MSc Dissertation

Privacy Preserving Neural Representation for Speech Audio

by

Nair Vellappally Aiswarya Sasi Kaladharan
SN 19180271

Supervised by:
Dr. Ehsan Shareghi

A dissertation submitted in partial fulfilment

of the requirements for the degree of

Master of Science in Integrated Machine Learning Systems

Department of Electronic and Electrical Engineering

University College London

2019/2020

Abstract

Several speech recognition technologies use cloud based deep learning networks to process voice commands. These voice commands are often encoded in the form of a representation before it is sent to the cloud. This representation might often contain sensitive data of users, which can be accessed by an eavesdropper for malicious purposes. Although considerable research has been done in privacy preservation with respect to images and text data, lot remains to be explored in the area of speech. This project aims to create a neural representation of speech data that can safeguard user privacy but at the same time preserve accuracy of given task. This will be experimented for 3 tasks: emotion classification, valence classification and activation classification. Privacy is implemented through Adversarial Training framework, whereby gender features are rendered indiscriminate, while preserving utility. It is hoped that this work helps towards creating a robust end-to-end system that guarantees user privacy across all three tasks without compromising on quality of service. [1]

[1]Code available on: https://github.com/aishvellappally/Privacy-Preserving-Neural-Network-for-Speech

Acknowledgements

I take this opportunity here to firstly thank my supervisor, Dr.Ehsan Shareghi, without whom this would not have been possible. Before the thesis began, I had very little understanding of how speech based systems work, or how privacy can be incorporated with deep learning. Working on this thesis was a huge learning curve and I am extremely grateful for my supervisor's patience, timely advices, guidance, references and regular brainstorming sessions, which ensured that I stayed on track towards completion of this work. It was really crucial, especially in the midst of uncertain times involving a global pandemic, and being away from all family and friends. I am indebted to my parents and sister back home, who kept inspiring and motivating me throughout this journey. Also, I would like to thank all my dear friends with whom I could share ideas and get feedback. Finally, I would like to thank the faculty of the Department of Electronics and Electrical Engineering, University College London for equipping me with the knowledge and giving me the opportunity to work on this thesis.

Declaration

I have read and understood the College and Department's statements and guidelines concerning plagiarism. I declare that all material described in this report is all my own work except where explicitly and individually indicated in the text. This includes ideas described in the text, figures and tables.
Name: Nair Vellappally Aiswarya Sasi Kaladharan
Date: 28th September 2020

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

As Artificial Intelligence makes waves around the world, the boundaries between humanity and technology are blurring. Today we have assistants who heed our voice commands and carry out various tasks. Statistics show that in 2019, there are about 3.25 billion digital voice assistants being used around the world [11]. Forecasts suggest this number could go up to 8 billion by 2023, exceeding the global population [11]. What has enabled the meteoric rise of voice technology is the advancements made in speech recognition, Natural Language Processing (NLP), deep learning and cloud-based training. The latter has enabled collection of user data from billions of devices across the world and using them to train complex deep learning algorithms on remote cloud servers. The deep neural networks will be then used to answer queries by users directed at the voice assistants.

To enable the cloud-based training scheme, transformed representations are generated locally and then uploaded for the target inference tasks, instead of sending the original voice data [12]. This intermediate representation is usually generated by a small neural network on the user's device. For these neural representations to be effective, there are two primary requirements, utility and privacy [12]. The utility requirement calls for the target prediction task to be performed correctly based on the neural representation, whereas the privacy requirement forces the leakage of private information to be minimized. For example, if the task is to respond to a user command, key content of speech is the only data required. Gender and emotions are sensitive attributes that the user might not want to be leaked. However, these representations may unintentionally leak sensitive information about the user without their permission that can be a target for attack by eavesdroppers, who are listening to the communication channel between the user and the cloud server [13]. This kind of attack can not only capture sensitive information, it can also possibly reconstruct the original speech data itself [13]. Such data can also be used to profile customers, which on one hand might drive innovations but on the other hand raises questions about individual privacy [14]. This profiling can lead to discrimination on the basis race, gender and age in credit ratings, policing and hiring [15].

Furthermore, the representation scheme should respect the limitations of processing and storage capabilities of user devices as well as the capacity of the communication channel over which they are sent to cloud servers [12]. This calls for a third requirement, i.e compactness of the neural representation.

Several approaches are being researched in the context of utility-privacy trade off. Some of the main approaches are based on differential privacy, cryptography and neural representations. In the realm of differential privacy, Abadi et al. showed how privacy can be ensured by adding random

noise to dataset [16]. This method is useful to control leakage of sensitive information at training phase because training phase consists of multiple data points, but at inference stage it is not useful as inference is based on only one data point [17]. Moreover, such a network can be exploited if an attacker has access to the network used to generate anonymity [18].

Bachrach et. al. proposed Cryptonets [19], which is based on Homomorphic Encryption to enforce privacy. These are Artifical Neural Networks (ANN) that can process encrypted data. However, encryption schemes only allow addition and multiplication operations and therefore ANNs must be modified to suit only these operations [19]. Also, encryption adds noise to dataset, and if layers of neural network get deeper, this noise will multiply across layers [19].

Neural representations of speech data are produced by deep learning architectures such as Long Short Term Memory (LSTMs) or Gated Recurrent Units (GRUs). Coavoux et.al. [13] and Jaiswal et. al. [20] have done pioneering work in this regard for text and multimodal data. Despite the research, several gaps exist. Firstly, speech data is more susceptible to sensitive information leakage than lexical data [20]. Learning privacy enhanced speech representation for multiple end tasks like emotion recognition and speaker identification is also a topic open to research [20]. Also, balancing the competing objectives of providing meaningful service quality while obfuscating sensitive information still remains an open problem [21].

The topic of this research is to create privacy-preserving neural representation for speech data. The idea is to build a neural network that can perform speech based tasks effectively (like emotion recognition), but without learning any sensitive information of users such as gender or emotions. Such representation must meet the following criteria:

1. A neural representation of speech data that can enforce privacy without costing utility, i.e., useful information must still be recoverable, but sensitive information must be hidden. For example if end system is just text to speech processing, emotion and gender are variables that are not useful and should be masked to protect privacy, but content of speech must still be recoverable from the neural representation.

2. The ability to obtain the privacy preserving representation in speech format itself, so as to fail attempts at reconstruction of original voice samples

3. Incorporate the idea of compactness of representation. The more compact the representation is, the better.

1.2 Problem Statement

For the scope of this thesis, we have explored the first part of the criteria, i.e., to create a privacy-preserving neural representation of speech. In this context, we define our problem in the following manner: We consider a scenario where a user wants to share a neural representation Z of the speech data X in a way that a latent information U can be inferred, but a sensitive latent information S is obfuscated. Here U represents utility, which can be quantified as the accuracy of end-task, whereas S represents privacy.

Goals

1. Find a representation of X as Z, in order to set upperbound performance for utility. The primary task defines our utility U. In our case, it is the task of emotion recognition.

2. Apply privacy preserving learning to obtain a neural representation that is resistant to gender classification, but maintains the benchmark of utility obtained in the first goal.

3. Achieve the above 2 goals across multiple aspects of emotion recognition - valence, activation and categorical emotions.

1.3 Outline of the Thesis

This thesis consists of 5 chapters, of which this is the first. Chapter 2 sets the stage by giving a brief understanding of speech recognition, neural networks and adversarial training. Chapter 3 reviews related literature, which has also helped us in evaluating and choosing the best methods for privacy learning. Chapter 4 describes in detail our experimental approach and model framework. In Chapter 5, we present our results and explanations for key observations. Finally in Chapter 6, we present a summary of the findings, limitations and future research directions.

Chapter 2

Background Theory

In the context of learning privacy-preserving representations of speech audio, one needs to be aware of deep learning and its various frameworks that can be used to model effective representations, as well as a preliminary understanding of speech feature extraction.The following sections are meant to explain these in a broad sense. This chapter also explains our chosen method of privacy learning based on intensive review of related literature.

2.1 Speech Recognition

In order for speech data to be processed by neural networks, we need to extract certain features. Similar to how pixels make up images, audio data is composed of various frequency components. The idea of a feature extraction method that is similar to how humans perceive sound is what led to the "Mel Scale". It was first developed by Stevens and Volkmann in their iconic paper "The Relation of Pitch to Frequency: A Revised Scale" [22] where they show that frequencies above 1000Hz are not perceived linearly by human. In fact, the human ear does not perceive much of a difference between 5000Hz and 10000Hz as it does between 1000Hz and 5000Hz. This relationship is captured by the Mel Scale, as shown in Fig.2.1. We can convert between Hertz (f) and Mel (m) using the following equation [23] :

$$m = 1125 * \ln(1 + \frac{f}{700}) \tag{2.1}$$

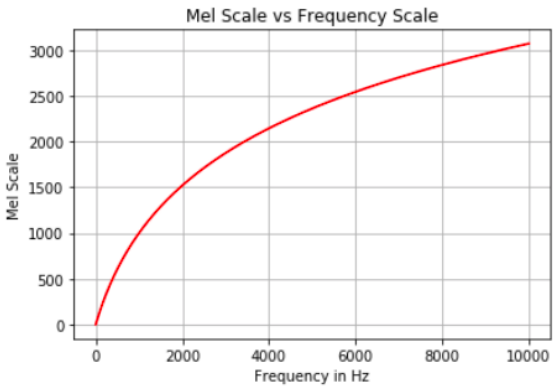


Figure 2.1: Graph showing relationship between frequency and Mel scale.

The coefficients used in converting f to m are not fixed, as what works for various speech recognition algorithms could be different [23].This is of relevance to us as later in the Experiments section we do try out two different speech libraries for Mel Filter Bank (MFB) feature extraction, yielding two different results.

Now that we understood the importance of Mel Scale, let us look at how it is used in feature extraction from speech signals. One of the most commonly used features, proven to be successful in speech recognition and emotion classification are Mel Filter Bank (MFB) energies [24]. In short, MFB energies are based on the Mel Scale, where we capture the energy content of raw speech at various levels of frequency in the Mel scale. This helps in discerning patterns of gender and emotion as different emotions express different energies at different frequencies [24]. Not only that, biologically speaking, voice of women contain higher energies at higher frequencies compared to men. Hence MFB energies can capture such differences, making them an ideal candidate for speech features. The following section describes in detail what goes into extracting MFB energies.

2.1.1 MFB Feature Extraction: Steps at a Glance

1. **Frame the signal**

   In this step we divide the signal into short duration frames, usually between 20-40ms. Framing is important because we would like to get portions of the signal that are stationary, at least statistically, in order to calculate a good spectral estimate. An audio signal is constantly changing in time, as shown in Fig.2.2, but in short frames, the variation in signal amplitude would be small and consistent, almost like periodic signals, as shown in Fig.2.3. This makes it ideal to calculate Fourier coefficients, which is a way of calculating the strength of various frequency components in the signal [23].

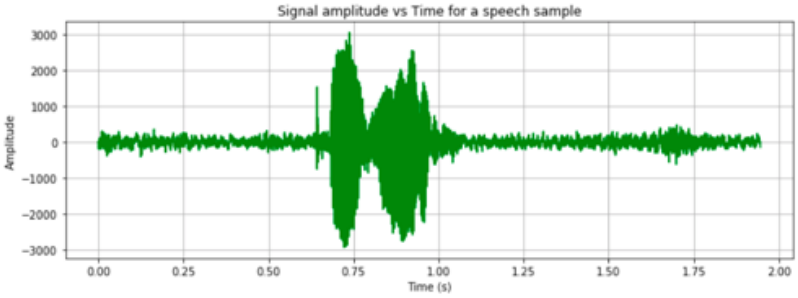


Figure 2.2: A sample of speech signal

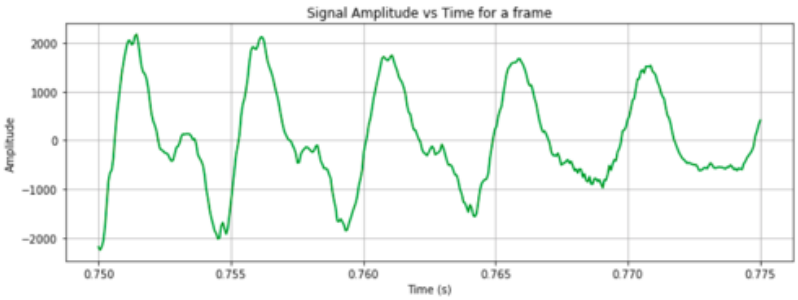


Figure 2.3: A frame from 0.75s to 0.775s from the above signal

2. **Calculate the power spectrum of each frame**

Different hairs in human cochlea, which is a part of our ears, vibrates depending on the frequencies present in the incoming sound wave, firing different nerves that carry the signal to the brain. This motivates us to compute the amount of energies associated with various frequencies in the frame, which is what a periodogram does.

In a periodogram, we calculate the Discrete Fourier Transform (DFT) of the ith frame,$S_i(k)$ as follows:

$$S_i(k) = \sum_{n=1}^{N} s_i(n)h(n)e^{-j2\pi kn/N} \qquad 1 \leq k \leq K \tag{2.2}$$

where $s_i(n)$ denotes the signal belonging to ith frame, $h(n)$ is an N sample long analysis window (e.g. Hamming window), K is the length of the DFT and j is the imaginary unit $\sqrt{-1}$. Multiplying the frame with a window results in the discontinuities of the frequency response being converted into transition bands between values on either side of the discontinuity and helps to smooth abrupt changes in signal, because a DFT cannot handle abrupt signal transitions [25]. The periodogram-based power spectral estimate for the speech frame is given by:

$$P_i(k) = \tfrac{1}{N}|S_i(k)|^2 \tag{2.3}$$

Squaring the DFT gives us power spectral density $P_i(k)$ contained in the frame at different frequencies, which is the periodogram of that frame. Fig.2.4 shows the periodogram estimate for the signal frame in Fig.2.3.

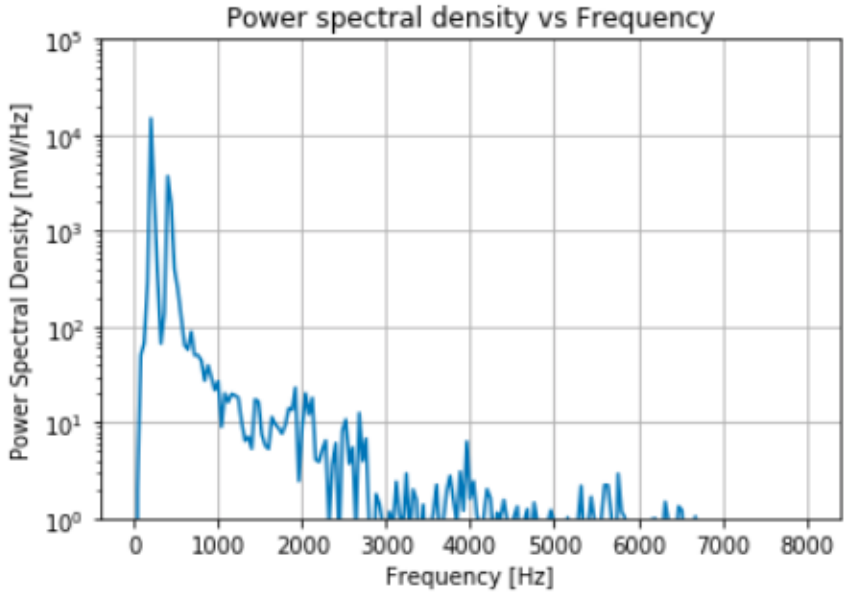


Figure 2.4: Periodogram estimate of the frame shown in Fig.2.3. Power Spectral Density is expressed in milliWatts per Hz.

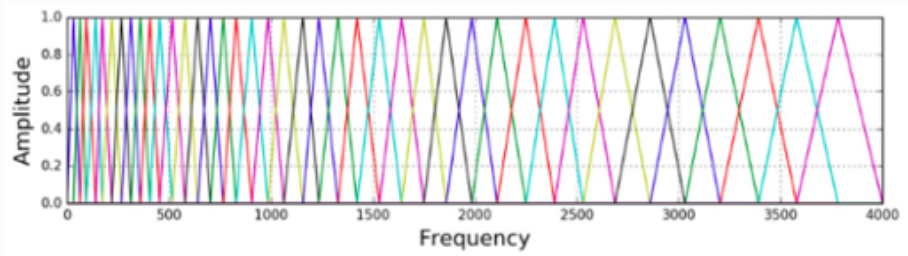


Figure 2.5: 40 filter banks spaced according Mel scale [1].

3. **Apply the Mel Filter Bank to the power spectra, sum the energy in each filter**

   As established in the beginning of this section, the cochlea can not discern the difference between two closely spaced frequencies. This effect becomes more pronounced as the frequencies increase,which is what necessitated a Mel scale. Also, as seen in Fig. 2.4, most of the power is concentrated near the lower frequencies.

   Hence we take clumps of periodogram bins and add the power in that region to get an idea of how much energy exists in that region. These regions of frequencies are spaced according to the Mel Scale - the first filter is very narrow, and as the frequency increases, the filters get wider as we are less concerned about variations at higher frequencies. An example of Mel Filter Bank is shown in Fig.2.5. Usually anywhere between 20 to 40 filter banks are used. In our experiment we have chosen 40 filter banks, to capture maximum variations at the lower frequencies.

4. **Take the logarithm of all filterbank energies**

   Once we have the filterbank energies, we take the logarithm of them. This is also motivated by human hearing: we don't hear loudness on a linear scale. Generally to double the percieved volume of a sound we need to put 8 times as much energy into it. This means that large variations in energy may not sound all that different if the sound is loud to begin with. This compression operation makes our features match more closely what humans actually hear.

2.2 Deep Learning

Neural Network is a machine learning (ML) technique that is inspired by the network of neurons that form our brain. It consists of processing units connected to each other in input, hidden and output layers [26]. What differentiates a deep learning algorithm from a typical neural network is the number of hidden layers: the former has more hidden layers. Connections between units in different layers are associated with a weight value. The product of the inputs and respective weights are taken and added at each unit, as shown in Fig.2.6. The sum is then transformed based on the activation function, which is in most cases is a simple non-linear function such as sigmoid function, tan hyperbolic or rectified linear unit (ReLU). Training a neural network involves adjusting these weights or parameters on the basis of a loss function L, which represents the penalty for mismatching the training data output with predicted output of network. The loss L(θ) on parameters θ is the average of the loss over all training examples $x_1, ..., x_N$, so $L(\theta) = \frac{1}{N_i} \sum L(\theta, x_i)$ [26]. Training essentially involves finding the set of parameters that yields as small a loss as possible.

There are various types of deep learning frameworks such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short Term Memory (LSTM) and Gated Recurrent Units (GRU) [26]. In the context of our project, we have used a combination of Convolutional

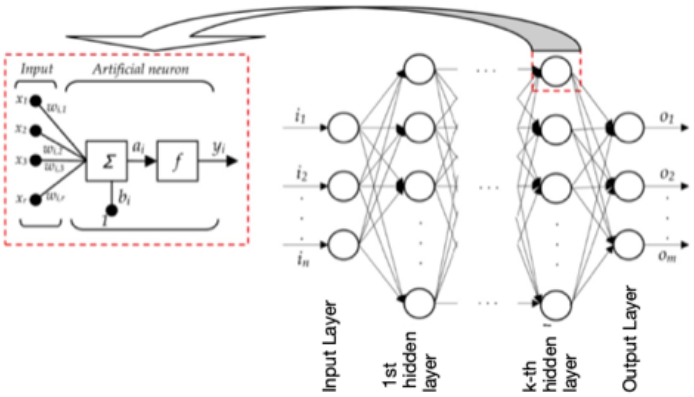


Figure 2.6: Architecture of a deep feed forward neural network [2]

Neural Network (CNN) with Gated Recurrent Units (GRU), which will be described in detail in the following sections.

2.2.1 Convolutional Neural Networks (CNN)

CNNs are essentially Neural Networks that have convolutional layers in them, which will be described in the following sections.

Convolution Layer

This layer is used for extracting relevant features from the input by means of convolution. Convolution in practical terms is performed by passing a filter/kernel over the input to produce a feature map. This kernel, which would be a matrix of weights of a certain width would be slid over the input. The size of the step it takes as it passes over the input is called stride. At every pass, a matrix multiplication is performed between the region of the input covered by the kernel and the kernel weights, and the results are summed onto a feature map. By using several number of filters having different weight structures, numerous convolutions can be performed, resulting in different kinds of feature maps that highlight different features.

Zero-value (or same value as the edge) can be added around the input features, so that the outputs of the convolution are not smaller than the input. This is called padding. While using a CNN, the important hyperparameters to choose are:

1. Kernel size
2. Filter count
3. Stride
4. Padding

In our project, we use frame-wise Mel Filter Bank energies as input, which is essentially a sequential or time-series input. Hence, we use 1D Convolution (1- Dimensional Convolution Layer), which performs convolution along the temporal axis alone as shown in Fig.2.7 The benefit of using CNNs for sequence classification is that they can learn from the raw time series data directly, thereby not having to manually extract features from raw MFB energy features. The model can

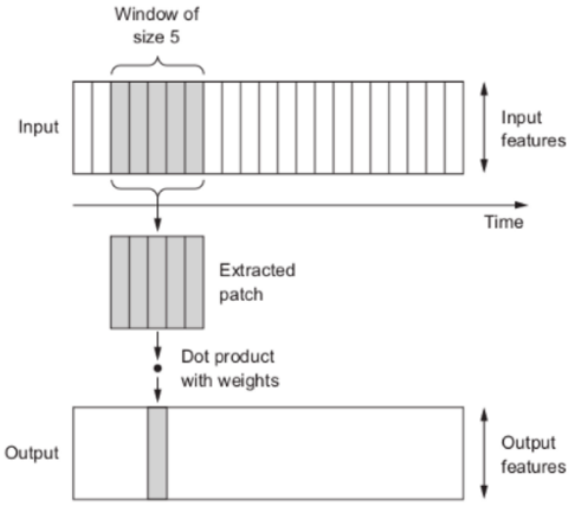


Figure 2.7: 1D Convolution works by producing an output for a temporal patch in the input sequence [3]

learn an internal representation of the time series data and ideally achieve comparable performance to models fit on a version of the dataset with manually engineered features [27].

2.2.2 Recurrent Neural Network

Although our network uses GRU, RNN and LSTMs have to be mentioned to get a full understanding of GRU as it is a variant of the RNN and LSTM.RNNs are neural networks that have "memory", i.e., the output of each unit is dependent not just on present input but also on a combination of previous inputs. This makes RNNs suitable to learn sequential information like text or speech, in which the each utterance or word depends on the previous word and so on.LSTMs and GRUs form a class of RNNs that can store long-term dependencies better than RNNs. In the context of speech this means that LSTMs have better memory of the initial parts of speech than RNNs have. LSTMs and GRUs are used by Google, Amazon and Apple in their voice recognition systems [26].

A typical RNN is illustrated in Fig.2.8. Every node x_i at a time step consists of an input from the previous node and it proceeds using a feedback loop. Each node produces a current hidden state and output by using current input and previous hidden state as in the following equations:

$$h_t = f(W_h h_{t-1} + V_h x_t + b_h) \tag{2.4}$$

$$o_t = f(W_o h_t + b_o) \tag{2.5}$$

where h_t denotes the hidden block of each time step (t), W and V are the weights for the hidden layers in recurrent connection, while b denotes the bias for hidden and output states as f represents an activation function (a non-linear function) applied on each node throughout the network.

Long Short Term Memory (LSTM)

The most common problem with RNN model is that as number of time steps increases, the network fails to learn enough context from time steps of states much before the current step. This is known as long term dependency [4]. Due to the deep layers and recurrent behavior of RNN, exploding

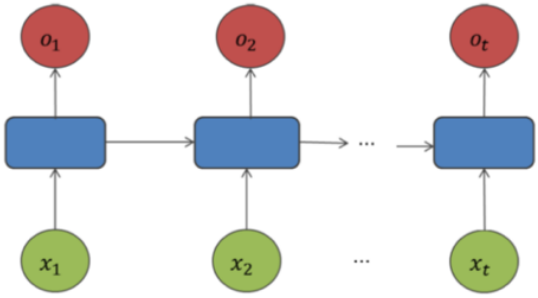
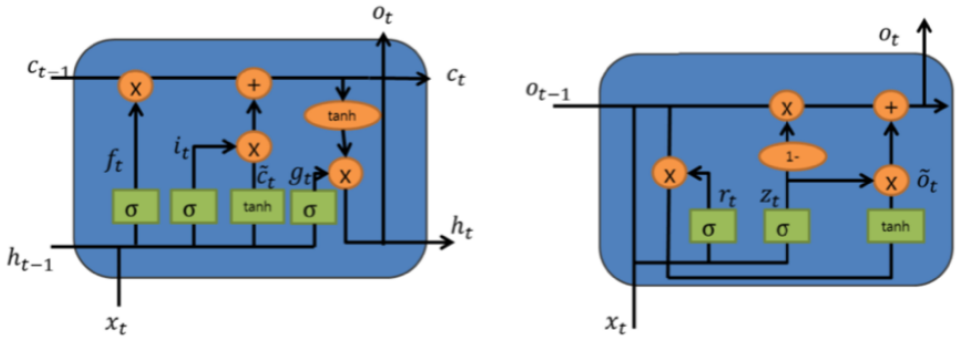


Figure 2.8: Recurrent Neural Networks [4]



(a) LSTM (b) GRU

Figure 2.9: LSTM and GRU cells [4]

and vanishing gradients problems are also encountered often [4]. In order to address this problem, LSTM models are introduced by including memory cells with several gates in the hidden layer. Fig.2.9a illustrates the block of a hidden layer with LSTM cell unit, and the three functions of gate controllers are as follows:

- Forget gate f_t decides which part of long-term state c_t should be omitted.
- Input gate controls which part of c_t should be added to long-term state c_t
- Output gate g_t determines which part of c_t should be read and outputs to h_t and o_t

The following equations show the long-term and short-term states of the cell and the output of each layer in time step in Fig. 2.9a.

$$f_t = \sigma(W_x^T f \cdot x_t + W_{hf}^T \cdot h_{t-1} + b_f) \tag{2.6}$$

$$i_t = \sigma(W_x^T i \cdot x_t + W_h^T i \cdot h_{t-1} + b_i) \tag{2.7}$$

$$o_t = \sigma(W_x^T o \cdot x_t + W_h^T o \cdot h_{t-1} + b_i) \tag{2.8}$$

$$g_t = \tanh(W_x^T g \cdot x_t + W_h^T g \cdot h_{t-1} + b_i) \tag{2.9}$$

$$c_t = f_t \otimes c_{t-1} + i_t \otimes \tilde{c}_t \tag{2.10}$$

$$o_t, h_t = g_t \otimes \tanh(c_t) \tag{2.11}$$

where $W_x f$, $W_x i$, $W_x o$, $W_x g$ denote the weight matrices for the corresponding connected input vector, $W_h f$, $W_h i$, $W_h o$, $W_h g$ represent the weight matrices of the short-term state of the previous time step, and b_f, b_i, b_o, b_g are bias terms.

Gated Recurrent Units (GRU)

In GRU cell unit, the two vectors in LSTM cell are combined into one vector o_t [4]. One gate controller controls both forget and input gates. When z_t outputs 1, the forget gate is opened and the input gate is closed, whereas z_t is 0, the forget gate is closed and the input gate is opened. In this fashion, the input of the time step is deleted every time the previous (t - 1) memory is stored. In the absence of an output gate, it can be said that the GRU is a different implementation of the delivery and combination of the information that LSTM wants to implement. Intuitively, the reset gate determines how to combine the new input with the previous memory, and the update gate decides how much of the previous memory information is retained to calculate the new state, see in Fig.2.9b. GRU use less training parameters and therefore use less memory, execute faster and train faster than LSTM's whereas LSTM is potentially more accurate on datasets with longer sequences [4].

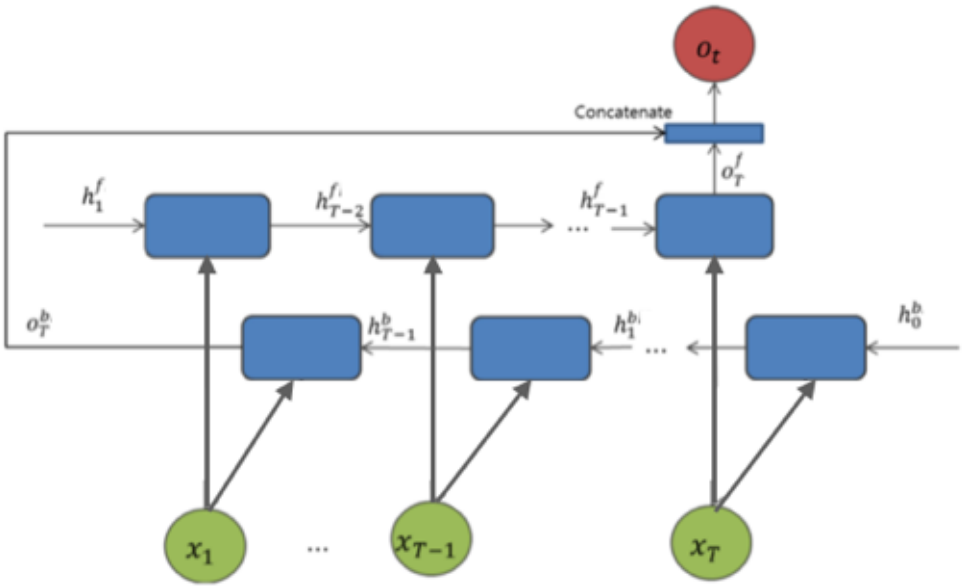


Figure 2.10: Bidirectional GRU

A bidirectional GRU is nothing but utilising two GRU cells in forward and backward directions. The segmented signal inputs $(x_1, ..., x_T)$ are fed into the network for each time step t (t = 1, ..., T) for each GRU cell. Each bi-directional cell contains a parallel of GRU tracks, forward and a backward sequence for utilizing context from the past and future to predict its corresponding class. At the final time step, two parallel tracks of backward and forward GRU tracks are concatenated into a single vector. The forward cell states h_0^f, the backward cell state h_0^b are initialized to zeroes for every layer N. The input x_t at time t, and previous cell states h_{t-1} to produce the output of the corresponding layer o_t , at time t either the through backward or forward stream given its parameter θ^f or θ^b can be defined as:

$$o_T^f, h_T^f = GRU(h_{T-1}^f, x_T; \theta^f) \tag{2.12}$$

$$o_T^b, h_T^b = GRU(h_{T-1}^b, x_T; \theta^b) \tag{2.13}$$

$$o_T = concat(o_T^f, o_T^b) \tag{2.14}$$

2.2.3 Other Layers

In the following sections, we are going to have a look at the other layers used in combination with the convolutional and GRU layers in order to make a complete deep learning network.

Activation Layer

The convolutional layers and fully connected layers are basically multiplication with weights followed by addition with biases. So in these layers, all the operations are linear (of the form wx + b). Activation layer is essentially a function that is used to introduce non-linearity into the network [26]. This function is multiplied with the output of previous layer.

Activation functions are usually functions like sigmoid, hyperbolic tangent (tanh), Rectified linear operation (ReLU), etc [26]. Without a non-linear activation function, even with many layers in the network, summing up these layers would just give a linear function, and hence cannot be used to create effective models for non-linearly separated data. We use ReLU activation function, which returns all positive features, while setting the negative values to zero as shown in Fig 2.11.

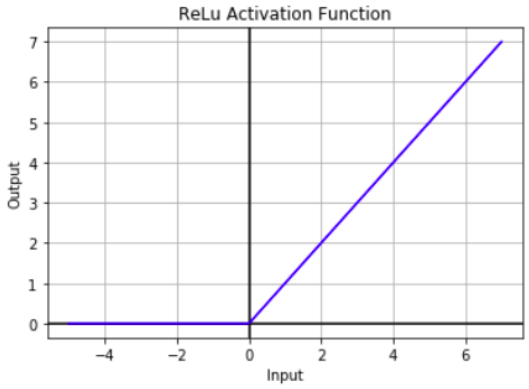


Figure 2.11: ReLU activation function

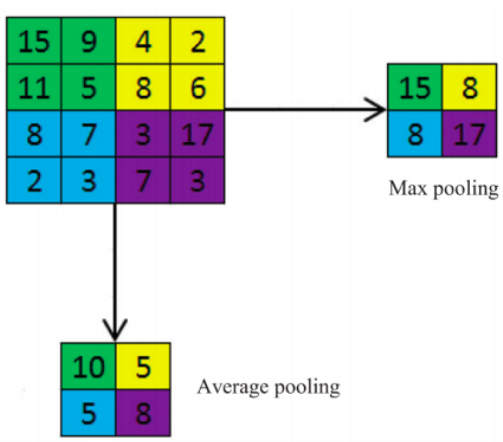


Figure 2.12: Average vs. Max Pooling. [5].

Pooling Layer

Pooling layers are commonly added between convolutions to serve the purpose of down-sampling (reducing the dimensionality). This serves the purpose of decreasing the number of parameters (weights and biases), and shortening training time. They also help in controlling over-fitting to an extent[5].

The most popular types of pooling are max-pooling and average-pooling, illustrated in Fig.2.12. In Max-Pooling, only the maximum value from the window is taken as the output, and the other values are discarded. In average(mean) pooling, the mean of all the values in the window is taken as the output. Max-Pooling is effective in the extracting sharp and most important features, where as average pooling does the extraction smoothly[5]. We use max-pooling with our convolutional layers and average pooling following BiGRU layers, which will be described in detail in the experiment section.

Fully Connected Layer

After going through convolution, pooling and activations, for the final high level reasoning in the network, fully connected layers , also called dense layers. This layer can be viewed as the nodes from a regular neural network, which perform a weighted sum of the inputs and add bias.

Nodes in this layer are connected to every single node in the previous layer (Fig.2.13). Theoretically, it is the final learning phase which maps the features extracted by all previous layers to desired outputs [5].

Softmax

Softmax layer is used as the final layer in classification problems with discrete class labels. This is because it is more appropriate to have one output p_j per class j. p_j is interpreted as the probability of class j for the input. Softmax assigns decimal probabilities that add up to 1, to each class (this is an m-dimensional probability vector, where m is the number of classes). The output p_j for node

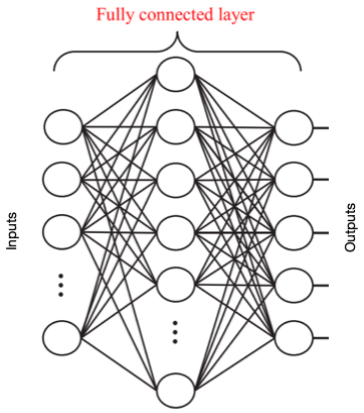


Figure 2.13: Fully connected layers

j is given by

$$p_j = \frac{e^{z_j}{}^L}{Q} \qquad Q = \sum_{k=1}^{m} e^{z_k}{}^L \tag{2.15}$$

where ${z_k}^L$ is the output of node k of the last layer (L). The node with the highest probability value is returned as the most likely class for the given input.

2.2.4 Loss Function

After the input goes through all the layers in the network followed by the softmax layer, model predictions are obtained at the output layer. The purpose of loss function is to estimate how accurate these predictions are, for the classification task at hand, and guides the training to update the model parameters accordingly. Specifically, the loss function compares the output with the target result and measures the amount of inconsistency between the value predicted by the network ($\widehat{y}$) and the actual label (y). Different types of loss functions can be used such as Mean Squared error, L2 loss, cross entropy etc. This information is passed on back wards into the network using backpropagation, an application of chain rule.

We use categorical cross entropy as our loss function, which is generally used for networks that produce multi-class outputs. If we use this loss, we will train a neural network to output a probability over the N classes as follows:

$$loss = -\sum_{i=1}^{N} y_i \log(\widehat{y}_i) \tag{2.16}$$

where y_i is the actual label for the ith class and y_i is the predicted label for the ith class.

2.2.5 Back-Propagation

Back-propagation is the algorithm used to train the weights in the network based on the loss function results. The gradient (direction of steepest ascent) of the loss function with respect to the network's weights is calculated. Since the gradient is required to move in the direction that minimizes the loss function, the weights and biases are updated by shifting them in the direction opposite of the gradient. This process is known as gradient descent. The gradient at the final

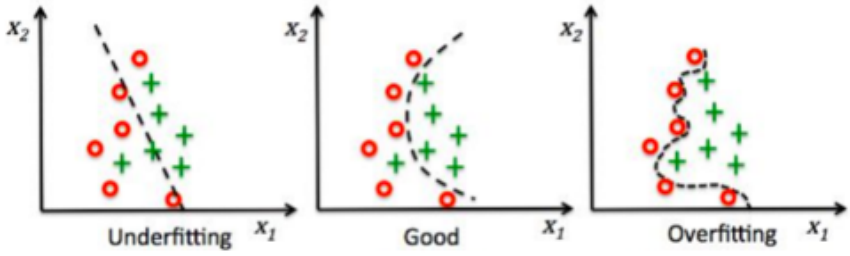


Figure 2.14: Illustration of Underfitting and Overfitting. The red o's and green +'s represent two different classes, between which a suitable decision boundary must be made [6].

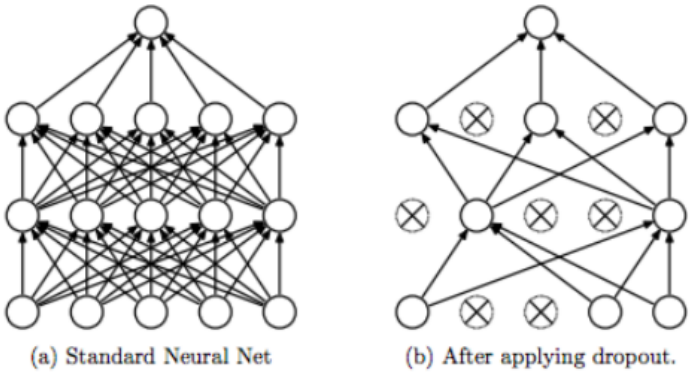


Figure 2.15: Illustration of using dropout in neural networks [7].

(output) layer is computed by taking the partial derivatives of the Loss function with respect to the weights in the preceding layer. In the inner layers, local gradients are calculated with respect to outputs of the next layers. Starting from the final layer, the loss is calculated. Based on the gradient of this, the previous layer weights are modified and local gradients are propagated backwards to allow the flow of loss information to the previous layers.

2.2.6 Regularisation

Over-fitting is one problem that could happen with deep networks with many layers. It happens when the model fits to the training data too closely, such that the model tries to fit even the outliers in training data, and in the process, the decision boundary becomes too convoluted and far from the real boundary (Fig.2.14). This leads to having a very high training accuracy but poor performance on unseen new samples. Thus the testing accuracy becomes low.

Dropout is one of the most effective regularization techniques used to deal with the problem [5]. In this method, every node in the layer except the ones in the output layer, behave like a draw from a Bernoulli distribution with probability p. Each node is present in the network with a probability $1 - p$. This results in us working with a different sub network in each batch, and the weights and biases obtained finally are an average of the weights and biases obtained from each sub network. In each batch during training, some randomly selected activation units are deactivated. i.e, their weights and biases become zero (Fig.2.15). This is done only during the training phase. During testing, the average of weights obtained by training all sub-networks is used.

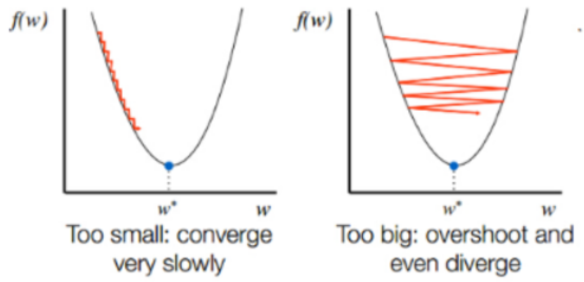


Figure 2.16: Influence of learning rate on gradient descent [8].

2.2.7 Optimization

One of the problems with gradient descent is that the optimization time is strongly affected by learning rate (Fig.2.16), as can be explained by the weight update rule in gradient descent:

$$w_{t+1} \leftarrow w_t - \alpha \frac{\partial L(\theta)}{\partial w_t} \tag{2.17}$$

where α is the learning rate, $L(\theta)$ is the loss function and w_{t+1} and w_t represents future and present weights respectively. If the learning rate is too high, the weights get shifted by a large amount, which leads to the weights overshooting the minimum, oscillating around it and never reaching the minimum. If the learning rate is too low, the weights make very small steps and take too long to reach the optimum [28]. More sophisticated optimizers adapt the learning rate automatically according to the curvature of the function. There are several algorithms in literature that do this - adagrad, Adam etc. Adam optimizer [28] is used in this thesis.

2.3 Domain Adaptation Adversarial Training

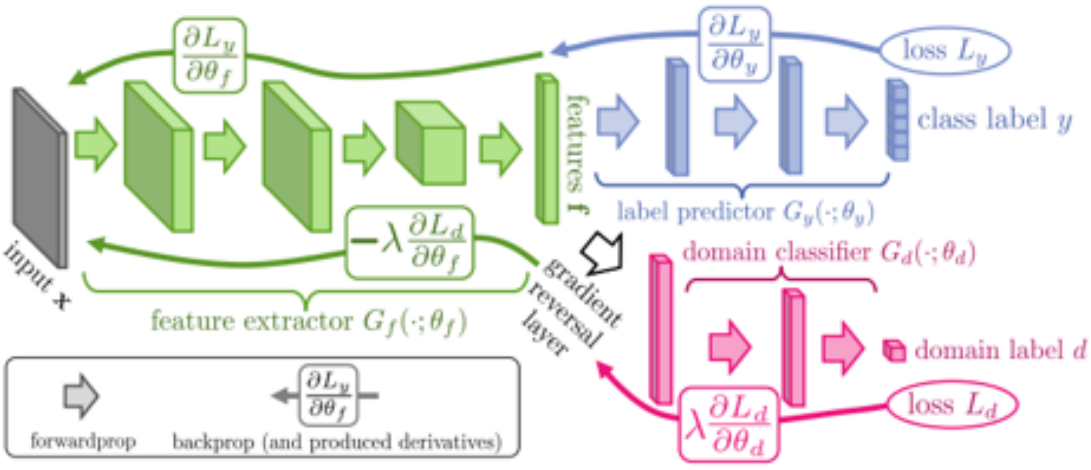


Figure 2.17: Domain Adversarial Neural Network [9].

Originally, Domain Adaptation is a learning paradigm where data at training and test time come from similar but different distributions. In [9], the authors come up with a Domain Adversarial Training of neural networks. The approach implements this idea in the context of neural network architectures that are trained on labeled data from the source domain and unlabeled data from the target domain (labeled target-domain data is not necessary). As the training continues, the approach promotes the development of features that are (i) discriminative for the main learning task on the source domain and (ii) indiscriminate with respect to the change between the domains.

The architecture includes a deep feature extractor (green) and a label predictor (blue) as shown in Fig.2.17. Unsupervised domain adaptation is achieved by adding a domain classifier (red), for the variable to be obfuscated. It is connected to the deep feature extractor through a Gradient Reversal Layer (GRL) which multiplies the gradient by a certain negative constant during the backpropagation. Gradient reversal ensures that the representation of data is as indistinguishable as possible for the domain classifier, thus resulting in the domain-invariant representation.

The Gradient Reversal Layer (GRL) has no parameters associated with it. During the forward propagation, the GRL acts as an identity transformation. During the backpropagation however, the GRL takes the gradient from the next layer and inverts its sign, by multiplying it by -1, or any other factor between 0 and -1, which we term as λ, before passing it to the preceding layer.

Chapter 3

Literature Review

Alepis and Patsakis in [29] showed how sending a raw speech data to a service provider's cloud servers for further analysis can reveal sensitive personal information. They studied the several risks of voice assistants in mobile devices, showing how urgent it is to develop privacy-preserving architectures for speech data by extracting the distinguish features from the speech without compromising individual privacy.

Earlier works on privacy research were based on rule-based systems, which would look at patterns in text and replace them with randomized tokens [30]. Other techniques involved adding noise or randomizing sentences [31]. These methods have the advantage of good interpretability, but they are difficult to scale to large datasets, and as complexity increases, more number of rules will be required [20]. Abadi et al. proposed adding random noise to defend against person identification through data [16]. But this method works better for structured data or images and often results in poorer task performance [20].

In the context of neural networks, works have focused on preventing unintentional memorization of all attributes of input data and their retrieval. Carlini et al. showed that attackers could retrieve sensitive information such as credit card numbers from models that are trained without setting guards against such unintended memorization [32]. They consider a situation where an attacker can access a trained language model and find that the attacker is able to recover sentences from the training data, since the model is trained to assign high probabilities to those sentences. Hence it shows that such memorization is problematic when the training data contains sensitive private information and personal data.

Li et al. in [12] studied the privacy of neural representations of images, and used peak signal to noise ratio to measure amount of information recovered in reconstruction by an attacker with respect to the original image. They were able to find a tradeoff between the privacy of the learned representations and the accuracy of an image classifier that processes these representations instead of original image.

In a recent study, Li et al. proposed a method based on GAN to improve privacy of neural networks in sentiment analysis tasks [33]. However, they only used only one adversary to modify the training of the main classifier and to evaluate the privacy of the representations, which could have led to overestimating privacy [13].

Coavoux et. al. [13] improves on the above two works by modeling an attacker scenario whereby they create an LSTM (Long Short Term Memory) based main classifier which performs classification tasks on the text as intended and an attacker classifier that obtains private information from hidden representations. This attacker classifier is used to measure privacy of the hidden representations based on the accuracy of the private information obtained. As for the defense mechanism,

the authors propose three methods. The first is adversarial classification (multidetasking), where an adversarial classifier that generates private information is trained alongside the main classifier, and the main classifier is penalized each time the attacker classifier is good at obtaining private information. The second is adversarial generation, where the attacker network is trained to reconstruct the original text itself, and not the private information. This type of training is useful to protect the network against any type of attack. The third is declustering, in which the learning process that enables similarity-based clustering of private variables in the representation space is penalized, making it difficult to reconstruct them. The paper proposes future work in the area of hyperparameter tuning in the loss functions to optimize the utility-privacy trade-off. The paper also asserts that multidetasking approach should be the preferred defense as it led to the most stable outcome and is computationally less expensive. Overall, the paper achieves its objective of improving the privacy of neural representations using defense mechanisms with only a very small effect, either positive or negative, on accuracy.

Apart from the above work, research has been done on task-specific privacy preservation for a particular feature in a dataset. For example, Elazar et.al. performed sentiment analysis on text data while preserving privacy [34], while Zhao et al. in [35] looked at gender classification as primary task, while masking ethnicity and age. Work done by Jaiswal et.al [20] is important in this regard. Most previous works on privacy preservation dealt with lexical data. But in [20], the authors tackled the problem of privacy preservation in multimodal representations for emotion recognition task. While the primary aim of most previous works has been to avoid unintentional learning by the application itself, they concentrated on minimizing the ability of an attacker to deliberately retrieve sensitive features from a representation. The work provides the following new contributions: 1) they analyzed how privacy of demographic data differed across modalities, and how adversarial paradigms can be used to improve this privacy, and 2) they were able to develop representations that protect against multiple privacy attacks, while still maintaining good accuracy on emotion classification task. The paper results suggest that audio data is more susceptible to gender information leakage than lexical data. The authors show that the model performs well even in case of protecting speaker identity, not just speakers gender identity. The model was able to achieve satisfactory performance in emotion recognition when protection against both gender and member identification were combined. What the paper leaves as a gap is to develop a model that works well not only for emotion recognition, but also multiple other tasks, all at the same time.

Bertran et. al. in [21] propose a information theoretic approach to enhance privacy. It is the first of its kind paper to study utility-privacy trade-offs using information metrics on higher dimensional data, i.e, visual or image data. The problem is formulated as a distribution-matching problem. The network is adversarial trained to obtain a sanitized representation of original images. Results on facial image data show that the framework is able to handle hard-to-model tasks such as hiding emotion recognition while enabling gender identification; and preserving subject verification capabilities on a subset of individuals, while disallowing this on non-consenting individuals. It also showed excellent performance on the challenging task of identity verification while obfuscating gender. This paper lays an excellent framework for creating obfuscated representation for speech data in the same domain.

Voice conversion is another method that has been used to support the preservation of user privacy by hiding the sensitive representation. The key concept of voice conversion is to convert the speech signal into target speaker while preserving the linguistic contents [36]. Aloufi et.al propose a privacy- preserving framework based on voice conversion to sanitize speech data [36]. It normalizes a sensitive part of the speech (such as emotion) while preserving the signal utility (speech content). CycleGAN architecture is used to transform acoustic features of original emotional utterances such that the modified speech results in a neutral utterance, masking all emotions.

The study of literature in this field shows that adversairal frameworks as explored in [13],[20] and [21] are suitable in implementing privacy in deep learning, which motivated our choice of using domain-adversarial training to enforce privacy in gender labels.

Chapter 4

Data and Experimental Setup

4.1 Goals, Aims and Objectives

We consider a scenario where a user wants to share a neural representation Z of the speech data X in a way that a hidden variable U can be inferred, but a sensitive variable S is obfuscated. X and Y can potentially be supported on the same domain,i.e, speech. Here U represents utility, which can be quantified as the accuracy of end-task, whereas S represents privacy.

Goals

1. Obtain baseline model for classifying categorical emotion,valence and activation from original data X. Determine upper bound for utility, which is classification accuracy for all 3 tasks.
2. Obtain a model that implements privacy in the above said classifier, while maintaining utility.
3. Obtain an attacker classifier that will be used to determine privacy of the original classifier.
4. Privacy could be said to be achieved if an attacker classifier makes predictions of the obfuscated variable with much lesser accuracy as compared to original baseline classifier. For example, emotion classification task on Y must give accuracy close to baseline accuracy, but very low accuracy for gender.

The objective is to classify Valence/Activation/Categorical Emotion labels while obfuscating Gender label. Different kinds of emotions elicit the similar variations in speech aspects, such as similar variation in pitch or loudness, which is why Cowie in [10] proposes just two dimensions, valence and activation, which motivates our objective to include them in our classification of emotions. Valence and Activation are two different aspects of emotion. Valence denotes how positive or negative an emotion is, whereas Activation denotes how suppressed or aroused the emotion is, i.e, it is a scale of "loudness" of the emotion. This scale is aptly illustrated in Fig.4.1, where the horizontal axis represents valence and the vertical axis represents activation.

4.2 Data and Features

4.2.1 Dataset

The first step towards implementing the project was to obtain a speech dataset that contains information on:

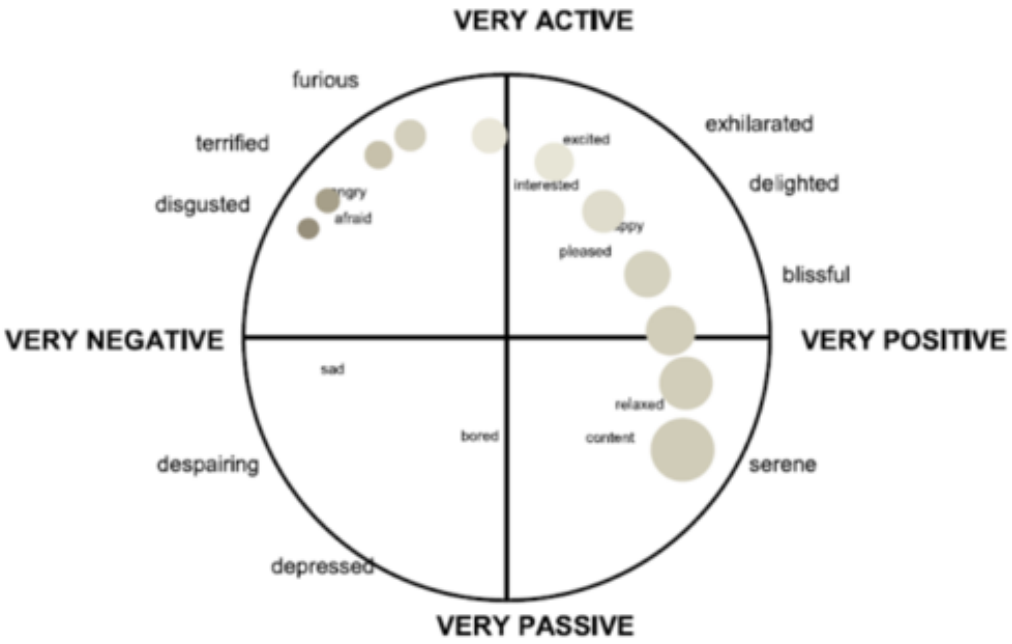


Figure 4.1: Illustration of categorical emotions on a space of valence-activation. X-axis represents scale of Valence and Y-axis represents scale of Activation [10].

1. Emotion aspects - Valence labels, Activation labels and Categorical Emotion labels
2. Gender
3. Speaker Identity

The speech datasets used in this project were IEMOCAP[37] and MSP-Improv[38].

1. The IEMOCAP dataset contains 10039 short duration speech utterances of 5 female and 5 male actors, with 10 different emotions: angry, happy, sad, neutral, disgusted, fearful, surprised, excited, frustrated and other. [37]. The IEMOCAP dataset was created to explore the relationship between emotion, gestures, and speech. Pairs of actors, one male and one female (five males and five females in total), were recorded over five sessions (either scripted or improvised) The data were segmented by speaker turn, resulting in a total of 10,039 utterances (5,255 scripted turns and 4,784 improvised turns). It contains audio, video, and associated manual transcriptions. Table 4.1 contain further information on IEMOCAP dataset.
2. The MSP-Improv dataset consists of 12 speakers in total- 6 males and 6 females. It was collected to capture naturalistic emotions from improvised scenarios while partially controlling for lexical content. The data of 8438 sentences were divided into 652 target sentences, 4,381 improvised turns (the remainder of the improvised scenario, excluding the target sentence), 2,785 natural interactions (interactions between recordings of the scenarios), and 620 read sentences (emotional readings of the target sentences). It contains audio, video, and transcriptions derived from automatic speech recognition (ASR).Table 4.2 presents the dataset statistics.

Table 4.1: IEMOCAP statistics.

ASPECT	CLASS	TOTAL	
		M	F
VALENCE	high	1496	
	middle	993	614
	low	2750	2776
ACTIVATION	high	2126	1996
	middle	1444	1125
	low	1669	1679
CATEGORICAL EMOTION	happy	268	327
	neutral	975	733
	sad	527	557
	anger	514	589
	frustration	1029	820
	excited	598	443
	fear	18	22
	disgust	2	0
	other	2	1
	surprise	51	56
	undefined	1255	1252

At this stage we prepare dataset such that information on gender, emotion and speaker identity are stored in Python's dataframe structure. Information on gender and speaker identity are not explicitly mentioned in the downloaded dataset, which is why we need to extract it from the name of speaker files and prepare it in dataframe format.

4.2.2 Labels

1. **Valence and Activation:** Utterances in IEMOCAP and MSP-Improv were annotated on a five-point Likert scale. The activation and valence values for were averaged over all annotations for an utterance and binned as: "low": [1, 2.75], "mid": (2.75, 3.25], "high": (3.25, max] and one-hot encoded. For example, a 'low' activation would be one-hot encoded as [1,0,0], 'mid' as [0,1,0] and 'high' as [0,0,1].

2. **Categorical Emotion:** A close observation of the dataset statistics reveals that most number of classified emotions belong to either of the 4 categories of Angry, Happy, Neutral or Sad. Many of the emotional attributes were also undefined. Hence only these 4 categories of emotions were considered while the remaining discarded while training categorical emotion classifier. They were also one-hot encoded as [1,0,0,0], [0,1,0,0] etc. in order to make it suitable for the softmax output of the emotion classifier.

3. **Gender:** Labels for Male and Female gender were also prepared and one-hot encoded.

Table 4.2: MSP-IMPROV statistics.

ASPECT	CLASS	TOTAL	
		M	F
VALENCE	high	1612	1689
	middle	1236	986
	low	1359	1556
ACTIVATION	high	2774	2596
	middle	788	918
	low	645	717
CATEGORICAL EMOTION	happy	1284	1360
	neutral	1907	1570
	sad	354	531
	anger	347	446
	other	37	48
	undefined	276	279

4.2.3 Feature Extraction

Mel Filter Bank energies are the chosen features for our project. In order to extract them from raw audio file, several pre-processing steps have to be done, as described in Section 2.1. The MFB features were extracted using two different libraries: Librosa's melspectrogram function [39] as well as python_speech_features library [40]. The two were used in an experiment to find which of the libraries can yield the best accuracies on emotion classification.

Table 4.3: Parameters for Mel Filter Bank feature extraction

	IEMOCAP	**MSP-Improv**
Sample rate	16000 Hz	44100 Hz
Frame length	25ms	25ms
Hop Length	10ms	10ms
Window	Hamming	Hamming
DFT length	400	1103
Min. Frequency	0 Hz	0 Hz
Max. Frequency	8000 Hz	22050 Hz
No. of Mel Filters	40	40

Table 4.3 presents the parameters used in extracting MFB energies using Librosa as well as python_speech_features library. Sample rate refers to the number of bits per second in the audio file. Frame length refers to the length of each frame, whereas hop length refers to the distance between subsequent frames as the window is slid over the audio sample. DFT length denotes the number K in the Discrete Fourier Transform formula in equation 2.2, which was used to compute the spectral estimate of each frame. The minimum and maximum frequency sets the range of frequencies to be considered in the audio : the maximum frequency is by default chosen as $samplerate/2$ according to Shannon's sampling theorem [41]. Finally, the log of the MFB energies were taken, as reasoned in Section 2.1. An example of log of MFB energies of one utterance is shown in Fig.4.2.

These features are z-normalised, as in equation 4.1 before passing to the training model, as feature normalisation helps in bringing all features around the same range of values and speeds up optimisation [42],

$$z = \frac{x-\mu}{\sigma} \tag{4.1}$$

where x represents original feature, μ represents mean of features in a batch and σ represents the standard deviation of features within a batch.

At this stage, we obtain 40 dimensional acoustic features in the form of an $k \times 40$ array where each $k \times 40$ array represents one speech sample, and k represents the number of frames in a sample. k is not a fixed number as different speech samples are of different lengths, hence having different number of frames.

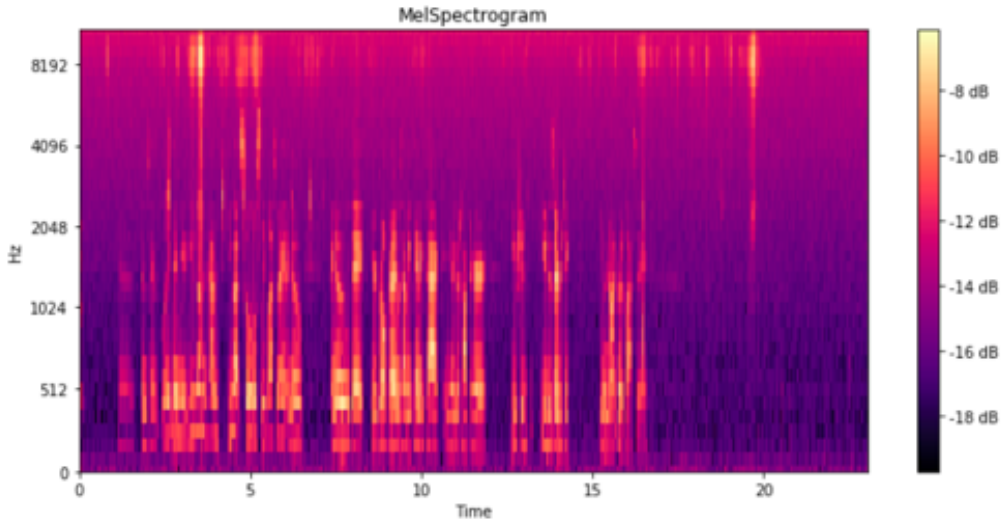


Figure 4.2: Illustration of the MFB features obtained after feature extraction

4.3 Experimental Setup

In this section, we describe in detail the model architecture we used, the recipe for training as well as the metrics to evaluate model performance.

4.3.1 Model Architecture

We have two networks for conducting our experiments: a main network and an attacker network. The main network which is trained to classify the 3 types of emotion labels- valence, activation and categorical emotion, while obfuscating gender at the same time. In other words, the objective of the main network is to maximize performance of emotion classifier while minimising performance of the gender classifier.

The attacker network is an independent classifier used to predict the privacy label, which is gender. Our end goal of privacy is said to be achieved if the attacker network can predict gender with as low accuracy as possible. The networks are described in detail in the following sections.

4.3.1.1 Main Network

The main network is based on state of the art model described in [43]. We pass the sequence of acoustic features through a number of layers of 1D convolution and 1D max-pooling to reduce the temporal resolution of the acoustic input sequence, in order to make training faster. We then pass the resulting sequence to bidirectional gated recurrent unit (GRU) layers [4] for temporal modeling. Previous work showed that GRUs can have comparable performance to that of long short-term memory (LSTM) units while using fewer parameters [4]. Fig.4.3 depicts the model architecture.

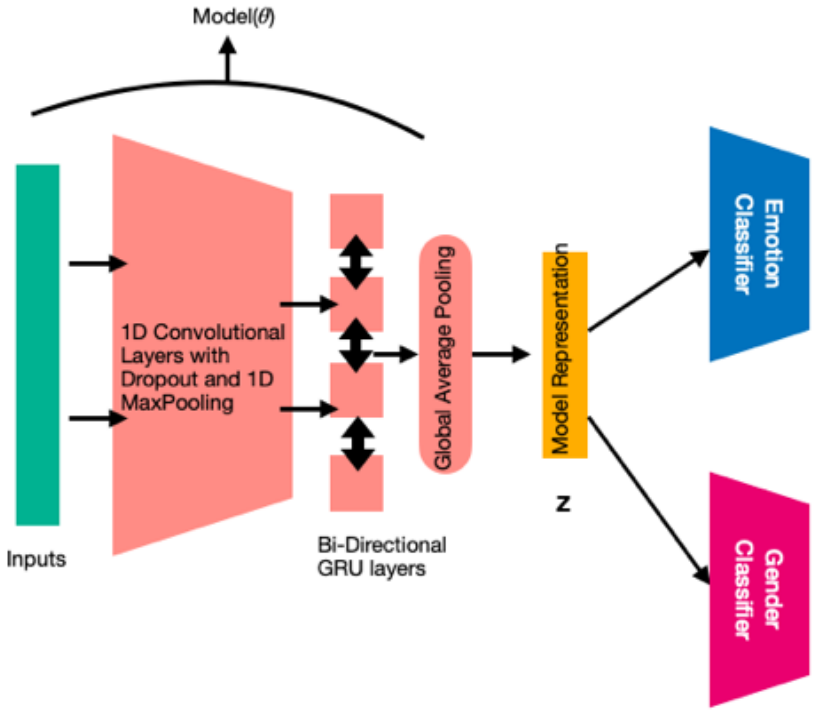


Figure 4.3: Model Architecture-without Privacy

The layers of the model are described in detail as follows:

Model(θ)

1. *1D Convolution Layer:* The acoustic features are passed through a 1D convolution layer that performs temporal convolution,as described in Section 2.2. We choose among $32, 64, 128$ number of kernels, and kernel width of 2,3 or 4 depending on which number gives the best performance for the task at hand.

2. *Activation:* We choose ReLU activation, which activates only the positive valued features.

3. *Dropout:* We choose a Dropout value of 0. This helps in regularising the model, as described in Section 2.2.6 and prevents overfitting.

4. *Batch Normalisation:* This layer makes sure that the output from Activation layer is normalised for the next set of convolution operation to be performed, according to equation 4.1.

5. *1D Max Pooling:* This layer reduces the dimensionality by half, thereby reducing the number of features after each passing layer and increasing training time.

   The above layers are stacked on top of each other, so that convolution-pooling operation is repeated 3 to 5 times, depending on which hyperparemeters yields the best performance.

6. *Bidirectional-GRU:* The output from the 1D max pooling layer from the last set of convolution-maxpooling operation is passed through 2 or 3 bi-directional GRUs, which performs temporal modeling of the feature map learnt by the convolutional layers, both in forward as well as backward directions. This helps in learning a better representation as the output is dependent both on past and future sequences. We choose between 32 or 64 as width of GRU layer.

7. *Global Average Pooling layer:* The output of the final Bidirectional GRU is passed through average pooling layer, which results in a fixed-length feature vector by averaging the sequential outputs, since it was shown that this can result in better discrimination between emotions when compared to only considering the output of the last layer of the bidirectional GRU [43].

   The above layers constitute the Model(θ) which generates the neural representation Z from the 40 dimensional acoustic input X.

Emotion Classifier

The emotion classifier takes in the representation Z as input and estimates valence or activation or categorical emotions using a set of fully connected layers, followed by a softmax layer having 3 nodes for valence/activation prediction and 4 nodes for categorical emotion prediction, since valence and activation consists of 3 classes and categorical emotion consists of 4 classes. The number of fully connected layers chosen is between 1 or 2, with a width of 32 or 64, depending on which yields better performance for the task at hand.

Gender Classifier

The emotion classifier takes in the representation Z as input and predicts gender a set of fully connected layers, followed by a sigmoid activated layer of 2 nodes, as the output is binary- male or female. The number of fully connected layers chosen is between 1 or 2, with a width of 32 or 64, depending on which yields better performance for the task at hand.

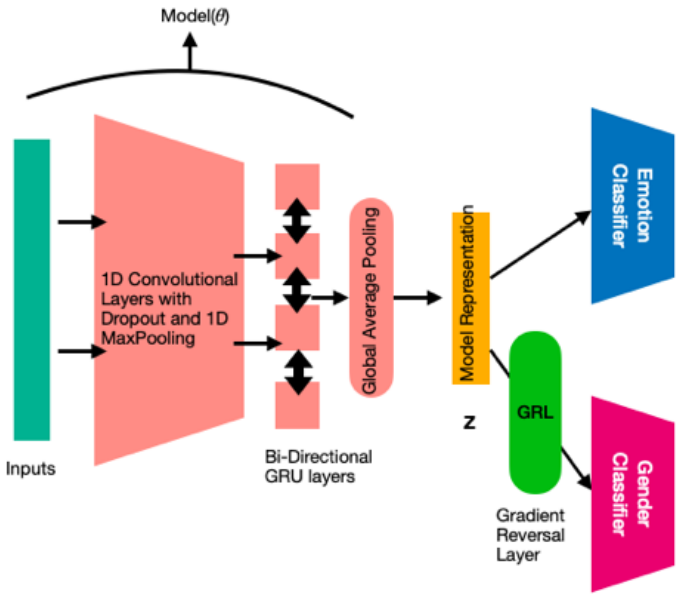


Figure 4.4: Model Architecture-with Privacy

Gender-Leakage

The main network is trained to unlearn gender. To achieve this goal, we use a Gradient Reversal Layer (GRL) [9] as described in Section 2.3. During the backward pass of the training phase, the GRL multiplies the backpropogated gradients by $-\lambda$ (i.e., the strength of the adversarial component). To make the network invariant to gender, we place the GRL function between Model(θ) and the gender classifier, as shown in Fig.4.4.

4.3.1.2 Attacker Network

We assume that the attacker has access to a held-out dataset (an unseen section of the original dataset) with known gender labels. In our case, we give the attacker access to 20% of the dataset, which the main network had not seen. The attacker generates representations for this dataset using the previously described Model(θ). The network then learns to predict gender label from the generated representations using a set of fully connected layers. Since the parameters used to construct the representation are fixed in Model(θ), the attacker only acts upon its own gender classifier parameters to optimize the gender prediction loss. The purpose of the attacker's network is to recover gender information from representations whose labels are unknown.

Though testing using a singular network isn't a guarantee of robustness of the representation to privacy attacks, for the scope of this work, we use a feed forward network, one of the powerful learning methods on a fixed size static representation.

4.3.2 Training

We implement models using the Keras library [44]. We use a weighted cross-entropy loss function [43] for each task and learn the model parameters using the Adam optimizer. We use mini-batches of 32 samples. The samples of our batches were chosen such that similar length samples were grouped together in a batch, which sped up training. We train the networks for a maximum of

Table 4.4: Train-test-validation statistics - Training : sessions 2 to 5, test-validation: session 1 for IEMOCAP dataset.

ASPECT	CLASS	TRAIN		DEV		TEST	
		M	F	M	F	M	F
VALENCE	high	1820	1829	118	96	103	102
	middle	640	413	59	41	90	49
	low	975	921	186	216	193	180
ACTIVATION	high	1438	1374	128	133	147	102
	middle	997	749	75	79	91	66
	low	1009	1031	160	141	148	163

50 epochs and monitor the validation loss for the emotion classifier after each epoch, stopping the training if the validation loss does not improve for five consecutive epochs. Once the training ends, we revert the network's weights to those that achieved the lowest validation loss on the emotion classification task.

For the privacy preserving classification model, we ensure that the chosen model yields as low a unweighted average recall (UAR) for the gender classification task on the validation set. The UAR cannot drop below chance UAR, which is 50%.

We use k-fold cross validation, where each fold consists of speakers from one session for validation and testing. For example, the IEMOCAP dataset consists of 5 sessions with 2 speakers each in a session, a male and female each per session, and the MSP-IMPROV dataset consists of 6 sessions in the same format. An example of data-split for train-validation-testing is shown in Table 4.4.

We use validation samples for hyper-parameter selection and early stopping. The hyper-parameters that we use for the main network include are tabulated in Table 4.5.

For the adversarial emotion classification setups, we use the hyper-parameters that maximize the validation emotion classification performance while minimizing the validation gender classification performance.

For the attacker's model, we use the following hyper-parameters: number of fully connected layers {2, 3, 4}, dense layers width {32, 64}.

As a metric, we report the Unweighted Average Recall (UAR) performance of our models, given the imbalanced nature of our data, as can be seen from Table 4.4. Before defining the UAR, let us look at what recall is defined as:

$$Recall = \frac{t_p}{t_p + f_n} \tag{4.2}$$

where t_p is the number of true positives and f_n is the number of false negatives. The recall is intuitively the ability of the classifier to find all the positive samples,i.e, the samples that actually belong to the class. When recall is extended to a multi-class paradigm, as we have in our case, we average the recall per class to obtain the UAR.

The UAR is defined as follows:

$$UAR = \frac{1}{N} \sum_{i=1}^{N} \frac{t_{p_i}}{t_{p_i} + f_{n_i}} \tag{4.3}$$

where N denotes the number of classes, t_{p_i} and f_{n_i} represents the number of true positives and false negatives per class i.

Table 4.5: Model Hyperparameters

Hyperparameters	Values
No. of conv kernels	{32,64,128}
Conv. kernel width	{2,3}
No. of conv layers	{4,5}
1D max pooling width	{2}
No. of GRU layers	{2,3}
GRU layers width	{32,64}
No. of dense layers	{1,2}
Dense layers width	{32,64}
GRL λ	{0.3,0.5,0.7,1}

4.3.3 Experiment Flow

Performance Metric We define performance for emotion recognition as the ability of the model to correctly classify either activation or valence into 3 categories: low, medium, and high, and categorical emotion into 4 classes: Happy, Angry, Neutral, Sad. We measure performance using UAR (chance is 0.33 for Valence and Activation, whereas 0.25 for categorical emotion).

Gender Leakage Leakage is defined as the ability of a trained gender classifier to predict gender from the representations which are obtained when the network is simultaneously trained to perform the primary task.

Gender Privacy Metric We define privacy metric as the inability of an attacker to be able to correctly predict gender from the representations trained on the primary task, which is emotion recognition. To test this, we use four part training.

1. We train the main network on a dataset (D_1) represented by the pair (x_{D_1}, y_{D_1}) where x is the data input while y is the gender label. We obtain representations from Model(θ) for this dataset as ($Z(x_{D_1})$). 80% of our main datasets will be used for training,testing and validating the main network performance using session wise k-fold validation.

2. We consider that the attacker has access to unused subset of the same dataset (D_2) represented by the pair (x_{D_2}, y_{D_2}). This subset is 20% of the dataset, which we did not use for training the main network. We generate representations $Z(x_{D_2})$ for (x_{D_2}, y_{D_2}) using the $Model(\theta)$ of the main network.

3. We then train the attacker model (M_{att}) to infer gender labels using the representations obtained in the previous step, represented as $M_{att}((Z(x_{D_2}), y_{D_2}))$.

4. Using the model obtained previously (M_{att}), we choose $Z(x_{D_1})$ as inputs, and measure the gender prediction capability of the attacker using UAR($M_{att}((Z(x_{D_1}), y_{D_1}))$. The Gender Privacy Metric of an attacker is then defined as $1 - UAR(M_{att})$.

The range of privacy metric goes from 0 (the attacker is always right) to 0.5 (the attacker has a chance UAR)- i.e, it has a 50% chance of predicting the correct gender.

Chapter 5

Results and Analysis

5.1 Effect of Speech Library

In order to extract Mel Filter Bank features efficiently, we decided to try two different libraries - Librosa [39] and Python_speech_features [40]. Both were tested on IEMOCAP dataset for the experiments in classifying Valence and Activation. The UAR for Valence and Activation are presented in Table 5.1.

Table 5.1: Results obtained for Valence and Activation using two different speech libraries on IEMOCAP datasets. Results reported in % UAR.

	Librosa	Python_speech_features
Valence	47	51.24
Activation	50	60

The results show a significant improvement in UAR when python_speech_features were used, compared to Librosa. Although Librosa is a popular tool for audio analysis and Automatic Speech Recognition, here it is proven otherwise. This could be attributed to how the features are calculated internally: the formula for calculating the Mel scale 2.1 can be used with different coefficients [23], which could yield different results in different implementations. Hence, we decided to use Python_speech_features library for all our experiments.

5.2 Main Classifier (Without Privacy)

5.2.1 Emotion Classifier

For the experiment where the emotion classifier was trained without implementing privacy, the results are tabulated in Table 5.2, Table 5.3 and Table 5.4 for Activation, Valence and Categorical Emotions respectively. UAR is the primary metric used, whereas per class recalls are also reported.

The first observation from the tables is the general imbalance in per class labels. For example, in Table 5.2, the middle class reports lower recall compared to the high and low. This reflects the trend of fewer data samples in middle class as can be seen in the data distribution in Table 4.1, Table 4.2, and Table 4.4.

The same goes for valence and categorical emotion as well, although the test UAR is far more imbalanced compared to Activation. A closer look at the data distribution in Table 4.1, Table 4.2,

and Table 4.4 reveals the reason, as valence and categorical emotion shows higher data imbalance between classes compared to activation.

Table 5.2: Activation classification results on training (TR) and test (TE) sets. Results are presented in % UAR as well as per class % recall.

aspect	UAR		Per Class Recall					
			high		middle		low	
	TR	TE	TR	TE	TR	TE	TR	TE
IEMOCAP	73.63	**60**	75.3	63.35	68.2	55.12	77.4	61.33
MSP-IMPROV	63.5	**61.1**	76.8	64.5	69.5	70.1	78.9	62

Table 5.3: Valence classification results on training (TR) and test (TE) sets. Results are presented in % UAR as well as per class % recall.

aspect	UAR		Per Class Recall					
			high		middle		low	
	TR	TE	TR	TE	TR	TE	TR	TE
IEMOCAP	67.67	**51.24**	56.3	51.3	77.3	41.9	69.4	55.5
MSP-IMPROV	55.5	**51.33**	68.7	51.3	78.9	50	70.1	55.6

Table 5.4: Categorical emotion label classification results on training (TR) and test (TE) sets. Results are presented in % UAR as well as per class % recall.

Dataset	UAR		Per Class Recall							
			angry		happy		neutral		sad	
	TR	TE	TR	TE	TR	TE	TR	TE	TR	TE
IEMOCAP	55.25	**51.5**	57	52.7	30	25	56	51.22	78	77.19
MSP-IMPROV	50.75	**40.7**	36.3	36.5	64.8	40.6	42.5	34.1	60	51.36

5.2.2 Gender Classifier

Table 5.5: Classifier results for IEMOCAP - U represents UAR for Utility (emotion classifier), L represents UAR for Gender classifier, P represents privacy. Note that the network is not trained for privacy in this case.

Aspect	U	L	P
Activation	60	67	36.3
Valence	51.24	63	40
Emotion	51.5	69	42.3

Table 5.6: Classifier results for MSP-IMPROV - U represents UAR for Utility (emotion classifier), L represents UAR for Gender classifier, P represents privacy. Note that the network is not trained for privacy in this case.

Aspect	U	L	P
Activation	61.1	70	37
Valence	51.3	58	42
Emotion	40.7	68	42.35

Results for the gender classifier, which was trained simultaneously with the emotion classifier are presented in Table 5.5 and Table 5.6 for IEMOCAP and MSP-Improv datasets respectively. The gender classifier is trained without the Gradient Reversal Layer, hence these are results when no privacy framework is implemented.

In all the tables, U is the unweighted average recall (UAR). Leakage in the model is shown by L, the lower the better, where chance leakage is 0.5. Privacy metric, represented by P ranges from [0, 0.5], and is the incapability of an attacker to obtain gender information from the representation, the higher the better.

Our results suggest that the network trained to only recognize emotion is generally discriminative for gender as well. For instance we obtain a leakage of 67% when training the for activation on IEMOCAP and of 70% when trained for activation on MSP-Improv dataset. This indicates that the model, although trained for emotion recognition, also learns features that enable it to discriminate between genders. The metric on Privacy, which is decided by how well attacker can classify gender on an unseen dataset, also shows quite low values in the range of 35% to 43%, whereas the ideal is 50%.

This high amount of gender leakage from a model for emotion classifier further necessitates our need to implement privacy preserving feature learning that can obfuscate gender.

5.3 Emotion Classifier (With Privacy)

Previous works have shown that a model trained invariant to gender or age leads to better protection from an attacker who tries to recover that information [13]. Previous works [34] have also shown that while the representations might be trained to reduce the leakage of sensitive information to minimum, the attacker may still be able to recover this information. Hence, we use this incapability of the attacker to infer sensitive information as our primary metric. To test this, we train the adversarial variants of the valence, activation and categorical emotion classifier for both datasets, while making sure that the gender leakage in the models is reduced to as low as possible and compare our results to those in Table 5.5 and Table 5.6. The results of adversarial training with Gradient Reversal Layer for both datasets are presented in Table 5.7 and Table 5.8 . The table also shows the values of UAR for emotion classifier and privacy metric for different values of GRL λ, the strength of the adversarial component. In the tables, a λ of 0.0 represents classifier without privacy, which is equivalent to the results presented in Table 5.5 and Table 5.6.

Our results suggest that:

- The privacy metric is much higher when the representations are trained adversarially with the Gradient Reversal Layer, compared to network trained without it.
- Even when leakage is adversarially reduced to a lower value,the attacker is still able to recover some information about gender.This can be observed from the fact that the Privacy metric never goes to 50%, even though it gets close to it. 50% is the highest possible privacy one can have for gender when defined in terms of UAR metric.

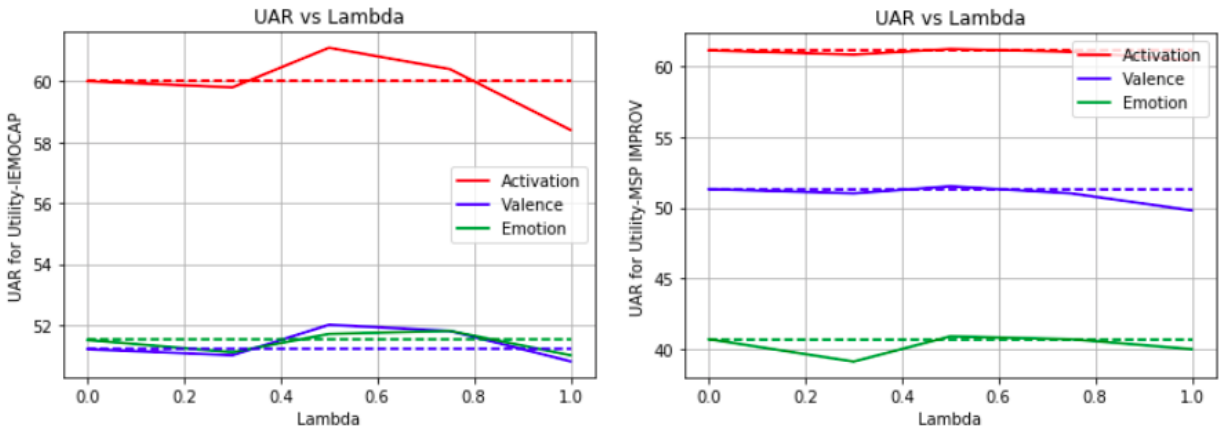
Table 5.7: Privacy on IEMOCAP. L-Leakage = UAR on gender classification by main classifier. P-Privacy = 1-UAR on gender classification by attacker classifier.

Aspect	Lambda														
	0.0			0.3			0.5			0.75			1.0		
	U	L	P	U	L	P	U	L	P	U	L	P	U	L	P
activation	60	67	36.3	59.8	58	47.5	61.1	55.5	48.5	60.4	52.4	48.8	58.4	53	40
valence	51.24	63	40	51	57	48	52	55	48.5	51.8	54	49	50.8	55.3	42
emotion	51.5	69	42.3	51.1	55.5	47.5	51.7	55	48	51.8	54.8	48.5	51	57	42

Table 5.8: Privacy on MSP-IMPROV. L-Leakage = UAR on gender classification by main classifier. P-Privacy = 1-UAR on gender classification by attacker classifier.

Aspect	Lambda														
	0.0			0.3			0.5			0.75			1.0		
	U	L	P	U	L	P	U	L	P	U	L	P	U	L	P
activation	61.1	70	37	60.8	55	46.5	61.2	54	47	61	54	48	60.4	58	44
valence	51.3	58	42	51	55	47.8	51.5	54	48.5	51	53	49	450.8	56	41
emotion	40.7	68	42.35	39.12	56	47.5	40.9	55	48	40.7	55.3	48.4	40	56	43

5.3.1 Effect of strength of adversarial component on Utility



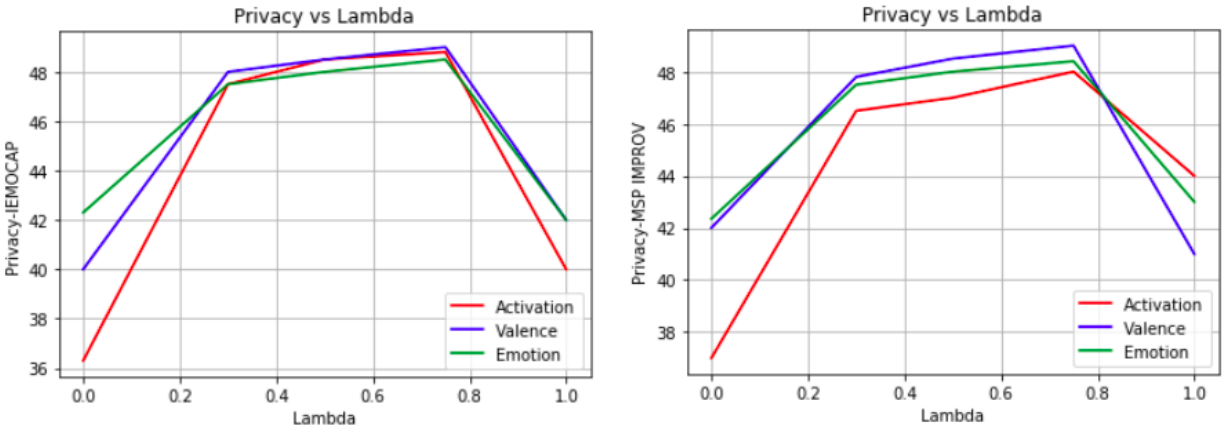
(a) IEMOCAP (b) MSP-IMPROV

Figure 5.1: Effect of λ on utility

Previous research has shown that training a model invariant to a dataset variable might lead to drop in performance on the primary task, especially when it is shown that features learnt for the primary task of emotion classification is also discriminative of the secondary task of gender classification [45].

Our results suggest that, in general there is no significant effect on the performance on utility, which is the performance of the primary task, when we train privacy preserving networks. We find that the performance is either maintained,or a slight increase or decrease, as shown in Fig.5.1. There is in fact a significant increase for activation and valence in IEMOCAP dataset for λ between 0.4 and 0.8 , implying that in some cases making the model invariant to gender increases its robustness by not learning replicable associations between gender and emotion label.

5.3.2 Effect of strength of adversarial component on Privacy



(a) IEMOCAP (b) MSP-IMPROV

Figure 5.2: Effect of λ on privacy metric

Fig.5.2 shows the effect of the strength of the adversarial component λ on privacy metric, based on the results presented in Table 5.7 and Table 5.8. One would expect that as the strength of the adversarial component increases, the network would learn features more invariant to gender. But the graph in Fig.5.2 shows that the relation between λ and privacy metric is not proportional, as one would expect.

We observe that the the attacker is usually less capable of predicting gender from representations when $= 0.50$ as compared to when $\lambda = 0.75$. But on the contrary, we see a decrease in the privacy metric when we move from $\lambda = 0.75$ to $\lambda = 1.00$ for activation, valence and categorical emotion. For example, the privacy metric for the audio system trained on IEMOCAP decreases from 48.8% to 40% as one goes from $\lambda = 0.75$ to $\lambda = 1.00$. The decrease in the privacy metric as $\lambda \to 1$ could be attributed to overfitting of data [46] when being trained for invariance to the gender variable, which the attacker network might be able to exploit [20]. This suggests that an increase in the strength of the adversarial component does not necessarily correlate to an increase in the privacy metric, and one must choose a λ based on trial and error.

5.4 Comparison with existing works

In [20], the authors implement privacy framework using Gradient Reversal Layer for multiple datasets, with activation and valence as primary tasks and gender as secondary task. They also used a CNN-GRU framework to model the MFB features. The comparison is presented in Table 5.9.

The comparison shows that we have done better in Privacy metric for both datasets, whereas for utility of primary task, we have done comparably and even slightly better for valence in case of MSP-Improv dataset, whereas we are worse off in both tasks in case of IEMOCAP dataset.

The difference in results could be due to the following reasons:

1. Differences in data pre-processing: The work in [20] does not describe in detail the steps involved in pre-processing of the speech data, nor have they specified the library they used for it. This could lead to differences in the features fed to the model in the first place, adversely impacting the performance in the primary task of emotion recognition, especially with the IEMOCAP dataset.

2. Difference in model framework and hyper-parameters : This could also cause slight variations in result.The work does not go into the details of training and validation procedure, or the split in dataset for the main-network - attacker-network framework.

Table 5.9: Comparison of Utility and Privacy with Jaiswal et.al.

Dataset		Utility		Privacy	
		Jaiswal et.al.	Our Work	Jaiswal et.al.	Our Work
IEMOCAP	Valence	60	51.24	45	49
	Activation	67	60	43	48.8
MSP-Improv	Valence	51	51.3	48	49
	Activation	63	61.1	44	48

3. The boost in privacy in our results could also be attributed to our model's inability to learn the primary task of emotion recognition as well as the work in [20], which could again be traced to the quality of features used in training.

Chapter 6

Conclusion and Future Work

This thesis has successfully explored it's titular goal of creating a privacy preserving neural representation for speech audio, in a experimental set up where the main task was emotion recognition and the variable to be obfuscated was gender. We showed that it is possible to obtain sufficient privacy in gender inference by a third-party network (the attacker network) while still maintaining or even improving accuracies on the primary tasks, using domain adversarial training paradigm [9]. Hence we have met our goal of preserving utility while maintaining privacy simultaneously. Comparison with previous research [20] also shows our network is able to perform well on the privacy aspect, although falling short on the utility aspect of emotion recognition, which was attributed to differences in feature extraction and training procedures.

6.1 Limitations and Assumptions

In this section, we would like to shed light on some of the limitations we had and assumptions made in the course of this work.

- The network was tested only on two datasets, due to lack of accessibilty to similar datasets with high number of samples. More testing needs to be done to identify any abnormalities or conclusive patterns.
- The attacker network here was assumed to have access to the model generating the neural representation, Model(θ).
- We tested against only one kind of attacker network, whereas a third party could develop a different kind of attacker network, which could yield different results.
- The work has only established utility in the context of emotion recognition, and privacy in the context of gender. It would be interesting to explore other facets such as speaker identification in the realm of privacy preservation.
- The output of the work may have suffered from not using the best tools for pre-processing speech data, which may affect the quality of the features extracted.

6.2 Directions for future work

We had stated three criteria for a privacy preserving representation in the Introduction of this thesis. This thesis has fulfilled the first criteria, which was to obtain a neural representation that

can enforce privacy without costing utility. The direction for future work is based on the second and third criteria -

- The ability to obtain the privacy preserving representation in speech format itself, so as to fail attempts at reconstruction of original voice samples.
- Incorporate the idea of compactness of representation. The more compact the representation is, the better.

The second criteria aims at creating a sanitised representation of speech.The sanitised representation is nothing but reconstructed audio, with privacy enforced in it. This could be done using a Autoencoder or Variational Auto Encoder. Autoencoders consists of encoder-decoder networks. The encoder learns a lower dimensional representation of the input called the latent space and the decoder gets a function that is very similar to the original input back from the latent space. In speech processing, autoencoders are used for speech denoising.

Variational autoencoder (VAE) is a type of autoencoder that encodes the input into a Gaussian distribution, i.e., the latent space is made to follow a Gaussian distribution. Samples from this latent space are used by the decoder to generate output samples that are representative of the original dataset. Hence VAEs are generative neural networks. Both autoencoders and VAEs are important in the context of this work as they are useful in learning obfuscated representations of speech data.

Also, most previous works have not taken compactness of representation with respect to communication channel constraints into account, which forms our third criteria. The size of representation generated by the encoder can be controlled by adding constrained optimization terms to the objective function, which is useful in obtaining compact representations of speech data.

The work is expected to take research in privacy preserving neural networks a step forward, especially in a difficult domain like speech. The methods developed by this work can be applied in a scenario where a user speaks into smart devices, such as Alexa or Siri, and the device has to send this data in an encoded format to external cloud servers, where a deep learning algorithm processes the speech representation. Common voice recognition systems use cloud based neural networks for speech processing. The impact of this work is to make any attempt to gain sensitive information from this representation a failure, whether the attempt is intentional (like an eavesdropper attack) or unintentional leakage. The research has huge application in speech recognition technologies such as Siri, Alexa etc to make it more robust to sensitive information and at the same time carry out voice based tasks effectively.The main beneficiaries of this project will be end users, who are guaranteed that they get the best of services in speech recognition technology without compromise on their sensitive data. In the day and age of growing concerns for data protection and privacy of users, works like this could contribute in some way to a safer user experience.

References

[1] Haytham M. Fayek, "Speech processing for machine learning: Filter banks, mel-frequency cepstral coefficients (mfccs) and what's in-between," 2016.

[2] "Artificial intelligence techniques for modelling of temperature in the metal cutting process," *Metallurgy–Advances in Materials and Processes*, pp. 153–176, 2012.

[3] Francois Chollet, *Deep Learning with Python*, Manning Publications Co., USA, 1st edition, 2017.

[4] H. M. Lynn, S. B. Pan, and P. Kim, "A deep bidirectional gru network model for biometric electrocardiogram classification based on recurrent neural networks," *IEEE Access*, vol. 7, pp. 145395–145405, 2019.

[5] Waseem Rawat and Zenghui Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural Computation*, vol. 29, pp. 1–98, 06 2017.

[6] Stanford, "Introduction to convolutional neural networks," https://web.stanford.edu/class/cs231a/lectures/intro_cnn.pdf, 2018.

[7] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[8] Sanjiv Das Subir Varma, "Chapter 7 training neural networks part 1," https://srdas.github.io/DLBook/GradientDescentTechniques.html, 2018.

[9] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky, "Domain-adversarial training of neural networks," 2015.

[10] Roddy Cowie and Randolph R Cornelius, "Describing the emotional states that are expressed in speech," *Speech communication*, vol. 40, no. 1-2, pp. 5–32, 2003.

[11] Statista, "Number of voice assistants in use worldwide 2019-2023," https://www.statista.com/statistics/973815/worldwide-, 2020.

[12] Meng Li, Liangzhen Lai, Naveen Suda, Vikas Chandra, and David Z. Pan, "Privynet: A flexible framework for privacy-preserving deep neural network training," 2017.

[13] Maximin Coavoux, Shashi Narayan, and Shay B Cohen, "Privacy-preserving neural representations of text," *arXiv preprint arXiv:1808.09408*, 2018.

[14] Centre for Data Ethics and Innovation, "Smart speakers and voice assistants," https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/831180/Snapshot_Paper_-_Smart_Speakers_and_Voice_Assistants.pdf, 2020.

[15] S. Hajian and J. Domingo-Ferrer, "A study on the impact of data anonymization on anti-discrimination," in *2012 IEEE 12th International Conference on Data Mining Workshops*, 2012, pp. 352–359.

[16] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 308–318.

[17] Daniel Kifer and Ashwin Machanavajjhala, "No free lunch in data privacy," in *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, 2011, pp. 193–204.

[18] M. Dias, A. Abad, and I. Trancoso, "Exploring hashing and cryptonet based approaches for privacy-preserving speech emotion recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 2057–2061.

[19] Ran Gilad-Bachrach, Nathan Dowlin, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing, "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy," in *International Conference on Machine Learning*, 2016, pp. 201–210.

[20] Mimansa Jaiswal and Emily Mower Provost, "Privacy enhanced multimodal neural representations for emotion recognition," 2019.

[21] Martin Bertran, Natalia Martinez, Afroditi Papadaki, Qiang Qiu, Miguel Rodrigues, Galen Reeves, and Guillermo Sapiro, "Adversarially learned representations for information obfuscation and inference," in *International Conference on Machine Learning*, 2019, pp. 614–623.

[22] S. S. Stevens and J. Volkmann, "The relation of pitch to frequency: A revised scale," *The American Journal of Psychology*, vol. 53, no. 3, pp. 329–353, 1940.

[23] S. Umesh, Leon Cohen, and Douglas Nelson, "Fitting the mel scale," 04 1999, vol. 1, pp. 217 – 220 vol.1.

[24] Carlos Busso, Sungbok Lee, and Shrikanth S Narayanan, "Using neutral speech models for emotional speech analysis," in *Eighth Annual Conference of the International Speech Communication Association*, 2007.

[25] Prajoy Podder, Tanvir Khan, Mamdudul Khan, and M. Rahman, "Comparative performance analysis of hamming, hanning and blackman window," *International Journal of Computer Applications*, vol. 96, pp. 1–7, 06 2014.

[26] A. Shrestha and A. Mahmood, "Review of deep learning algorithms and architectures," *IEEE Access*, vol. 7, pp. 53040–53065, 2019.

[27] T. Sainath, Brian Kingsbury, Abdel rahman Mohamed, and B. Ramabhadran, "Learning filter banks within a deep neural network framework," *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pp. 297–302, 2013.

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

[29] Efthimios Alepis and Constantinos Patsakis, "Monkey says, monkey does: security and privacy on voice assistants," *IEEE Access*, vol. 5, pp. 17841–17851, 2017.

[30] Jose Maria Gomez-Hidalgo, Jose Miguel Martin-Abreu, Javier Nieves, Igor Santos, Felix Brezo, and Pablo G Bringas, "Data leak prevention through named entity recognition," in *2010 IEEE Second International Conference on Social Computing*. IEEE, 2010, pp. 1129–1134.

[31] Alexandre Evfimievski, "Randomization in privacy preserving data mining," *ACM Sigkdd Explorations Newsletter*, vol. 4, no. 2, pp. 43–48, 2002.

[32] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song, "The secret sharer: Evaluating and testing unintended memorization in neural networks," in *28th {USENIX} Security Symposium ({USENIX} Security 19)*, 2019, pp. 267–284.

[33] Yitong Li, Timothy Baldwin, and Trevor Cohn, "Towards robust and privacy-preserving text representations," *arXiv preprint arXiv:1805.06093*, 2018.

[34] Yanai Elazar and Yoav Goldberg, "Adversarial removal of demographic attributes from text data," *arXiv preprint arXiv:1808.06640*, 2018.

[35] Han Zhao, Jianfeng Chi, Yuan Tian, and Geoffrey J Gordon, "Adversarial privacy preservation under attribute inference attack," *arXiv preprint arXiv:1906.07902*, 2019.

[36] Ranya Aloufi, Hamed Haddadi, and David Boyle, "Emotionless: Privacy-preserving speech analysis for voice assistants," *arXiv preprint arXiv:1908.03632*, 2019.

[37] Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N Chang, Sungbok Lee, and Shrikanth S Narayanan, "Iemocap: Interactive emotional dyadic motion capture database," *Language resources and evaluation*, vol. 42, no. 4, pp. 335, 2008.

[38] C. Busso, S. Parthasarathy, A. Burmania, M. AbdelWahab, N. Sadoughi, and E. M. Provost, "Msp-improv: An acted corpus of dyadic interactions to study emotion perception," *IEEE Transactions on Affective Computing*, vol. 8, no. 1, pp. 67–80, 2017.

[39] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto, "librosa: Audio and music signal analysis in python," 2015.

[40] James Lyons, Darren Yow-Bang Wang, Gianluca, Hanan Shteingart, Erik Mavrinac, Yash Gaurkar, Watcharapol Watcharawisetkul, Sam Birch, Lu Zhihe, Josef Hölzl, Janis Lesinskis, Henrik Almér, Chris Lord, and Adam Stark, "jameslyons/python_speech_features: release v0.6.1," Jan. 2020.

[41] C.E. Shannon, "Communication in the presence of noise," *Proceedings of the IRE*, vol. 37, no. 1, pp. 10–21, jan 1949.

[42] Dalwinder Singh and Birmohan Singh, "Investigating the impact of data normalization on classification performance," *Applied Soft Computing*, p. 105524, 2019.

[43] Zakaria Aldeneh, Soheil Khorram, Dimitrios Dimitriadis, and Emily Mower Provost, "Pooling acoustic and lexical features for the prediction of valence," in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, 2017, pp. 68–72.

[44] François Chollet, "keras," https://github.com/fchollet/keras, 2015.

[45] Z. Meng, J. Li, Z. Chen, Y. Zhao, V. Mazalov, Y. Gong, and B. Juang, "Speaker-invariant training via adversarial learning," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 5969–5973.

[46] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry, "Adversarially robust generalization requires more data," in *Advances in Neural Information Processing Systems*, 2018, pp. 5014–5026.