Chapter 5

A Deep Dive Into Deep Learning Techniques for Solving Spoken Language Identification Problems

Himanish Shekhar Das and Pinki Roy
Department of Computer Science and Engineering, National Institute of Technology Silchar, Silchar, India

5.1 Introduction

Automatic language identification (LID) is a challenging research area in the domain of speech signal processing. It is the identification of a language from a random spoken utterance. The process of spoken LID is a front end for many applications such as multilingual conversational systems, spoken language translation, multilingual speech recognition, spoken document retrieval, human machine interaction through speech, etc. automatic speech recognition (ASR) systems are generally language dependent. Once a spoken LID system correctly identifies a language, an ASR system can take over to process the speech.

In his paper, Muthusamy [1], explained one critical use of automatic LID. It was about distress calls made to 911 operators. Since there are many U.S. citizens from different ethnic groups, it was found that in times of crisis, people used their own native language when calling for help. It is therefore necessary to understand what language the person is using before forwarding the call to an interpreter who can understand the message. AT&T introduced Language Line Services, which employed interpreters who could handle 140 languages. However, Muthusamy found that when a person who spoke Tamil, (a language spoken in India and Sri Lanka) called Language Line Services, there was a 3-min delay before the language was identified, and a Tamil interpreter was brought online. In such crucial circumstances, LID can play a crucial role.

Deep learning is currently used in many lines of. This is an era dominated by artificial intelligence (AI), specifically deep learning techniques such as in bioinformatics [2], forecasting the future of the mobile number portability [3], and advancement in multimedia content [4]. In Spoken LID too many deep learning techniques have been used with significant success. Prominent among the deep learning techniques are feed-forward deep neural network (FF-DNN) commonly referred to as multilayer perceptron (MLP), convolutional neural network (CNN), long short term memory-recurrent neural network (LSTM-RNN), etc. These contemporary types of deep neural network techniques have overshadowed conventional methods such as Gaussian mixture model (GMM) and hidden Markov model (HMM). These techniques showed significant improvement in recognition performance over various parameters. It is the ability of deep neural networks techniques to perform complex correlation among speech signal features, which enhances their performance over traditional approaches.

5.2 Spoken Language Identification

While a human identifying a spoken language is commonplace, mimicking the process for machines is not. Let us first try to understand how a human identifies a language.

Humans are trained in one or more languages soon as they start learning. It is easy to identify the language(s) they learned if a person knows the phonemes, syllables, words, and sentence structure. The person get the cues about the language from multiple prompts, viz. phonetic, phonotactic, prosodic, and lexical. It is also important to note that even if a language that is similar in phonetic features to a language people know, they can make an educated guess about the language. In most cases, such a language belongs to the same family of languages. For example, if a person knows Bengali, languages like Assamese, Odiya, and Hindi (from Indo-Aryan language family), it will not very difficult for the person to identify even with very little hearing (small training set). However, it will not be that easy for the person to understand languages like Tamil, Telegu, and Kannada (from Dravidian language family), because these languages belong to another family of languages. Identifying other languages belonging to other families will require a higher degree of training. Then again, if somehow one can correlate some phones of the unknown language to a known language(s), the person can make out syllables and map them according to the known language. This is also relevant in case of written languages. How often do we see, (mostly in social media) that one can use English to construct Bengali words? Even if the person on the other end has very basic knowledge of English, one can interpret those words to Bengali and have an understanding. This is possible because phones of Bengali can be expressed in terms of English phones. In addition, we humans are excellent at comprehending multilingual speeches. Often we use a word from other languages in our speech, but it does not create a problem for a listener to identify the primary language of the speaker as well as tag those foreign words and separately identify them, too (only if one has previous knowledge about those foreign words). It can be concluded that we humans use acoustic models as well as language models apart from other sources of information to identify a language.

It is not that simple for a machine though. It is widely known in the speech processing community that making a machine proficient in all languages will evidently make the machine an excellent language identifier. The flip side is that it will incur a lot of expenses in terms of time, money, and labor. This is why different strategies have been formulated to identify a language by a machine.

All the spoken LID systems can be broadly classified into two groups: explicit and implicit LID systems. The systems, which require segmented and labeled speech corpus, are known as explicit LID systems. The systems that do not require phone-recognizers (thus no need for segmented and labeled speech data) are known as implicit LID systems. In other words, implicit LID systems require only the raw speech data along with the true identity of the language spoken.

5.3 Cues for Spoken Language Identification

As discussed earlier, there must be some unique feature(s), which will separate one language from another. Researchers like Muthusamy et al. [1]; Zissman and Berkling [5]; Li et al. [6]; Lee [7]; in their respective papers, almost unanimously categorized the cues into the following groups:

- *Acoustic Phonetics:* Whatever we say, at its smallest possible structure can be termed as a phone. According to Coxhead [8], a phone can be defined as a "unit sound" of a language. It is a "unit" sound because the whole of the phone must be substituted to make a different word. Phones can be again (for convenience) divided into monophones, diphones, and triphones among others. A monophone refers to a single phone, a diphone is an adjacent pair of phones and a triphone is simply a group of three phones. Acoustic phonetics is driven by the belief that a set of phones are different in different languages, and even if languages have identical phones, the frequency of occurrence will vary. The phones that comprise the word "celebrate" might be [s eh 1 ax bcl b r ey q]. Acoustic phonetic features are extracted by several methods. Some popular methods are linear predictive coding (LPC), linear prediction cepstral coefficients (LPCC), perceptual linear prediction coefficients (PLP), mel-frequency cepstral coefficient (MFCC), filter bank (FB), and shifted delta coefficients (SDC).

LPC is based on the source-filter model of speech signal. One can think of LPC as a coding method; a way of encoding information in a speech signal into a smaller space for transmission over a restricted channel. LPC encodes a signal by finding a set of weights on earlier signal values that can predict the next signal value. The LPC calculates a power spectrum of the signal. For a medium or a low bit rate coder, LPC is most widely used. While we pass the speech signal from speech analysis filter to remove the redundancy in signal, a residual error is generated as an output. It can be quantized by a smaller number of bits compared to the original signal. A parametric model is computed based on the least mean square error theory; this technique is known as linear prediction (LP). By this method, the speech signal is approximated as a linear combination of its previous samples. In this technique, the acquired LPC coefficients describe the formants. The frequencies at which the resonant peak occur are called formant frequencies. Thus with this method, the locations of the formants in a speech signal are estimated by computing the linear predictive coefficients over a sliding window and finding the peaks in the spectrum of the resulting LP filter.

Another important extension of the LPC is the LPCC, a short-term cepstral representation. The cepstral analysis is used to decorrelate dependences among variables of the acoustic features.

PLP models human speech based on the concept of psychophysics of hearing. PLP discards irrelevant information of speech and thus improves the speech recognition rate. PLP is identical to LPC except that its spectral characteristics have been transformed to match characteristics of the human auditory system. PLP approximates three main perceptual aspects namely: the critical-band resolution curves, the equal-loudness curve, and the intensity-loudness, power-law relation, which are known as cubic-root.

MFCC is the most used spectral feature extraction method. MFCCs are based on frequency domain using the mel scale, which is based on the human ear scale. MFCC is a representation of the real cepstral of a windowed short-time signal derived from the fast Fourier transform (FFT) of that signal. The difference from the real cepstral is that a nonlinear frequency is used. The MFCC coefficients are represented in a number of frames centered at equally spaced times, during a constant sampling period.

The human ear perceives sound in a nonlinear fashion. Filterbank analysis provides a straightforward route to obtaining the desired nonlinear frequency resolutions. An array of band-pass filters are used to separate the input signal into multiple components.

The SDCs, which are computed from the deltas of MFCCs, can capture a wide range of dynamics of an utterance. Therefore they showed significant improvement in performance of spoken LID.

- *Phonotactics:* Phones are a unit sound, but they do not exist as individual entities in speech. Every utterance is composed of multiple phones together in various combinations. Phonotactics describes the rules of combination of phones. It is believed that for every language, the rules of the combination of phones are different. For instance, the phone cluster /sr/ is very common in Tamil, but not in English.
  1. *Prosodics:* Prosodics are speech features such as stress, tone, or word juncture that accompany or are added over constants and vowels; these features are not limited to single sounds but often extend over syllables, words, or phrases. It is believed that prosody is useful for distinguishing

between broad language classes. Mandarin and Vietnamese have very different intonation characteristics than English.

2. *Vocabulary and Syntax:* Vocabulary is the collection of words, specific to a language. There will be words in every language that are unique and can directly point to the language. Even if there are common words between languages, their syntactic way to form a sentence will not be the same.

5.4 Stages in Spoken Language Identification

The process of spoken LID can be divided into two major blocks, feature extraction, and classification. However, there can be many other operations that can be done in speech signal preprocessing like voice activity detection and noise removal, feature extraction and classification are the most important stages of the activity.

In feature extraction (FE) stage cues which will help in discrimination between languages are extracted as features (Fig. 5.1). These features are then used in the classification stage to obtain a score, which helps in identifying the spoken language. Currently, there is a trend of end-to-end spoken LID systems. In end-to-end systems, there is no distinction between FE and classification stages; both of them are merged into the systems (Fig. 5.2).

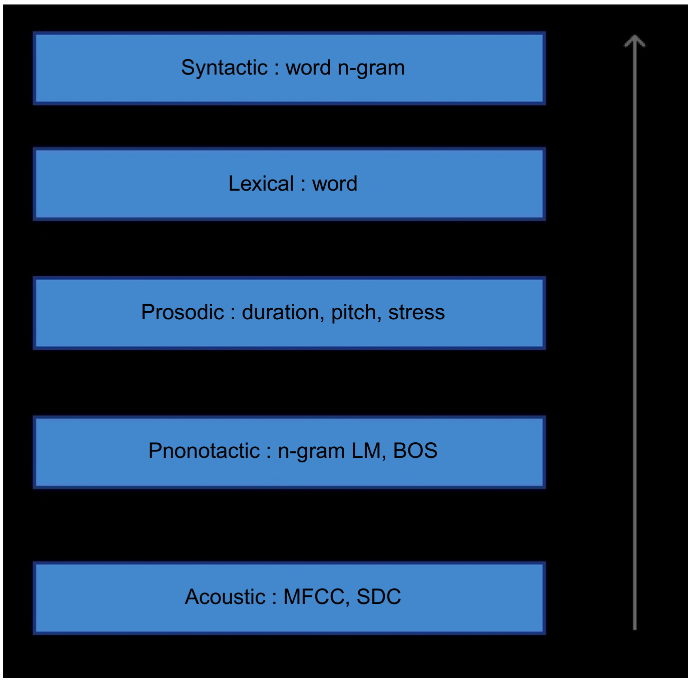
Syntactic : word n-gram
Lexical : word
Prosodic : duration, pitch, stress
Pnonotactic : n-gram LM, BOS
Acoustic : MFCC, SDC

FIG. 5.1 Features hierarchy.

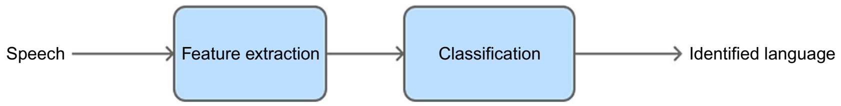

FIG. 5.2 LID process.

5.5 Deep Learning

In machine learning (ML), knowledge is acquired from data representations and not from specific formulae and algorithms. When learning is done, it is not in a single layer of computation, rather it spans across two or more layers. They type of learning is known as deep learning (DL). In the present scenario, deep learning and deep neural network are almost synonymous. If people feel interested in many facets of deep learning, they can read the Stanford University UFLDL tutorial [9], *Deep Learning*, by Goodfellow et al. [10], or *Neural Networks and Deep Learning* by Nielsen [11].

Since deep learning is inspired by biological neural network and human is still the best intelligence when it comes to identify a person in a picture or melody in a song or whether it is to an extent safe to jump over a ditch. It is for this reason that deep learning is thought to be suitable over traditional machine learning algorithms. Since most of traditional machine learning concepts use a domain expert approach, how the machine will learn is determined by the logic handcrafted by humans. The algorithms are extremely important in decision making. Whereas in deep learning, very little formulation is done and learning is instead, dependent on data. Availability of more data generally results in better accuracy. Deep learning does a remarkable job when the problem is complex and data availability is high.

5.6 Artificial and Deep Neural Network

An artificial neural network (ANN) is an aspect of AI that is focused on emulating the learning approach that humans use to gain certain types of knowledge. Like biological neurons, which are present in the brain, ANN also contains a number of artificial neurons, and uses them to identify and store information. ANN consists of input and output layers, as well as (in most cases) one or more hidden layer(s) consisting of units that transform the input into something that the output layer can use (Fig. 5.3).

The human brain contains approximately 86 billion neurons [12]. These neurons are connected to each other like a mesh. Stimuli from the external environment or inputs from sensory organs are accepted by entities known as dendrites. These inputs create electric impulses, which quickly travel through the neural network. A neuron can then send messages to other neuron(s) to handle the issue or does not send it forward. This is known as activation of the neuron. A neuron is connected to thousands of other neurons by axons.

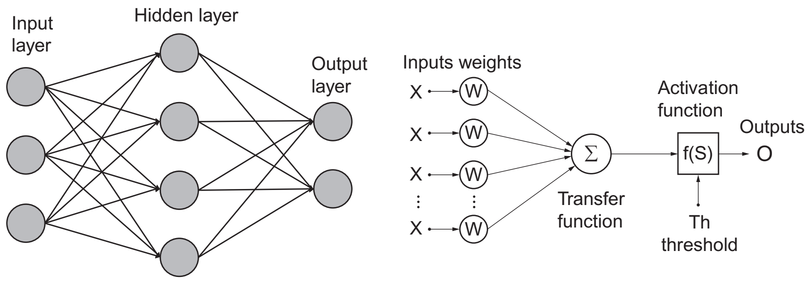

FIG. 5.3 A typical ANN and a typical artificial neuron.

ANNs are composed of multiple artificial neuron nodes, which imitate biological neurons of the human brain. Unlike biological neurons, there is only one type of link that connects one neuron to others. The neurons take input data and simple operations are performed on those data. The result of these operations is passed to other neurons. Whether the result will be passed, is determined by the activation function. The activation function plays an important role for both feature extraction and classification.

In the biological neural network, the size of dendrites varies in accordance with the importance of inputs. In ANN, this is achieved by using weights and biases. Earlier experimental work checked the performance of ANN with SVM and found that ANN performs better than SVM [13].

ANN can be of many types. Prominent among them are feed-forward neural network, radial basis function neural network, Kohonen self-organizing neural network, recurrent neural network, convolutional neural network and modular neural network.

A deep neural network (DNN) can be considered as stacked neural networks, i.e., networks composed of several layers.

- *FF-DNN:* FF-DNN, also known as multilayer perceptrons (MLP), are as the name suggests DNNs where there is more than one hidden layer and the network moves in only forward direction (no loopback). These neural networks are good for both classification and prediction. For spoken LID, we use the classification approach. When the FF-DNN is used as a classifier, the input and output nodes will match the input features and output classes.

The most important concepts in a FF-DNN are weights, biases, nonlinear activation and backpropagation. Let us try to understand them. Below is a sample FF-DNN (Fig. 5.4).

The input layer has four elements I-1, I-2, I-3, and I-4. They define an input, I={I-1, I-2, I-3, I-4}. What we need is to find one or more patterns from the entities of the input, so that those patterns can be used to classify one output from the other. In order to do that, we devise a number of hidden units with activation function. Nonlinear activation functions like Sigmoid, tanh, ReLU are used to form a pattern of active hidden units.

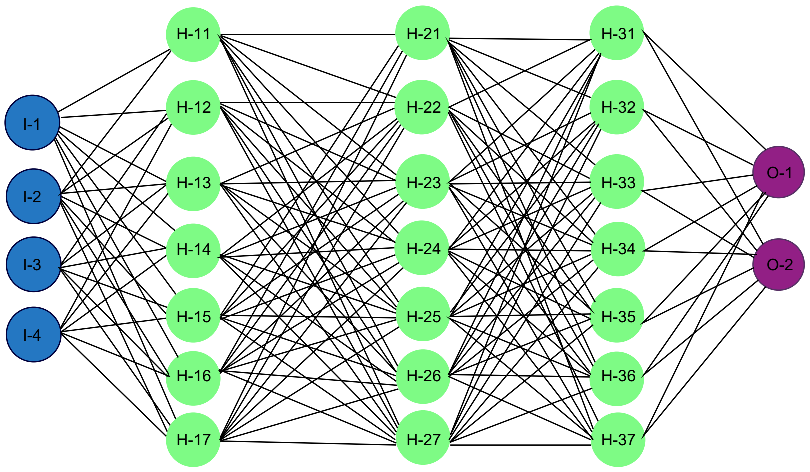

FIG. 5.4 Feed-forward deep neural network with three hidden layers.

Let us assume that, H-13, H-14, H-17, H-22, H-23, H-26, H-32, H-33, H-36, and H-37 are needed for active output O-1. To achieve this, we need to adjust the weights and biases such that, these hidden nodes are activated by the activation function. Initially, weights and biases are randomly initialized. Then, we train the network with tens of thousands of inputs. We use backpropagation of error to adjust weights and biases, so that suitable values of them will activate the hidden neurons.

The set of weights and biases, which will be used to uniquely identify a particular output, are called feature set or kernel.

It is believed that when the pattern used for discrimination is so complex that traditional statistical and numerical approaches do not work, FF-DNN is the key to solution.

- *Radial Basis Function Neural Network:* The radial basis function neural networks (RBF-NN) are inspired by biological neural systems, in which neurons are organized hierarchically in various pathways for signal processing. They are tuned to respond selectively to different features/characteristics of the stimuli within their respective fields. RBF-NN is structurally the same as MLP. RBF-NN has an input layer, a hidden layer, and an output layer. RBF-NN is strictly limited with exactly one hidden layer. The hidden units provide a set of functions that constitute an arbitrary basis for the input patterns. Hidden units are known as radial centers and the hidden layers are feature vectors. There are different radial functions like, Gaussian radial function, thin plate spline, quadratic, inverse quadratic, etc. The most popular radial function is Gaussian activation function. RBF is used in pattern classification and regression.

- *Kohonen Self-Organizing Neural Network:* Self-organizing neural networks can cluster groups of similar patterns into a single set. They assume a topological structure among their cluster units effectively mapping weights to input data. Kohonen self-organizing map neural network is one of the basic types of self-organizing maps. The ability to self organize provides new possibilities-adaptation to formerly unknown input data. It seems to be the most natural way of learning, which is used in our brains, where no patterns are defined. These patterns take shape during the learning process. Self-organizing networks can be either supervised or unsupervised. Unsupervised learning is a means of modifying the weights of a neural network without specifying the desired output for any input patterns. The advantage is that it allows the network to find its own solution, making it more efficient with pattern association. The disadvantage is that other programs or users have to figure out how to interpret the output. The functioning of a self-organizing neural network is divided into three stages: construction, learning, and identification.
- *Recurrent Neural Network:* FF-DNN only moves in the forward direction. No feedback loop is used. But there are many elements like speech that works in sequence. You cannot make out the meaning of a sentence without all the words, and even if you have them, you need them to be sequenced to find the correct meaning. Recurrent networks, unlike FF-DNN, take as their input not just the current input, but also state perceived previously in time. The idea behind RNNs is to make use of sequential information. In theory RNNs can make use of information in arbitrarily long sequences, but in practice they are limited to looking back only a few steps (Fig. 5.5).

The left side shows a representative form of a folded RNN, which means the network is available for n iterations in time, while the right side shows RNN states in times $t-1$, t, and $t+1$. The right side is also known as unrolled RNN and represents a full network. Let us understand how RNN works. x_t is the input at time step t. s_t is the hidden state at time step t. s_t contains a previous

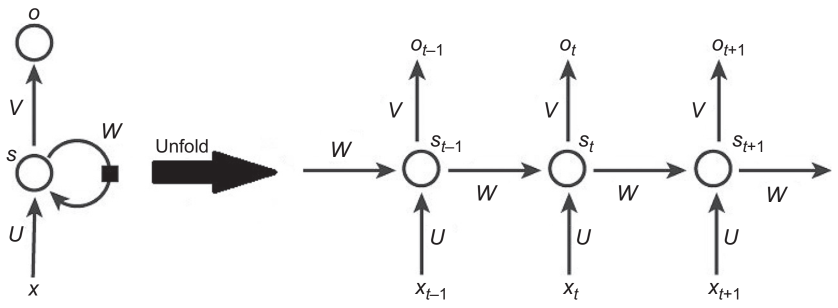

FIG. 5.5 Recurrent neural network.

hidden state, and is calculated taking consideration of the previous hidden state along with the input at the current step. $s_t = f(Ux_t + Ws_{t-1})$. The function f is usually a nonlinearity such as tanh or ReLU. $s-1$, which is required to calculate the first hidden state, and is typically initialized to all zeroes. o_t is the output at step t. unlike a traditional deep neural network, which uses different parameters at each layer, an RNN shares the same parameters, U, V, and W; across all steps. This reflects that we are performing the same task at each step, just with different inputs. This greatly reduces the total number of parameters needed to be learned. Training an RNN is similar to training a traditional neural network. A backpropagation algorithm is used with a little twist. This is called backpropagation through time (BPTT). One of the most popular implementations of RNN is long short-term memory (LSTM) which solves the vanishing/exploding gradient problem.

- *CNN:* CNN works a little differently from FF-DNN and RNN where a neuron is supposed to do an activation. In case of a CNN, a neuron is a result of multiple convolution operations before getting activated for feature extraction or classification. A CNN has multiple stages of operation, viz., convolution, pooling, nonlinearity. These stages can be repeated several times to make a deep CNN. In the convolution stage, convolution operation is performed, which can be simply stated as the element-wise multiplication between the filter elements and the input. The input is represented as a two-dimensional matrix for that purpose. A filter or kernel is also a two-dimensional matrix, but of a smaller size. There may be a weight filter and a bias filter. The elements of the weight filter are multiplied with input nodes and the elements of the bias filter are added. To reduce the dimension of the output, a pooling operation is done. The most widely used pooling operation is max-pooling. Then a nonlinearity like ReLU or Sigmoid or tanh is used as an activation function. Finally, when the layers are done, a fully connected (FC) layer is used for classification (Fig. 5.6).
- *Modular Neural Network:* A modular neural network is a combination of independent neural networks. Each independent neural network serves as a module to the system as a whole. The task at hand is divided into subtasks, which are optimized by individual modules. The reason behind is

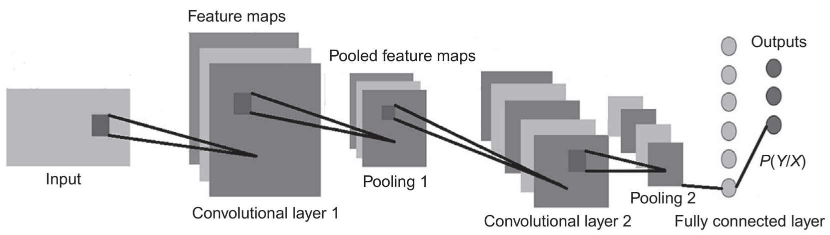

FIG. 5.6 Convolutional neural network.

that every neural network has its strength, and if it is required to improve the performance of the overall system, all its components need to be optimized in their operation.

5.7 Comparison of Spoken LID System Implementations with Deep Learning Techniques

- *FF-DNN for Spoken LID:* Perhaps the most cited FF-DNN with an acoustic-phonetic feature is that of Lopez-Moreno et al. [14]. They used Google 5M LID corpus and NIST LRE 2009 corpora. The target test utterance was 3 s. ReLU was the activation function. Two and eight hidden layers were used in two different test sets. Each hidden layer contained 2560 units. One extra output node was provided for out-of-set languages. Thirty-nine PLP ($13+\triangle+\triangle\triangle$) features were used in frame level. The input layer was fed with 21 frames formed by stacking the current processed frame and its ± 10 left-right context. Training was done using asynchronous stochastic gradient descent (SGD) within the DistBelief framework. The learning rate was 0.001, and minibatch size was 200 samples. The output scores were computed at an utterance level by respectively averaging the log of the softmax output of all its frames. C_{avg} (average cost) and EER (equal error rate) was used as performance metrics. It was claimed that the best performance was achieved by the DNN systems, where the eight hidden layers of DNN architecture yielded up to a $\sim$70% relative improvement in C_{avg} terms with respect to the best reference system with Google 5M LID corpus. A relative improvement of $\sim$43% in EER was obtained with eight-hidden layers of DNN, trained with 200 h of NIST LRE 2009 corpora. It was concluded with a heavy amount of training data (>20 h) per language is a key for the DNN to outperform i-vector system.

Richardson et al. [15] proposed another FF-DNN for language identification. PLP features were chosen as the feature. Samples of speech were converted into 20 ms fragments. A context of $\pm$5–10 frames around the current input frame was used. DNN posteriors and DNN bottleneck features were used. For the experiments, a subset of six languages of the NIST 2009 LRE corpora was used. Twenty hours of training per language was done resulting in 120 h total training. Samples of speech at 3, 10, and 30 s were selected for experimentation. For a direct approach, DNN was trained using the training partition of the data set and consisted of 819 input nodes (stack of 21 frames of 13 Gaussianised PLP with their first- and second-order derivatives). Two hidden layers with 2560 nodes per layer and six output nodes were selected. The activation function was chosen to be Sigmoid. The learning rate was kept at 0.2. Degradation in performance was observed relative to the baseline system at the 30 and 10 s durations,. There was a slight gain at 3 s duration for two-layer DNN. BNF together with GMM posterior provided the best performance. It was observed that increasing the amount of training data and the number of parameters both up to a factor of three did not significantly improve performance. Appending delta features to the PLP features gave a small but significant performance gain. It was mentioned that without Guassianisation on the input PLP features, the LID performance degraded by well over a factor of two. Fusion did not always show a better identification rate. The best domain score fusion showed a performance gain of almost 20% relative to the BNF/GMM system alone, while the best out of domain score fusion gave a relative gain of only about 9%.

Herry et al. [16] proposed a different method based on language pair discrimination using MLP as classifiers of acoustic features. OGI MLTS Corpus of 11 languages was used. The focus was on 3 s signals. Several MLPs were used where each of them discriminated a couple of languages. The theory behind it is that less output choices improves accuracy, so a language is paired with one of the rest of the 10 languages to make one detection phase. Finally, all the results were merged to conclude the identified language. The front end was RASTA processing to generate acoustic vectors of dimension 24 of relevant spectrum representation. RASTA processing on 32 ms signal with an overlap of 50% was used. The MLPs used sigmoidal activation functions, 50 hidden cells and two outputs. The learning algorithm used was the stochastic back propagation algorithm. The obtained results highlighted scores ranging from a 75.1% to an 82% detection rate.

- *CNN for Spoken LID:* Montavon [17] proposed a CNN for spoken LID. He used VoxForge and RadioStream dataset. Each sample corresponded to a speech signal of 5 s. One shallow CNN-TDNN and one three-layer deep CNN-TDNN were used for experiment. 30 mel-frequency filterbank between 0 and 5 kHz was used. The whole 5 s sample was used as an input. The classifier was trained with SGD. A confusion matrix was used for performance measurement. It was claimed that deep architecture was 5%–10% more accurate than the shallow counterpart. It was also concluded that the overall accuracy of a spoken LID system can vary considerably depending on the selected subset of languages to be identified. It was surmised that deep learning, compared to techniques based on hand-coded features, is still a mystery. It was suggested that accuracy can be improved by collecting more samples and extending the time dependence in order to learn higher level language features.

Lei et al. [18] proposed a CNN implementation in noisy conditions. In one of the approaches, the state posterior counts from the CNN are used directly for language identification. The approach was inspired by both the phoneme posteriogram and PRLM approaches. They used Defense Advanced Research Projects Agency (DARPA) Robust Automatic Transcription of Speech program (RATS) data. Speech samples taken at 3, 10, 30 and 120 s are used for the experiment. Three models, posterior extraction using CNN, CNN/i-vector and CNN posterior, were used. Forty filterbanks with a context of 15 frames were deployed, the height of the convolutional filter was kept at eight. Two hundred convolutional filters were used to model the data in more detail. The output vectors of the different filters are concatenated into a long vector that is then input to a traditional DNN. This DNN usually includes five to seven hidden layers. CNN trained for ASR is used to extract the posterior for every frame. Then the posteriors from the CNN are used to estimate the zeroth and first order statistics for i-vector model training. This is an acoustic way CNN/i-vector system. A CNN posterior system focus is on modeling the sequence of phonetic units given by a phone recognizer. This approach only uses zeroth order statistics to estimate the low-dimensional vector. Five target languages and one that was from the set language were taken for the study. 200×8 convolutional filters of 40 log mel filterbank coefficients with a context of 7 frames from each side of the center frame were chosen as input. The subsequent DNN included five hidden layers with 1200 nodes per hidden layer, and the output layer with 3353 nodes of representation of the senones. The neural network back-ends with a single hidden layer of 200 nodes were used for all presented system. All input vectors to the NN back-end were size 400. The output layer was a size 6. To optimize the performance on all durations, the original dataset was separated into 8 and 30 s segments with a 50% overlap. The performance was evaluated on three measurements, including EER, for all target languages. The miss rate where the false alarm rate was equal to 1% and the C_{avg}. fusion with linear logistic regression and twofold cross-validation were also done. Relative improvements between 23% and 50% on average EER with respect to a state-of-the-art UBM/i-vector system across different duration conditions. Fusion of both approaches gives to a 20% relative performance gain over the best individual system.

Ganapathy et al. [19] also supported using CNN for robust spoken LID. Though, they confined the use of CNN as a feature extractor and not as a classifier. The CNN models used were trained on noisy data provided under the RATS program for Arabic Levantine (ALV) and Farsi (FAS) KWS. Three hundred hours of data for each language was used for training. The CNNs were trained on 32 dimensional log-mel spectra augmented with Δ and $\Delta\Delta$s. The log-mel spectra were extracted by first applying the mel scale integrators on power spectral estimates in short analysis windows (25 ms) of the signal followed by the log transform. Each frame of speech was also appended temporally with a context of 11 frames. The CNNs used two convolutional layers with 512 hidden nodes. The first convolutional layer was processed with 9×9 filters while the second convolutional layer was processed with 4×3 filters. The nonlinear outputs from the second convolutional layer were then input to a fully connected DNN. Three hidden layers with 2048 units, followed by a bottleneck layer with 25 activations were deployed. They concluded that fusing bottleneck features along with acoustic features provided good results. The CNN BN features provided robust representations which were useful for the spoken LID tasks. The BN features provided about 21% relative improvement in the evaluation set and about 25% in the development set, as claimed by the authors.

- *RNN for Spoken LID:* Gonzalez-Dominguez et al. [20] proposed an LSTM RNN implementation for spoken LID. NIST LRE 2009 corpora was used for training and testing. Eight representative languages were selected with up to 200 h of audio available. Focus was given to 3 s utterances. The LSTM RNN architecture used contained 512 memory cells. The input to the network was 39-dimensional PLP features that were calculated at a given time step with no stacking of acoustic frames. The total number of parameters N, ignoring the biases, was $N = n_i \times n_c \times 4 + n_c \times n_c \times 4 + n_c \times n_o + n_c \times 3$, where n_c is the number of memory cells, n_i is the number of input units and n_o is the number of output units. The LSTM RNN model was trained using asynchronous stochastic gradient descent (ASGD), and the truncated backpropagation through time (BPTT) learning algorithm within a distributed training framework. Activations are forward propagated for a fixed-step time of 20 over a sub sequence of an input utterance; the cross entropy gradients are computed for this subsequence and backpropagated to its start. For better randomization of gradients in ASGD and stability of training, the training utterances were split into random chunks of length between 2.5 and 3 s. The same language identification was set sparsely for a chunk, one in every five frames for the experiments. The errors were only calculated for the frames for which a target language identification were set. One hundred machines were used for distributed training, and in each machine four concurrent threads processed a batch of four subsequences. An exponentially decaying learning rate of le-04 was used. For scoring, an utterance level score was computed for each target language by averaging the log of the softmax outputs for that target language of all the frames in an utterance. In order to assess the performance, two different metrics were used, C_{avg} and ERR. Two major findings were concluded. It was observed that LSTM RNN architecture provided better performance than FF-DNN with four hidden layers. The fact is particularly interesting as the proposed LSTM RNN contained 20 times fewer parameters than four hidden layers FF-DNN. Also, the C_{avg} values indicated that the scores produced by the LSTM RNN model were better calibrated than those produced by FF-DNN or i-vector systems. Both neural network approaches (FF-DNN and LSTM-RNN) surpass the i-vector system performance by $\sim$47% and $\sim$52% in EER and C_{avg}, respectively.
- *Hybrid NN Models for Spoken LID:* Every ANN has its strengths. When applied in a particular task, one NN provides better performance than others do. Keeping that in consideration, researchers tried hybrid NN models for the spoken LID.

Jin et al. [21] proposed an end-to-end DNN-CNN classification for spoken LID. The authors introduced two end-to-end DNN-CNN neural network variants, which utilized high-order LID-senone statistics. Both systems combine the benefits of both the high-order Baum-Welch statistics calculation of i-vector systems with the natural discriminating attributes of neural networks. In LID-net2, high-order statistics were obtained through an O2P method borrowed from fine-grained visual recognition, whereas in LID-bilinear-net, the statistics were obtained using the outer product operation from two different layers and pooled to obtain an utterance representation. Experiments were conducted on a full 23 languages of NIST LRE 2009 corpora, and performances were compared to the state-of-the-art DBF/i-vector systems. First, a six-layer DNN (48×21-2048-2048-50-2048-2048-3020) with an internal bottleneck layer was trained. Then parameters were transferred from the first three layers to DNN layers 1–3 of LID-net, which was trained. Finally, all layer parameters below the SPP layer were transferred to LID-net2 and LID-bilinear-net and the two networks were trained separately. In LID-net2, the high-order statistics of LID-senones were obtained through second-order pooling (O2P) of posteriors pooled from the CNN convolution layer prior to the FC mapping network. In LID-bilinear-net, the network utilized posteriors pooled from two different CNN layers instead of one. Each network was trained and tested independently for 30, 10 and 3 s data. Each network had six convolutional layers. The feature maps from CNN layers 1–5 had 512 channels and the feature maps after layer six were evaluated with between 64 and 512 channels. Each convolutional layer was followed by a batch normalization layer and first and second order LID-senone statistics were evaluated. It was observed that both systems outperformed the baseline system for 3 and 10 s utterances in terms of EER. Only 30 s utterances were a slightly behind in terms of performance.

Ranjan et al. [22] proposed two-stage training and transfer of learning in their research. They used the dataset from the NIST 2015 LRE i-vector machine learning challenge. The training set contained 300 i-vectors per language corresponding to each of the 50 in set target languages along with a test set and a development set. The speech utterances corresponding to the training set i-vectors were chosen so that their durations would exhibit a log-normal distribution with a mean duration of 35.15 s. In the two-step DNN training strategy, the initial DNN was trained using only in set labeled training data. The initial DNN was then used to estimate out-of-set labels from the development data. Next, a second DNN is trained for LID with both in set and estimated out-of-set labels. A FF-DNN with sigmoid activation function was used in first training. A new FF-DNN is trained using the i-vectors in the second stage. The mini-batch SGD algorithm was used with a mini-batch size of 256. The DNN had two hidden layers with 1024 units each. The input layer had 400 nodes corresponding to the 400-dimensional i-vectors. A dropout factor of 0.2 was used. The output layer had 50 nodes corresponding to all in set languages. A cost value of 26.82 was reported for 1024 hidden layers.

In their paper, Ferrer et al. [23] compared three approaches using combinations of DNN with other systems to identify the spoken language. The first two approaches used DNN as a feature extractor, while in the third case, DNN is used as classifier. In the first approach, zeroth-order statistics for i-vector extraction were estimated using a DNN, and the method was named DNN/iv modeling approach. The second approach, named DNN/post, used a DNN output layer to create features for language recognition. The posteriors for each senone at each frame were processed to generate a single vector per utterance, which is then modeled with standard back-end techniques. A third approach was to train a DNN with a bottleneck architecture, and then use the outputs of the bottleneck layer as features within an i-vector framework. NIST LRE 2009 and RATS SLR were used as corpus. The authors claimed improvements between 40% and 70% relative to a state of art GMM i-vector system on test durations from 3 to 120 s.

Bartz et al. [24] used a hybrid convolutional recurrent neural network (CRNN) that operates on spectrogram images of audio snippets. In their work, they tried to utilize the power of CNNs to capture spatial information, and the power of the RNNs to capture information through a sequence of time steps for identifying the language from a given audio snippet. They created their own dataset. Data was collected for six different languages. The network architecture consisted of two parts. The first part was the convolutional extractor that took a spectrogram image representation of the audio file as input. This feature extractor convolved the input image in several steps and produced a feature map with a height of one. The feature map was then sliced along the x-axis, and each slice was used as a time step for the subsequent BLSTM network. The network used five convolutional layers, each layer followed by the ReLU activation function, batch normalization and 2×2 max pooling with a stride of 2. The BLSTM consisted of two single LSTMs with 256 outputs units each. Finally, both outputs were concatenated to a vector of 512 dimensions and fed into a fully connected layer with 4/6 output units serving as the classifier. The authors claimed an accuracy of 92% and an F1 score of 0.92 (Table 5.1).

5.8 Discussion

A total of 11 highly appreciated and cited Research Papers on automatic LID with deep learning are discussed above. This leads to an understanding of a few very useful criterions for a successful deep learning-based sutomatic LID. First, in order to attract more researchers to implement deep learning-based language identification system, we need a standardized, free and openly available data corpus. NIST is currently being used widely, but its availability is a factor that hinders more research in this area [15, 16, 23]. OGI MLTS is of age and a new consortium must be formed to make available a free and open speech corpora. Second, PLP and Filterbanks are still used widely, even though MFCC and its acceleration coefficients are considered as de facto standard. Third, almost every author is looking for a fusion of approaches, and fused approaches are providing better accuracies. Ferrer et al. [23] refers to a fused i-vector and

TABLE 5.1 Summary of the Works

Author Reference	Feature Extraction Technique	Classification Technique	Corpora Used	No. of Languages	Recognition Accuracy
13	39 PLP (13 + △ + △△) Stack of 21 frames	FF-DNN with 2–8 hidden layers	Google 5M and NIST LRE 2009	Google 5M – 25 languages +9 Dialects NIST LRE 2009 – 8 languages	EER – 9.58 (average) for DNN_8_200h
14	39 PLP (13 + △ + △△) Stack of 21 frames	FF-DNN with 2 hidden layers	NIST LRE 2009	6 languages	C_{avg} – 2.56 (for 30 s) 4.24 (for 10 s) 10.3 (for 3 s)
15	RASTA	MLP with 50 hidden cells	OGI MLTS	11 languages	Average Competitive Rate – 77%
16	End-to-end		VoxForge and RadioStream	3 languages	VoxForge – 91.2% (known speakers) 80.1% (new speakers) RadioStream – 87.7% (known radios) 83.5% (new radios)
17	40 Filterbanks	i-vector, posterior	RATS data	5 languages	EER – CNN/posterior – 14.08 (3 s) CNN/i-vector – 13.60 (3 s)

Continued

TABLE 5.1 Summary of the Works—cont'd

Author Reference	Feature Extraction Technique	Classification Technique	Corpora Used	No. of Languages	Recognition Accuracy
18	Multiple features including CNN-BF	GMM/i-vector	RATS data	5 languages	EER – BN-ALV – 15.3 (3 s) BN-FAS – 15.0 (3 s)
19	39 PLP	LSTM RNN	NIST LRE 2009	8 languages	EER –8.35 (average) C_{avg} – 0.0944
20	End-to-end		NIST LRE 2009	23 languages	EER – LID-net-256 – 7.57 (3 s) LID-bilinear-net-512-relu – 6.86 (3 s)
21	300 i-vector	DNN	NIST 2015 LRE i-vector machine learning challenge	50 languages	Cost (progressive subnet) DNN-2-1024 – 26.82
22	Mixed	Mixed	NIST LRE 2009 and RATS data	4 languages	DNN-eng/post – 14.18 (3 s)
23	End-to-end		Own dataset	4 languages	Accuracy – 92%

DNN BN system, which gives the best accuracy values in all 3, 10 and 30 s segments. Bartz et al. [24] also used a fusion of four systems to get the best C_{avg} values. A similar approach is observed in other papers, too. The reason is that all approaches have their strengths and weaknesses, and no single one can claim to be the best approach. The fusion of approaches is thus the only contender for the best approach. Fourth, its the 3 and 10 s segments that are critical for applications. Most of the work is done to find the suitable approach, which can result in best 3 s accuracy value. Finally, end-to-end systems are making headway and may be the future of automatic LID.

5.9 Conclusion

It is important to understand the role of spoken LID in speech technology. It is neither a complete ASR, nor does it involve the intricacies of NLP systems. Though it needs to be accurate, at the same time, it should also be fast and lightweight. Today, the spoken LID scenario can be divided into two branches, end-to-end and hybrid NN systems. Both have their advantages. In end-to-end, both feature extraction and classification are done with the same system, reducing the processing time and resource. But hybrid NNs tend to perform better. There has to be a compromise between accuracy and performance. Even though PPRLM and i-vector systems are still state-of-the-art, DNNs are catching on fast, and some results show that they even surpassed the other systems. Yet there is the question of how will the systems work when placed in a real-time environment. We have seen different sets of speech data corpus; NIST LRE is predominant, but even they do not cover all the languages of the world. People still need to create their own dataset for training and testing. An enormous repository of speech data is an absolute necessary before the systems can be used for field trials. There has been many parameters for accuracy but C_{avg} and EER are almost a benchmark now. Among the trending technologies, hybrid end-to-end DNNs seem to be the solution for the future. Since speech is a collection of sequences and not stationary, we may see variants of RNN and other sequence-training NNs in the future. We, the human beings, are mostly trained like a PPRLM model and that is why deep models that can emulate PPRLM, is something to look for in the future.

References

[1] Y.K. Muthusamy, E. Barnard, R.A. Cole, Automatic language identification: a review/tutorial, IEEE Signal Process. Mag. 11 (4) (1994) 33–41.

[2] K. Lan, D.T. Wang, S. Fong, L.S. Liu, K.K. Wong, N. Dey, A survey of data mining and deep learning in bioinformatics, J. Med. Syst. 42 (8) (2018).

[3] S. Hu, M. Liu, S. Fong, W. Song, N. Dey, R. Wong, Forecasting China future MNP by deep learning, in: Behavior Engineering and Applications, Springer, 2018, pp. 169–210.

[4] Dey, N., Ashour, A. S., & Nguyen, G. N., Deep Learning for Multimedia Content Analysis, Mining Multimedia Documents, CRC Press. ISBN 9781138031722.

[5] M.A. Zissman, K.M. Berkling, Automatic language identification, Speech Comm. 35 (1–2) (2001) 115–124.

[6] H. Li, B. Ma, K.A. Lee, Spoken language recognition: from fundamentals to practice, Proc. IEEE 101 (5) (2013) 1136–1159.

[7] C.-H. Lee, Principles of spoken language recognition, in: Springer Handbook of Speech Processing, Springer, Berlin, Heidelberg, 2008, pp. 785–796.

[8] P Coxhead, Natural Language Processing & Applications, In lecture Notes, NLP Resources.

[9] UFLDL Tutorial, Stanford University, http://deeplearning.stanford.edu/wiki/index.php/UFLDL_Tutorial.

[10] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016.

[11] M. Nielsen, Neural Networks and Deep Learning, http://neuralnetworksanddeeplearning.com.

[12] Wikipedia, https://en.wikipedia.org/wiki/List_of_animals_by_number_of_neurons.

[13] P. Roy, P.K. Das, S.K. Gupta, Comparison of SVMs and NNs approach for automatic identification of Indian languages, in: 1st International Science & Technology Congress, IEMCONGRESS-2014, August 28–31, Science City, Kolkata, India, 2014, ISBN 978-93-5107-248-5, pp. 39–44.

[14] I. Lopez-Moreno, J. Gonzalez-Dominguez, D. Martinez, O. Plchot, J. Gonzalez-Rodriguez, P.J. Moreno, On the use of deep feedforward neural networks for automatic language identification, Comput. Speech Lang. 40 (2016) 46–59.

[15] F. Richardson, D. Reynolds, N. Dehak, Deep neural network approaches to speaker and language recognition, IEEE Signal Process. Lett. 22 (10) (2015) 1671–1675.

[16] S. Herry, B. Gas, C. Sedogbo, J.-L. Zarader, Language detection by neural discrimination, in: Proc. ICSLP, 2004.

[17] G. Montavon, Deep learning for spoken language identification, in: NIPS Workshop on Deep Learning for Speech Recognition and Related Applications, 2009, pp. 1–4.

[18] Y. Lei, L. Ferrer, A. Lawson, M. McLaren, N. Scheffer, Application of convolutional neural networks to language identification in noisy conditions, in: Proc. Odyssey-14, Joensuu, Finland, vol. 41, 2014.

[19] S. Ganapathy, K. Han, S. Thomas, M. Omar, M. Van Segbroeck, S.S. Narayanan, Robust language identification using convolutional neural network features, in: *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[20] J. Gonzalez-Dominguez, I. Lopez-Moreno, H. Sak, J. Gonzalez-Rodriguez, P.J. Moreno, Automatic language identification using long short-term memory recurrent neural networks, in: Fifteenth Annual Conference of the International Speech Communication Association, 2014.

[21] M. Jin, Y. Song, I. McLoughlin, End-to-end DNN-CNN Classification for Language Identification, in: Proceedings of the World Congress on Engineering, vol. 1, 2017.

[22] S. Ranjan, C. Yu, C. Zhang, F. Kelly, J.H.L. Hansen, Language recognition using deep neural networks with very limited training data, IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2016) 5830–5834.

[23] L. Ferrer, Y. Lei, M. McLaren, N. Scheffer, Study of senone-based deep neural network approaches for spoken language recognition, IEEE/ACM Trans. Audio Speech Language Process. 24 (1) (2016) 105–116.

[24] C. Bartz, T. Herold, H. Yang, C. Meinel, Language identification using deep convolutional recurrent neural networks, in: International Conference on Neural Information Processing, Springer, Cham, 2017, pp. 880–889.