Spring 3-16-2018

Speech Emotion Recognition using Convolutional Neural Networks

Somayeh Shahsavarani
University of Nebraska-Lincoln, bahar@huskers.unl.edu

SPEECH EMOTION RECOGNITION USING CONVOLUTIONAL NEURAL NETWORKS

by

Somayeh Shahsavarani

A THESIS

Presented to the Faculty of
The Graduate College at the University of Nebraska
In Partial Fulfillment of Requirements
For the Degree of Master of Science

Major: Computer Science

Under the Supervision of Professor Stephen D. Scott

Lincoln, Nebraska

March, 2018

SPEECH EMOTION RECOGNITION USING CONVOLUTIONAL NEURAL NETWORKS

Somayeh Shahsavarani, M.S.

University of Nebraska, 2018

Advisor: Stephen D. Scott

Automatic speech recognition is an active field of study in artificial intelligence and machine learning whose aim is to generate machines that communicate with people via speech. Speech is an information-rich signal that contains paralinguistic information as well as linguistic information. Emotion is one key instance of paralinguistic information that is, in part, conveyed by speech. Developing machines that understand paralinguistic information, such as emotion, facilitates the human-machine communication as it makes the communication more clear and natural. In the current study, the efficacy of convolutional neural networks in recognition of speech emotions has been investigated. Wide-band spectrograms of the speech signals were used as the input features of the networks. The networks were trained on speech signals that were generated by the actors while acting a specific emotion. The speech databases with different languages were used to train and evaluate our models. The training data on each database were augmented with two levels of augmentations. The dropout technique was implemented to regularize the networks. Our results showed that the gender-independent, language-independent CNN models achieved the state-of-the-art accuracy, outperformed previously reported results in the literature, and emulated or even surpassed human performance over the benchmark databases. Future work is warranted to examine the capability of the deep learning models in speech emotion recognition using daily-life speech signals.

Acknowledgements

Working on this thesis has been a fascinating adventure for which I am extremely grateful as it has pushed my boundaries and broadened my horizons. For what I have accomplished, I would first like to express my deepest and sincerest gratitude to my advisor, Dr. Stephen Scott, whose knowledge, commitment, and pursuit of high standards have been and will be an endless source of inspiration to me. As well as his solicitous guidance in research, I am profoundly appreciative of his brilliant lectures that were one of the most instructive and encouraging resources for me to learn algorithms and machine learning. I am honored to be one of his Graduate students and indebted for all his support and guidance. I would also like to thank my committee members, Dr. Thomas Carrell and Dr. Vinodchandran Variyam. I am greatly thankful to them for consenting to be my committee members, reading this thesis, and providing me with their suggestions and feedback. I extend my special thanks to Dr. Thomas Carrell for his expertise and thoughtful insight that greatly enriched and improved my work. Special thanks go for Dr. Monita Chatterjee and her colleagues, in Auditory Prostheses and Perception Laboratory at Boys Town National Research Hospital, for sharing their speech emotion database with us. Also, I would like to express my appreciation to all the faculty and staff members. I extent my special thanks to Ms. Deb Heckens whose kindness and in-time help always delighted me when I was desperate. Last but not least, I would like to pay special warmth and appreciation to my beloved husband, Dr. Dr. Omid Zandi, my mother, Soheila Azimi, my father, Esmaeil Shahsavarani, my family, and my friends whose love, care, support, patience, and encouragement have constantly motivated me to move forward.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The past several decades have witnessed a wealth of studies on understanding the human brain and building systems that mimic human intelligence [1, 2, 3, 4, 5, 6]. The human brain is an intricate organ that has been a lasting inspiration for research in Artificial Intelligence (AI). The neural networks of human brain are strongly competent to learn high-level abstract concepts from experiencing low-level information processed by sensory periphery. Learning language, understanding speech, and recognizing faces are some examples that manifest the remarkable power of the human brain in learning high-level concepts. The main goal of AI is to develop intelligent systems that are able to generate rational thoughts and behaviors similar to human thought and performance [1]. There are a variety of study fields that are considered as the sub-fields of AI. Computer vision, natural language processing, automated reasoning, robotics, machine listening, and machine learning are some of these major areas in AI research.

Developing machines that interact with humans by understanding speech pave the way for building systems that are equipped with human-like intelligence. Speech is the most natural and convenient way by which humans communicate, and understanding speech is one of the most intricate processes that human brain performs. Automatic speech recognition (ASR) has been an active field of AI research aiming to generate machines that communicate with people via speech [7, 8]. The early ASR systems mainly focused on the linguistic properties of speech to understand spoken utterances [9, 10, 11, 12]. Speech is an information-rich signal that contains paralinguistic information as well as linguistic information. Iden-

tity, gender, intention, mood, and emotion are some key paralinguistic information that are conveyed by speech and they were less investigated by the classical ASR framework [13]. The human brain employs all linguistic and paralinguistic information to understand the underlying meaning of the utterances and have effective communication. In fact, any deficit in the perception of paralinguistic features has an adverse effect on the quality of communication. It has been argued that children who are not able to understand the emotional states of the speakers develop poor social skills and in some cases they show psychopathological symptoms [14, 15]. This highlights the importance of recognizing the emotional states of speech in effective communication. Hence, developing machines that understand paralinguistic information such as emotion is imperative for establishing a clear, effective, and human-like communication.

Emotion recognition has been the subject of research for years. Detection of emotion from facial expressions and biological measurements such as heart beats or skin resistance formed the preliminary framework of research in emotion recognition [16]. More recently, emotion recognition from speech signal has received growing attention. The traditional approach toward this problem was based on the fact that there are relationships between acoustic features and emotion. In other words, emotion is encoded by acoustic and prosodic correlates of speech signals such as speaking rate, intonation, energy, formant frequencies, fundamental frequency (pitch), intensity (loudness), duration (length), and spectral characteristic (timbre) [13, 17]. There are a variety of machine learning algorithms that have been examined to classify emotions based on their acoustic correlates in speech utterances. Hidden Markov models (HMM), Gaussian mixture models, nearest neighborhood classifiers, linear discriminant classifiers, artificial neural networks, and support vector machines are some examples that have been widely used to classify emotions based on their acoustic features of interest [13, 17, 18, 19, 20]. The performance of these classifiers mostly depends on the feature extraction techniques and the features that are considered salient for a specific emotion. Given

the fact that the acoustic correlates of emotion in speech signals vary across speakers, genders, language, and cultures [13], there is no universal agreement on the acoustic correlates of emotions [21, 22]. This results in a myriad of "hand-crafted" features depending on the speech corpus. Using deep learning models is one sensible approach to tackle this problem.

Speech emotion recognition is inherently a multimodal process. Although speech modality conveys a large portion of the emotional information, it is not sufficient for recognizing affective states of humans in daily-life situations. Other modalities such as visual or linguistic modality also contribute to convey necessary information required for recognizing emotions. That is, in addition to speech, people use other paralinguistic cues such as facial expression, body language, semantic, or context to identify emotions in others. In fact, it has been debated that 55% of the message is conveyed by body language when people communicate with one another [23]. We limited our study to speech modality and have not considered other modalities. As a result, the databases that were acted by actors were used in the current study because the emotions were expressed with exaggeration by actors, which potentially compensates for the lack of information provided by other modalities. This allows us to explore the effectiveness of deep learning models with greater control compared with using daily-life utterances. On the other hand, the broad availability of the acted speech benchmarks in the speech community enables us to compare our work with previously explored models. In the current study, we investigated the capability of convolutional neural networks in classifying speech emotions using acted speech databases within and across four different languages: German, Italian, British English, and American English. The specific contribution of this study is using wide-band spectrograms instead of narrow-band spectrograms as well as assessing the effect of data augmentation on the accuracy of our models across widely-used benchmark databases. Our results revealed that wide-band spectrograms and data augmentation equipped CNNs to achieve the state-of-the art accuracy and surpass human performance.

The remainder of this thesis is organized as follows. Chapter 2 briefly reviews the previous work related to the current study. Chapter 3 explains the basic machine learning concepts that are essential to understand the experiments conducted in the current work. Chapter 4 describes the experimental setup including the architecture of the networks that were implemented, and the databases that were administered in this study. Chapter 5 delineates the results and discusses the outcomes. Chapter 6 summarizes the current study and suggests potential work for future.

Chapter 2

Related Work

Deep learning is a modern machine learning technique that emerged to deal with big databases and complex systems. The advent of deep learning brought with it a wave of novel algorithms that diminished the need for "hand-crafted" features prior to classification [24]. That is, deep learning models can learn low-level features from training data in their lower layers and build high-level representation in the upper layers based upon the proceeding layers. As a result, the deep learning models are able to extract the features automatically. Recently, a rapid growth has been observed in using deep learning models to classify speech emotions. The efficacy of deep learning models in speech emotion recognition has been examined during last years in various studies. Stuhlsatz et al. [25] compared the performance of a Generalized Discriminant Analysis (GerDA) based on deep neural networks (DNNs) with support vector machines (SVM) on classifying speech utterances using their emotional dimensions such as arousal and valence [26]. In a specific preprocessing procedure, the static acoustic features were extracted and fed into the classifiers as the input data. Their results showed that the DNN highly outperformed the SVM in detection of emotional dimensions of speech utterances. Li et al. [27] compared the performance of the hybrid deep neural network-hidden Markov model (DNN-HMM) classifier with the hybrid Gaussian mixture model-hidden Markov model (GMM-HMM) classifier in speech emotion recognition. The DNN in the DNN-HMM was used to extract the discriminative features that were later used by the HMM to classify speech emotions. Their results showed that the DNN-HMM outperformed the GMM-HMM in classifying speech emotions. Mao

et al. [28] used convolutional neural networks, which are a specific kind of deep learning models, to automatically learn discriminative features from the narrow-band spectrograms of speech signals. Subsequently, they used an SVM to classify the features learned by CNN. Their results demonstrated a superior performance in learning high-level discriminative features from the low-level spectrographic representation of the speech signals. Fayek et al. [29] implemented a deep neural network (DNN) to classify speech emotions. The one-second narrow-band spectrograms of the speech signals were used as the input of the DNN. Their results showed an improvement over classic machine learning algorithms. Zheng et al. [30] implemented a convolutional neural network to learn discriminative features from narrow-band log-spectrograms of speech signals. Similar to previous studies, their results showed a performance improvement over the methods that used hand-crafted features to classify speech emotions. Trigeorgis et al. [31] proposed an end-to-end deep learning model in which they combined a convolutional neural network with a recurrent neural network. Recurrent neural networks (RNNs) are sequence models that deal with sequential data such as speech, text, or video [32]. Trigeorgis et al. [31] took advantage of the spatial resolution of the CNNs and the temporal resolution of the RNNs to extract discriminative features of emotions from the raw data without any preprocessing; that is, they used the raw audio signals as the inputs of their model. Their results also confirmed the efficacy of the deep learning models in learning salient features to recognize affective states of humans using their speech utterances. Recently, Papakostas et al. [33] assessed the capability of deep convolutional neural networks in classifying speech emotions using publicly available databases. In doing so, they compared the performance of deep neural models with SVM classifiers. Their results manifested the superior performance of the deep neural networks over SVM classifiers in recognizing emotional states of the speech sounds. Specifically, Papakostas et al. [33] developed convolutional neural networks with four convolutional layers, each followed by a max-pooling layer, and two fully connected layers. The stochastic gradient

descent (SGD) algorithm was used to train the networks with 5000 training iterations. The models were tested and withing and across languages using F1-score. In addition, dropout technique and data augmentation were employed to combat overfitting the training data. To augment training data, background noise with three different levels of signal to noise ratio was added to speech signals. The 250×250 narrow-band spectrogram images of the speech signals were fed into the deep neural networks as the input. In our work, we also examined the efficacy of convolutional neural networks in decoding emotions in speech signal. Our work differs from the work by Papakostas et al. [33] from several aspects. For instances, we used wide-band spectrogram images as the input of the convolutional neural networks in lieu of the narrow-band spectrogram images; also, we used a different learning algorithm and slightly different data augmentation. Further, we developed convolutional neural networks with significantly smaller number of hyperparameters and notably faster than the models developed by Papakostas et al. [33]. Above all, our convolutional neural networks outperformed the previous models including the work by Papakostas et al. [33]. Prior to diving into the details of our work, next chapter reviews some basic concepts in machine learning and deep learning.

Chapter 3

Deep Artificial Neural Networks

Machine learning is an area of study in Computer Science and Artificial Intelligence. It primarily focuses on developing algorithms that can automatically learn a task or a skill, and gain knowledge by experience such as observing training data. Subsequently, it is desirable that machine learning algorithms generalize well their learned knowledge from these observations to new, unseen data. In doing so, different learning strategies such as supervised learning, unsupervised learning, or reinforcement learning can be employed. In what follows, we give a brief introduction on supervised learning as it has been applied in the current study. To learn about other types of learning, we refer the reader to Mitchell et al. [34], Bishop [35], Suton and Barto [36], and James et al. [37].

In supervised learning, the training data incorporate the desired response, called labels; that is, for each observation (training sample or instance), there is a corresponding label. The goal of learning is to predict the label of each training/test instance as correctly as possible. Classification is one example of supervised learning wherein the machine learning algorithm learns to classify the input data into two or more categories. To do so, the algorithm learns the discriminant features or attributes across different categories or classes based on observing training data. These features are later used to classify new test input data. Artificial neural networks (ANNs) are one well-known instance of supervised learning algorithms although some ANNs can be trained by unsupervised learning [38]. ANNs are inspired by the way biological neural networks, such as human central nervous system, work. That is, they consist of a highly interconnected processing

units, called neurons. ANNs are the basic building blocks of deep learning, which is a strong modern machine learning technique. In fact, deep learning models are ANNs with a plethora of neurons and layers. Deep learning models have achieved remarkable successes in various machine learning applications such as classification. The key concept that makes deep learning models efficacious is their ability to learn complex features out of simple features [24]. That is, the first layers of deep learning models represent simple and basic features of the training data. The deeper layers build a complex representation by using these low-level features. This ability of building high-level features out of low-level features can potentially reduce the amount of preprocessing required for extracting hand-crafted features before designing classifiers. In the current study, we have implemented a deep learning model, called convolutional neural network, to classify emotional states of speech signals.

This chapter provides background knowledge that is necessary to understand the experimental setup (Chapter 4) and results (Chapter 5) of the current work. The remainder of this chapter is organized as follows. In Section 3.1, we introduce the multi-layer perceptron network, which is a popular ANN architecture. In Section 3.2, we introduce the convolutional neural network, which is a well-known and effective deep learning model, especially in the field of speech and image processing. Finally, in Section 3.3, we describe k-fold cross-validation, which is a commonly used model assessment technique to evaluate the performance of machine learning models.

3.1 Multi-layer Perceptron Networks

The network architecture is one of the factors that characterizes artificial neural networks (ANNs). It determines the way neurons, the basic processing units, are connected to one another. The multi-layer perceptron (MLP) is a long-established ANN architecture that is composed of neurons called linear threshold units (LTUs) [38]. Figure 3.1 illustrates a linear threshold unit. As demonstrated, an LTU

receives weighted inputs from different neurons (here from three neurons) and computes the linear combination of the inputs as $z = w_1x_1 + w_2x_2 + w_2x_2 + b$, where b is the bias term. Then, a step function (e.g., Heaviside function $H(x)$ or Sign function $sgn(x)$) is applied on the linear combination to generate the output as $y = f(z)$ where f is a step function. If the weighted sum is greater than a threshold value (which is affected by the bias term), the LTU will generate an output.

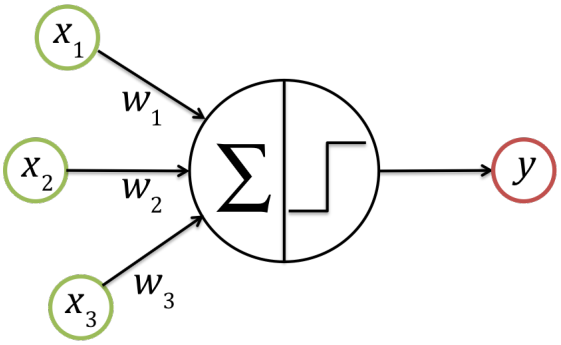


Figure 3.1: Linear threshold unit. It processes the inputs by computing the linear combination of the inputs and applying a step function.

Generally, an MLP incorporates one input layer, one or more LTU layers (called hidden layers), and one output layer. The information flows from the input layer (lower level) toward the output layer (higher level). That is why they are called feedforward artificial neural networks. The input layer represents the values of one training sample in different dimensions. This can be the amplitude of an audio signal at different sampling points or the intensity of an image at different pixels. The input is usually denoted as a vector $\vec{x}$ whose length indicates the number of dimensions (e.g., the number of sampling points in an audio signal or the number of pixels in an image). The output is either a real number, y, or a vector, $\vec{y}$, which shows the label of the input. An MLP is mainly configured as the layers of the neurons denoted as $l^{[0]}, l^{[1]}, l^{[2]}, \ldots, l^{[n]}, l^{[n+1]}$, where $l^{[0]}$ is the input layer, $l^{[1]}, l^{[2]}, \ldots, l^{[n]}$ are the n hidden (LTU) layers, and $l^{[n+1]}$ is the output layer. Every neuron within the layer $l^{[j]}_{1 \leq j \leq (n+1)}$ (all layers except the input layer) directly receives the weighted input from every neuron within a layer that is one level low,

i.e., $l^{[j-1]}$. Figure 3.2 demonstrates an MLP with one hidden layer where W_1 and W_2 are matrices associated with the values that weight the inputs of the neurons in the hidden layer and the output layer, respectively. The vectors $\vec{b_1}$ and $\vec{b_2}$ are the weights associated with bias terms.

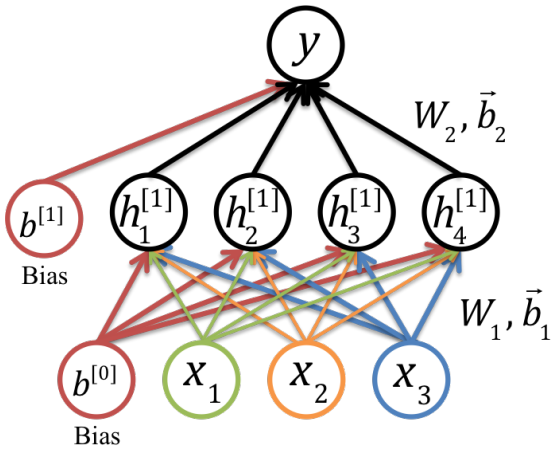


Figure 3.2: Multi-layer perceptron. W_1 and $\vec{b_1}$ denote the weight matrix and the bias vector associated with the input layer. W_2 and $\vec{b_2}$ are the weight matrix and the bias vector associated with the hidden layer.

In fact, the weight matrices (W_1 and W_2) and the bias vectors ($\vec{b_1}$ and $\vec{b_2}$) are the parameters of interest. That is, the network learns to classify the input data by adjusting the values of these parameters. There are several learning techniques that can find the optimum values of these parameters. Backpropagation is one example of these learning methods that has been widely used to train MLP networks. The basis of the backpropagation algorithm is gradient descent, which is briefly introduced in the next section.

3.1.1 Gradient Descent

Searching for the optimum values of the weight parameters can be viewed as an optimization problem. Having the error function or loss function, the goal is to find the parameters of interest in a way that minimize the error function. There are several ways to define the error function. Equation 3.1 displays a well-established error function called sum of squared errors [34] where W, d, and D stand for weights, one training instance, and all training data, respectively. As

shown, the squared of error between the actual label ($\vec{t_d}$) and the predicted label ($\vec{y_d}$) is summed over all training data. The goal is to search the weight space for the values that minimize this function.

$$E(W) = \frac{1}{2}\sum_{d\in D}(\vec{t_d} - \vec{y_d})^2. \tag{3.1}$$

Gradient descent is a common optimization algorithm that is used to minimize the error function. Suppose we have a linear unit that computes the weighted sum of its inputs as follows:

$$z = w_1x_1 + w_2x_2 + \cdots + w_px_p + b \quad = \vec{w}\cdot\vec{x} + b, \tag{3.2}$$

where b, $\vec{w}$, $\vec{x}$, and z are the bias term, weight vector, the input vector, and the output of the linear unit, respectively. This unit's error function is

$$E(\vec{w}) = \frac{1}{2}\sum_{d\in D}(t_d - y_d)^2. \tag{3.3}$$

Gradient descent, which is an iterative technique, begins with a random initial values of the weight parameters and updates the weights in the opposite direction of the gradient of the error function as presented in equation 3.4 [34]

$$w_i = w_i - \eta\frac{\partial E}{\partial w_i}, \tag{3.4}$$

where w_i is the weight associated with the ith dimension of the input and η is the learning rate that determines the step of updates.

The backpropagation algorithm employs gradient descent to find the local minima of the error function of the multi-layer perceptron (MLP) networks. The error function of the MLP networks is not convex as it is in a linear unit. As a result, finding global minima is not guaranteed. Since we do not have access to the output of the hidden neurons, the backpropagation algorithm takes advantage of the chain rule of calculus and computes the contribution of hidden neurons to the

output error to update the weights associated with the hidden layers [24, 38]. It should be noted that the step function of the MLP networks poses challenges for taking the derivative with respect to the weights. Therefore, the step function is replaced with the sigmoid function, $\sigma(x) = \frac{1}{1+e^{-x}}$,which is differentiable. For more detail, we refer the reader to [34, 24, 38].

There are several variants of gradient descent such as gradient descent with momentum [39] or root mean square propagation (RMSprop) [40] that increase the rate of convergence. Gradient descent with momentum uses an exponentially decaying weighted average of gradients instead of gradients to update the weights. That is, the gradient across the number of iterations can be viewed as a time-series signal, $\frac{\partial E}{\partial w}(t)$, where t stands for the number of iterations. The gradients can be replaced with the exponentially weighted average of gradients as $v_t = \beta v_{t-1} + (1-\beta)\frac{\partial E}{\partial w}(t)$, where $v_0 = 0$ and $\beta \in [0, 1]$, to update the weights. RMSprop optimization computes the exponentially decaying weighted average of the squared gradient, $m(t) = \beta m_{t-1} + (1-\beta)(\frac{\partial E}{\partial w}(t))^2$, where $m_0 = 0$ and $\beta \in [0, 1]$, and use its square root to scale the learning rate as $\eta\prime = \frac{\eta}{\sqrt{m(t)}}$. These variants of gradient descent can be used to smooth out the oscillation around local optima and increase the rate of convergence. Adam algorithm introduced by Kingma and Ba [41] is another gradient-based optimization algorithm that combines the merits of gradient descent with momentum and RMSprop optimization to minimize the loss function [42]. That is, it employs the first order momentum as in gradient descent with momentum and the second-order momentum as in RMSprop optimization to update the weights. The name of this algorithm roots from "adaptive moment estimate". This algorithm is very effective and has been widely used in deep learning models.

3.2 Convolutional Neural Networks

Convolutional neural networks (CNNs) are one of the most popular deep learning models that have manifested remarkable success in the research areas such as

object recognition [43], face recognition [44], handwriting recognition [45], speech recognition [46], and natural language processing [47]. The term convolution comes from the fact that convolution—the mathematical operation—is employed in these networks. Generally, CNNs have three fundamental building blocks: the convolutional layer, the pooling layer, and the fully connected layer. Following, we describe these building blocks along with some basic concepts such as softmax unit, rectified linear unit, and dropout.

3.2.1 Convolutional Layer

Convolutional layers in CNNs use convolution instead of multiplication to compute the output. As a result, the neurons in the convolutional layers are not connected to all the neurons in their preceding layers. This architecture is inspired by the fact that neurons of the visual cortex have local receptive field [48, 49]. That is, the neurons are specialized to respond to the stimuli limited to a specific location and structure. As a result, using convolution introduces sparse connectivity and parameter sharing to CNNs, which decreases the number of parameters in deep neural networks drastically.

Figure 3.3 demonstrates the convolution of a kernel, which is a 2×2 matrix, with a one-channel 3×3 image. The output is a volume of $2 \times 2 \times 1$. Generally, the size of output is $(n_h - f + 1) \times (n_w - f + 1) \times n_f$, where n_h is the height of the input, n_w is the width of the input, and n_f is the number of kernels. The depth of the kernel is determined by the depth of the input. For the example demonstrated in Figure 3.3, the depth of the input is $n_c = 1$. As a result the depth of kernel is 1. Also, the depth of the output is 1 since there is only one kernel. As can be seen, each output neuron is the weighted sum of the input neurons within the corresponding receptive field, which introduces sparse connectivity in CNNs. Further, the kernel is shared across the layer, which introduces parameter sharing in CNNs. The step by which the kernel slides along the input is called stride. In our example (Figure 3.3), the stride is $s = 1$, which means that the kernel shifts

one step over the image. It should be noted that the input volume shrinks after each convolutional layer. To avoid this decrease, we can pad the outer edge of the input with zero. Overall, the height and width of the output is determined by $\lceil \frac{n_h+2p-f}{s} \rceil + 1 \times \lceil \frac{n_w+2p-f}{s} \rceil + 1$, where p is the number of zero padding and s is the number of strides [50].

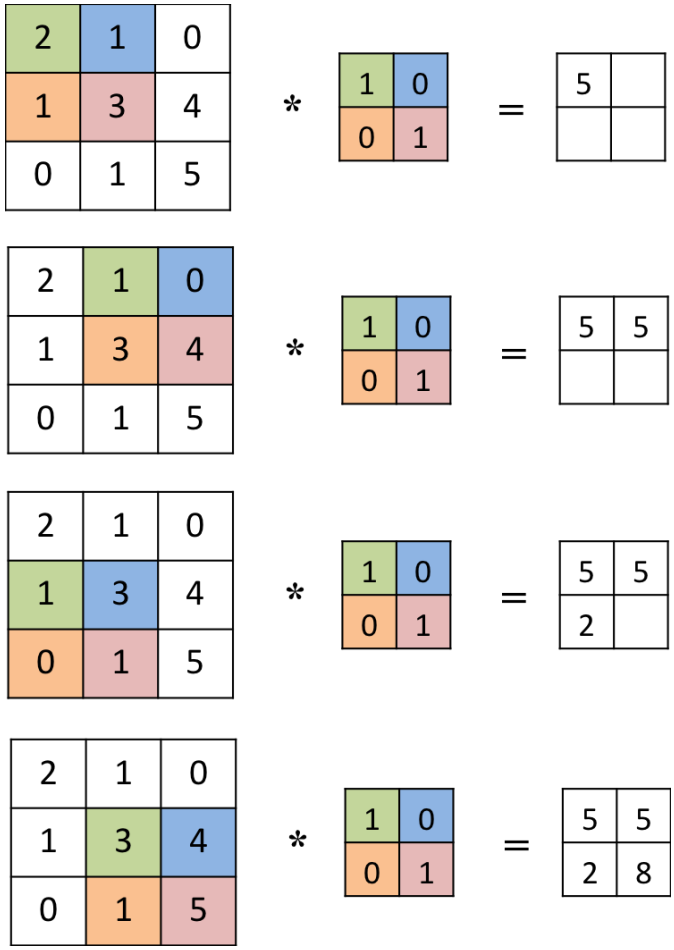


Figure 3.3: The convolution of a 3×3 image by a 2×2 kernel with a stride of 1.

The local filtering that happens in convolutional layers allows detecting different basic low-level features of interest and generating various feature maps. The deep layers use these feature maps to construct the high-level representation of the inputs.

3.2.2 Pooling Layer

The second important building block of CNNs is a pooling layer. This layer is used to make the outputs less sensitive to the local variation in the inputs.

This invariance to small local translation can decrease the spatial resolution and lead to underfitting in some applications. When accurate spatial features are not required, pooling can improve the performance of CNNs in extracting the features of interest. Further, pooling can reduce overfitting since it decreases the number of dimensions and parameters [24]. In a sense, pooling takes subsamples from the outputs [38]. Similar to convolutional layers, pooling layers use a kernel (a rectangular receptive field) to apply an aggregation function such as maximum, average, $L2$-norm, or weighted average to summarize the values of the neurons within the pooling kernel. To have a pooling layer in CNNs, we need to determine the size of pooling kernels, the step of shifting, and the number of padding. Figure 3.4 depicts max pooling over a 3×3 matrix where the size of pooling kernel is 2×2 and the kernel shifts one pixel over the matrix (i.e., stride of 1).

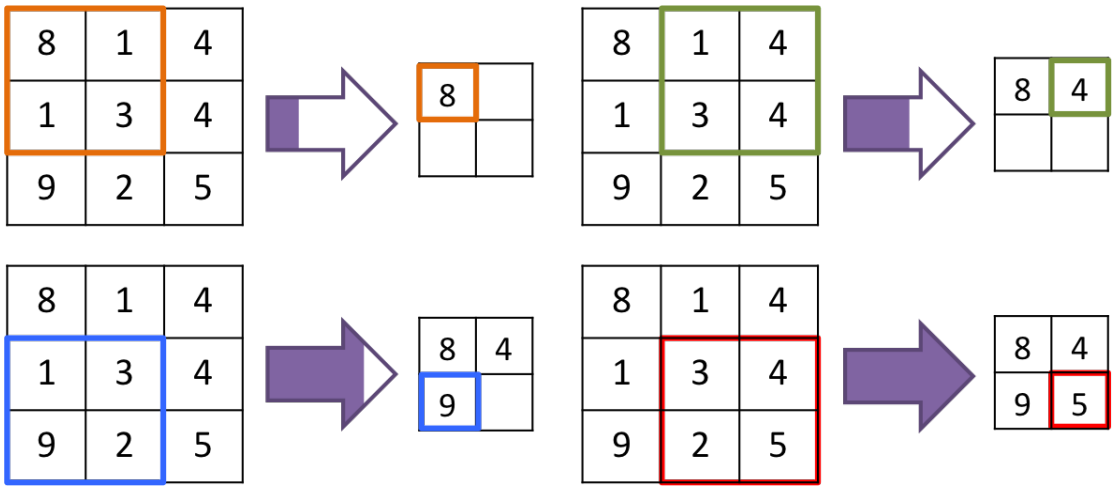


Figure 3.4: Pooling of a 3×3 image using a 2×2 kernel with a stride of 1; the maximum value of each window is supsampled.

3.2.3 Fully Connected Layer

A typical CNN consists of several convolutional layers where each convolutional layer is followed by a pooling layer. The last building block of CNNs is the fully connected layer, which is basically a traditional MLP. This component is used to either make a more abstract representation of the inputs by further processing of the features or classify the inputs based on the features extracted by preceding

layers [51].

3.2.4 Softmax Unit

A softmax unit is usually used as the output of the fully connected layer. A softmax unit employs softmax function (normalized exponential) to represent the probability distribution of k classes. In fact, softmax function is a generalization of the sigmoid function, $\sigma(z) = \frac{1}{1+e^{-z}} = \frac{e^z}{1+e^z}$, which manifests the probability distribution of two possible classes [24]. Equation 3.5 displays the softmax function

$$\text{softmax}(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} \quad , \tag{3.5}$$

where $\vec{z}$ is the output of the k-way softmax unit, $\text{softmax}(\vec{z})_i$ is the probability that the input instance is in class i, and z_i is the ith element of the vector $\vec{z}$.

3.2.5 Rectified Linear Unit

The activation functions have been the linchpin of artificial neural networks (ANNs). That is, they incorporate nonlinearity into ANNs, which makes ANNs an effective tool to learn complex models. Sigmoid function $g(z) = \sigma(z) = \frac{1}{1+e^{-z}}$ and hyperbolic tangent function $g(z) = \tanh(z) = 2\sigma(2z) - 1$ are two popular activation functions, especially in traditional ANNs (see Figure 3.5).

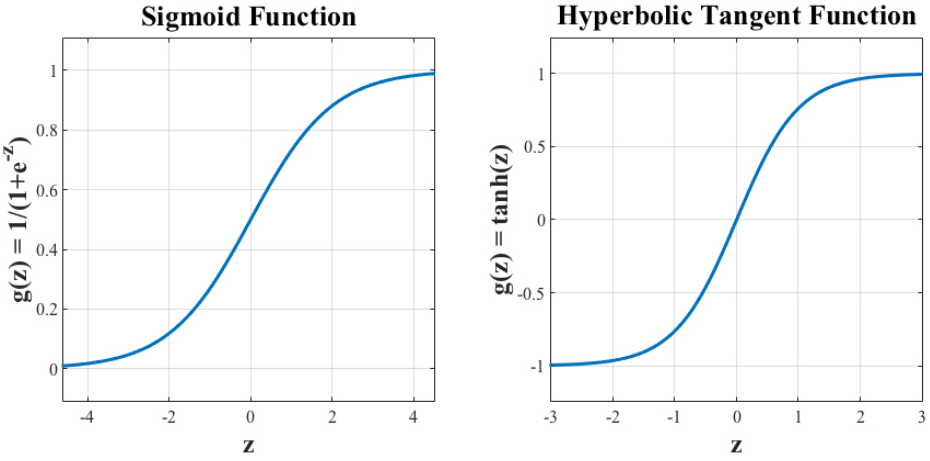


Figure 3.5: Sigmoid activation function and hyperbolic tangent activation function.

One disadvantage of these functions is that they saturate for zs that have high absolute values, which adversely affects the gradient-based learning. To overcome this shortcoming, the rectified linear unit (ReLU)—$g(z) = \max(z, 0)$—can be employed as an activation function [24]. Figure 3.6 shows the ReLU function. Despite its nonlinearity property around $z = 0$, it behaves linear for $z > 0$ and $z < 0$. ReLU functions provide a compromise between the nonlinearity, needed to learn complex models, and linearity, needed to facilitate gradient-based learning. As a result, ReLU functions are widely used as the activation function in deep learning models. In convolutional layers of CNNs, ReLU functions are applied on the feature maps before subsampling of the pooling layers. The main drawback of the ReLU functions is the zero output for the zs with negative value. This problem is known as the dying ReLUs [38]; that is, the output of the neurons becomes zero during the training and remains zeros. As a result, the neurons become ineffective. To eliminate this problem, a variant of ReLU function, named leaky ReLU has been introduced, $\text{ReLU}_\alpha(z) = \max(\alpha z, z)$, where α determines the slope of the ReLU function for $z < 0$ [52, 38]. Previous research has shown that using leaky ReLU may increase the likelihood of overfitting the training data for the small number of the training instances [38]. This is because the number of parameters will increase, which consequently increases the sensitivity of the network to the nuisance variances.

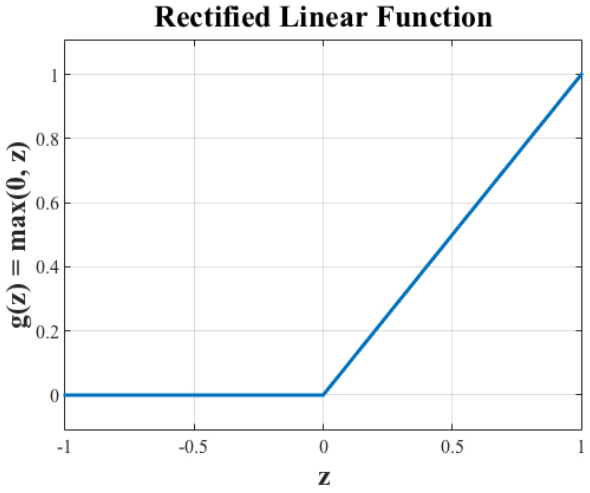


Figure 3.6: Rectified linear function.

3.2.6 Cross-entropy

The cross-entropy between labels, provided by training data, and the outputs, generated by softmax function, is used as a measure of the loss function. The cross entropy is

$$H(l, s) = H(l) + D_{KL}(l||s), \tag{3.6}$$

where l denotes the true probability distribution of the labels and s denotes the estimated probability distribution of the labels by softmax unit. $H(l)$ is the entropy of l and D_{KL} is the Kullback-Leibler divergence of s from l.

In communication theory, entropy is a measure of information that reflects the degree of uncertainty or choice in a system; that is, a greater randomness or uncertainty is associated with a larger degree of freedom of choice that indicates the greater information in a system [53]. The definition of entropy is

$$H = \sum_{x \in X} p(x) \log \frac{1}{p(x)}, \tag{3.7}$$

where X is a random variable with probability distribution of p and possible values of $\{x_1, x_2, \ldots, x_n\}$.

The Kullback-Leibler divergence measures how well the classifier estimates l by using s [54]. The more s diverges from l, the more the predictions are uncertain. The objective of learning is to set the parameters such that they minimize the cross-entropy or D_{KL}. The discrete entropy can be written as

$$H(l, s) = -\sum_{x} l(x) \log s(x). \tag{3.8}$$

In convolutional neural networks (CNNs) where softmax unit is used to estimate the probability distribution of the classes, the cross-entropy is considered a better loss function than the mean squared error (MSE) [24] mostly because MSE is more effective in the regression problems when the outputs have real continuous value.

3.2.7 Mini-batch Learning

The large number of training examples in deep learning models hampers the speed of learning and poses challenges for computing resources. The idea of using mini batches of examples from the training set seeks to ameliorate the problem of the enormity of the data [24]. That is, the network is trained over several batches instead of one batch of training data for each iteration. Specifically, the optimization problem occurs on several error subfunctions instead of one single error function. Mini-batch learning lies between two extremes of batch learning and stochastic learning. In batch learning, to which we alluded earlier, all training instances are used to train the model for each iteration. However, in stochastic learning, every instance is used separately to form an error function to train the model for each iteration. The primary benefit of stochastic and mini-batch learning is preserving time and computing resources. However, they never converge to the global optimum; instead, the stochastic or mini-batch algorithms either approach or fluctuate around the global optimum [42, 38]. The mini-batch learning is more accurate than stochastic learning since it takes a larger portion of training data into consideration while optimizing the error functions. Albeit, stochastic learning can escape local optima easier than mini-batch leaning [38]. The size of mini batches is usually set to 64, 128, 256, 512, or 1024 in deep leaning models such as convolutional neural networks (CNNs) [42].

3.2.8 Dropout

Variance and bias are two competing properties of gradient-based learning [37]. The estimated underlying model of a system can be changed depending on the training set used for learning. Variance is a concept that refers to these changes. Ideally, the goal is to have a model that is a general representative of the system behavior and, hence, it is less sensitive to the the specific training set. Models with high variance follow the noise-induced changes in training set and overfit the training data. Generally, complex models suffer from high variance. On the other

hand, the models built over the assumptions that simplify the real behavior of the system have high bias. These models fail to capture the underlying complexity of the system and underfit the training data.

There is a high risk of overfitting the training set (high variance) in deep learning models due to the excessive amount of training data and model parameters of deep neural networks. In traditional machine learning, there is a trade-off between variance and bias; that is, decreasing variance leads to increasing bias and vice versa. However, training deep architectures, along with big training sets, has resolved this long-standing variance-bias dilemma. So long as the deep networks are properly regularized, the variance can be reduced without hurting the bias [42].

Models with high variance perform well on the training set, but they are unsuccessful to generalize well on the test set. $L1$-norm regularization and $L2$-norm regularization are two common traditional approaches to remedy overfitting. That is, the objective function incorporates weight decay into the the loss function as shown below for $L1$-norm regularization and $L2$-norm regularization [42, 24], respectively

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} \mathscr{L}(l^{(i)}, s^{(i)}) + \frac{\lambda}{2m} \|w\|_1 , \tag{3.9}$$

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} \mathscr{L}(l^{(i)}, s^{(i)}) + \frac{\lambda}{2m} \|w\|_2^2 , \tag{3.10}$$

where m is the number of training data, $\mathscr{L}$ denoted the loss function such as cross-entropy or mean squared error (MSE), $l^{(i)}$ is the true label of the ith training instance, $s^{(i)}$ is the predicted label of the ith training instance, w denotes the parameters of interest, and λ is a hyperparmeter that controls the strength of the regularization. Incorporating the regularization term constraints the optimization to the small values of w. This may yield weight assignment to a smaller number of features, which consequently decreases the complexity of the model.

Dropout is an algorithm that is widely used in deep learning models as a

powerful regularization technique to reduce overfitting. Dropout is a strong and highly effective algorithm that regulates the deep neural networks by randomly omitting the neurons in the hidden layers during training. The neurons are omitted with an assigned probability, say, p [55]. The outputs and the error function of the networks can be cast as random variables. In other words, dropout approximates bagged ensemble training where training data is used to train multiple networks [24]. Therefore, the output of the network with dropout can be viewed as an average ensemble of multiple networks with different architectures [38]. Baldi and Sadowski [56] have proven that the expected value of the error function of the network with dropout includes a regularization term similar to weight decay in $L1$-norm and $L2$-norm regularization. Further, they have argued that the highest level of regularization can be achieved when the hidden neurons are omitted with the probability of $p = 0.5$.

3.2.9 Data Augmentation

Deep learning models strive for data because their actual power is magnified when they are trained with large data sets [57]. In fact, increasing the size of training set has been an effective way to fight overfitting and to improve generalizability of deep learning models [24]. However, acquiring new data is an expensive, time-consuming endeavor. To tackle the data collection problem, data augmentation—which is a regularization technique—is employed to artificially synthesize new training data and increase the size of training sets [38]. Recent years have witnessed a major success of data augmentation in several machine learning tasks, especially in classification [24, 58, 59, 60, 43, 61]. In supervised learning, data augmentation includes transformation techniques to which the classifier of interest is invariant. That is, the class label of data is not affected by data augmentation. For instance, rotation of images for object recognition task or embedding speech signals in background noise for speech emotion recognition task effectively boosts the size of training sets without vitiating the labels of instances.

3.3 k-Fold Cross-Validation

Cross-validation is a model assessment technique that is used to evaluate the performance of machine learning models. In this technique, the data is randomly divided into k distinct non-overlapping subsets, or folds with approximately equal sizes. The first fold is used as the validation set to evaluate the performance of the model whereas the remaining $k-1$ folds are used as the training set to train the model. This procedure is repeated k times wherein the kth fold is treated as the validation set and the remainder of the folds are treated as the training test for each kth trial. Ultimately, the average of performance accuracy across these k test sets is used to estimate the performance of the model [37, 38, 24].

Chapter 4

Experimental Setup

In the current study, we implemented convolutional neural networks (CNNs) to classify speech utterances based on their emotional contents. In addition to three widely used benchmarks for recognition of emotion from speech utterances [62, 63, 64], we used a private database to train and evaluate our models. We used TensorFlow (an open-source library written in Python and `C++` [65]) as the programming framework to implement our CNN models. This chapter describes the experimental setup of the current work. The first section introduces the databases administered in our study (summarized in Table 4.1). The second section explains the preprocessing procedure. The third section describes the training and test setup used to train and evaluate our models. Finally, this chapter ends by introducing the baseline architecture of the CNNs implemented in the current work.

4.1 Databases

4.1.1 EMODB: German Database

The Berlin Database of Emotional Speech (EMODB)[1] is a public German speech database that incorporates audio files with seven emotions: happiness, sadness, anger, fear, disgust, boredom, and neutral [62]. The German utterances were recorded by five men and five women (9 of them had passed an acting schooling) producing 10 utterances for each emotion. Twenty listeners evaluated the emotional labels of the utterances. The sentences had emotionally neutral contents and were recorded with a sampling rate of 48 kHz and downsampled to 16 kHz.

[1] `http://emodb.bilderbar.info/start.html`

The sentences with recognition accuracy of greater than 60% were chosen for further analysis. This lead to 71 happy, 62 sad, 127 angry, 69 scared, 46 disgusted, 81 bored, and 79 neutral speech utterances for a total of 535 speech samples.

4.1.2 SAVEE: British English Database

The Surrey Audio-Visual Expressed Emotion (SAVEE)[2] is a public British English speech database that has audio files with seven emotion labels: happiness, sadness, anger, fear, disgust, surprise, and neutral [63]. Four English male actors generated 15 utterances for each emotion. Three of these utterances were common among all emotions. Two of them were emotion specific. The remaining utterances were generic sentences that were different across six emotions but neutral. The three common and two emotion specific sentences of each emotion were used to record neutral emotions. The sampling rate of all recordings was 44.1 kHz. Overall, there are 60 utterances for each emotion except neutral and 120 utterances for neutral for a total of 480 utterances. We randomly selected 60 utterances from 120 neutral utterance. Therefore, we used 420 utterances of this database.

4.1.3 EMOVO: Italian Database

EMOVO[3] is a public Italian speech database that includes audio files with seven emotion labels: happiness, sadness, anger, fear, disgust, surprise, and neutral [64]. Six Italian actors (three female and three male) generated 14 utterances for each emotion. The sentences were either emotionally neutral or nonsense and were consistent across emotions. All sentences were recorded with sampling rate of 48 kHz. There are 84 utterances for each emotion for a total of 588 utterances.

[2] http://personal.ee.surrey.ac.uk/Personal/P.Jackson/SAVEE/
[3] http://voice.fub.it/activities/corpora/emovo/index.html

4.1.4 BTNRH: American English Database

BTNRH is a private American English speech database that was developed in Auditory Prosthesis and Perception Laboratory at Boys Town National Research Hospital, furnished by Dr. Monita Chatterjee (personal communication). It has audio files with five emotional states: happiness, sadness, anger, fear, and neutral. We used utterances generated by 12 American speakers (seven female and five male) in our experiments. The speakers uttered 12 sentences with each of the 5 emotion states. We used 144 utterances for each emotion for a total of 720 utterances. The sentences were emotionally neutral and were recorded with sampling rate of 44.1 kHz.

4.1.5 All-Inclusive: Language-Independent Database

All databases were integrated to form a language-independent database across German, Italian, British English, and American English languages. In doing so, we first identified the shared emotion classes across all languages. All databases had happy, sad, angry, scared, and neutral speech utterances. Therefore, the bored and disgusted speech instances were removed from the EMODB database, and the surprised and disgusted speech examples were removed from the SAVEE and EMODB databases while the speech sounds with joint emotion classes were preserved. This led to the all-inclusive database with 5 emotional states.

Table 4.1: A summary of the databases used in the current study—the number of utterances for each emotional state and the total number of utterances.

Database	Happy	Sad	Angry	Scared	Neutral	Disgusted	Surprised	Bored	Total
EMODB	71	62	127	69	79	46	-	81	535
SAVEE	60	60	60	60	60	60	60	-	420
EMOVO	84	84	84	84	84	84	84	-	588
BTNRH	144	144	144	144	144	-	-	-	720
All-Inclusive	359	350	415	357	367	-	-	-	1848

4.2 Preprocessing

In order to have a consistent sampling rate across all databases, all utterances were resampled and filtered by an antialiasing FIR lowpass filter to have frequency rate of 16 kHz prior to any processing. All audio utterances were then converted into spectrograms. A spectrogram is an image that displays the variation of energy at different frequencies across time. The vertical axis (ordinate) represents frequency and the horizontal axis (abscissa) represents time. The energy or intensity is encoded either by the level of darkness or by the colors. There are two general types of spectrograms: wide-band spectrograms and narrow-band spectrograms. Wide-band spectrograms has a higher time resolution than narrow-band spectrograms. This property enables the wide-band spectrograms to show individual glottal pulses. In contrast, narrow-band spectrograms have higher frequency resolution than wide-band spectrograms. This feature enables the narrow-band spectrograms to resolve individual harmonics [66, 67]. Figure 4.1 depicts the wide-band and narrow-band spectrogram images of a speech utterance.

Considering the importance of vocal fold vibration, along with the fact that glottal pulse is associated with one period of vocal fold vibration [66], we decided to convert all utterances into wide-band spectrograms. In doing so, the length of Hamming windows were set to 5 ms with 4.4 ms overlap. The number of DFT points was set to 512. Also, we discarded the frequency information greater than 4 kHz from spectrograms since frequencies below 4000 Hz are sufficient for speech perception in many situations [68]. In pilot studies, eliminating energy above 4000 Hz improved the performance of the algorithms. This gave 129 frequency points. All spectrogram images were, first, resized to have 129×129 pixels and, then, z-normalized to have zero mean and standard deviation close to one.

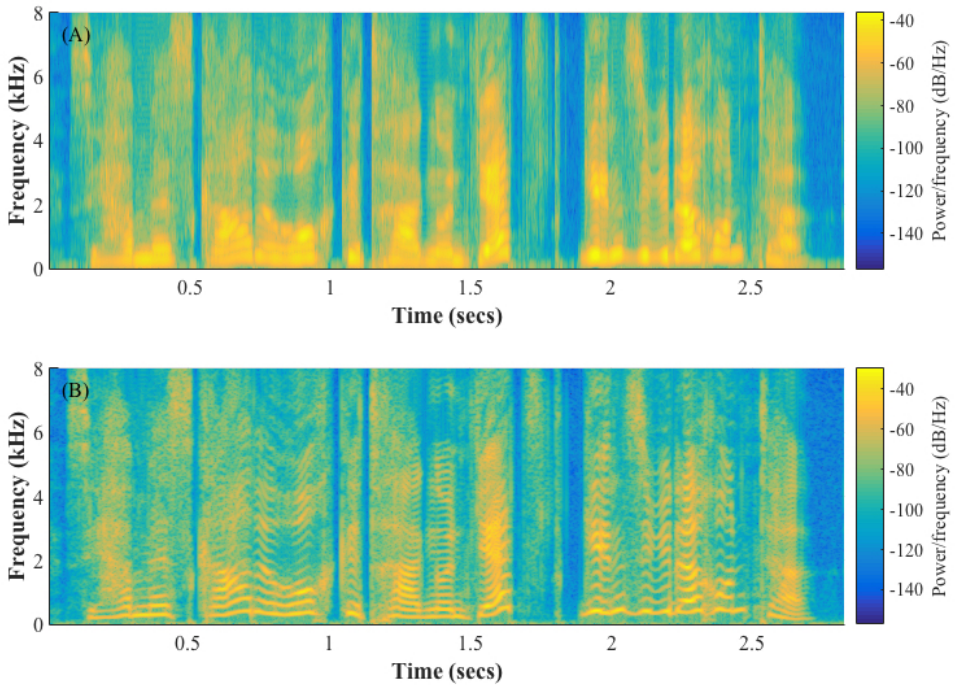


Figure 4.1: (A) Wide-band spectrogram with 5 ms Hamming window; (B) narrow-band spectrogram with 25 ms Hamming window.

4.3 Training and Test Sets

Our models were trained and evaluated using 5-fold cross-validation. That is, the data were partitioned into 5 folds. The first fold was used as a test set whereas the other folds were used to train our models. Then, the second fold was used to test our models while the remainder of the folds were used for training, and so on. To reduce overfitting and the adverse effect of small size of databases, the data sets were augmented by adding white Gaussian noise with +15 signal to noise ratio (SNR) to each audio signal either 10 times or 20 times. The SNR is defined as $10 \log_{10}(\frac{P_{speech}}{P_{noise}})$, where P is the average power of the signal. The data augmentation resulted in two types of data sets used to train our models: data sets with 10 times augmentation (10x) and data sets with 20 times augmentation (20x). We used the original data without noise to test our models. The augmented data were used only for training. Finally, the labels of the training and the test data were encoded as one-hot vectors. Table 4.2 shows the class labels of each

database. The number of training epochs was varied between 100 to 4000. The favorable training epoch was set to 100 due to the computation and time expenses.

Table 4.2: Class labels of each database.

	Stimulus Type			
Emotion	EMODB	SAVEE	EMOVO	BTNRH
happy	1	1	1	1
sad	2	2	2	2
angry	3	3	3	3
scared	4	4	4	4
bored	5	-	-	-
surprised	-	5	5	-
disgusted	6	6	6	-
neutral	7	7	7	5

4.4 Architecture

The baseline architecture of the deep neural network that was implemented in the current study was a convolutional neural network with two convolutional layers and one fully connected layer with 1024 hidden neurons. Depending on the number of classes, either a 5-way or a 7-way softmax unit was used to estimate the probability distribution of the classes. Every convolutional layer was followed by either a max-pooling or average-pooling layer. Rectified Linear Units (ReLU) were used in convolutional and fully connected layers as activation functions to introduce nonlinearity to the model. The initial kernel size of convolutional layers was set to 5×5 with stride of 1. The initial number of kernels was set to 8 and 16 for the first and the second convolutional layers, respectively. The kernel size of pooling layers was set to 2×2 with stride of 2. Cross-entropy was used as the loss function and the Adam optimizer was employed to minimize the loss function over the mini batches of the training data. The size of the mini batches was set to 512. The number of training iteration was 100. The networks developed in this study took between 30 minutes to 2 days to be trained using Graphics Processing Units (GPUs). Broadly speaking, GPUs are used instead of CPUs to accelerate the speed

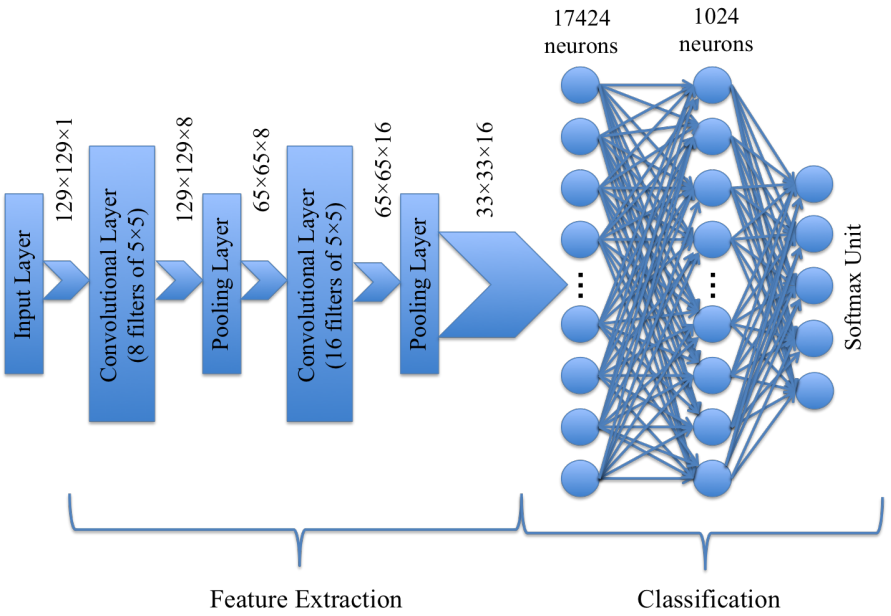


Figure 4.2: The baseline architecture of the CNN used in the current study to classify speech utterances based on their emotional states.

of computation since GPUs have several cores and can handle a large number of concurrent threads. We used the Crane cluster of the Holland Computing Center at University of Nebraska-Lincoln to run our experiments. Further, we incorporated the dropout algorithm into the fully connected layer to improve the performance of our networks whenever the symptoms of overfitting were diagnosed. Figure 4.2 illustrates the fundamental building blocks of the CNN model in the current work.

Chapter 5

Results & Discussion

We ran several language-dependent, gender-independent experiments on each database. We embarked on our study by implementing the baseline CNN architecture introduced in Chapter 4. Subsequently, we modified the hyperparameters such as the size of convolution kernels and the deletion probability of the dropout algorithm hing on the performance of the models. This chapter aims to present the results of these experiments and to discuss the outcomes.

5.1 Results

5.1.1 EMODB: German Database

We ran several experiments on the EMODB database. In all experiments, our networks learned the training data with accuracy 100%. However, the accuracy on the test data was varied across different architectures. Table 5.1 summarizes the architectures and their corresponding accuracy on the test data. The accuracy on the test data is the average of accuracy of the test data across the 5 folds of the cross-validation evaluation. Figure 5.1 demonstrates the accuracy of the CNN models over training and test for each iteration of learning.

As demonstrated, the first significant boost in the performance of our network occurred when dropout was employed. Applying dropout increased the test accuracy from 58.62% to 78.01%. Increasing the window size of the first convolutional layer from 5×5 to 10×10 also improved the performance of the network. Further, using average pooling instead of max pooling enhanced the performance. The second significant boost occurred by increasing the number of augmentation

Table 5.1: The summary of the architectures and the results of the experiments ran on the EMODB database; $f1$ is the size of the kernels in the first convolutional layer, $f2$ is the size of the kernels in the second convolutional layer, $p_{dropout}$ is the deletion probability of dropout, epoch denotes the number of training iterations, and CV stands for cross-validation.

f_1	f_2	pooling	$p_{dropout}$	augmentation	epoch	CV accuracy (%)
5×5	5×5	Max	0	10x	100	58.62
5×5	5×5	Max	0.5	10x	100	78.01
10×10	5×5	Max	0.5	10x	100	85.29
10×10	5×5	Average	0.5	10x	100	87.58
5×5	5×5	Max	0.5	20x	100	95.84
10×10	5×5	Max	0.5	20x	100	96.61
10×10	**5×5**	**Average**	**0.5**	**20x**	**100**	**96.78**
10×10	**5×5**	**Average**	**0.5**	**20x**	**800**	**99.5**
10×10	**5×5**	**Average**	**0.5**	**20x**	**4000**	**99.83**

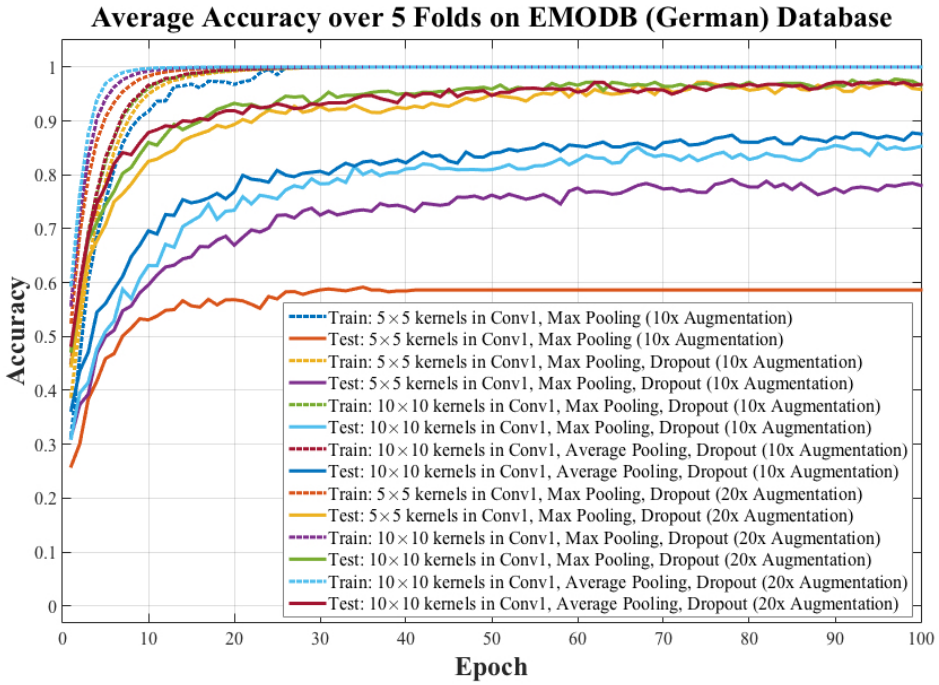


Figure 5.1: The performance accuracy of the CNN models that were trained and tested on the EMODB database.

from 10 times augmentation to 20 times augmentation. This increase yielded a test accuracy of 95.84%. Increasing the window size of the first convolutional layer enhanced the performance from 95.84% to 96.61%. Using average pooling in lieu of max pooling changed the performance from 96.61% to 96.78%. Taken together, the highest performance was associated with the architecture with 10×10 kernels

in the first convolutional layer, 5×5 kernels in the second convolutional layer, average pooling, dropout with $p = 0.5$, and training data with 20x augmentation.

Figure 5.2 illustrates the color-map confusion matrix for the architecture with the highest accuracy after 100 training epochs. Table 5.2 shows the corresponding numerical values of this color map. As can be seen, the network classified unseen angry utterances with 100% accuracy. Bored and disgusted emotions were classified with the poorest accuracy compared with other emotions.

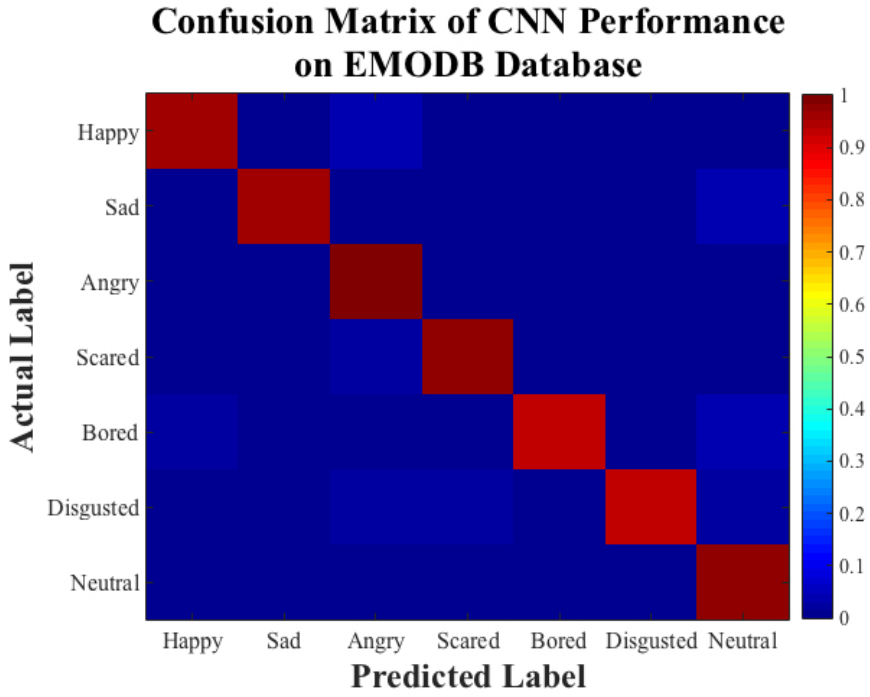


Figure 5.2: The color-map confusion matrix of the CNN with the highest performance and 100 training epochs on the EMODB database.

Moreover, the architecture with the highest performance, as given in Table 5.1 (10×10 kernels in the first convolutional layer, 5×5 kernels in the second convolutional layer, average pooling, dropout with $p = 0.5$), was trained and tested with 800 and 4000 training epochs. The results indicated an improvement in the test accuracy as the number of training epochs increased. Figures 5.3 and 5.4 illustrates the color-map confusion matrices of the CNN with 800 and 4000 training epochs, respectively. Tables 5.3 and 5.4 show the corresponding numerical values of these color maps. As demonstrated, increasing the number of training improved the performance of the model on the test data.

Table 5.2: The numerical confusion matrix of the CNN with the highest performance and 100 training epochs on the EMODB database. The diagonal numbers show the percent of each class that was correctly identified. The off-diagonal numbers displays the percent of each class that was incorrectly identified as other classes.

		Predicted Labels						
		happy	sad	angry	scared	bored	disgusted	neutral
Actual Labels	happy	**95.77**	0	4.23	0	0	0	0
	sad	0	**96.77**	0	0	0	0	3.23
	angry	0	0	**100**	0	0	0	0
	scared	0	0	2.9	**97.10**	0	0	0
	bored	2.47	0	0	0	**92.59**	1.23	3.70
	disgusted	0	0	2.17	2.17	0	**93.48**	2.17
	neutral	0	0	0	1.27	1.27	0	**97.47**

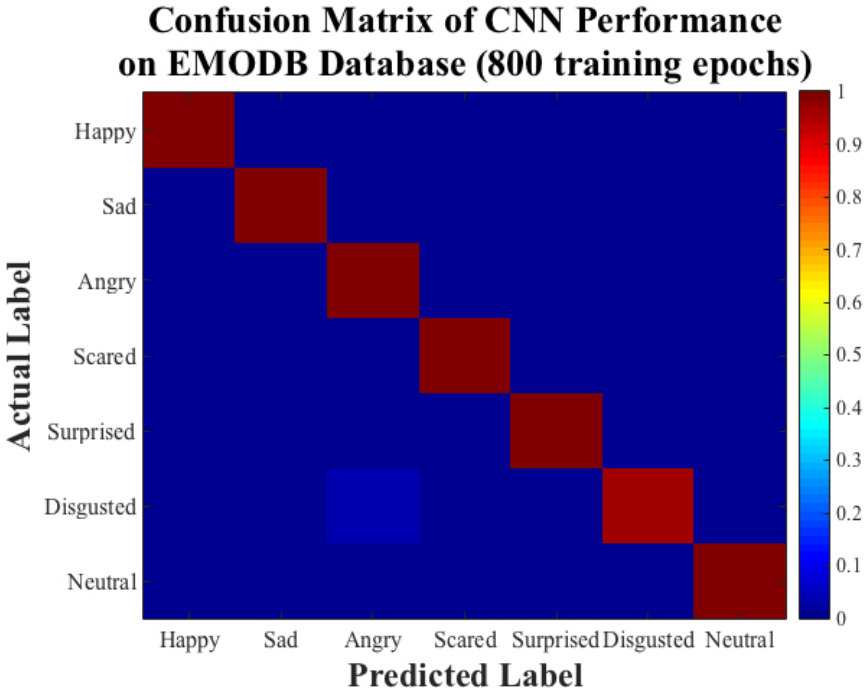


Figure 5.3: The color-map confusion matrix of the CNN with the highest performance and 800 training epochs on the EMODB database.

Further, to underline the effect of data augmentation on the performance of the CNN model, the CNN architecture with the highest performance (10×10 kernels in the first convolutional layer, 5×5 kernels in the second convolutional layer, average pooling, dropout with $p = 0.5$) was trained on the original EMODB database and the augmented (20x) EMODB database, separately. Figure 5.5 depicts the accuracy of the CNN models on the test data. As demonstrated, the CNN that

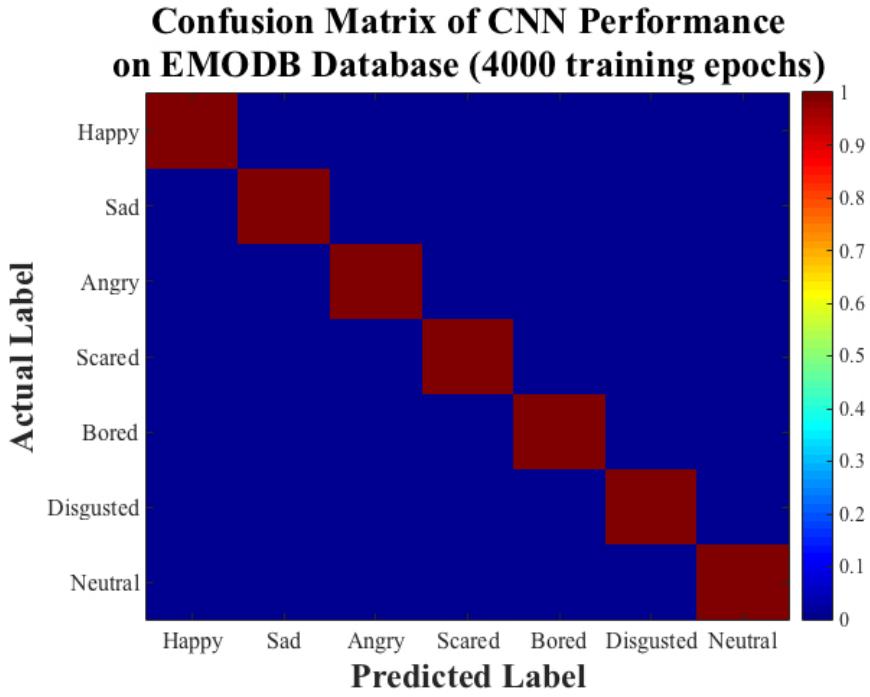


Figure 5.4: The color-map confusion matrix of the CNN with the highest performance and 4000 training epochs on the EMODB database.

Table 5.3: The numerical confusion matrix of the CNN with the highest performance and 800 training epochs on the EMODB database. The diagonal numbers show the percent of each class that was correctly identified. The off-diagonal numbers displays the percent of each class that was incorrectly identified as other classes.

		Predicted Labels						
		happy	sad	angry	scared	bored	disgusted	neutral
Actual Labels	happy	**98.59**	0	1.41	0	0	0	0
	sad	0	**100**	0	0	0	0	0
	angry	0	0	**100**	0	0	0	0
	scared	0	0	0	**100**	0	0	0
	bored	0	0	0	0	**100**	0	0
	disgusted	0	0	4.35	0	0	**95.65**	0
	neutral	0	0	0	0	0	0	**1**

was trained on the original database classified the test data with accuracy of approximately less than 55% whereas the CNN that was trained on the augmented database classified the same test data with accuracy close to 99.5%. This highlights the crucial role of data augmentation in enhancing the CNN performance.

Table 5.4: The numerical confusion matrix of the CNN with the highest performance and 4000 training epochs on the EMODB database. The diagonal numbers show the percent of each class that was correctly identified. The off-diagonal numbers displays the percent of each class that was incorrectly identified as other classes.

		Predicted Labels						
		happy	sad	angry	scared	bored	disgusted	neutral
Actual Labels	happy	**100**	0	0	0	0	0	0
	sad	0	**100**	0	0	0	0	0
	angry	0.79	0	**99.21**	0	0	0	0
	scared	0	0	0	**100**	0	0	0
	bored	0	0	0	0	**100**	0	0
	disgusted	0	0	0	0	0	**100**	0
	neutral	0	0	0	0	0	0	**100**

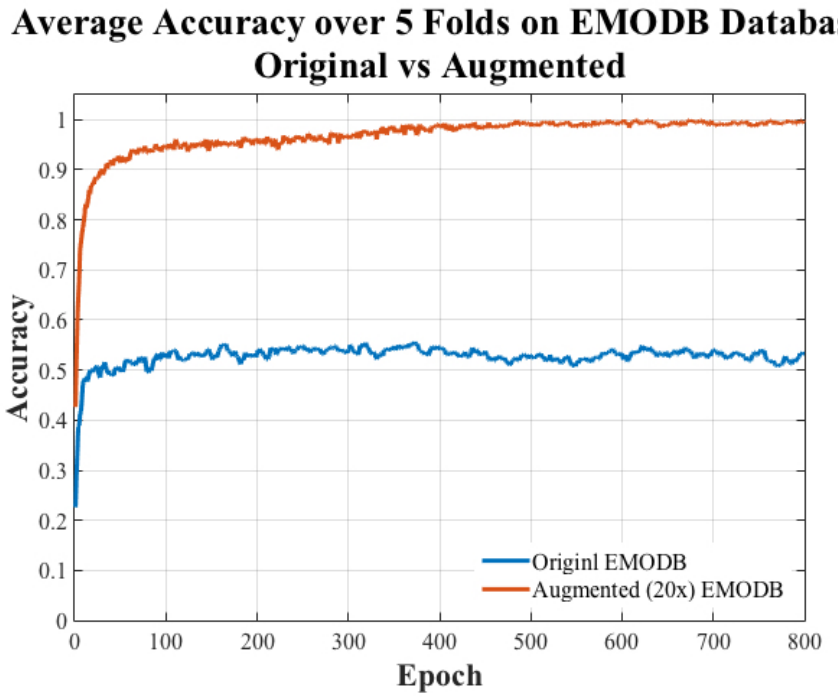


Figure 5.5: A comparison between the CNN performance on the original EMODB database and the augmented EMODB database.

5.1.2 SAVEE: British English Database

Table 5.5 summarizes the experiments performed on the SAVEE database. The outcomes reveal that the network with 11×11 kernels in the first convolutional layer, 5×5 kernels in the second convolutional layer, average pooling, dropout with $p = 0.5$, and 20 times augmentation of the training data had the best performance

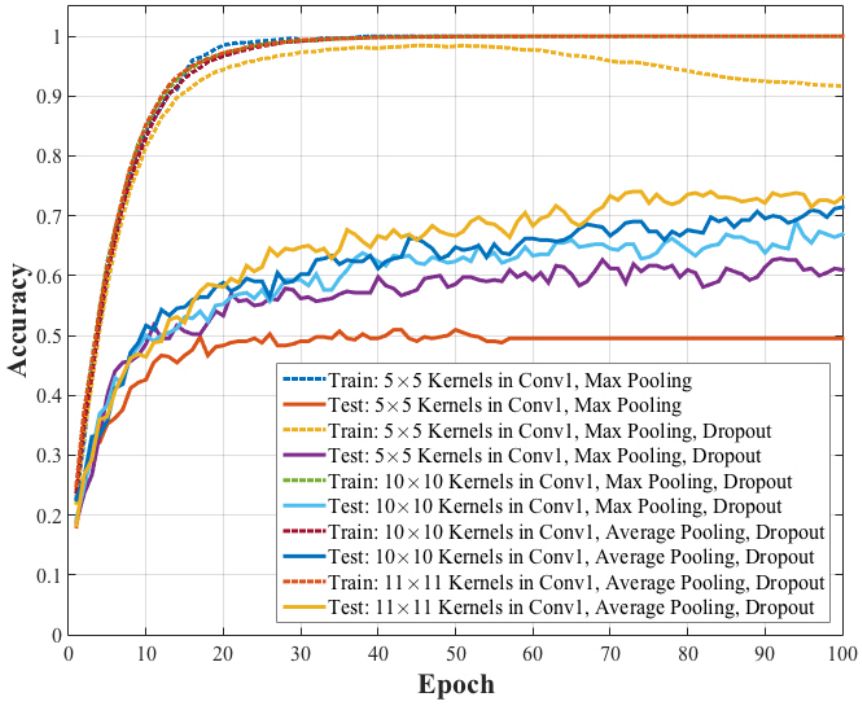
on this database. Figure 5.6 demonstrates the average performance accuracy of the CNNs over 5 folds of cross-validation on the data with 10 times augmentation (image (a)) and 20 times augmentation (image (b)) across the learning iterations. As can be seen, our networks with various architectures could learn the training data. However, the performance of the networks on the test data varies across different architectures. Similar to the EMODB database, integrating dropout into the architecture boosted the performance of the network. Further, increasing the size of kernel's window in the first convolutional layer, together with average pooling, enhanced the network performance, especially when the training data were augmented 20 times.

Table 5.5: The summary of the architectures and the results of the experiments ran on the SAVEE database; $f1$ is the size of the kernels in the first convolutional layer, $f2$ is the size of the kernels in the second convolutional layer, $p_{dropout}$ is the deletion probability of dropout, epoch denotes the number of training iterations, and CV stands for cross-validation.

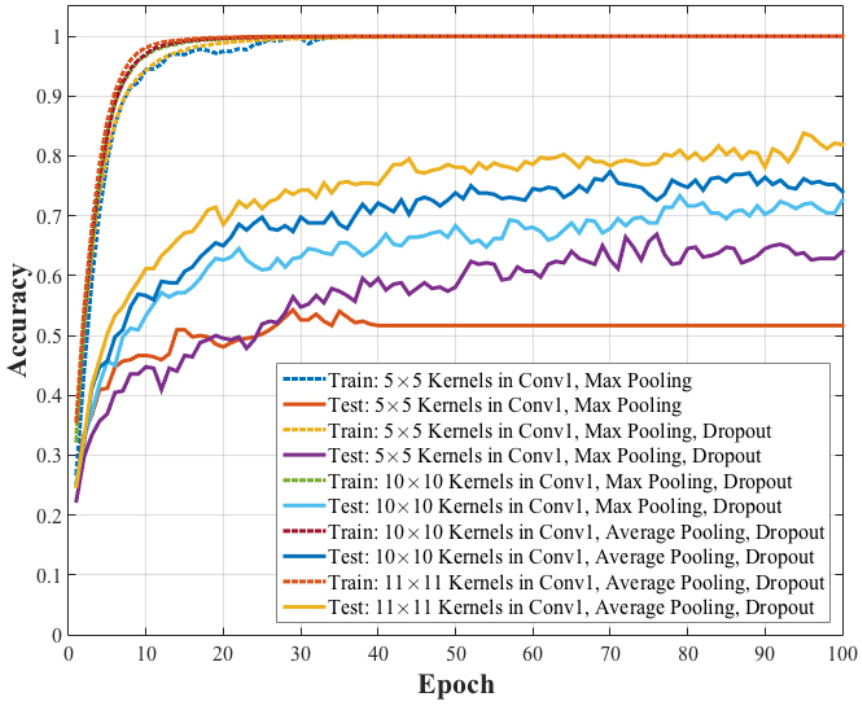
f_1	f_2	pooling	$p_{dropout}$	augmentation	epoch	CV accuracy (%)
5×5	5×5	Max	0	10x	100	49.5
5×5	5×5	Max	0.5	10x	100	60.9
10×10	5×5	Max	0.5	10x	100	66.9
10×10	5×5	Average	0.5	10x	100	71.4
11×11	5×5	Average	0.5	10x	100	73.1
5×5	5×5	Max	0	20x	100	51.67
5×5	5×5	Max	0.5	20x	100	64.05
10×10	5×5	Max	0.5	20x	100	72.62
10×10	5×5	Average	0.5	20x	100	74.05
11×11	**5×5**	**Average**	**0.5**	**20x**	**100**	**81.9**
11×11	**5×5**	**Average**	**0.5**	**20x**	**800**	**95.48**
11×11	**5×5**	**Average**	**0.5**	**20x**	**4000**	**98.33**

Figure 5.7 illustrates the color-map confusion matrix associated with the architecture that had the best performance on the SAVEE database with 100 training epochs. Table 5.6 displays the corresponding numerical values of the color map. The network classified the disgusted and neutral utterances with the highest accuracy of 90% whereas it classified the scared and surprised utterances with the lowest accuracy of 70%.

Similar to the EMODB database, increasing the number of training epochs



(a) 10x Augmentation



(b) 20x Augmentation

Figure 5.6: The average performance accuracy of the CNN models over 5 folds trained and tested on the SAVEE database.

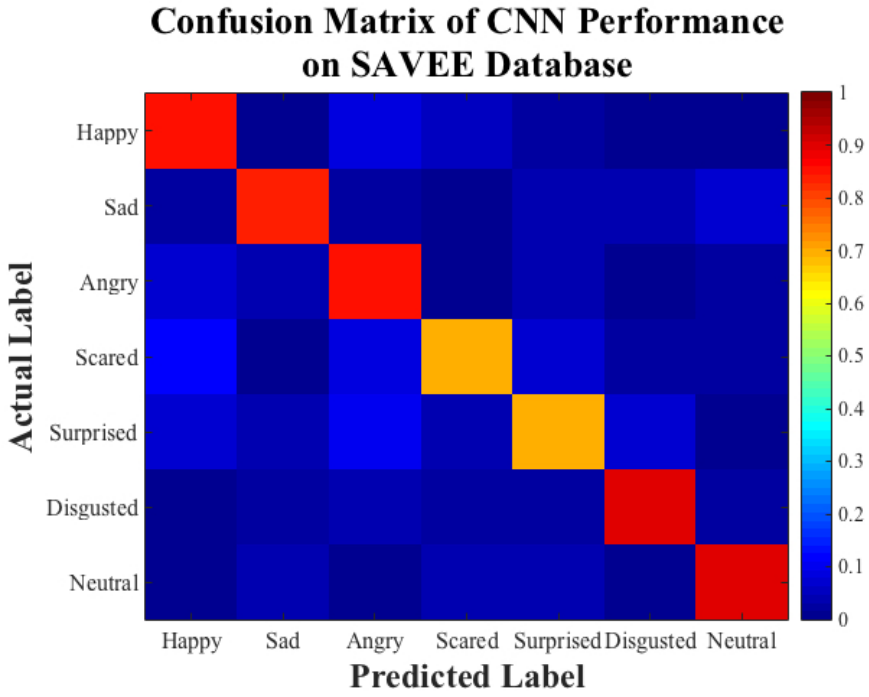


Figure 5.7: The color-map confusion matrix of the CNN with the highest performance and 100 training epochs on the SAVEE database.

Table 5.6: The numerical confusion matrix of the CNN with the highest performance and 100 training epochs on the SAVEE database. The diagonal numbers show the percent of each class that was correctly identified. The off-diagonal numbers displays the percent of each class that was incorrectly identified as other classes.

		Predicted Labels						
		happy	sad	angry	scared	surprised	disgusted	neutral
Actual Labels	happy	**85**	0	8.33	5	1.67	0	0
	sad	1.67	**83.33**	1.67	0	3.33	3.33	6.67
	angry	6.67	3.33	**85**	0	3.33	0	1.67
	scared	1.167	0	8.33	**70**	6.67	1.67	1.67
	surprised	6.67	3.33	10	3.33	**70**	6.67	0
	disgusted	0	1.67	3.33	1.67	1.67	**90**	1.67
	neutral	0	3.33	0	3.33	3.33	0	**90**

enhanced the performance of the CNN on the test data. Figures 5.8 and 5.9 depict the color-map confusion matrices associated with the architecture that had the best performance on the SAVEE database with 800 and 4000 training epochs, respectively. Tables 5.7 and 5.8 show the corresponding numerical values of these color maps.

Further, to assess the effect of data augmentation, the CNN architecture with

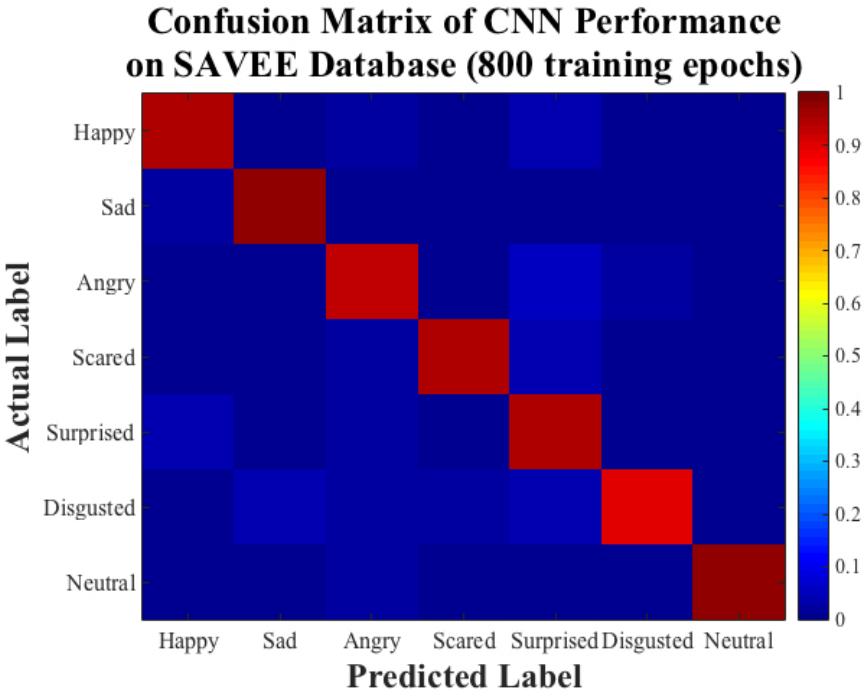


Figure 5.8: The color-map confusion matrix of the CNN with the highest performance and 800 training epochs on the SAVEE database.

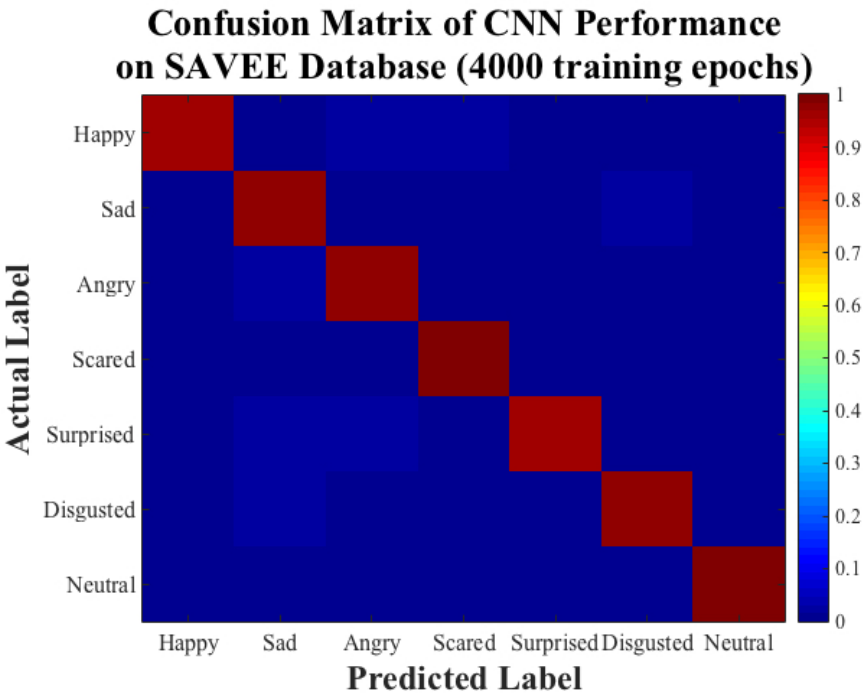


Figure 5.9: The color-map confusion matrix of the CNN with the highest performance and 4000 training epochs on the SAVEE database.

the highest performance, as shown in Table 5.6 (11×11 kernels in the first convolutional layer, 5×5 kernels in the second convolutional layer, average pooling, dropout with $p = 0.5$), was trained on the original SAVEE database and the aug-

Table 5.7: The numerical confusion matrix of the CNN with the highest performance and 800 training epochs on the SAVEE database. The diagonal numbers show the percent of each class that was correctly identified. The off-diagonal numbers displays the percent of each class that was incorrectly identified as other classes.

		Predicted Labels						
		happy	sad	angry	scared	surprised	disgusted	neutral
Actual Labels	happy	**95**	0	1.67	0	3.33	0	0
	sad	1.67	**98.33**	0	0	0	0	0
	angry	0	0	**93.33**	0	5	1.67	0
	scared	0	0	1.67	**95**	3.33	0	0
	surprised	3.33	0	1.67	0	**95**	0	0
	disgusted	0	3.33	1.67	1.67	3.33	**90**	0
	neutral	0	0	1.67	0	0	0	**98.33**

Table 5.8: The numerical confusion matrix of the CNN with the highest performance and 4000 training epochs on the SAVEE database. The diagonal numbers show the percent of each class that was correctly identified. The off-diagonal numbers displays the percent of each class that was incorrectly identified as other classes.

		Predicted Labels						
		happy	sad	angry	scared	surprised	disgusted	neutral
Actual Labels	happy	**96.67**	0	1.67	1.67	3.33	0	0
	sad	0	**98.33**	0	0	0	1.67	0
	angry	0	1.67	**98.33**	0	0	0	0
	scared	0	0	0	**100**	0	0	0
	surprised	0	1.67	1.67	0	**96.67**	0	0
	disgusted	0	1.67	0	0	0	**98.33**	0
	neutral	0	0	0	0	0	0	**100**

mented (20x) SAVEE database separately. Figure 5.10 displays the accuracy of the CNN models on the test data. As can be seen, the CNN that was trained on the original database classified the test data with accuracy approximately less than 40% whereas the CNN that was trained on the augmented database classified the same test data with accuracy of around 95%. Similar to the EMODB database, the augmentation significantly enhanced the performance of the CNN model on the test data.

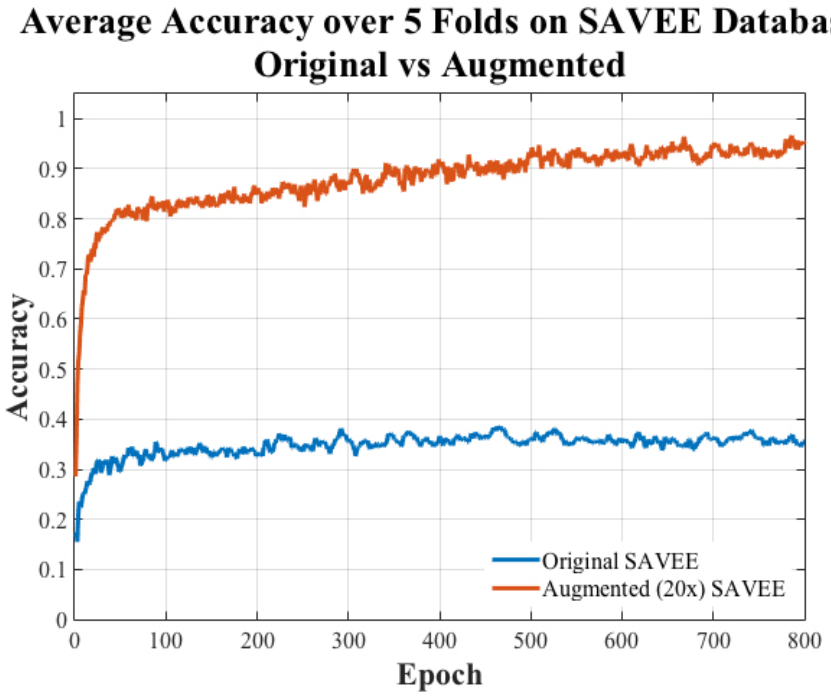


Figure 5.10: A comparison between the CNN performance on the original SAVEE database and the augmented SAVEE database.

5.1.3 EMOVO: Italian Database

Table 5.9 summarizes the architectures and the average accuracy of the networks implemented using the EMOVO database. Figures 5.11a and 5.11b demonstrate the average accuracy of these networks over 5 folds of cross-validation for the data with 10 times augmentation and the data with 20 times augmentation, respectively. As can be seen, all networks performed well on the training data regardless of the level of data augmentation. However, the performance of the networks on the test data varied depending on the level of data augmentation. Generally, the networks trained on the data with 20 times augmentation had a better generalization accuracy. The best accuracy achieved by the architecture with 10×10 kernels, max-pooling layer, and dropout with $p = 0.5$ trained on the data with 20 times augmentation.

Figure 5.12 and Table 5.10 illustrate the color-map and numerical confusion matrices associated with the architecture that outperformed other architectures trained on the EMOVO database with 100 training epochs. As displayed, our

Table 5.9: The summary of the architectures and the results of the experiments ran on the EMOVO database; $f1$ is the size of the kernels in the first convolutional layer, $f2$ is the size of the kernels in the second convolutional layer, $p_{dropout}$ is the deletion probability of dropout, epoch denotes the number of training iterations, and CV stands for cross-validation.

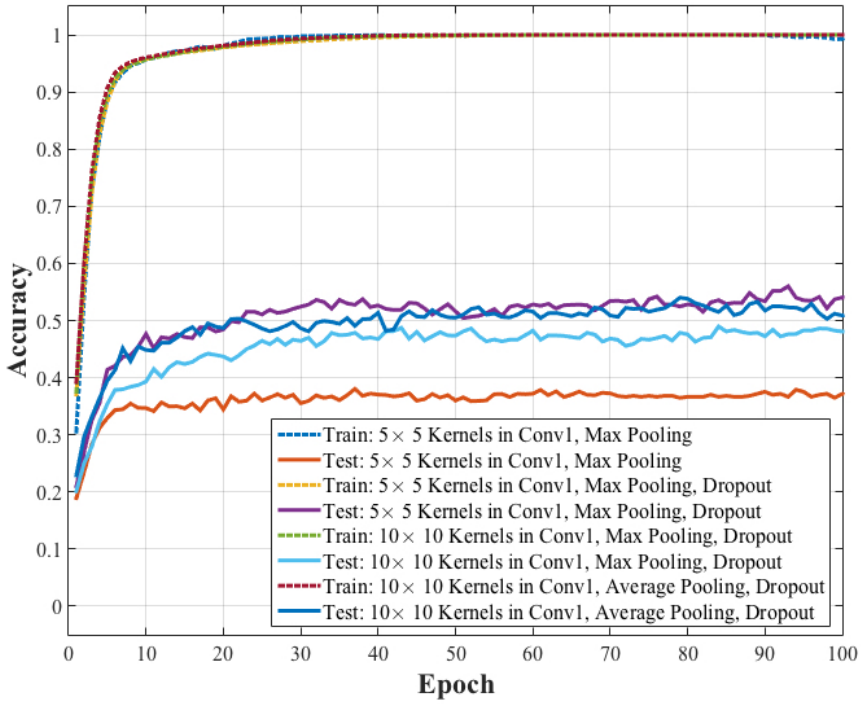
f_1	f_2	pooling	$p_{dropout}$	augmentation	epoch	CV accuracy (%)
5×5	5×5	Max	0	10x	100	37.18
5×5	5×5	Max	0.5	10x	100	54.04
10×10	5×5	Max	0.5	10x	100	48.07
10×10	5×5	Average	0.5	10x	100	50.86
5×5	5×5	Max	0	20x	100	58.21
5×5	5×5	Max	0.5	20x	100	73.64
10×10	**5×5**	**Max**	**0.5**	**20x**	**100**	**81.07**
10×10	5×5	Average	0.5	20x	100	78.04
10×10	**5×5**	**Max**	**0.5**	**20x**	**800**	**91.61**
10×10	**5×5**	**Max**	**0.5**	**20x**	**4000**	**97.68**

model classified the angry speech utterances with the highest accuracy of 90.48% compared with other emotions.

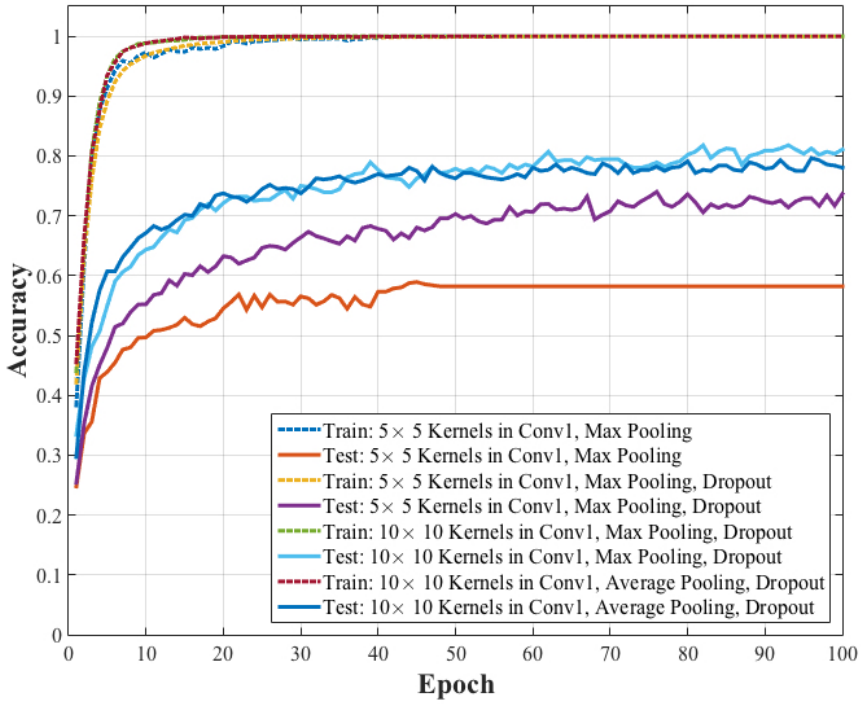
Table 5.10: The numerical confusion matrix of the CNN with the highest performance and 100 training epochs on the EMOVO database. The diagonal numbers show the percent of each class that was correctly identified. The off-diagonal numbers displays the percent of each class that was incorrectly identified as other classes.

		Predicted Labels						
		happy	sad	angry	scared	surprised	disgusted	neutral
Actual Labels	happy	**79.76**	0	9.52	2.38	3.57	1.19	3.57
	sad	3.57	**76.19**	2.38	3.57	10.71	1.19	2.38
	angry	3.57	0	**90.48**	0	4.76	0	1.19
	scared	1.19	2.38	5.95	**78.57**	5.95	0	5.95
	surprised	1.19	2.38	8.33	4.76	**78.57**	1.19	3.57
	disgusted	5.95	1.19	3.57	3.57	7.14	**75**	3.57
	neutral	5.95	0	5.95	4.76	5.95	1.19	**76.19**

Similar to the previous databases, the performance of the CNN enhanced as the number of training epochs increased from 100 to 800 and 4000 epochs. Figures 5.14 and **??**, together with Tables 5.11 and 5.12 display the color-map and numerical confusion matrices associated with the architecture that outperformed other architectures trained on the EMOVO database with 800 and 4000 training epochs.



(a) 10x Augmentation



(b) 20x Augmentation

Figure 5.11: The average performance accuracy of the CNN models over 5 folds trained and tested on the EMOVO database.

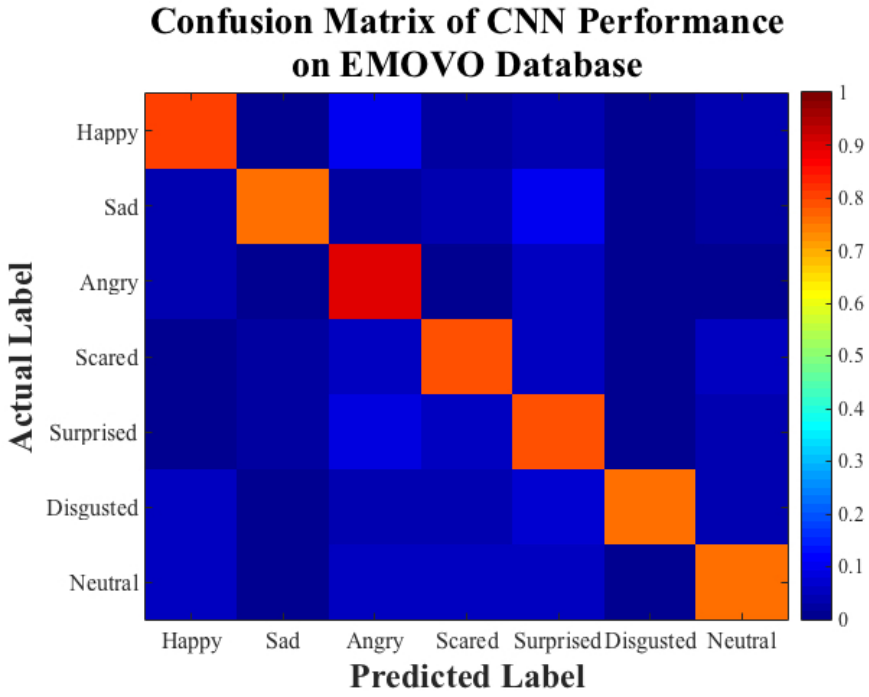


Figure 5.12: The color-map confusion matrix of the CNN with the highest performance and 100 training epochs on the EMOVO database.

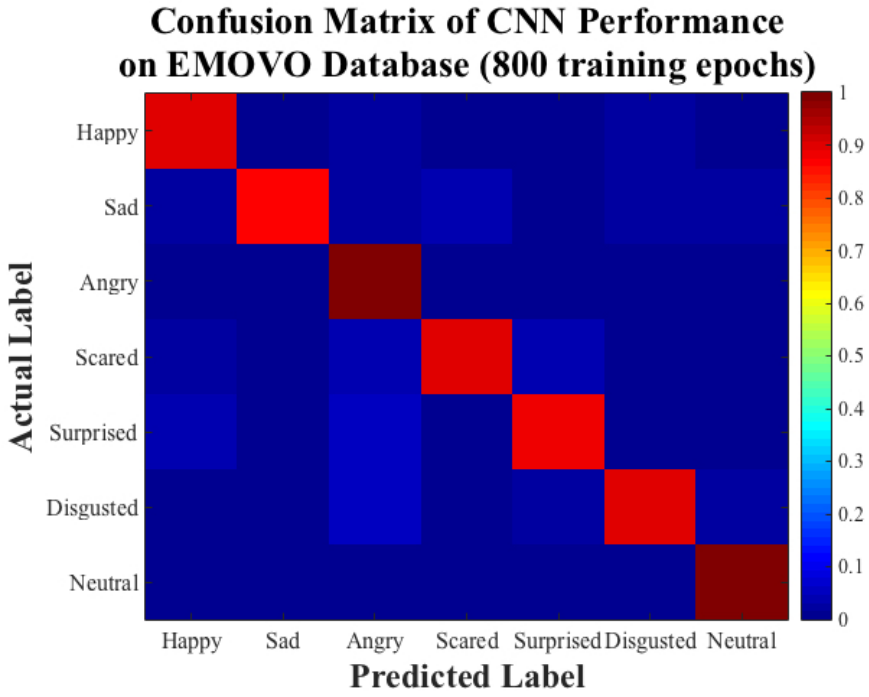


Figure 5.13: The color-map confusion matrix of the CNN with the highest performance and 800 training epochs on the EMOVO database.

Figure 5.20 demonstrates the effect of data augmentation on the performance of the CNN model. The CNN model with the highest performance illustrated in Table 5.9 (10×10 kernels in the first convolutional layer, 5×5 kernels in the

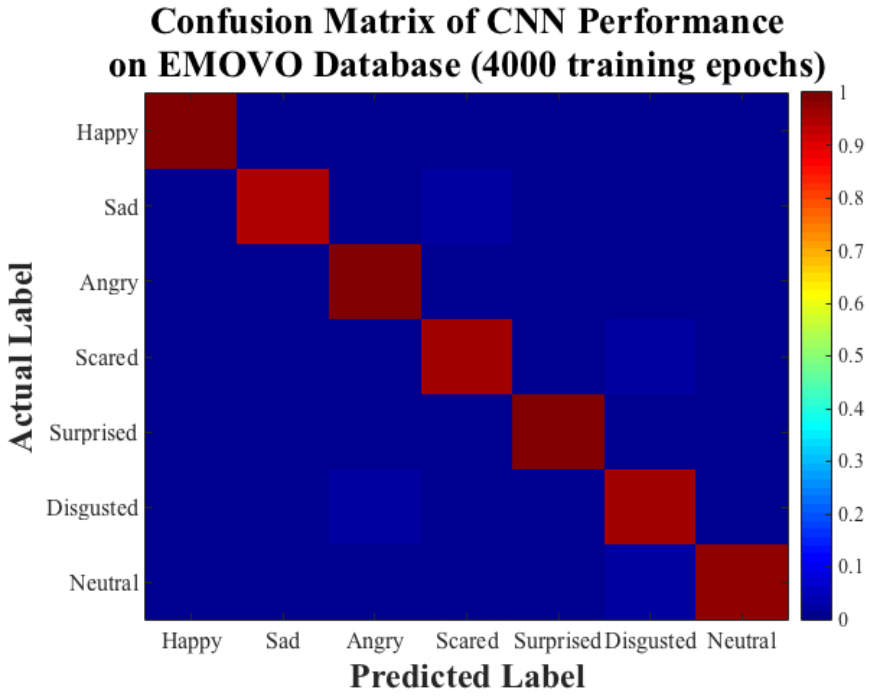


Figure 5.14: The color-map confusion matrix of the CNN with the highest performance and 4000 training epochs on the EMOVO database.

Table 5.11: The numerical confusion matrix of the CNN with the highest performance and 800 training epochs on the EMOVO database. The diagonal numbers show the percent of each class that was correctly identified. The off-diagonal numbers displays the percent of each class that was incorrectly identified as other classes.

		Predicted Labels						
		happy	sad	angry	scared	surprised	disgusted	neutral
Actual Labels	happy	**90.12**	1.23	2.47	1.23	1.23	2.47	1.23
	sad	2.38	**86.90**	2.38	3.57	0	2.38	2.38
	angry	1.19	0	**98.81**	0	0	0	0
	scared	2.38	0	3.57	**89.29**	3.57	1.19	0
	surprised	3.66	1.22	4.88	0	**89.02**	0	1.22
	disgusted	1.2	0	4.82	0	2.41	**89.16**	2.41
	neutral	1.2	0	0	0	0	0	**98.8**

second convolutional layer, max pooling, dropout with $p = 0.5$) was trained on the original EMOVO database and the augmented (20x) EMOVO database separately. The results showed that the CNN model that was trained on the original database classified the test data with accuracy approximately less than 35% whereas the CNN model that was trained on the augmented database classified the same test data with accuracy approximately 92% after 800 training epochs. This underlines

Table 5.12: The numerical confusion matrix of the CNN with the highest performance and 4000 training epochs on the EMOVO database. The diagonal numbers show the percent of each class that was correctly identified. The off-diagonal numbers displays the percent of each class that was incorrectly identified as other classes.

		Predicted Labels						
		happy	sad	angry	scared	surprised	disgusted	neutral
Actual Labels	happy	**100**	0	0	0	0	0	0
	sad	1.19	**95.24**	0	2.38	0	1.19	0
	angry	1.19	0	**98.81**	0	0	0	0
	scared	0	0	0	**96.43**	1.19	2.38	0
	surprised	0	0	0	0	**98.81**	0	1.19
	disgusted	0	0	2.38	1.19	0	**96.43**	0
	neutral	0	0	0	0	0	2.38	**97.62**

the crucial role of data augmentation in improvement of the CNN performance.

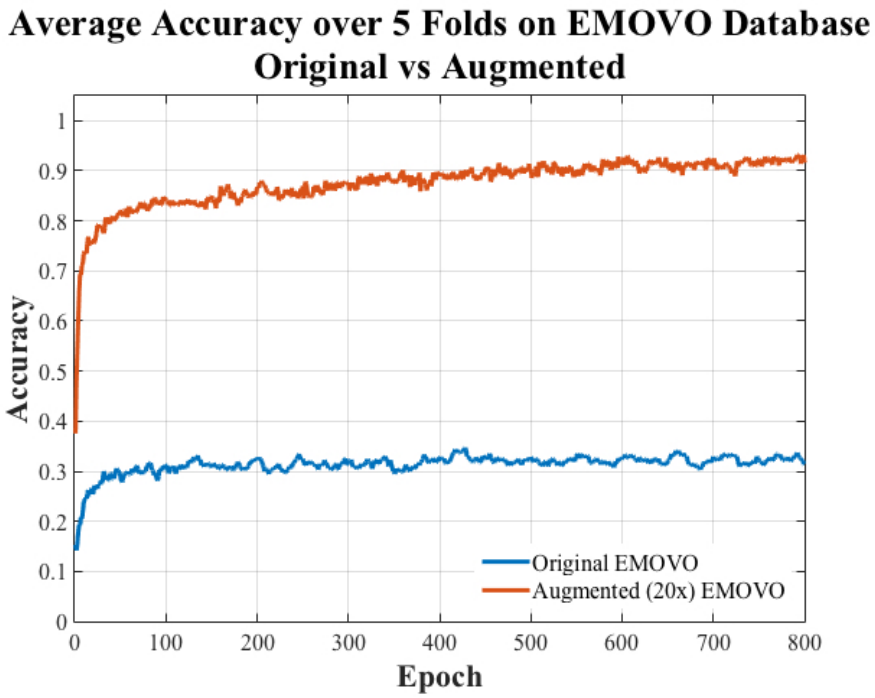


Figure 5.15: A comparison between the CNN performance on the original EMOVO database and the augmented EMOVO database.

5.1.4 BTNRH: American English Database

Table 5.13 summarizes the architectures of the CNN models and their corresponding average accuracy using the BTNRH database. Figures 5.16a and 5.16b

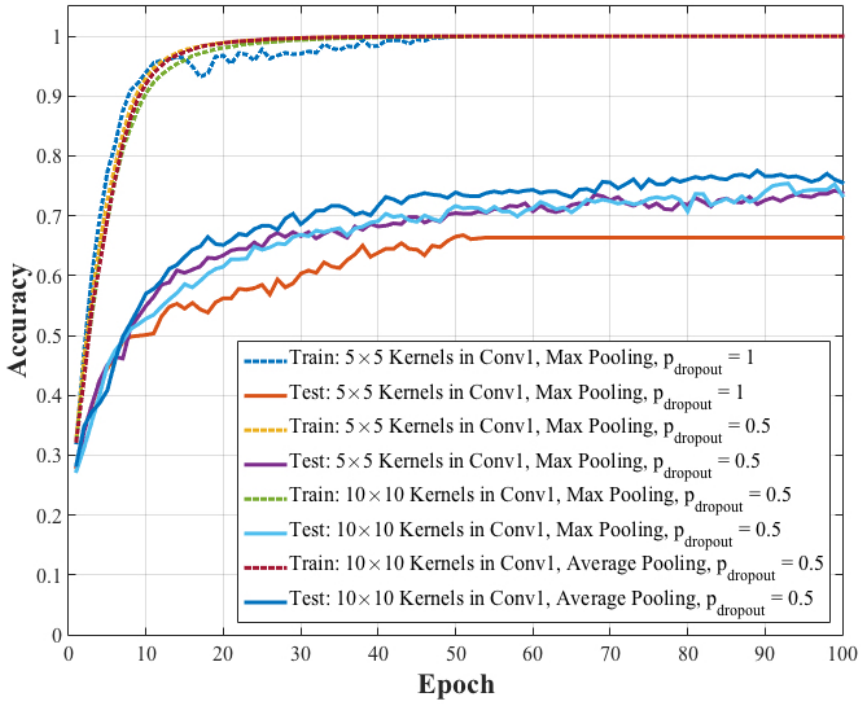
depict the average performance accuracy on the data with 10 times augmentation and 20 times augmentation, respectively. The average accuracy was computed over the 5 folds of the cross-validation assessment.

Table 5.13: The summary of the architectures and the results of the experiments ran on the BTNRH database; $f1$ is the size of the kernels in the first convolutional layer, $f2$ is the size of the kernels in the second convolutional layer, $p_{dropout}$ is the deletion probability of dropout, epoch denotes the number of training iterations, and CV stands for cross-validation.

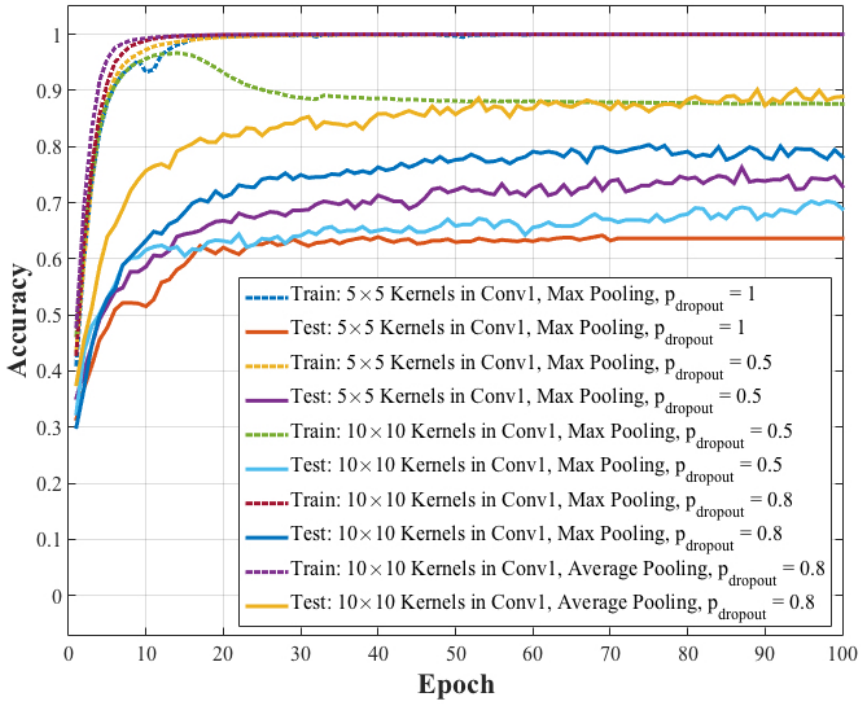
f_1	f_2	pooling	$p_{dropout}$	augmentation	epoch	CV accuracy (%)
5×5	5×5	Max	0	10x	100	66.36
5×5	5×5	Max	0.5	10x	100	73.81
10×10	5×5	Max	0.5	10x	100	73.38
10×10	5×5	Average	0.5	10x	100	75.55
5×5	5×5	Max	1	20x	100	63.64
5×5	5×5	Max	0.5	20x	100	72.91
10×10	5×5	Max	0.5	20x	100	68.9
10×10	5×5	Max	0.2	20x	100	78.17
10×10	**5×5**	**Average**	**0.2**	**20x**	**100**	**88.91**
10×10	**5×5**	**Average**	**0.2**	**20x**	**800**	**92.63**
10×10	**5×5**	**Average**	**0.2**	**20x**	**4000**	**94.51**

As demonstrated, data augmentation, along with increasing the size of kernels in the first convolutional layer and employing dropout, boosted the performance of the CNN on this database up to 88.91%. The results manifest that the degree of regularization required to improve the generalizability of the CNN on the BTNRH database was slightly different from the other databases. That is, the CNN needed less regularization, i.e., $p = 0.8$, using the BTNRH database compared with using the other databases, i.e., $p = 0.5$.

Figure 5.17 shows the color-map confusion matrix related to the architecture with the highest performance and 100 training epochs on the BTNRH database. Table 5.14 presents the numerical values corresponding to this color map. The model classified happy speech utterances with the highest accuracy of 95.33% and sad speech utterances with the lowest accuracy of 86%. Similar to the previous databases, increasing the number of training epochs from 100 to 800 and 4000 improved the test accuracy. Figures 5.19 and **??**, together with Tables 5.15 and 5.16,



(a) 10x Augmentation



(b) 20x Augmentation

Figure 5.16: The average performance accuracy of the CNN models over 5 folds trained and tested on the BTNRH database.

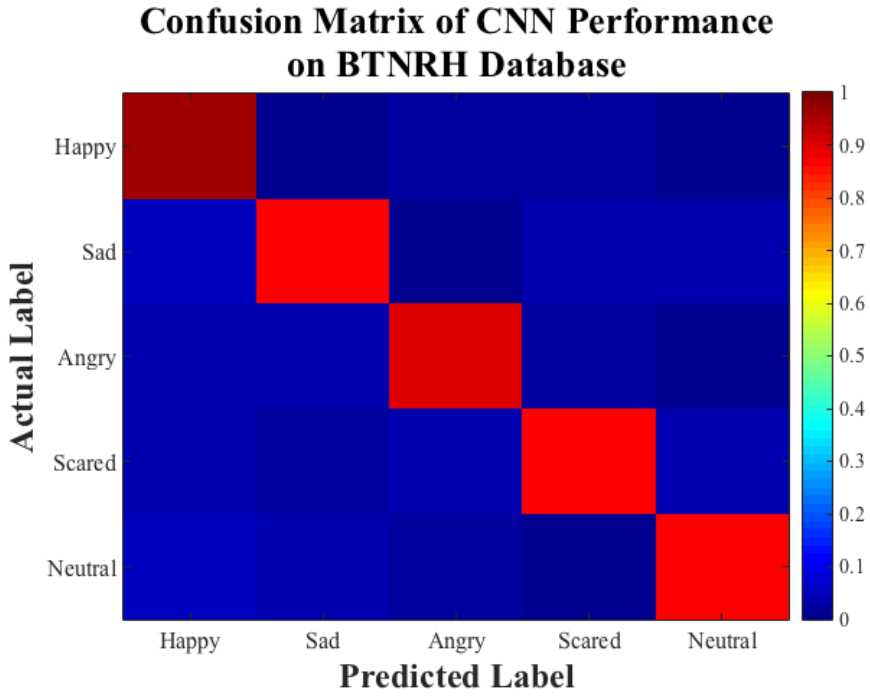


Figure 5.17: The color-map confusion matrix of the CNN with the highest performance 100 training epochs on the BTNRH database.

Table 5.14: The numerical confusion matrix of the CNN with the highest performance and 100 training epochs on the BTNRH database. The diagonal numbers show the percent of each class that was correctly identified. The off-diagonal numbers displays the percent of each class that was incorrectly identified as other classes.

		Predicted Labels				
		happy	sad	angry	scared	neutral
Actual Labels	happy	**95.33**	0	2.67	2	0
	sad	6	**86**	6.67	3.33	4
	angry	4.03	3.36	**89.93**	2.68	0
	scared	4.03	2.68	3.36	**86.58**	3.36
	neutral	5.33	4	2	1.33	**87.33**

show the color-map confusion matrices and the numerical values corresponding to these color maps related to the architecture with the highest performance and 800 and 4000 training epochs on the BTNRH database.

To underscore the effect of data augmentation on the CNN performance, the CNN with the highest performance, as shown in Table 5.13 (10×10 kernels in the

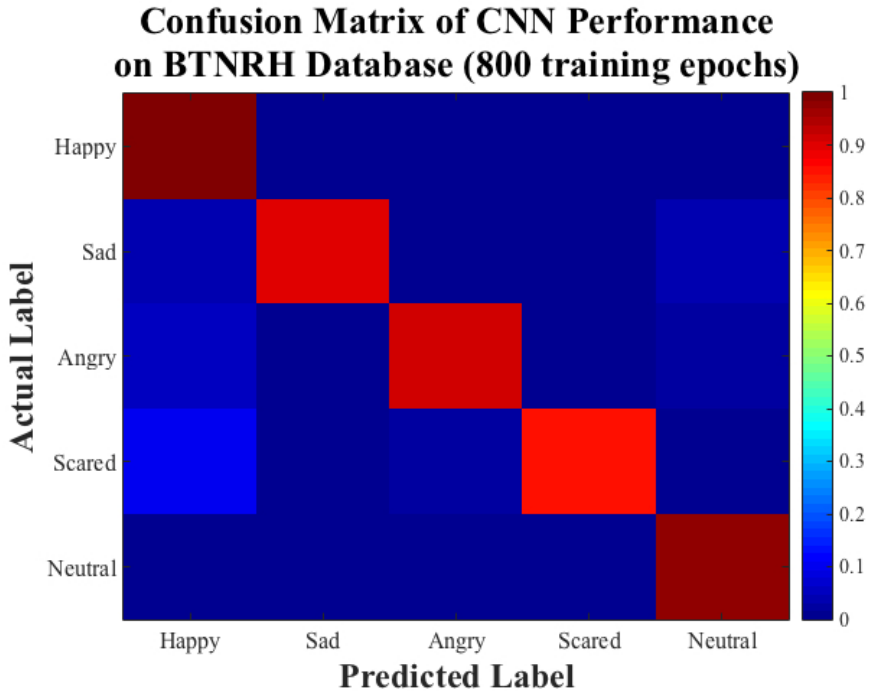


Figure 5.18: The color-map confusion matrix of the CNN with the highest performance 800 training epochs on the BTNRH database.

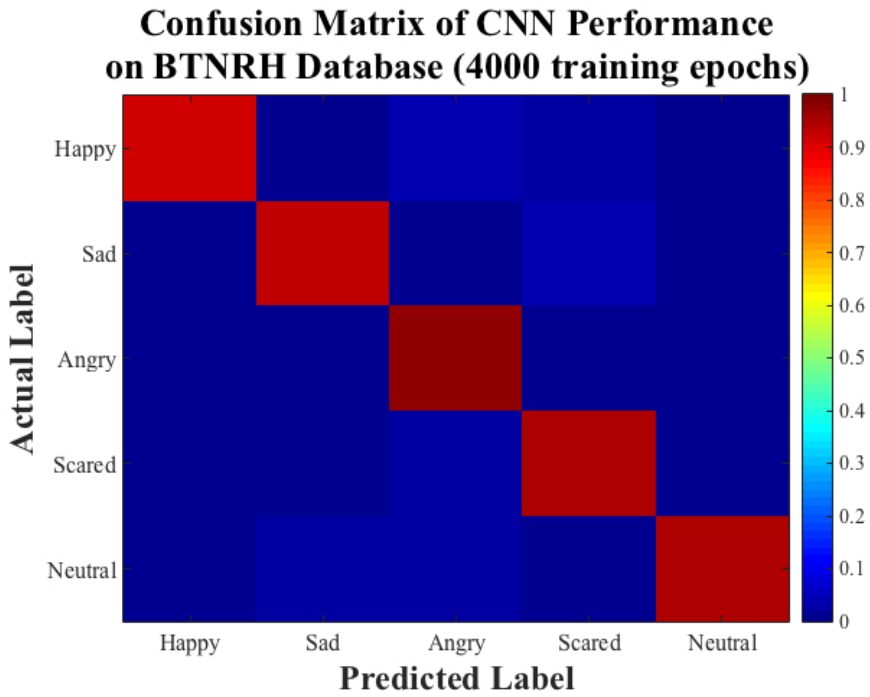


Figure 5.19: The color-map confusion matrix of the CNN with the highest performance 4000 training epochs on the BTNRH database.

first convolutional layer, 5×5 kernels in the second convolutional layer, average pooling, dropout with $p = 0.8$), was trained on the original BTNRH database and the augmented (20x) BTNRH database separately. Figure **??** shows the accuracy

Table 5.15: The numerical confusion matrix of the CNN with the highest performance and 800 training epochs on the BTNRH database. The diagonal numbers show the percent of each class that was correctly identified. The off-diagonal numbers displays the percent of each class that was incorrectly identified as other classes.

		Predicted Labels				
		happy	sad	angry	scared	neutral
Actual Labels	happy	**99.33**	0	0	0	0.67
	sad	4.67	**90**	0.67	1.33	3.33
	angry	5.33	0.67	**90.67**	0.67	2.67
	scared	10.74	0.67	2.01	**85.91**	0.67
	neutral	0.67	0.67	0.67	0.67	**97.33**

Table 5.16: The numerical confusion matrix of the CNN with the highest performance and 4000 training epochs on the BTNRH database. The diagonal numbers show the percent of each class that was correctly identified. The off-diagonal numbers displays the percent of each class that was incorrectly identified as other classes.

		Predicted Labels				
		happy	sad	angry	scared	neutral
Actual Labels	happy	**92**	0.67	4	2.67	0.67
	sad	0.67	**92.67**	0.67	4.67	1.33
	angry	0	0	**98**	1.33	0.67
	scared	0.67	1.34	2.69	**95.3**	0
	neutral	0.67	2	2.67	0.67	**94.67**

of the CNN model. As can be seen, the accuracy of the CNN on the test that was trained on the original database was approximately less than 35% whereas the accuracy of the CNN on the same test set that was trained on the augmented database was approximately around 93%. Consistent to the previous databases, data augmentation notably improved the performance of the CNN on the test data.

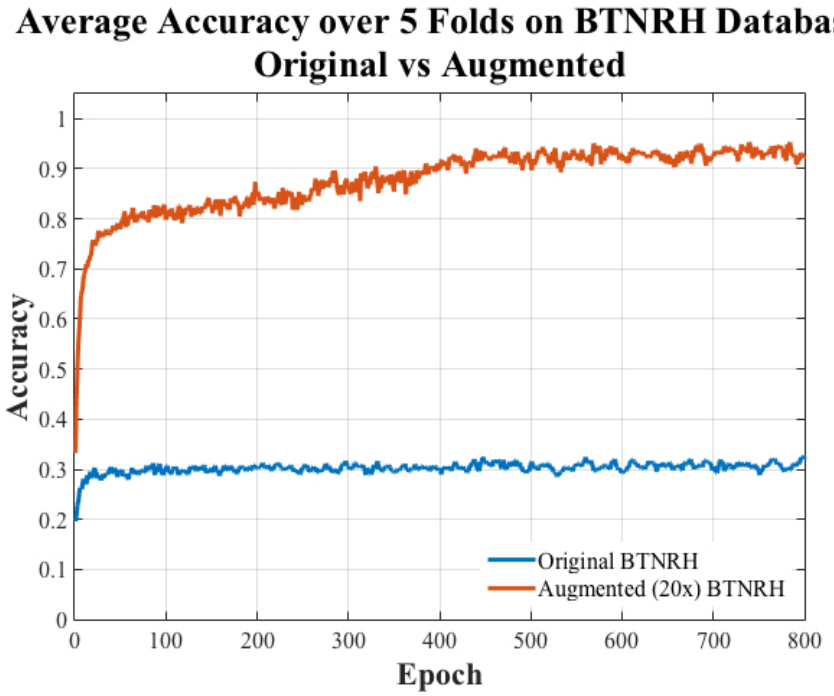
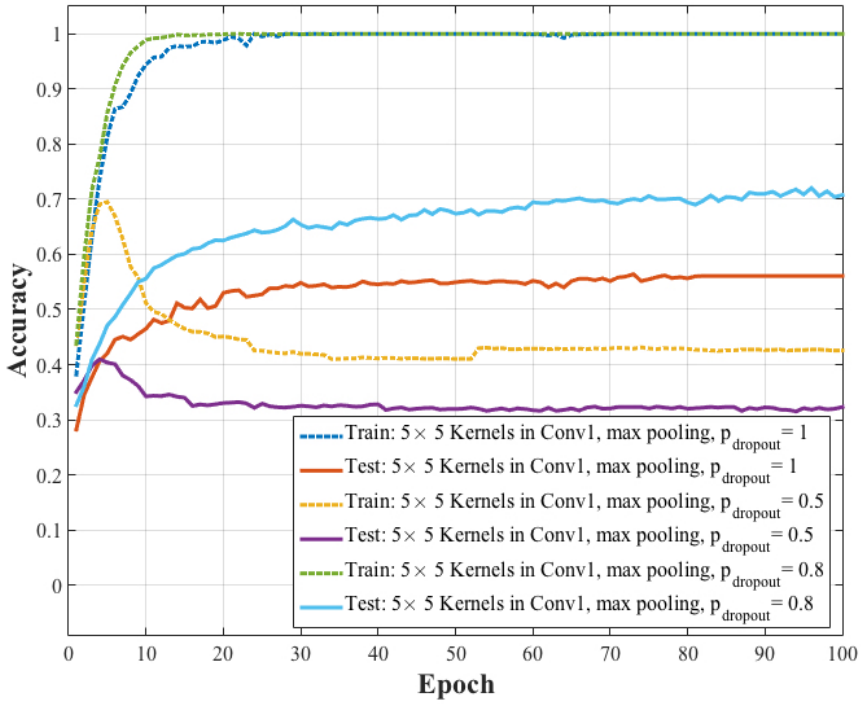


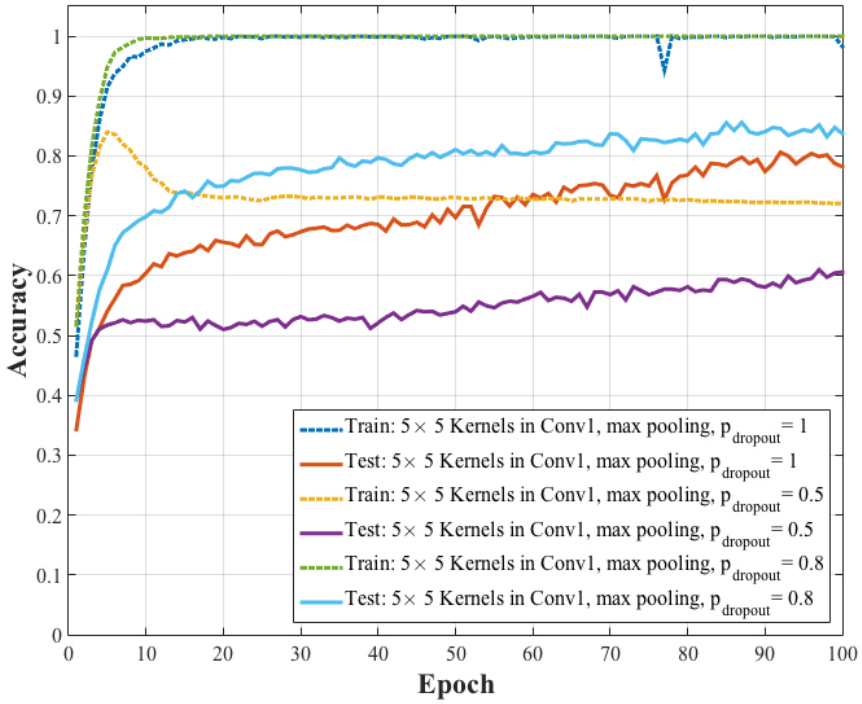
Figure 5.20: A comparison between the CNN performance on the original BTNRH database and the augmented BTNRH database.

5.1.5 All-Inclusive: Language-Independent Database

All four databases used in the current study were consolidated into a one big database. The same 5-fold cross-validation paradigm as the previous experiments was employed to train and test the CNN models. Figures 5.21a and 5.21b illustrate the average performance accuracy on the data with 10 times augmentation and 20 times augmentation, respectively. Similar to the previous experiments, the average accuracy was computed over the 5 folds of the cross-validation assessment. As demonstrated, increasing the augmentation rate from 10 times to 20 times improved the performance of the CNN. This result is consistent with the language-dependent experiments. Also, it can be noticed that employing dropout with the probability of 0.5 caused the model significantly underfit the training data. In a trial and error approach, we found that the dropout probability of 0.2 compromises the best between overfitting and underfitting the data. This result is in contrast with the language-dependent experiments where employing dropout generally improved the performance of the CNN models.



(a) 10x Augmentation



(b) 20x Augmentation

Figure 5.21: The average performance accuracy of the CNN models over 5 folds trained and tested on the all-inclusive, language-independent database with 100 training epochs.

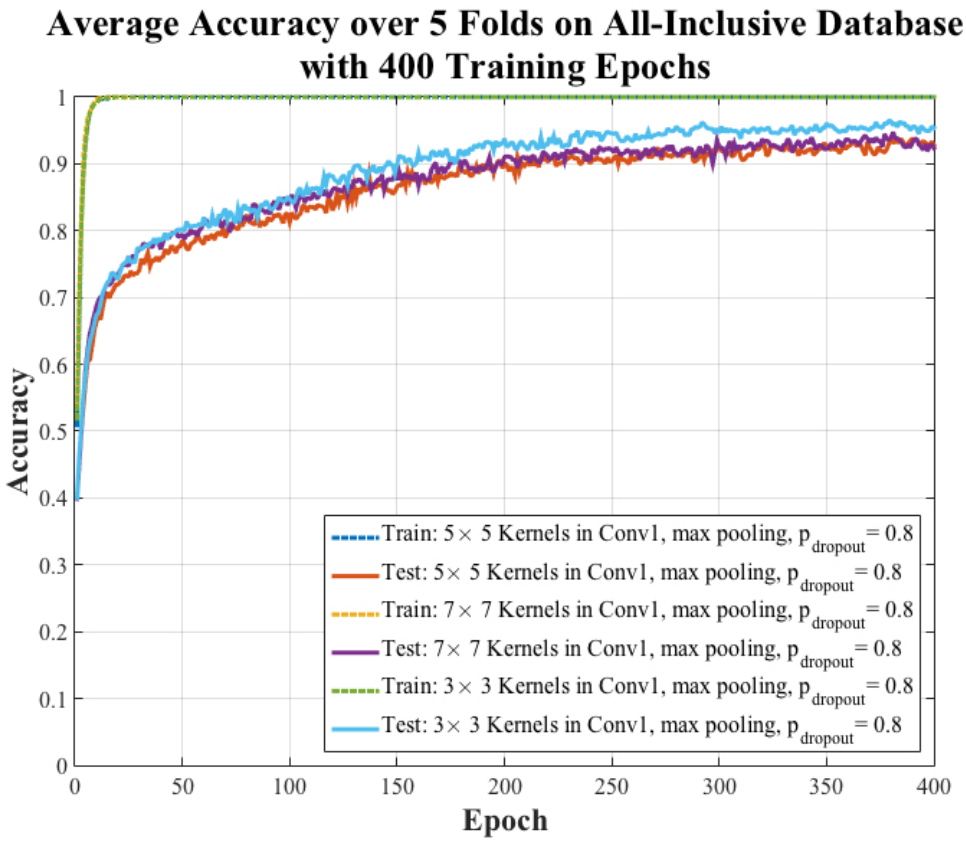


Figure 5.22: The average performance accuracy of the CNN models over 5 folds trained and tested on the all-inclusive, language-independent database with 400 training epochs.

Further, as displayed in Figure 5.22, a decrease in the generalization error was observed as the number of the training epochs increased. This effect was consistent with the language-dependent experiments. Moreover, the results showed that decreasing the size of the kernels from 5×5 to 3×3 improved the generalization accuracy. This was opposite to the results in the language-dependent experiments where increasing the size of the kernels enhanced the performance of the CNNs. Moreover, the CNN models with max pooling had higher performance than the CNN models with average pooling. This also contradicts the results on language-dependent models where average pooling, in many instances, enhanced the test accuracy.

Table 5.17 summarizes the architectures of the CNN models, the number of training epochs, the augmentation rate, and their corresponding average accuracy using the all-inclusive, language-independent database.

Table 5.17: The summary of the architectures and the results of the experiments ran on the BTNRH database; $f1$ is the size of the kernels in the first convolutional layer, $f2$ is the size of the kernels in the second convolutional layer, $p_{dropout}$ is the deletion probability of dropout, epoch denotes the number of training iterations, and CV stands for cross-validation.

f_1	f_2	pooling	$p_{dropout}$	augmentation	epoch	CV accuracy (%)
5×5	5×5	Max	0	10x	100	56.08
5×5	5×5	Max	0.5	10x	100	32.32
5×5	5×5	Max	0.2	10x	100	70.76
5×5	5×5	Max	0	20x	100	78.2
5×5	5×5	Max	0.5	20x	100	60.6
5×5	5×5	Max	0.2	20x	100	83.72
5×5	5×5	Max	0.2	20x	400	92.83
7×7	5×5	Max	0.2	20x	400	92.36
3×3	5×5	Max	0.2	20x	400	95.38
3×3	5×5	Max	0.2	20x	800	96.55
3×3	5×5	**Max**	**0.2**	**20x**	**4000**	**97.48**

Table 5.18: The numerical confusion matrix of the CNN with 4000 training iterations on the all-inclusive, language-independent database. The diagonal numbers show the percent of each class that was correctly identified. The off-diagonal numbers displays the percent of each class that was incorrectly identified as other classes.

		Predicted Labels				
		happy	sad	angry	scared	neutral
Actual Labels	happy	**98.08**	0	0.55	1.1	0.27
	sad	0.84	**96.06**	0.28	2.54	1.13
	angry	1.43	0	**98.34**	0.24	0
	scared	0.55	0.55	0.83	**96.96**	1.10
	neutral	1.07	0	.8	0	**98.12**

Figure 5.23 shows the color-map confusion matrix related to the architecture with 4000 training iterations on the all-inclusive database. Table 5.18 presents the numerical values corresponding to this color map. The model classified neutral speech utterances with the highest accuracy of 98.12% and sad speech utterances with the lowest accuracy of 96.06%.

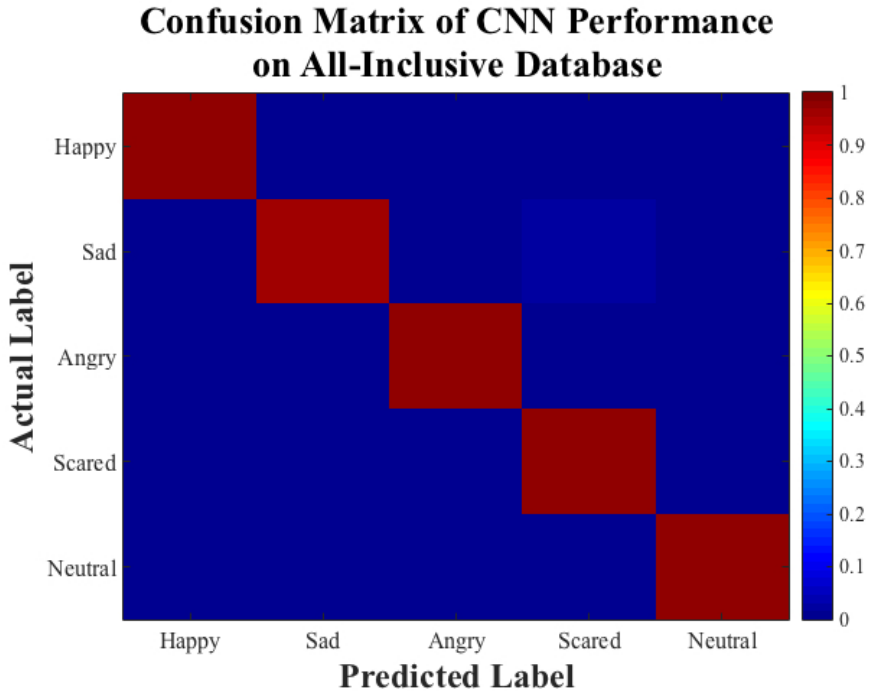


Figure 5.23: The color-map confusion matrix of the CNN with 4000 training iterations on the all-inclusive, language-independent database.

5.2 Discussion

The CNN model developed in the current study achieved remarkable results in classifying speech emotions using the EMODB database. The CNN, together with data augmentation, surpassed human performance on the EMODB database that was previously reported by Burkhardt et al. [62]. Tables 5.19 and 5.20 compare the classification accuracy of each emotion resulted from our network with human performance. As demonstrated, our network outperformed human performance. This outperformance is even greater for the network with 4000 training epochs. Further, our network manifested a superior performance over the hybrid DNN-HMM classifier used by Li et al. [27]. They used deep neural networks (DNN) to extract discriminative features and used hidden Markov models (HMM) to classify the speech emotions based on the extracted features. The best results that they achieved on the EMODB database was 77.92%, which is significantly less than the best result that has been achieved by our network with 100 training epochs, i.e., 96.78%. Moreover, our network has outperformed the model proposed by Mao

et al. [28]. They used a convolutional neural network with one convolutional layer, which was followed by an average-pooling layer, and one fully connected layer to extract discriminative features from the narrow-band spectrogram images. In doing so, the kernels were first pre-trained using an unsupervised auto-encoder. Subsequently, they used an SVM classifier to decide the emotional states of speech utterances base on these extracted features. This model had an average accuracy of 85.2% on the EMODB database, which also is less than the best accuracy of our network with 100 training epochs. Our results showed that a convolutional neural network with two convolutional layers, together with the wide-band spectrogram images, data augmentation, and sufficient training epochs, could classify speech emotion with a state-of-the-art accuracy using the EMODB database.

Table 5.19: A comparison between the classification accuracy by human listeners and the convolutional neural network using the EMODB database with 100 training epochs; the human performance reported by [62], the convolutional neural network implemented in the current study had two convolutional layers followed by the average pooling, and one fully connected layer with dropout ($p = 0.5$); the EMODB database was augmented to train the network.

Emotion	Human Performance Accuracy	Our Network Accuracy (%)
happiness	83.7	95.7
sadness	80.7	96.7
anger	96.9	100
fear	87.3	97.1
boredom	86.2	92.5
disgust	79.6	93.4
neutral	88.2	97.4

Although the performance of our networks on the SAVEE database was inferior to the performance on the EMODB database, the best accuracy achieved in our experiments after 100 training epochs surpassed the human performance on this database that was reported by Haq et al. [63]. It was stated that human subjects recognized emotions of speech utterances from the SAVEE database by the average accuracy of 66.5%. In contrast, our network could classify the speech emotion of this database with 81.9% accuracy after 100 training epochs. Fur-

Table 5.20: A comparison between the classification accuracy by human listeners and the convolutional neural network using the EMODB database with 4000 training epochs; the human performance reported by [62], the convolutional neural network implemented in the current study had two convolutional layers followed by the average pooling, and one fully connected layer with dropout ($p = 0.5$); the EMODB database was augmented to train the network.

Emotion	Human Performance Accuracy	Our Network Accuracy (%)
happiness	83.7	100
sadness	80.7	100
anger	96.9	99.21
fear	87.3	100
boredom	86.2	100
disgust	79.6	100
neutral	88.2	100

ther, Haq et al. [63] developed a Gaussian classifier to classify emotions using the SAVEE database. They employed a traditional machine learning approach to classify speech emotions. That is, they extracted pitch, energy, duration, and spectral features from audio signals and subsequently applied principal component analysis (PCA) and linear discriminant analysis (LDS) to reduce the feature size. A Gaussian classifier was designed to use these features to distinguish speech emotions from one another. The best accuracy of their model was 56.3%, which was less than the best accuracy we achieved in our experiments after 100 training epochs, i.e., 81.9%. Furthermore, Mao et al. [28], whose model was described above, implemented a deep learning model to classify emotions using the SAVEE database. The best accuracy they achieved was 73.6%, which was less than the best accuracy of our networks. Moreover, Fayek et al. [29] developed a deep neural network with 5 layers and used the narrow-band spectrogram images as the inputs. They augmented the training data by re-sampling the audio utterances with different sampling rate. Prior to any processing, all utterances used in their study were re-sampled to have sampling rate of 8 kHz. The total classification accuracy of their model on the SAVEE database was 59.7%. All in all, our network outperformed the previous models reported in the literature. Table 5.21 outlines the accuracy

of previous models and our work on the EMODB and SAVAEE databases.

Table 5.21: A comparison between the convolutional neural network implemented in the current study and some previous models using the EMODB and SAVEE databases. "Our Network-100" stands for the network developed in the current thesis with 100 training epochs. Likewise, "Our Network-4000" stands for the network with 4000 training epochs developed in the current thesis.

Study	Database	Performance Accuracy (%)
Li et al. [27]	EMODB	77.92
Mao et al. [28]	EMODB	85.2
Our Network-100	**EMODB**	**96.7**
Our Network-4000	**EMODB**	**99.83**
Haq et al. [63]	SAVEE	56.3
Fayek et al. [29]	SAVEE	59.7
Mao et al. [28]	SAVEE	73.6
Our Network-100	**SAVEE**	**81.9**
Our Network-4000	**SAVEE**	**98.33**

The classification accuracy of the network developed on the EMOVO database with 100 training epochs was 81.07%. Costantini et al. [64] previously reported that the overall recognition accuracy of this database by human listeners was 80.57%. This shows that the performance of the CNN, developed in our study, was comparable to human performance. Figure 5.24 compares the color-map confusion matrix between our model with 100 training epochs and human performance. As demonstrated, our model outperformed human listeners in classifying happy and disgusted speech utterances whereas human listeners outperformed our model in recognizing sad and neutral speech sounds. Albeit, the overall performance of our model was close to human performance. However, our results demonstrates that increasing the number of training epochs from 100 to 4000 improved the performance of the network such that it surpassed the human performance. Figure 5.25 compares the color-map confusion matrix between our model with 4000 training epochs and human performance. Table 5.22 displays the numerical confusion matrix corresponding to human performance.

Table 5.23 summarized the classification accuracy of each emotion resulted from

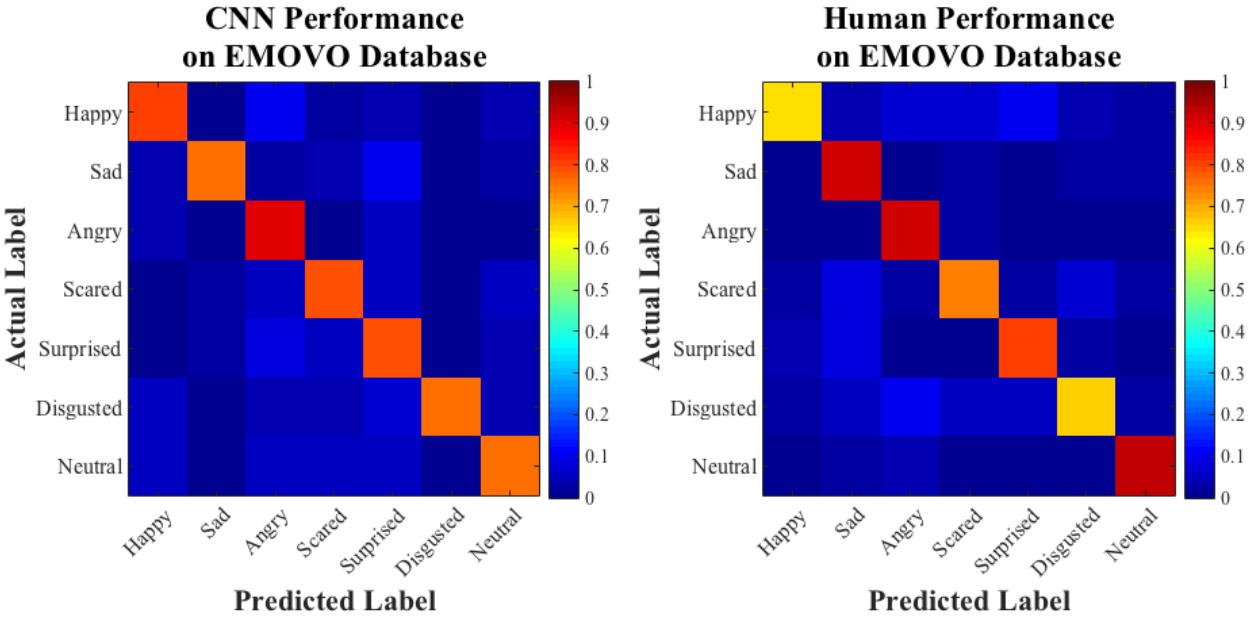


Figure 5.24: Convolutional neural network with 100 training epochs *vs* human on the EMOVO database. The color-map confusion matrices of emotion recognition by the CNN implemented in our study and the human listeners reported by [64].

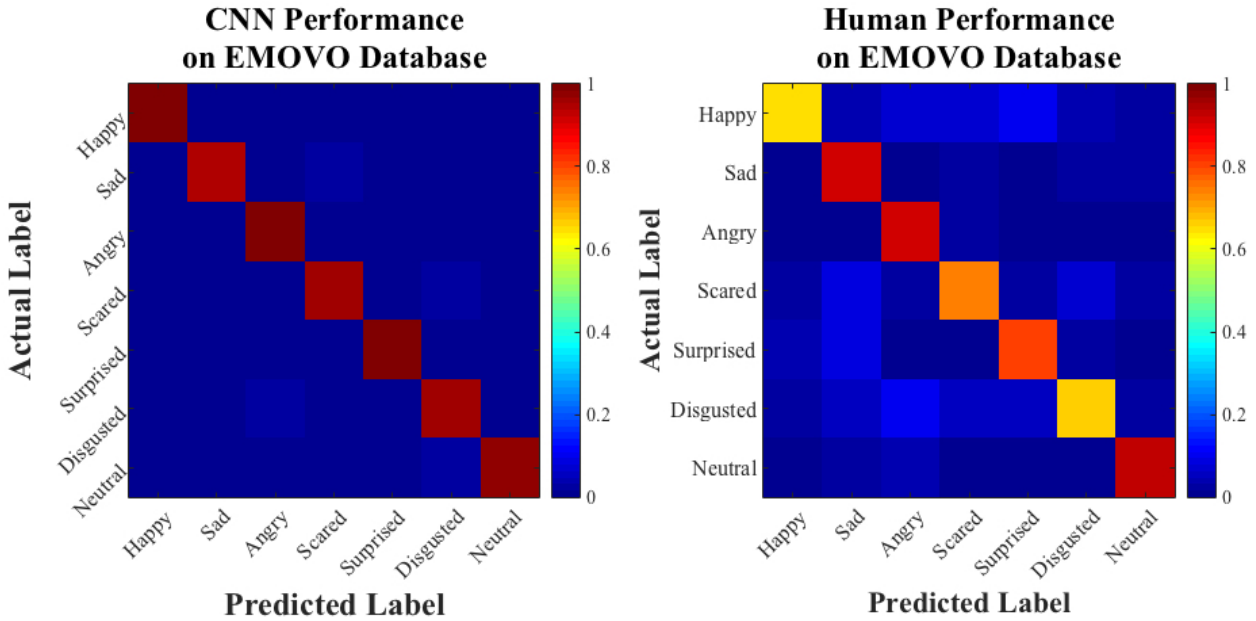


Figure 5.25: Convolutional neural network with 4000 training epochs *vs* human on the EMOVO database. The color-map confusion matrices of emotion recognition by the CNN implemented in our study and the human listeners reported by [64].

our network trained with 100 training epochs on the EMOVO database and human performance. As alluded to earlier, our model had higher accuracy in classifying the happy, disgusted, and scared speech utterances whereas human listeners had higher accuracy in recognizing the other emotional states. Both CNN and human

Table 5.22: The numerical confusion matrix of the Human performance on the EMOVO database reported by [64].

		Predicted Labels						
		happy	sad	angry	scared	surprised	disgusted	neutral
Actual Labels	happy	**65**	4	7	7	1	4	2
	sad	1	**92**	0	3	0	2	2
	angry	1	1	**92**	3	1	1	1
	scared	2	9	3	**74**	3	7	2
	surprised	4	9	1	1	**81**	3	1
	disgusted	2	6	10	6	6	**67**	3
	neutral	0	2	4	0	0	1	**93**

performance had the highest accuracy in classifying the angry speech sounds. Likewise, Table 5.24 compares the classification accuracy of the emotions between our network with 4000 training epochs and human performance on the EMOVO database. As demonstrated, our network outperformed human performance in recognizing all the emotional states after 4000 training epochs.

Table 5.23: A comparison between the classification accuracy by human listeners and the convolutional neural network with 100 training epochs using the EMOVO database; the human performance reported by [64], the convolutional neural network implemented in the current study had two convolutional layers followed by the max pooling, and one fully connected layer with dropout ($p = 0.5$); the EMOVO database was augmented to train the network.

Emotion	Human Performance Accuracy	Our Network Accuracy (%)
happy	65	79.76
sad	92	76.19
angry	92	90.48
scared	74	78.57
surprised	81	78.57
disgusted	67	75
neutral	93	76.19

The BTNRH database was privately developed for the research in Speech and Hearing Science [14]. We are the first machine learning group who used this database to develop deep learning models. The outcomes of the experiments on this database were similar to other databases. That is, augmentation, dropout, and increasing the number of training epochs boosted the performance of the CNN

Table 5.24: A comparison between the classification accuracy by human listeners and the convolutional neural network with 4000 training epochs using the EMOVO database; the human performance reported by [64], the convolutional neural network implemented in the current study had two convolutional layers followed by the max pooling, and one fully connected layer with dropout ($p = 0.5$); the EMOVO database was augmented to train the network.

Emotion	Human Performance Accuracy	Our Network Accuracy (%)
happy	65	100
sad	92	95.24
angry	92	98.81
scared	74	96.43
surprised	81	98.81
disgusted	67	96.43
neutral	93	97.62

models on this database significantly. Further, increasing the size of kernels had a regularization effect on the CNN performance.

The result on all-inclusive, language-dependent database indicated that the model faced both underfitting and overfitting problems. That is, although data augmentation improved the performance of the data, but dropout with the highest effect of regularization significantly exacerbated the performance of the CNN models. Further, the language-independent models required more training iterations than language-dependent models to learn the discriminant features and disentangle the confound variances such as genders and languages. Overall, the results showed that the CNN decodes emotions in acted speech signals, incorporating different genders and different languages, with a state-of-the-art accuracy of 96.55%. It should be noted that we removed the speech instances pertinent to the emotion classes that were not present in all databases. More specifically, we removed bored and disgusted utterances from the EMODB database as well as the surprised and disgusted utterances from the SAVEE and EMOVO database. This naturally raises the question whether the improvement on all-inclusive database happened because these removed classes were unusually hard to classify. To approach this question, the F1-score of each emotion against other emotions were measured

for the language-dependent models that were trained on EMODB, SAVEE, and EMOVO databases as follow:

$$F_1 = 2 \times \frac{\text{precision} \times \text{sensitivity}}{\text{precision} + \text{sensitivity}}. \tag{5.1}$$

The F1-score takes into account both sensitivity and precision of recognition. The sensitivity quantifies the proportion of the instances of the specific emotion that were recognized correctly, i.e., $\frac{\text{true positives}}{\text{true positives+false negatives}}$. On the other hand, the precision quantifies the contribution of the emotion to false negatives of other emotions, i.e., $\frac{\text{true positives}}{\text{true positives+false positives}}$. In other words, the precision reflects the degrees of confusion a specific emotion introduces to the system—the proportion of other emotions recognized as the emotion of interest. Tables 5.25, 5.27, and **??** summarize the measured F1-scores of the language-dependent and language-independent CNN models with 100, 800, and 4000 training epochs, respectively. As can be seen in ables 5.25 (related to the network with 100 training epochs), the F1-scores of disgusted and bored classes are not less than other emotions. However, the F1-scores of the surprised classes in boht SAVEE and EMOVO databases are smaller than other classes. This suggests that the surprised class has posed a challenge to the classifiers with 100 training epochs and removing it from the all-inclusive database explains a part of the observed improvement. Inspecting Tables 5.25, 5.27, and **??** reveals that decoding bored, surprised, or disgusted emotions in speech signals were not unusually hard for the CNN models after 800 and 4000 training epochs although it posed a challenge to the CNN classifiers that were trained with 100 training epochs. As a result, the improvement observed in all-inclusive database is, in part, due to the greater availability of emotion instances that facilitated emotion recognition irrespective of the language. All the F1-scores measured in our work were greater than F1-scores reported by [33]. Their CNN models were more complex than ours as they had deeper convolutional and fully connected layers with a larger numbers of kernels. Further, their models were trained longer than our models. Albeit, our CNN models outperformed the

CNN models in [33].

Table 5.25: F1-scores of each emotion for all language-dependent and all-inclusive models with 100 training epochs.

	F1-Score							
Database	happy	sad	angry	scared	bored	surprised	disgusted	neutral
EMODB	0.9496	0.9752	0.9653	0.9565	0.9875	-	0.9778	0.9693
SAVEE	0.8031	0.8547	0.7846	0.7636	-	0.7368	0.8926	0.8926
EMOVO	0.7929	0.8366	0.8	0.7952	-	0.7253	0.8344	0.7758
BTNRH	0.8882	0.8776	0.9024	0.8836	-	-	-	0.8973
All-inclusive	0.7911	0.8343	0.8437	0.8199	-	-	-	0.8474

Table 5.26: F1-scores of each emotion for all language-dependent and all-inclusive models with 800 training epochs.

	F1-Score							
Database	happy	sad	angry	scared	bored	surprised	disgusted	neutral
EMODB	0.9929	1	0.9883	1	1	-	0.9778	1
SAVEE	0.95	0.9752	0.9256	0.9661	-	0.9048	0.9391	0.9916
EMOVO	0.9102	0.9182	0.9121	0.9202	-	0.9202	0.9202	0.9535
BTNRH	0.9003	0.9375	0.9347	0.9110	-	-	-	0.9511
All-inclusive	0.9646	0.9642	0.9739	0.9498	-	-	-	0.9756

Table 5.27: F1-scores of each emotion for all language-dependent and all-inclusive models with 4000 training epochs.

	F1-Score							
Database	happy	sad	angry	scared	bored	surprised	disgusted	neutral
EMODB	0.9930	1	0.9960	1	1	-	1	1
SAVEE	0.9831	0.9672	0.9752	0.9917	-	0.9831	0.9833	1
EMOVO	0.9882	0.9756	0.9822	0.9643	-	0.9881	0.9474	0.9880
BTNRH	0.9485	0.9424	0.9423	0.9342	-	-	-	0.9595
All-inclusive	0.9741	0.9785	0.9810	0.9656	-	-	-	0.9786

Further, Figure 5.26 illustrates the average accuracy on the test data for all language-dependent and the language-independent models after 4000 training epochs. As can be seen, the network trained on the EMODB database always had the higher average accuracy on the test data. The network trained on the all-inclusive database outperformed the language-dependent networks trained on the SAVEE, EMOVO, and BTNRH databases when the training epoch was set

to 100 and 800. However, after 4000 training epochs, the performance of the language-dependent networks on the SAVEE and EMOVO databases converged into the performance of the language-independent all-inclusive database. The network trained on the BTNRH database had the poorest accuracy for all training epochs. Also, the test accuracy curves plateaued for 4000 training epoch. As a result, we did not try the training epochs greater than 4000.

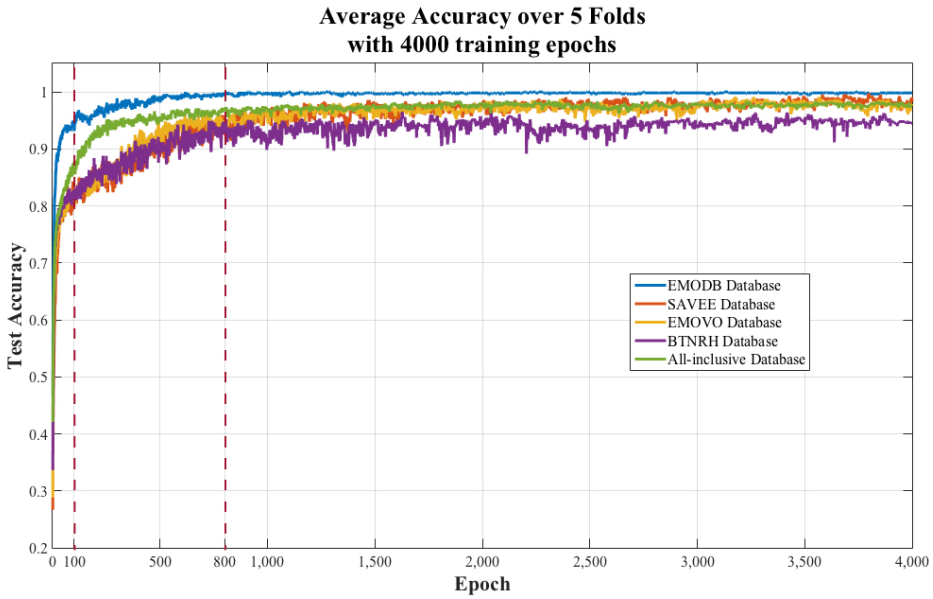


Figure 5.26

Taken together, the results of the current study manifest the efficacy of the convolutional neural networks in classifying the emotion of the speech utterances that were acted by actors; this effectiveness was independent of the gender and the language of the speakers. Further, the outcomes underscore the role of the data augmentation, dropout, and learning time in improving generalizability of the CNN models. Moreover, the results revealed that increasing the size of the kernels in the first convolutional layer had a regularization effect on the language-dependent models; that is, it mitigated overfitting the training data and improved the generalization accuracy. Likewise, the results showed that decreasing the size of the kernels in the first convolutional layer, along with increasing the number of training iterations, equipped the language-independent models to cope with underfitting the training data.

Chapter 6

Conclusion & Future Work

In this study, we implemented convolutional neural networks (CNNs) to classify emotional states of acted speech utterances using three commonly used benchmark databases [62, 63, 64] and one private database (personal communication). The CNNs had two convolutional layers, each followed by a pooling layer, and one fully connected layer. The k-way softmax unit was used to estimate the probability distribution of the classes in the output layer. The cross-entropy was used as the loss function to measure the estimation error and Adam optimizer was used to minimize the loss function. All speech signals were converted into wide-band spectrograms and fed into the CNNs as the inputs. Previous literature mostly used narrow-band spectrograms, which have higher frequency resolution than wide-band spectrograms and resolve individual harmonics. On the other hand, wide-band spectrograms have higher time resolution than narrow-band spectrograms and show individual glottal pulses, which are associated with fundamental frequency and pitch. Due to the importance of pitch in emotion recognition, this study used wide-band spectrogrmas in lieu of narrow-band spectrograms. Further, the training data were augmented by adding white noise to audio signals. Dropout was employed with various deletion probabilities to regularize the CNNs. The size of convolution kernels were manipulated whenever required.

The CNNs performed well on the training data for all databases. However, the performance on the test data varied across different architectures and databases. The results showed that dropout improved the performance of the networks on the test sets to various extents depending on language dependencies. That is, the

language-dependent models required a greater degree of regularization compared with the language-independent models. However, data augmentation decreases the test errors significantly for both language-dependent and language-independent models. Our results also revealed that increasing the size of convolution kernels had a regularization effect on the language-dependent models and increased the accuracy of the networks on the test data. On the other hand, decreasing the size of convolution kernels assisted the language-independent models to fight underfitting the model. Increasing the number of training epochs boosted the test accuracy for both language-dependent and language-independent models. Incorporating dropout, data augmentation, and wide-band spectrograms, the CNNs achieved the state-of-the-art accuracy, outperformed previously reported results in the literature, and emulated or even surpassed human performance over the benchmark databases.

The results of the current study manifested the competency of CNNs in learning the underlying emotional features of speech signals from their low-level representation using wide-band spectrograms irrespective of the gender and the language of the speakers. Despite the remarkable success of our networks, there are still opportunities for further enhancements. As alluded to earlier, wide-band spectrograms cannot resolve individual harmonics; that is, it lacks some fine-grained spectral information. For future work, we suggest to incorporated the narrow-band spectrograms into the training data as the second channel of the input and form a multi-channel training data. This might reduce the number of training epochs or the amount of data augmentation that was required to achieve high performance. Further, although there is almost no room remained for EMODB database [62] to be improved, there still exists a noticeable gap between the training and the test accuracy of the networks on other databases with 100 training epochs. To eliminate this gap, we suggest, instead of random initialization, to pre-train the CNNs by transfer learning [69] from the architecture trained on the EMODB database to the architectures that are trained on other databases. More-

over, as explained in Chapter 1, emotions are encoded in several modalities such as audio, visual, and linguistic. Using information from different modalities will facilitate approaching the problem of automatic emotion recognition in daily life situations and under adverse listening conditions. For future work, we suggest to use audio-visual databases or audio-visual-linguistic databases to train deep leaning models where facial expressions and semantic information are taken into account as well as speech signals. Finally, k-fold cross-validation was used to train all-inclusive, language-independent models. For future, we suggest using leave-one-out cross-validation to train and test the performance of the CNN models and compare the results with k-fold cross-validation.

References

[1] Stuart Jonathan Russell, Peter Norvig, John F Canny, Jitendra M Malik, and Douglas D Edwards. *Artificial intelligence: a modern approach*, volume 2. Prentice hall Upper Saddle River, 2003.

[2] Demis Hassabis, Dharshan Kumaran, Christopher Summerfield, and Matthew Botvinick. Neuroscience-inspired artificial intelligence. *Neuron*, 95 (2):245–258, 2017.

[3] Marcel Van Gerven. Computational foundations of natural intelligence. *Frontiers in Computational Neuroscience*, 11:112, 2017.

[4] Paul R Cohen and Edward A Feigenbaum. *The handbook of artificial intelligence*, volume 3. Butterworth-Heinemann, 2014.

[5] Ray Kurzweil. *The singularity is near*. Gerald Duckworth & Co, 2010.

[6] Marvin Minsky. *The emotion machine: Commonsense thinking, artificial intelligence, and the future of the human mind*. Simon and Schuster, 2007.

[7] Dong Yu and Li Deng. *AUTOMATIC SPEECH RECOGNITION*. Springer, 2016.

[8] Lawrence R Rabiner and Biing-Hwang Juang. *Fundamentals of speech recognition*, volume 14. PTR Prentice Hall Englewood Cliffs, 1993.

[9] Lalit R Bahl, Frederick Jelinek, and Robert L Mercer. A maximum likelihood approach to continuous speech recognition. In *Readings in speech recognition*, pages 308–319. Elsevier, 1990.

[10] Stephen E Levinson, Lawrence R Rabiner, and Man Mohan Sondhi. An introduction to the application of the theory of probabilistic functions of a markov process to automatic speech recognition. *The Bell System Technical Journal*, 62(4):1035–1074, 1983.

[11] Su-Lin Wu, ED Kingsbury, Nelson Morgan, and Steven Greenberg. Incorporating information from syllable-length time scales into automatic speech recognition. In *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, volume 2, pages 721–724. IEEE, 1998.

[12] Vaibhava Goel and William J Byrne. Minimum bayes-risk automatic speech recognition. *Computer Speech & Language*, 14(2):115–135, 2000.

[13] Louis Ten Bosch. Emotions, speech and the asr framework. *Speech Communication*, 40(1-2):213–225, 2003.

[14] Monita Chatterjee, Danielle J Zion, Mickael L Deroche, Brooke A Burianek, Charles J Limb, Alison P Goren, Aditya M Kulkarni, and Julie A Christensen. Voice emotion recognition by cochlear-implanted children and their normally-hearing peers. *Hearing research*, 322:151–162, 2015.

[15] Nancy Eisenberg, Tracy L Spinrad, and Natalie D Eggum. Emotion-related self-regulation and its relation to children's maladjustment. *Annual review of clinical psychology*, 6:495–525, 2010.

[16] Harold Schlosberg. Three dimensions of emotion. *Psychological review*, 61 (2):81, 1954.

[17] Thomas S Polzin and Alex Waibel. Detecting emotions in speech. In *Proceedings of the CMC*, volume 16. Citeseer, 1998.

[18] Chul Min Lee and Shrikanth S Narayanan. Toward detecting emotions in spoken dialogs. *IEEE transactions on speech and audio processing*, 13(2): 293–303, 2005.

[19] Dimitrios Ververidis and Constantine Kotropoulos. Emotional speech recognition: Resources, features, and methods. *Speech communication*, 48 (9):1162–1181, 2006.

[20] Tim Polzehl, Shiva Sundaram, Hamed Ketabdar, Michael Wagner, and Florian Metze. Emotion classification in children's speech using fusion of acoustic and linguistic features. In *Tenth Annual Conference of the International Speech Communication Association*, 2009.

[21] Moataz El Ayadi, Mohamed S Kamel, and Fakhri Karray. Survey on speech emotion recognition: Features, classification schemes, and databases. *Pattern Recognition*, 44(3):572–587, 2011.

[22] Björn Schuller, Anton Batliner, Stefan Steidl, and Dino Seppi. Recognising realistic emotions and affect in speech: State of the art and lessons learnt from the first challenge. *Speech Communication*, 53(9-10):1062–1087, 2011.

[23] Albert Mehrabian et al. *Silent messages*, volume 8. Wadsworth Belmont, CA, 1971.

[24] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[25] André Stuhlsatz, Christine Meyer, Florian Eyben, Thomas Zielke, Günter Meier, and Björn Schuller. Deep neural networks for acoustic emotion recognition: raising the benchmarks. In *Acoustics, speech and signal processing (ICASSP), 2011 IEEE international conference on*, pages 5688–5691. IEEE, 2011.

[26] Arianna Mencattini, Eugenio Martinelli, Giovanni Costantini, Massimiliano Todisco, Barbara Basile, Marco Bozzali, and Corrado Di Natale. Speech emotion recognition using amplitude modulation parameters and a combined feature selection procedure. *Knowledge-Based Systems*, 63:68–81, 2014.

[27] Longfei Li, Yong Zhao, Dongmei Jiang, Yanning Zhang, Fengna Wang, Isabel Gonzalez, Enescu Valentin, and Hichem Sahli. Hybrid deep neural network–hidden markov model (dnn-hmm) based speech emotion recognition. In *Affective Computing and Intelligent Interaction (ACII), 2013 Humaine Association Conference on*, pages 312–317. IEEE, 2013.

[28] Qirong Mao, Ming Dong, Zhengwei Huang, and Yongzhao Zhan. Learning salient features for speech emotion recognition using convolutional neural networks. *IEEE Transactions on Multimedia*, 16(8):2203–2213, 2014.

[29] Haytham M Fayek, Margaret Lech, and Lawrence Cavedon. Towards real-time speech emotion recognition using deep neural networks. In *Signal Processing and Communication Systems (ICSPCS), 2015 9th International Conference on*, pages 1–5. IEEE, 2015.

[30] WQ Zheng, JS Yu, and YX Zou. An experimental study of speech emotion recognition based on deep convolutional neural networks. In *Affective Computing and Intelligent Interaction (ACII), 2015 International Conference on*, pages 827–831. IEEE, 2015.

[31] George Trigeorgis, Fabien Ringeval, Raymond Brueckner, Erik Marchi, Mihalis A Nicolaou, Björn Schuller, and Stefanos Zafeiriou. Adieu features? end-to-end speech emotion recognition using a deep convolutional recurrent network. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 5200–5204. IEEE, 2016.

[32] Andrew Ng. Sequence models. Deeplearning.ai on Coursera, February 2018.

[33] Michalis Papakostas, Evaggelos Spyrou, Theodoros Giannakopoulos, Giorgos Siantikos, Dimitrios Sgouropoulos, Phivos Mylonas, and Fillia Makedon. Deep visual attributes vs. hand-crafted audio features on multidomain speech emotion recognition. *Computation*, 5(2):26, 2017.

[34] Tom M Mitchell et al. Machine learning. wcb, 1997.

[35] C Bishop. Pattern recognition and machine learning (information science and statistics), 1st edn. 2006. corr. 2nd printing edn. *Springer, New York*, 2007.

[36] RS Suton and Andrew G Barto. Reinforcement learning: an introduction. adaptive computation and machine learning, 2002.

[37] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 112. Springer, 2013.

[38] Aurélien Géron. Hands on machine learning with scikit-learn and tensorflow, 2017.

[39] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

[40] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

[41] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[42] Andrew Ng. Improving deep neural networks: Hyperparameter tuning, regularization and optimization. Deeplearning.ai on Coursera, October 2017.

[43] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[44] Steve Lawrence, C Lee Giles, Ah Chung Tsoi, and Andrew D Back. Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks*, 8(1):98–113, 1997.

[45] Yann LeCun, Bernhard E Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne E Hubbard, and Lawrence D Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in neural information processing systems*, pages 396–404, 1990.

[46] Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig. The microsoft 2016 conversational speech recognition system. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 5255–5259. IEEE, 2017.

[47] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657, 2015.

[48] David H Hubel. Single unit activity in striate cortex of unrestrained cats. *The Journal of physiology*, 147(2):226–238, 1959.

[49] David H Hubel and Torsten N Wiesel. Receptive fields of single neurones in the cat's striate cortex. *The Journal of physiology*, 148(3):574–591, 1959.

[50] Nikhil Buduma and Nicholas Locascio. *Fundamentals of deep learning: designing next-generation machine intelligence algorithms*. " O'Reilly Media, Inc.", 2017.

[51] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[52] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.

[53] Claude Elwood Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55, 2001.

[54] Reuven Y Rubinstein and Dirk P Kroese. *Simulation and the Monte Carlo method*, volume 10. John Wiley & Sons, 2016.

[55] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[56] Pierre Baldi and Peter J Sadowski. Understanding dropout. In *Advances in neural information processing systems*, pages 2814–2822, 2013.

[57] Andrew Ng. Neural networks and deep learning. Deeplearning.ai on Coursera, September 2017.

[58] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[59] Justin Salamon and Juan Pablo Bello. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, 24(3):279–283, 2017.

[60] Navdeep Jaitly and Geoffrey E Hinton. Vocal tract length perturbation (vtlp) improves speech recognition. In *Proc. ICML Workshop on Deep Learning for Audio, Speech and Language*, volume 117, 2013.

[61] Martin A Tanner and Wing Hung Wong. The calculation of posterior distributions by data augmentation. *Journal of the American statistical Association*, 82(398):528–540, 1987.

[62] Felix Burkhardt, Astrid Paeschke, Miriam Rolfes, Walter F Sendlmeier, and Benjamin Weiss. A database of german emotional speech. In *Interspeech*, volume 5, pages 1517–1520, 2005.

[63] Sanaul Haq, Philip JB Jackson, and J Edge. Speaker-dependent audio-visual emotion recognition. In *AVSP*, pages 53–58, 2009.

[64] Giovanni Costantini, Iacopo Iaderola, Andrea Paoloni, and Massimiliano Todisco. Emovo corpus: an italian emotional speech database. In *LREC*, pages 3501–3504, 2014.

[65] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin

Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

[66] Brian CJ Moore. *An introduction to the psychology of hearing*. Brill, 2012.

[67] David Pisoni and Robert Remez. *The handbook of speech perception*. John Wiley & Sons, 2008.

[68] Adam K Bosen and Monita Chatterjee. Band importance functions of listeners with cochlear implants using clinical maps. *The Journal of the Acoustical Society of America*, 140(5):3718–3727, 2016.

[69] Andrew Ng. Convolutional neural networks. Deeplearning.ai on Coursera, October 2017.