, standardization is less prone to outliers in the LLDs; however, the resulting range of standardized values is not fixed. Both techniques can be conducted in an *online approach*, in which the required parameters (e.g. in the case of standardization, mean and standard deviation for each LLD) are computed from all LLDs in the training data and then stored in the system to be used with all sample instances that need to be classified. *Postprocessing* in Figure 33.4 refers to the aforementioned term frequency weighting and histogram normalization methods.

33.3 DECODING

Once a suitable representation of all acoustic and/or linguistic information in a speech segment has been found, the representation needs to be *decoded* in order to reach a prediction for the respective target (e.g. the emotion). The most critical aspect is typically the availability and the proper selection of the training data. As shown in the following sections, each machine-learning model needs a large number of examples in each category (in case of a classification task) or in different ranges (in case of a regression task). If the training data has not been chosen carefully, the resulting model will usually not generalize well enough. This could be the case if, for instance, training data has been recorded only in silent environments but the system is supposed to work also in noisy or more generally speaking 'mismatched' environments. Another problem can arise if the

subjects have not been well selected. Speakers should be balanced in both age and gender, and ideally represent all ages and types of the general population, if expected to work for these. However, the low amount of available training data is usually the most problematic bottleneck.

Decoding can be done on different levels: (1) on *frame level*; (2) on *segment level*; and (3) even on the plain signal, usually referred to as *end-to-end learning*. This section gives a brief review of machine-learning methods that can be employed on the aforementioned levels.

33.3.1 Frame-Level Decoding

In frame-level decoding, the sequence of LLDs—or the sequence of audio words—is directly fed into a machine-learning algorithm. In this section, two types of models that can be used with sequential data are introduced—*hidden Markov models* and (recurrent) *neural networks*.

33.3.1.1 Hidden Markov Models

Hidden Markov models (HMMs) (Rabiner, 1989) are one of the most used learning algorithms in many speech-processing tasks. They are based on *Markov chains*, a statistical process given by a number of states and transition probabilities between all states. The main characteristic of Markov chains is that the subsequent state depends only on a finite number of previous states (in a first-order Markov chain, only the current state). In HMMs, the current state cannot be observed directly, but only estimated through its *emission* (also called *observation*) based on a second probabilistic process in addition to the probabilistic state transitions. HMMs are especially suitable for ASR, where the spoken words are modelled on different levels: on the first level, there is often an HMM for each phoneme where the emissions are the measured LLDs (e.g. Mel-frequency cepstral coefficients) (Gales & Young, 2008); on the second level, an HMM can, for example, model a linguistic word with the phonemes as their emissions.

Formally, an HMM is given by a set of states (S) and a set of possible emissions (X). The transition probabilities between all states are described by a matrix (A), and the probability, that a state emits a certain element of X is defined by another matrix (B). For each HMM, the two Markov properties apply: (1) the probability of the following state depends only on the current state, and (2) the probability of the current emission depends only on the current state.

Different models of HMMs have been introduced so far, differing in the number of states and in the positivity of their state transition probabilities. In Figure 33.5, a three-state *linear left–right model* is shown. A linear left–right model ('stay or move') is commonly used in ASR as linguistic information in speech is encoded in an ordered sequence of phonemes. By having self-transitions (loops) in the model, different durations of, for example, spoken sounds, or simply different speech rates, can be captured. Thus, an HMM is able to *warp* the signal in time. The 'start' and 'end' state in the model shown in Figure 33.5 have been introduced to simplify the further calculations.

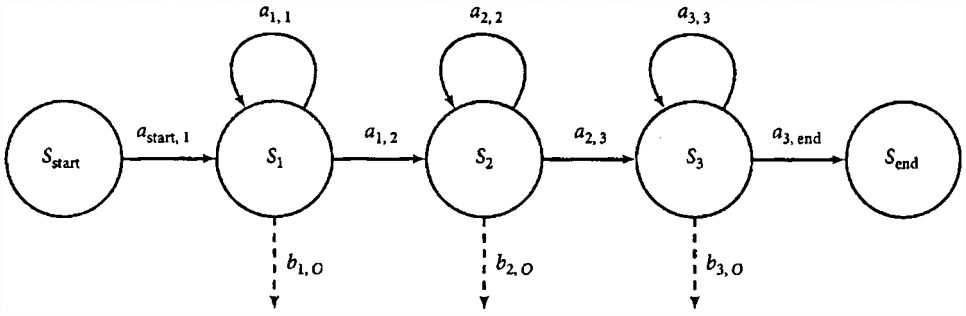

FIGURE 33.5 Linear left–right hidden Markov model with three hidden states.

In general, each HMM represents the conditional probability $p(\mathbf{x}|c)$, which denotes the probability of observing the sequence $\mathbf{x}$ under the hypothesis that it belongs to class c. Given a sequence of T observations in an audio segment, this probability can be computed as

$$p(\mathbf{x}|c) = \sum_{\text{seq}} a_{s_{\text{start}},s_1} \prod_{t=1}^{T} b_{st,xt} a_{st,st+1}, \tag{5}$$

where the summation is executed over all possible sequences of states. Alternatively, exploiting the *Viterbi algorithm*, a speed-up is achieved while still obtaining good results (Jelinek, 1997). In this case, the conditional probability turns into the following equation, taking only the most probable state sequence into account:

$$\hat{p}(\mathbf{x}|c) = \left\{ a_{s_{\text{start}},s_1} \prod_{t=1}^{T} b_{st,xt} a_{st,st+1} \right\}. \tag{6}$$

So far, the assumption has been made that there is a finite number of observations (e.g. audio words) and, therefore, the probabilities in matrix B are discrete. In audio-recognition tasks, however, the emission probabilities in B are usually modelled continuously by *Gaussian mixture models* (GMMs) (Jelinek, 1997) as, for example, the LLDs are continuously valued. The emission probability density functions (PDFs) for each state s are determined by

$$b_s(x_t) = \sum_{m=1}^{M} c_{s,m} \mathcal{N}(x_t; \mu_{s,m}, \Sigma_{s,m}), \tag{7}$$

for a GMM consisting of M mixtures, mixture weights $c_{s,m}$, and the PDF of the multivariate Gaussian distribution $\mathcal{N}(\cdot; \mu, \Sigma)$ with mean vector μ and covariance matrix Σ.

In order to train the model, its parameters (i.e. the elements of matrices A and B) need to be learned based on a given training data set of audio sequences and corresponding labels. In the case of ASR, the labels are the transcriptions of the speech segments (one HMM is then usually trained per phoneme or word); in the case of emotion recognition, the emotion

labels (one HMM is then usually trained per emotion class). For parameter estimation in HMMs, the *Baum-Welch* algorithm is usually employed, which is an instance of the previously mentioned expectation maximization algorithm. For further details, we refer to the corresponding literature (Baum et al., 1970; Schuller, 2013).

In the recognition phase, all HMMs are evaluated applying, for example, the Viterbi algorithm. Consequently, the HMM with the highest posterior probability is supposed to be the correct model for the respective target (e.g. a word, an emotion, further speaker state or trait). Classification can be realized by the *maximum a posteriori* (MAP) estimate,

$$\hat{c} = \arg\max_{c} \; p(c|\mathbf{x}), \quad c = 1, \ldots, C, \tag{8}$$

where $\hat{c}$ is the most likely out of C classes, and $\mathbf{x}$ is the sequence of observations. Exploiting Bayes' theorem,

$$p(c|\mathbf{x}) = \frac{p(\mathbf{x}|c) \cdot p(c)}{p(\mathbf{x})} \tag{9}$$

and as the prior probability $p(\mathbf{x})$ is the same for all classes, Equation 8 can be transformed into

$$\hat{c} = \arg\max_{c} p(\mathbf{x}|c) \cdot p(c), \quad c = 1, \ldots, C \tag{10}$$

If all classes are presumed to have the same probability, the MAP is identical to the *maximum likelihood* (ML) estimate.

In the case of ASR, the term in Equation 10 splits up into an HMM, denoted by $p(x|c)$ and defining the *acoustic model* (AM), and a prior probability $p(c)$, defining the *language model* (LM). In fact, the acoustic model usually consists of several HMMs. Also, the language model can be refined by using the aforementioned n-grams with linguistic words. Then, a whole sequence of words W is modelled as

$$p(W) = \prod_{i=1}^{n} p(w_i \,|\, w_{i-(n-1)}, \ldots, w_{i-1}). \tag{11}$$

This is especially advantageous for homophones (e.g. 'piece' and 'peace') or words with similar pronunciations (e.g. 'affect' and 'effect'). As an example, the language model would favour 'I read a book' over 'I red a book'.

Though HMMs are most popular in ASR, they can be used for arbitrary classification problems (e.g. emotion recognition in speech) (Schuller et al., 2003). As mentioned, each emotional state is then represented by an HMM, where the one with maximum likelihood is chosen based on the given speech signal. The language model can then model the probability of emotion transitions, such as whether it is more likely to change from angry to neutral to happy, or to change from angry to happy to angry, etc. Several open-source toolkits have been published providing training and decoding of HMMs, such as *Sphinx* (Walker et al., 2004), *HTK* (Young et al., 2006), and *Kaldi* (Povey et al., 2011).

33.3.1.2 *Artificial Neural Networks*

In recent years, however, research on computational paralinguistics (and ASR) has mostly abandoned HMMs due to their susceptibility to background noise and other factors such as the fact that discriminative learning is harder to realize. The prevailing research is now refocusing on another fundamental approach in machine learning—*artificial neural networks* (ANNs). The basic idea is to create a model which is inspired by the information processing in the human brain, or more generally, in the human nervous system.

First research in this domain was published in the 1940s by McCulloch and Pitts, who were proposing a model of *neurons* as simple cells providing the processing of binary signals (McCulloch & Pitts, 1943). A common neuron model is displayed in Figure 33.6. The elements of an M-dimensional input vector x are weighted individually and summed up, and a bias w_0 is added. Then, a *non-linear activation function* is applied to the result, which can be realized by, for example, a *step function* or a *sigmoid function* $f(u) = \frac{1}{1+\exp(-\alpha u)}$ with the steepness parameter α. The optimum activation function depends on the task at hand. The whole (multilayer) ANN is then a network consisting of several neurons, where the output of each neuron ($\hat{y}$) is an input to another neuron:

$$\hat{y} = f\left(\sum_{m=0}^{M} w_m x_m\right); \quad x_0 = 1. \tag{12}$$

As displayed in Figure 33.7, the cells are arranged in different layers, where all outputs of one layer are propagated to all inputs to the next layer in the case of a *feed forward* neural network. In a *recurrent* neural network (RNN), the outputs are also propagated to the same or even previous layers. The step of one propagation is then synchronized with a clock where, in each clock cycle, a new input feature vector is given to the input layer of the recurrent neural network. Altogether, the ANN involves a sequence of non-linear transformations of the input data, providing the final predictions in the output layer. The input layer usually simply

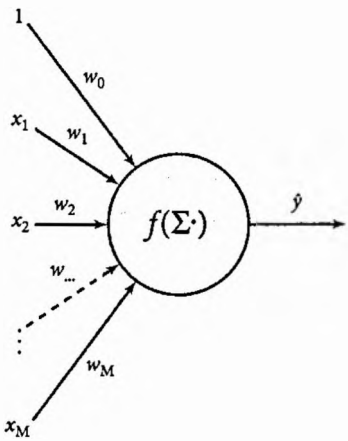

FIGURE 33.6 Neuron—the basic component of artificial neural networks.

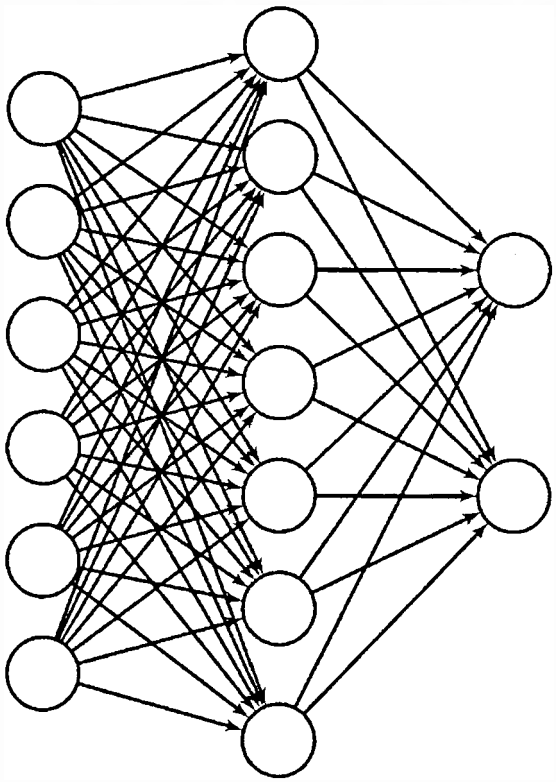

FIGURE 33.7 A feedforward neural network with one hidden layer. Each circle represents a neuron.

propagates the input values to the second layer and involves no weighting, summation, or non-linearity. The layers, which are neither input nor output, are called *hidden layers*. The output layer usually has a dimensionality according to the number of possible targets. Thus, if it handles a binary decision task (two classes), there are two neurons in the output layer, each one providing a likelihood for the corresponding class. Suited functions such as the *softmax* function can be used to normalize the output to the range of 0 to 1, and thus reflect posterior probabilities per class. In the case of a (single-task) regression problem, there is usually one neuron per predicted variable.

The process of learning the weights of all neurons is time-consuming and computationally intensive. The common practice is the *back propagation* method with *stochastic gradient descent* (Rumelhart et al., 1986).

Whilst research on ANN was on the fringes from the 1970s onwards, it has experienced a powerful revival since the first decade of the twenty-first century. This has been driven mainly by two factors: (1) the increasing performance of computers, especially parallel architectures such as graphics processing units (GPUs), that are a fundamental requirement to the time-consuming training of ANNs; and (2) the proposal of new architectures, particularly *long short-term memory recurrent neural networks* (LSTM-RNNs) (Hochreiter & Schmidhuber, 1997). In this context, new methods of (pre-)training 'deep'

networks (i.e. up to hundreds of hidden layers) such as layer-wise unsupervised learning and 'drop-out' learning (randomly omitting neurons during iterations), or suited activation functions such as 'rectified linear units', are to be mentioned.

Nowadays, ANNs are mostly *deep neural networks*, which have at least two hidden layers. *Deep learning* has evolved as a new research field of machine learning in computer science and is applied to a large variety of tasks, such as handwriting recognition, ASR, medical image analysis, and music synthesis, to mention only a few. Popular modern architectures of deep learning (LeCun et al., 2015) comprise *restricted Boltzmann machines* (RBMs) (Hinton & Salakhutdinov, 2006), *convolutional neural networks* (CNN) (Simard et al., 2003), and the aforementioned LSTM-RNN. CNNs are mostly used in the context of spatially distributed data (e.g. images). In this approach, local regions in the signal are iteratively processed in a convolutional layer of neurons with globally equal weights. By a successive *pooling* layer, the outputs of the convolutional layers are downsampled. The final layers of a CNN are then fully connected, so that the compressed information from all regions can be combined.

In contrast, recurrent neural networks are preferably used with sequential data, such as speech signals, as they can model both the long-term and short-term evolution of the signal, based on the sequence of LLDs. The core of LSTM-RNNs are the *long short-term memory cells*. They are able to store activations from an earlier time instant in the signal for an arbitrary time. For this reason, they overcome the *vanishing gradient problem*, which was one of the major drawbacks of recurrent neural networks in audio recognition. The fundamental principle of an LSTM cell is displayed in Figure 33.8. A common neuron (shown at the top of the cell) is complemented by a memory and three *gate neurons*, controlling the flow of the activation by scaling it with their outputs at different stages of the cell. The *input gate* scales the input to the memory. The incoming activation is then stored in the so-called *error carousel*, where it can remain for an arbitrary number of time steps. This is accomplished by a loop which is controlled by the *forget gate*, scaling the recurring activation of the cell memory. The actual output of the LSTM cell is controlled by the third gate, the *output gate*. Besides the activations from outside the cell, the stored activation itself is also passed to all

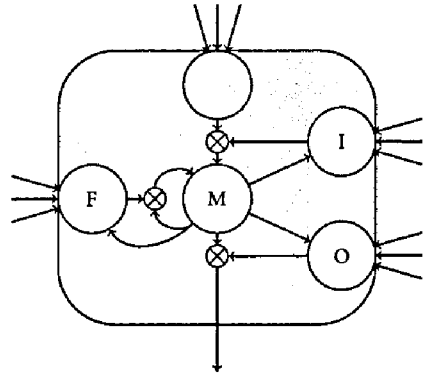

FIGURE 33.8 Long short-term memory cell. I: input gate, F: forget gate, O: output gate, M: memory.

gates as an input. The behaviour of the gate neurons has to be learned in addition to the original neuron. However, the same set of three gates can control several neurons to reduce the number of free parameters to be learned from data.

LSTM-RNNs have successfully been used for a great variety of audio-recognition tasks, such as voice-activity detection (Eyben, Weninger, Squartini, & Schuller, 2013), social-signal detection (Brückner & Schuller, 2014), acoustic-novelty detection (Marchi et al., 2015), emotion recognition (Wöllmer, Kaiser, et al., 2013), and feature enhancement (Zhang et al., 2016) in paralinguistics. It seems noteworthy that dynamic-sequence learning is possible with recurrent neural networks, similar to the warping abilities in HMMs.

In LSTM-RNNs, all weights in each cell of the network must be learned, based on the given training data. Several open-source tools that are able to train LSTM-RNNs and other neural architectures have been published, including *CURRENNT* (Weninger et al., 2015) and *TensorFlow* (Abadi et al., 2016). This simplifies the application tremendously as efficient implementation is a challenge.

33.3.2 Segment-Level Decoding

In segment-level decoding, the feature vector representing the segment (e.g. the vector of functionals and/or the bag-of-words derived per spoken unit, such as words or phrases) is fed into the machine-learning model to predict either a discrete class or a continuous label. Formally spoken, in the case of classification, the class $c^{\wedge}$ with the highest posterior probability given the feature vector x is chosen according to a MAP estimation (see Equation 8).

Many different learning methods have emerged (e.g. *Bayes classifier, k-nearest neighbour, decision trees, random forests)* and each one has its pros and cons. However, probably the most commonly used method in the field of speaker-state and trait analysis, including emotion recognition from speech, is the *support vector machine* (SVM) for classification tasks and its counterpart *support vector regression* (SVR) for regression tasks (Cortes & Vapnik, 1995). In the following, only SVM will be further discussed, but most of the theory can easily be adapted to derive SVR (Schuller, 2013).

The principle of SVMs is to find an optimum separating hyperplane between two classes in the feature space. The feature space is constructed from the feature vectors x in the training data. The goal of an SVM is now to find the hyperplane in the feature space which separates the classes with the widest 'channel' between the instances of both classes. Likewise as in ANNs, training is discriminative, as in general, an ideal separation is not feasible due to outliers in the training data; *slack variables* have been introduced, through which a certain number of outliers are allowed. The amount of error can be controlled by the so-called *complexity* parameter of an SVM. The basic principle is exemplified in Figure 33.9. The margins on both sides of the hyperplane are represented as dashed lines and the area in between should ideally be free of instances. The instances on the margins on both sides are named *support vectors* and define the equation of the hyperplane in the feature space. With the binary labels $y_i \in \{-1, +1\}$, the feature vectors x_i where $a_i > 0$ for the support vectors, a scalar bias b, and the size of the training set L, the prediction for a sample instance x_t is based on the following decision function:

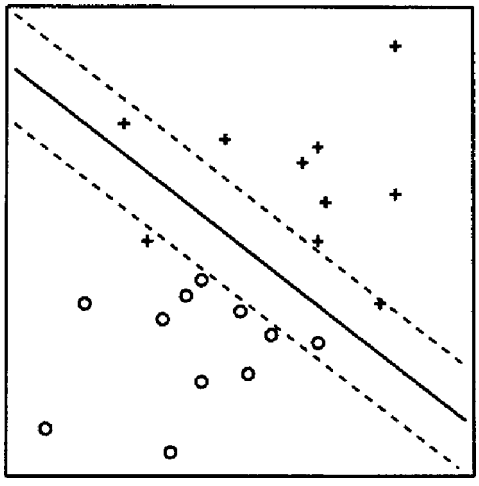

FIGURE 33.9 Support vector machine classifier in a binary classification task based on a two-dimensional feature space. Outliers are present in the training data.

$$\hat{y}(x_t) = \operatorname{sign}\left(\sum_{l=1}^{L} y_l a_l x_l \cdot x_t + b\right), \tag{13}$$

where

$$\operatorname{sign}(u) = \begin{cases} +1 & \text{if } u > 0, \\ 0 & \text{if } u = 0, \\ -1 & \text{if } u < 0. \end{cases} \tag{14}$$

As shown in the example, it is usually better to tolerate some training samples within the margin in order to get a wider channel. The result is consequently called a *soft margin*. If no outliers were allowed between the margins on both sides, the result would be a *hard margin* and a narrower channel. Such a model normally does not generalize well and is subject to overfitting. That is why the complexity parameter must always be tuned in order to be sure to obtain a good model. This process is referred to as *hyperparameter optimization*, in which several SVMs are trained with different complexities, typically in a certain range of values ≤ 1. Each SVM is then evaluated on independent validation data and the SVM with the best performance (the lowest error) is finally selected as the optimum model.

So far, we were making the assumption that, disregarding the outliers, the feature space was somehow linearly separable into two parts. This might not always be the case though, especially if the space is low-dimensional. In order to tackle this problem, the feature vectors can be transformed into a space of higher dimension, using a non-linear transformation. In an SVM, (suited) transformations do not need to be computed explicitly as in the final decision function; only the dot product between the training instances and a sample instance is present (see Equation 13). Such suited functions describing the dot product between the

transformed feature vectors are named *kernels* and the way of using them is called *kernel trick*. A multitude of kernel functions have so far been introduced, and the most established ones are the *linear*, the *polynomial*, and the *Gaussian* (RBF) kernel. Selection of the most suitable one is normally subject to trial and error (Hsu et al., 2003). Empirically, we can say that the linear kernel normally works well with large-feature vectors (e.g. with the aforementioned ComParE feature set). However, combinations of multiple kernels can also be used.

Finally, one might need to build a classifier for more than two classes. As SVMs can, in their usually preferred 'basic' version, only do binary decisions, multiple SVMs are trained and their outputs are combined. The two most popular schemes applied are referred to as *one-vs-one* and *one-vs-all* (Hsu & Lin, 2002), where each class is trained against each other or all remaining ones.

So far, many implementations of SVM have been published and are available to the research community. One popular example is the open-source library LIBSVM (Chang & Lin, 2011) which supports several kernels, optimization methods, multi-class classification, and regression.

Deep neural networks (e.g. LSTM-RNN) can also be employed for classification or regression on the segment level. In the special case of recurrent neural networks, the information from adjacent segments is usually also taken into account for the predictions at the present moment in time, similar to a language model.

33.3.3 End-to-End Learning

Besides exploiting the extracted acoustic features from the speech signal, a machine-learning model can also be learned directly on the raw audio signal, in a procedure called *end-to-end learning*. It has successfully been applied to speech-based emotion recognition (Trigeorgis et al., 2016) and is also of great interest in ASR (Graves & Jaitly, 2014), even though most approaches that have been proposed, so far, still use a handcrafted time-frequency transform in the first step. It has, however, already been shown that it is feasible to automatically learn meaningful representations from the speech signal by methods of deep learning (Jaitly & Hinton, 2011). This process is usually referred to as *feature learning*.

An architecture for end-to-end learning for emotion recognition, similar to that proposed by Trigeorgis et al. (2016), is shown in Figure 33.10. It learns a suitable representation of the

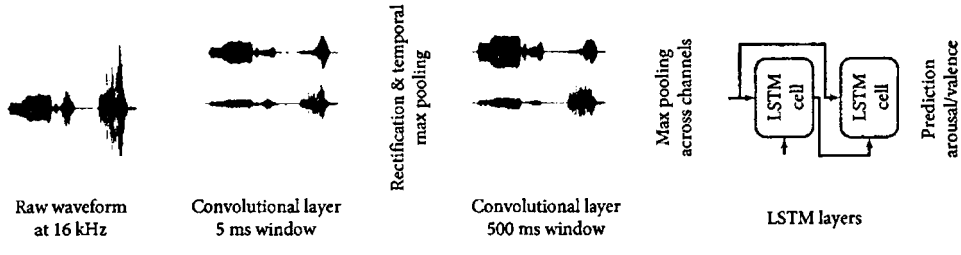

FIGURE 33.10 End-to-end learning architecture for speech-based emotion and further states and traits recognition.

signal incorporating all information relevant to the emotion. Hence, the first layers in the network replace the conventional handcrafted acoustic features in a certain way. The first step is a temporal convolutional layer consisting of forty parallel filters with a window size of 5 ms, extracting fine-scale spectral information. Then, a half-wave rectifier is applied, which means that all parts of the band signals that are below zero are set to zero. This step is motivated by the rectifying property of the cochlear transduction in the human ear. The rectified band signals are then pooled by a factor of two, resulting in a downsampled representation. Subsequently, there is another convolutional layer of forty parallel filters, this time with a window size of 500 ms, in order to capture characteristics of the speech signal on a longer scale. The output signals are then again maximum pooled, this time across the channels, which results in a huge dimensionality reduction. Finally, the remaining compressed representation is fed into a conventional LSTM-RNN, which provides the emotion predictions in terms of the two dimensions, arousal and valence. The number of LSTM cells found to be sufficient for this task was 128.

Interestingly, the outputs of some cells within a neural network trained for emotion recognition were observed as closely correlated with 'typical' prosodic features such as loudness and fundamental frequency (Trigeorgis et al., 2016). This accounts for the finding that affective states are encoded in prosody (Gunes et al., 2011). It has also been found that the filters learned by deep neural networks have a bandpass behaviour which is very similar to the filtering process in the human inner ear (Tüske et al., 2014).

33.4 Conclusion

Finally, it must be stated that there is an almost infinite number of methods with regard to both features describing the content of a speech signal and models to decode the audio information. The best-suited approach always depends on many parameters, such as the final application, the recording environment, and the amount of available training data, to mention only a few. This chapter is, by far, not exhaustive and accounts only for a reasonable selection of ways to proceed in machine recognition of human voice. More information about specific topics is found in the corresponding references.

In the future, one can assume a trend towards cross-lingually, cross-culturally, and environmentally robust speaker-state and trait analysers. These will likely target a whole range of speaker attributes simultaneously rather than assessing, for example, emotion in isolation, to exploit interdependencies of the states and traits such as between emotion and personality. Their high interdependence is obvious, as all different states and traits impact on the same one vocal-production mechanism and the cognitive process choosing the words to be said. Likely, these engines will be trained on huge amounts of data which will be increasingly labelled in weakly supervised ways such as by active, semi-supervised, and reinforcement learning. Likely, deep learning or similar approaches of 'holistic' end-to-end learning will play an increasing role, ultimately leading to the advent of such technology in a broad range of everyday applications, such as, for example, emotionally sensitive virtual agents, health monitoring, and personalized recommendation engines.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... Zheng, X. (2016). TensorFlow: large-scale machine learning on heterogeneous distributed systems. *arXiv preprint*, arXiv:1603.04467

Baum, L. E., Petrie, T., Soules, G., & Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics*, 41(1), 164–171.

Brückner, R. & Schuller, B. (2014). *Social Signal Classification Using Deep BLSTM Recurrent Neural Networks*. Proceedings of ICASSP, IEEE, Florence, Italy, pp. 4856–4860.

Burkhardt, F., Paeschke, A., Rolfes, M., Sendlmeier, W., & Weiss, B. (2005). *A Database of German Emotional Speech*. Proceedings of INTERSPEECH, ISCA, Lisbon, Portugal, pp. 1517–1520.

Chang, C.-C. & Lin, C.-J. (2011). LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3), 27.

Cortes, C. & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297.

Ekman, P. (1999). Basic emotions. In: T. Dalgleish & M. J. Power (eds) *Handbook of Cognition and Emotion* (pp. 301–320). New York: John Wiley & Sons.

Eyben, F., Scherer, K., Schuller, B., Sundberg, J., André, E., Busso, C., ... Truong, K. (2016). The Geneva minimalistic acoustic parameter set (GeMAPS) for voice Research and Affective Computing. *IEEE Transactions on Affective Computing*, 7(2), 190–202.

Eyben, F., Weninger, F., Groß, F., & Schuller, B. (2013). *Recent Developments in openSMILE, the Munich Open-Source Multimedia Feature Extractor*. Proceedings of MM 2013, ACM, Barcelona, Spain, pp. 835–838.

Eyben, F., Weninger, F., & Schuller, B. (2013). *Affect Recognition in Real-life Acoustic Conditions—A New Perspective on Feature Selection*. Proceedings of INTERSPEECH, ISCA, Lyon, France, pp. 2044–2048.

Eyben, F., Weninger, F., Squartini, S., & Schuller, B. (2013). *Real-life Voice Activity Detection with LSTM Recurrent Neural Networks and an Application to Hollywood Movies*. Proceedings of ICASSP, IEEE, Vancouver, Canada, pp. 483–487.

Fei-Fei, L. & Perona, P. (2005). *A Bayesian Hierarchical Model for Learning Natural Scene Categories*. Proceedings of CVPR, Vol. 2, IEEE, San Diego, CA, USA, pp. 524–531.

Gales, M. & Young, S. (2008). The application of hidden Markov models in speech recognition. *Foundations and Trends in Signal Processing*, 1(3), 195–304.

Graves, A. & Jaitly, N. (2014). *Towards End-to-End Speech Recognition with Recurrent Neural Networks*. Proceedings of ICML, Vol. 14, Beijing, China, pp. 1764–1772.

Grzeszick, R., Plinge, A., & Fink, G. A. (2015). *Temporal Acoustic Words for Online Acoustic Event Detection*. Proceedings of GCPR, Aachen, Germany, pp. 142–153.

Gunes, H., Nicolaou, M. A., & Pantic, M. (2011). Continuous analysis of affect from voice and face. In: A. A. Salah & T. Gevers (eds) *Computer Analysis of Human Behavior* (pp. 255–291). Springer.

Hinton, G. E. & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, 313(5786), 504–507.

Hochreiter, S. & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780.

Hsu, C.-W., Chang, C.-C., & Lin, C.-J. (2003). *A Practical Guide to Support Vector Classification*. Technical report, Department of Computer Science, National Taiwan University, Taipei, Taiwan.

Hsu, C.-W. & Lin, C.-J. (2002). A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2), 415–425.

Jaitly, N. & Hinton, G. (2011). *Learning a Better Representation of Speech Soundwaves Using Restricted Boltzmann Machines*. Proceedings of ICASSP, IEEE, Prague, Czech Republic, pp. 5884–5887.

Jelinek, F. (1997). *Statistical Methods for Speech Recognition*. MIT Press.

Joshi, J., Goecke, R., Alghowinem, S., Dhall, A., Wagner, M., Epps, J., Parker, G., & Breakspear, M. (2013). Multimodal assistive technologies for depression diagnosis and monitoring. *Journal on MultiModal User Interfaces*, 7(3), 217–228.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444.

Lim, H., Kim, M. J., & Kim, H. (2015). *Robust Sound Event Classification Using LBP-HOG Based Bag-of-Audio-Words Feature Representation*. Proceedings of INTERSPEECH, ISCA, Dresden, Germany, pp. 3325–3329.

Liu, Y., Zhao, W.-L., Ngo, C.-W., Xu, C.-S., & Lu, H.-Q. (2010). *Coherent Bag-of Audio Words Model for Efficient Large-Scale Video Copy Detection*. Proceedings of CIVR, ACM, Xi'an, China, pp. 89–96.

Marchi, E., Vesperini, F., Weninger, F., Eyben, F., Squartini, S., & Schuller, B. (2015). *Non-Linear Prediction with LSTM Recurrent Neural Networks for Acoustic Novelty Detection*. Proceedings of IJCNN, IEEE, Killarney, Ireland.

McCulloch, W. S. & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4), 115–133.

O'Shaughnessy, D. (1987). *Speech Communication: Human and Machine*. Addison-Wesley Publishing Co.

Pancoast, S. & Akbacak, M. (2012). *Bag-of-Audio-Words Approach for Multimedia Event Classification*. Proceedings of INTERSPEECH, ISCA, Portland, USA, pp. 2105–2108.

Pancoast, S. & Akbacak, M. (2013). *N-Gram Extension for Bag-of-Audio-Words*. Proceedings of ICASSP, IEEE, Vancouver, Canada, pp. 778–782.

Plinge, A., Grzeszick, R., & Fink, G. A. (2014). *A Bag-of- Features Approach to Acoustic Event Detection*. Proceedings of ICASSP, IEEE, Florence, Italy, pp. 3732–3736.

Pokorny, F., Graf, F., Pernkopf, F., & Schuller, B. (2015). *Detection of Negative Emotions in Speech Signals Using Bags-of-Audio-Words*. Proceedings of ACII, AAAC, IEEE, Xi'an, China, pp. 879–884.

Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., ... Vesely, K. (2011). *The Kaldi Speech Recognition Toolkit*. Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding, Big Island, HI, USA.

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of IEEE*, 77(2), 257–286.

Rawat, S., Schulam, P. F., Burger, S., Ding, D., Wang, Y., & Metze, F. (2013). *Robust Audio-Codebooks for Large-Scale Event Detection in Consumer Videos*. Proceedings of INTERSPEECH, ISCA, Lyon, France, pp. 2929–2933.

Riley, M., Heinen, E., & Ghosh, J. (2008). *A Text Retrieval Approach to Content-Based Audio Hashing*. Proceedings of ISMIR, Philadelphia, PA, USA, pp. 295–300.

Ringeval, F., Amiriparian, S., Eyben, F., Scherer, K., & Schuller, B. (2014). *Emotion Recognition in the Wild: Incorporating Voice and Lip Activity in Multimodal Decision Level Fusion*. Proceedings of ICMI, ACM, Istanbul, Turkey, pp. 473–480.

Ringeval, F., Marchi, E., Méhu, M., Scherer, K., & Schuller, B. (2015). *Face Reading from Speech—Predicting Facial Action Units from Audio Cues*. Proceedings of INTERSPEECH, ISCA, Dresden, Germany, pp. 1977–1981.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323, 533–536.

Schmitt, M., Janott, C., Pandit, V., Qian, K., Heiser, C., Hemmert, W., & Schuller, B. (2016). *A Bag-of-Audio-Words Approach for Snore Sounds' Excitation Localisation*. Proceedings of ITG Speech Communication, ITG/VDE, IEEE, Paderborn, Germany, pp. 230–234.

Schmitt, M., Ringeval, F., & Schuller, B. (2016). *At the Border of Acoustics and Linguistics: Bag-of-Audio-Words for the Recognition of Emotions in Speech*. Proceedings of INTERSPEECH, ISCA, San Francsico, CA, USA, pp. 495–499.

Schmitt, M. & Schuller, B. (2016). OpenXBOW—introducing the Passau open-source crossmodal bag-of-words toolkit. *arXiv preprint*, arXiv:1605.06778

Schuller, B. (2012). The computational paralinguistics challenge. *IEEE Signal Processing Magazine*, 29(4), 97–101.

Schuller, B. (2013). *Intelligent Audio Analysis, Signals and Communication Technology*. Springer.

Schuller, B., Batliner, A., Steidl, S., & Seppi, D. (2009). *Emotion Recognition from Speech: Putting ASR in the Loop*. Proceedings of ICASSP, IEEE, Taipei, Taiwan, pp. 4585–4588.

Schuller, B., Mousa, A. E.-D., & Vasileios, V. (2015). Sentiment analysis and opinion mining: on optimal parameters and performances. *WIREs Data Mining and Knowledge Discovery*, 5, 255–263.

Schuller, B., Rigoll, G., & Lang, M. (2003). *Hidden Markov Model-Based Speech Emotion Recognition*. Proceedings of ICASSP, Vol. II, IEEE, Hong Kong, China, pp. 1–4.

Schuller, B., Steidl, S., Batliner, A., Epps, J., Eyben, F., Ringeval, F., Marchi, E., & Zhang, Y. (2014). *The INTERSPEECH 2014 Computational Paralinguistics Challenge: Cognitive and Physical Load*. Proceedings of INTERSPEECH, ISCA, Singapore, pp. 427–431.

Schuller, B., Steidl, S., Batliner, A., Hantke, S., Hönig, F., Orozco-Arroyave, J. R., ... Weninger, F. (2015). *The INTERSPEECH 2015 Computational Paralinguistics Challenge: Degree of Nativeness, Parkinson's and Eating Condition*. Proceedings of INTERSPEECH, ISCA, Dresden, Germany, pp. 478–482.

Schuller, B., Steidl, S., Batliner, A., Nöth, E., Vinciarelli, A., Burkhardt, F., ... Weiss, B. (2012). *The INTERSPEECH 2012 Speaker Trait Challenge*. Proceedings of INTERSPEECH, ISCA, Portland, OR, USA, pp. 254–257.

Schuller, B., Steidl, S., Batliner, A., Vinciarelli, A., Scherer, K., Ringeval, F., ... Kim, S. (2013). *The INTERSPEECH 2013 Computational Paralinguistics Challenge: Social Signals, Conflict, Emotion, Autism*. Proceedings of INTERSPEECH, ISCA, Lyon, France, pp. 148–152.

Simard, P. Y., Steinkraus, D., & Platt, J. C. (2003). *Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis*. Proceedings of ICDAR, IEEE, Edinburgh, UK, pp. 958–962.

Sivic, J., Russell, B. C., Efros, A. A., Zisserman, A., & Freeman, W. T. (2005). *Discovering Object Categories in Image Collections*. Technical report, MIT CSAIL.

Trigeorgis, G., Ringeval, F., Brueckner, R., Marchi, E., Nicolaou, M. A., Schuller, B., & Zafeiriou, S. (2016). *Adieu Features? End-to-End Speech Emotion Recognition using a Deep Convolutional Recurrent Network*. Proceedings of ICASSP, IEEE, Shanghai, China, pp. 5200–5204.

Tüske, Z., Golik, P., Schlüter, R., & Ney, H. (2014). *Acoustic Modeling with Deep Neural Networks using Raw Time Signal for LVCSR*. Proceedings of INTERSPEECH, ISCA, Singapore, Singapore, pp. 890–894.

Walker, W., Lamere, P., Kwok, P., Raj, B., Singh, R., Gouvea, E., Wolf, P., & Woelfel, J. (2004). *Sphinx-4: A Flexible Open Source Framework for Speech Recognition*. Technical report, Sun Microsystems, Inc.

Weninger, F., Bergmann, J., & Schuller, B. (2015). Introducing CURRENNT: the Munich open-source CUDA RecurREnt neural network toolkit. *Journal of Machine Learning Research*, 16, 547–551.

Wiggins, J. S. (ed.) (1996). *The Five-Factor Model of Personality: Theoretical Perspectives.* New York: Guilford.

Wöllmer, M., Kaiser, M., Eyben, F., Schuller, B., & Rigoll, G. (2013). LSTM-modeling of continuous emotions in an audiovisual affect recognition framework. *Image and Vision Computing*, 31(2), 153–163.

Wöllmer, M., Weninger, F., Geiger, J., Schuller, B., & Rigoll, G. (2013). Noise robust ASR in reverberated multisource environments applying convolutive NMF and long short-term memory. *Computer Speech and Language*, 27(3), 780–797.

Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Liu, X., ... Woodland, P. (2006). *The HTK Book (v3.4).* Cambridge University.

Zhang, Z., Ringeval, F., Han, J., Deng, J., Marchi, E., & Schuller, B. (2016). *Facing Realism in Spontaneous Emotion Recognition from Speech: Feature Enhancement by Autoencoder with LSTM Neural Networks.* Proceedings of INTERSPEECH, ISCA, San Francsico, CA, USA, pp. 3593–3597.

Zwicker, E. & Fastl, H. (2013). *Psychoacoustics: Facts and Models, Vol. 22.* Springer Science & Business Media.