Performance analysis of meta-learning and contrastive learning for speech emotion recognition

King Gandhi, Raeshak

Award date:
2022

Technische Universiteit
Eindhoven
University of Technology

Department of Mathematics and Computer Science
Interconnected Resource-aware Intelligent Systems Research Group

Performance analysis of meta-learning and contrastive learning for speech emotion recognition

Master Thesis

Raeshak King Gandhi

Supervisors:
Prof. Dr. ir. Nirvana Meratnia
MSc. Vasileios Tsouvalas

Eindhoven, June 2022

Acknowledgement

This thesis is an opportunity provided by the Department of Mathematics and Computer Science at Eindhoven University of Technology. I am extremely grateful to the support provided by various people for the duration of my master's thesis. First of all, I would like to express my utmost gratitude to Prof. Dr. ir. Nirvana Meratnia for providing me the opportunity to work on this wonderful thesis. I would also like to thank her for the peerless guidance provided by her for the duration of my thesis. Next, I would like express my deepest thanks to Vasileios Tsouvalas for the extensive help and support he provided throughout the thesis, both technical and theoretical, no matter the time or place. I would not have been able to complete the thesis without their help nor would I have developed a research oriented mindset.

I would also like to thank my parents for providing me with abundant financial and emotional support which enabled me to brave tough waters throughout my Master's here at TU/e. Finally, I would like to thank my friends for their constant encouragement during my time entire time abroad.

Abstract

Emotion recognition has a variety of applications in many fields. Audio is one of the means of detecting emotions, the others being facial and body movement recognition, and measuring biometric signals. Speech Emotion Recognition (SER), however, it is not extremely reliable. One of the problems faced by SER is the problem of Domain Generalization (DG). When a SER model is trained in one language corpus, it does not perform well in predicting data from another corpus of the same language, much less on a corpus from another language. In this thesis, DG methods perfected for use on images are used to perform DG on Log-Mel spectrogram of audio emotion data from different datasets belonging to various languages. Two methods are selected, i.e., Meta-Learning Domain Generalization (MLDG) [29] and Self-supervised Contrastive Regularization (SelfReg) [48]. MLDG and SelfReg are used to test performance of generalization of emotions over different languages. In case of low performance, this thesis tries to identify which part of the model is the cause of this low performance. It also tries to identify different avenues in which models can be improved in the future. Finally, the input to the DG models are varied by performing augmentations and/or using embeddings from another pre-trained model, called [77].

Keywords: Domain Generalization, Meta-learning, Contrastive learning, deep learning, Speech emotion recognition, augmentation.

Contents

List of Figures

List of Tables

List of Acronyms

CNN Convolutional Neural Network

CCE Categorical Cross-Entropy

MLDG Meta-Learning Domain Generalization

SelfReg Self-supervised Contrastive Regularization

DG Domain Generalization

SER Speech Emotion Recognition

GAN Generative Adversarial Networks

DL Deep Learning

IDCL Inter-Domain Curriculum Learning

RNN Recurrent Neural Networks

SGD Stochastic Gradient Descent

STFT Short-Time Fourier Transform

Chapter 1

Introduction

Emotion recognition has many applications ranging from simple mood recognition for selecting a music playlist, to a more complex utility in monitoring behavior of people in a certain place to prevent illegal activities. Recognizing positive emotions may be used in smartphones to capture pictures or to receive a review of a movie on completion. Recognizing negative emotions, for example, may help in healthcare domain to alert officials regarding the current (negative) state of the patient in psychiatric institutions. Emotions could be determined from one's facial movements and expressions, tracking one's biometric data such as pulse and EEG data, or via audio. Audio is considerably less invasive, when compared to biometric data or use of video recordings, and one can retain privacy to a higher extent, when compared to video. Authors of [81] have stated that Speech Emotion Recognition (SER) can be utilised in applications that require response to emotional states, such as patient care, geriatric nursing, call centres, psychological consultation, and human communication. However, SER has its own drawbacks, such as input audio being susceptible to noise.

Hidden Markov Models, such as [13] and [14], Gaussian mixture models [38] and SVMs (such as[16]) used to be famous emotion detection models. Nowadays, deep learning (DL) models are trending when it comes to emotion detection from audio. Here, a variety of audio features are extracted from raw audio signals, with Mel spectrogram and MFCCs being two of the most dominant ones. DL models are trained on these audio features, usually extracted from a single audio collection database (an audio dataset). Hence, most of the existing deep learning approaches are either speaker dependent or language dependent or both. In this master thesis, we try to overcome the aforementioned dependencies by utilizing meta-learning and contrastive learning in SER.

The speaker and language dependencies of models are often affected by domain shifts. Specifically, the models are trained on selected features which are exclusive to speakers or the language. However, when a different speakers or languages are used, the models' performance deteriorates. To prevent this, models could learn to recognize speaker and language invariant features by utilizing deep learning techniques which perform significantly better when faced with the problem of domain shifts. In particular, meta-learning, which is trained over multiple sampled tasks, and contrastive learning, with specific losses are defined for latent space alignment of encoded features from different classes, could be employed. Meta-learning tries to optimize the optimizer of the model itself (the meta data) depending on a defined meta objective, which could be utilized to learn speaker and language invariant features. In addition, with contrastive learning, models learn from similar and dissimilar pairs of data in batches depending on a defined contrastive loss, which could again be defined to improve domain generalisation. Authors of [29, 48] utilize meta-learning and contrastive learning, respectively, to address domain generalisation. In this master thesis, we aim to adapt both techniques for SER.

1.1 Problem statement

Speech emotion recognition (SER) has its unique set of challenges due to the personalized nature of how emotions are expressed. One such challenge is creating a model which can perform well across multiple corpora, even if they are of the same language. This is due to various factors such as label ambiguities, different data distributions, different recording environments, language acquisition, and varying speakers. To obtain a robust model resistant to the problems caused by domain shift (due to difference in speakers, languages etc), specific techniques that perform Domain Generalization (DG) are required. Some models showing high accuracies while being trained and tested with one corpus, while their performance deteriorates once they had been tested on another. In particular, the average accuracy of such models does not exceed 65%, (as for example reported in [72, 73, 74]) when being tested across multiple corpora. This points us to our first problem statement, caused by domain shift, stated as follows:

Problem Statement 1: Models which perform well while being trained and tested on a single corpus do not perform well on other corpora, even those of the same language.

Each language may have its own feature considered important for emotion recognition, while the same feature may not be important for another language. Adding differences in prosody, speaking rate, the difference in pronunciations and enunciations between different languages, the accuracies of SER models drop drastically when tested on different languages.

As mentioned above, a possible way to decrease the effect of lack of generalization is to train models on multiple SER tasks. This is done, for example, in [20] and [21], where an ensemble of multiple trained models is used to create a generalized model. However, this approach significantly increases both time and resource consumption. In this regard, we investigate the use of meta-learning to train tasks related to SER and train a generalized model using DG learning techniques tackling the aforementioned problem statement. Some of the advantages of meta-learning include:

- *Training generalized models*: Meta-learning can be trained to perform multiple tasks such as identify emotions over multiple speech corpora, with proper definition of the required tasks.
- *Fast adaptation with fewer data requirements*: Meta-learning models can train over large number of classes with significantly lower amount of data samples per class.

Another possible way to tackle the problem of domain shift is the use of contrastive learning, by using losses specifically designed to perform alignment of data between different domains and learn domain invariant features. We investigate the use of contrastive learning to overcome domain shifts. Advantages of contrastive learning include:

- *Self-supervised*: Contrastive learning can utilize data from multiple datasets without needing labels to train. Classification is performed as a downstream task.

However, while performing experiments, it was seen that Domain generalisation models which perform well in the field of image recognition did not perform very well when it was utilized on log mel spectrograms of different speech data depicting emotions. This is explained in detail in the upcoming chapters.

Problem Statement 2: Domain generalization methods with good performance across image domains do not achieve similar performance on speech emotion recognition tasks.

In this regard, experiments will be run to determine whether the problem arose from the model itself, from the backbone network, or the data.

1.2 Research Questions

From the above-mentioned problem statements, we define the following research questions:

To what extent can the performance and generalization (over different languages) of deep-learning backbone SER models be improved?

Answers to the following sub-research questions, collectively, help us answer our main research question:

1. To what extent can the performance of deep learning backbone SER models, generalized over multiple speech corpora of a single language, be improved by using meta-learning and contrastive learning techniques?
2. To what extent can the performance of deep learning backbone SER models, generalized over multiple speech corpora of multiple languages, be improved by using meta-learning and contrastive learning techniques?
3. In case of low performance of the model, does the problem lie on the model's algorithm, the backbone network or the input data?
4. What is the performance of a different form of input, such as an augmented input, obtained from different speech emotion dataset combinations, used to train models to perform domain generalization?

1.3 Approach

To answer the above research questions, we investigate different meta-learning and contrastive learning methods. Literature related to state-of-art topics on audio pre-processing, deep learning models (for backbone networks), and augmentation techniques were investigated to try and improve performance of the chosen models. The contribution of this thesis can be summarized as follows:

- Studying domain generalization by training on different combinations of speech emotion datasets and testing on an entirely different speech emotion dataset, by using meta-learning and contrastive learning.
- Improving existing models by replacing outdated parts such as losses and augmentation tactics with state-of-the-art techniques.
- Explaining different experiments performed to test the effectiveness of augmentation and different forms of inputs to the models obtained from audio pre-processing.
- Performing extensive experiments to identify the reason in case of low performance and accuracy results.
- Solving this performance issues by investigating the effect of different backbone networks and inputs.

1.4 Thesis outline

The thesis is organized as follows:

- **Literature survey** (Chapter 2): The literature survey begins with a brief description of emotions and its different representations as viewed in different literature. This is followed by elaborating on different audio pre-processing techniques utilized, especially when it comes

to emotion detection. Next, a list of widely used speech emotion datasets is presented. Finally, a taxonomy of state-of-the-art deep learning techniques used for detecting emotions is presented.

- **Preliminaries** (Chapter 3): In this chapter, domain generalization and the chosen techniques i.e. Meta-Learning Domain Generalization (MLDG) [29] and Self-supervised Contrastive Regularization (SelfReg) [48] are explained in detail.
- **Methodology** (Chapter 4): This chapter includes our approach including adoption of MLDG and SelfReg from the image domain to the audio domain, the architecture of the models, and our pre-training strategy.
- **Experiments** (Chapter 5): Extensive experiments that we performed on domain generalization are presented in this chapter together with their observations and results.
- **Conclusion** (Chapter 6): This chapter concludes the thesis and elaborates on the future work that could be built upon this thesis.

Chapter 2

Literature survey

This chapter provides basic information about state-of-the-art research being done on related fields, such as in the area of emotion and Speech Emotion Recognition (SER). Compiling this information helped identify the problem addressed by this Thesis in the field of SER.

2.1 Emotions

Emotions, according to [1], are impulses human beings feel. Those tend to influence the way they act and can range from positive emotions such as happiness to negative emotions such as disgust. In addition, emotions can range from extremely active emotions such as excitement or anger, to passive emotions such as sadness. According to [12], emotions can be represented using a two-dimensional model as shown in 2.1, the two dimensions representing arousal and valence. Valence, usually represented in the x-axis, indicates how negative or positive (pleasant or unpleasant) an emotion is. Arousal, usually represented in the y-axis, indicates the excitation levels of emotions. Most of the emotion datasets usually take these values into account. They help in providing a clear, finer distinction between emotions. Another manner of representation of emotions is the "palette theory" as specified in [57]. It utilizes a number of core emotions (Anger, fear, joy, surprise, sadness and disgust). Different combinations of these core emotions could give rise to new, derived emotions called compound emotions.

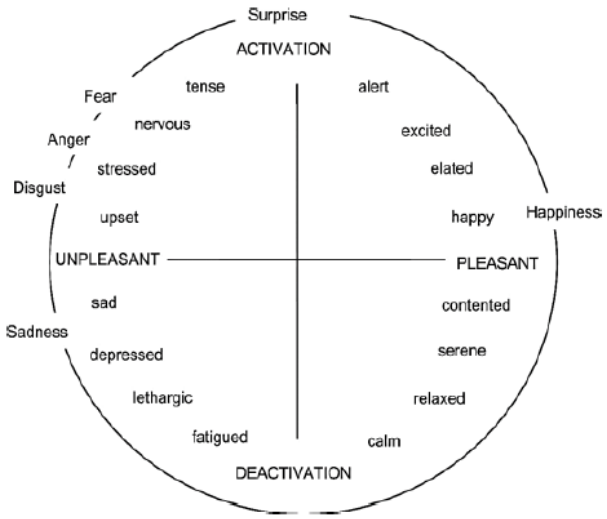


Figure 2.1: Circumflex model of emotions, taken from [1]

2.2 Audio pre-processing

Before extracting audio features, audio signals must be processed to produce meaningful data. There may be numerous steps depending on the audio feature to be extracted and the input audio signal.

2.2.1 Audio segmentation

The most basic processing steps involve filtering the active signal fragments from those that depict silence using zero-crossing detectors and intensity peak detectors (to distinguish between actual speech and environmental noise) [7]. These active signal fragments are then framed, with each frame consisting of an equal number of samples (considering digital audio signal, else sampling must be performed). Furthermore, to ensure a smooth transition, these frames have some degree of overlap. Framing can be done using different windows, such as Hamming, Hann (Hanning), Triangular, Gauss, Welch, Blackman, and Bartlett [8]. In addition to segmentation, by choosing proper windowing function, noise reduction and isolation of important audio frequencies can also be achieved. The audio segmentation pre-processing steps are shown in Figure 2.2.

2.2.2 Audio features

Choosing adequate audio features to be extracted from audio signals to detect emotions is an equally important step during the pre-processing of audio data for SER. According to [1], audio signals could be analyzed using three approaches - temporal, prosodic and frequential.
The temporal approach involves detecting parameters such as windows, zero crossings, peaks, and kurtosis. The number of windows is related to the number of segments of voiced activity detected. In zero crossings the weighted average of times that the speech signal changes the sign in a window of time is calculated [2]. Furthermore, the kurtosis parameter depicts the deviation from the normal distribution. A higher value means a distribution with a higher peak, which implies larger impulsiveness of sound. The prosodic approach involves measuring prosodic features of the speech such as rhythm, stress, and intonations.

The most dominant audio features used in SER research originate from the frequency domain. Those features are extracted from the spectrum of the analyzed signal, such as the Power spectral density. A common frequency-based acoustic feature is the Mel Frequency Cepstral Coefficient (MFCC). It involves mapping the spectra of a signal into the Mel scale using triangle or cosine overlapping windows, followed by taking a log of the powers at each Mel frequency and compressed by performing a Direct Cosine Transform on these log Mel powers [3]. In this way, the resulting MFCCs are the magnitude of the resulting spectrum. In particular, the first coefficient represents

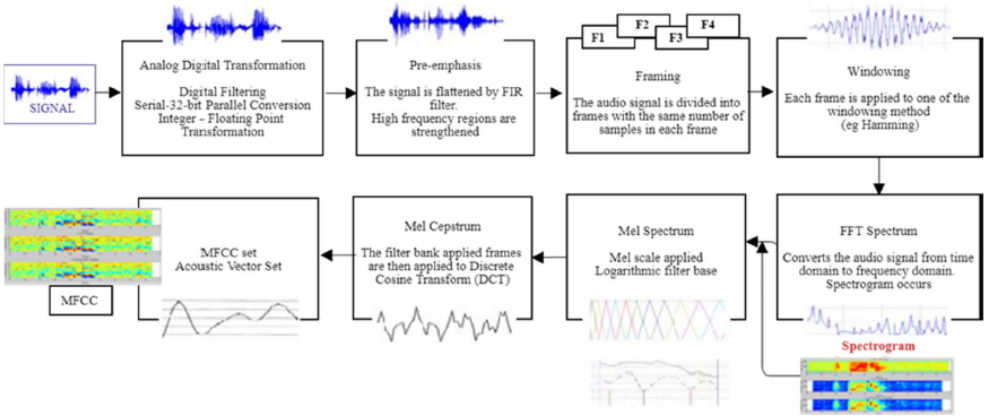


Figure 2.2: Pre-processing steps and MFCC extraction, taken from [32]

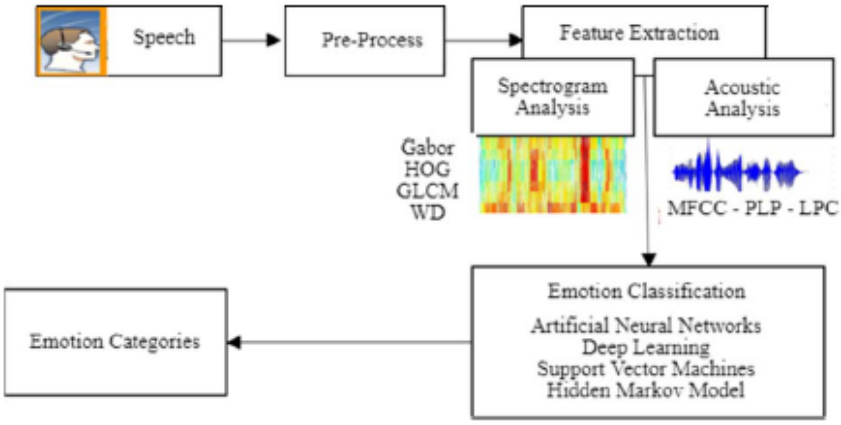


Figure 2.3: Basic SER model, taken from [32]

the average power in the spectrum, the second approximates the shape of the spectrum, while the remaining ones contain finer details of the spectrum [5]. The extraction process is illustrated in Figure 2.2. However, the DCT compression step may remove auditory information and disrupt spatial relations [4]. By omitting the DCT compression, the filter banks values, known as the log-Mel spectrum, can retain the spatial relations and thus are attracting considerable attention in recent SER research. In addition, the Mel scale represents the relation between actual frequency and the frequency of sound perceived by humans, whereas MFCCs only capture the timbral characteristics of sound. Another disadvantage of MFCC is that it is susceptible to additive noise.

Other than MFCCs and log-Mel Spectrograms, there are numerous other frequency-based features, such as Chromagram, Spectral contrast feature, Tonnetz representation, perceptual linear predictive cepstral coefficients, linear predictive coefficients, line spectral frequencies. After extracting the audio features, these can then be fed as inputs to a deep learning model to learn the underlining relationship between these features and the emotional state present in the audio signal. The log-Mel spectrogram features are often treated as images and are fed to popular image recognition models, such as ResNet or VGG16 [19, 33, 17, 18]. Finally, audio feature extraction tools, such as openSMILE [6], have also been developed to extract a series of low-level descriptors, including MFCCs. An abstract example of an end-to-end SER system is shown in Figure 2.3.

2.3 Emotion Recognition Models

SER models could be broadly classified into traditional machine learning models and deep learning models. Pre-processed audio data or spectrograms could be used as inputs and classified into different emotions either by predicting emotions directly or by determining the arousal and valence values. Different SER models have been categorized and presented in Figure 2.4.

2.3.1 Non-deep learning models

In the past, traditional ML techniques determined a probability distribution over a predefined set of emotions using hand-crafted features from audio signals. Among those, Hidden Markov models were widely used due to their ability to capture the temporal effects of speech in the form of a concatenation of states resulting in high accuracies [13, 14]. Another popular approach was combining the prediction of different classifiers, such as a naive Bayes classifier [15] and Support Vector Machines (SVMs) [16] for a given set of audio feature, like MFCCs. With the advent of deep learning, these classifiers were then combined with deep learning techniques. DL techniques

were responsible to extract meaningful representations from raw audio signal such as in [17]. Nonetheless, all of the aforementioned approaches were speaker-dependent and had low accuracies when experimented across different datasets.

2.3.2 Deep learning models

Recently, the research focus shifted towards deep learning approaches for SER. Deep learning SER networks can be used as feature extractors from segmented audio features, such as log-Mel spectrograms, after which a downstream classifiers, such as Random forest classifiers [17] or clustered using KNN clustering [18], is used for classification. In addition, deep learning end-to-end SER networks can be used to perform both feature extraction and classification of input data. In [17] and [18] a CNN is utilized to determine relations between a sequence of inputs and produce a vector with smaller dimensions. In this way, the newly compressed vector is able to effectively capture local features and characteristics of the input data. Furthermore, for the case of log-Mel spectrograms when treated as images, CNN-based models have proven to be particularity successful in SER, following their exceptional representation power in image classification task. A considerable amount of research has also been directed to overcome the dependency of emotion recognition models on a speaker's voice characteristics. Few papers, such as [19], have used LSTMs to capture long-term contextual dependencies as well as local information, which showed great improvements in speaker-independent recognition, but only within specific datasets, as they are still dependent on the utterances and language of that dataset. Authors of [20] and [21] used ensemble methods to improve speaker independence. In particular, [20] have proposed an ensemble of three deep learning models, each of which are tuned to learn the arousal, valence and categorical emotions. Following a different approach, [21] has created an ensemble of deep learning models which first detect "difficult" emotions, such as boredom and disgust, and then proceed to detect "easily" distinguishable emotions, such as happiness, anger, sadness, fear and neutral. This ensemble method's performance is similar to [19].

Deep learning was also used in other manners in the area of SER, such as in [22] providing attention to only the important parts of the speech, such as verbal sounds and non silent or active regions, and using LSTM encoders with attention modules. In addition, DL models were also used to create embeddings using autoencoders to aid in their clustering (for semi-supervised learning) [23] and models which takes into account the accuracy of annotators and uses them as auxiliary inputs to reduce the effects of faulty labels[24].

2.3.2.1 Meta-learning

Meta-learning is a relatively new concept recently developed in deep learning. A normal deep learning technique involves making multiple training passes to compute the gradients of the model's parameters and optimizing them to converge to an appropriate model. Meta-learning, however, is different as it tries to optimize the parameters of the optimizer itself (the meta data). It could involve learning the relation between two inputs by computing their distance in metric space for few shot learning (metric based). It could also imply learning optimal initialization parameters rather than random parameters for faster convergence (optimization based). Or it could entail replacing the optimizing an optimizer of a model with a RNN for faster convergence (model-based).

An advantage of meta-learning is its capability of few shot learning. Since a meta-learning model is able to learn and adapts quicker, it requires significantly less data for training. Another advantage is that meta-learning optimizes over different tasks. This means that [29] is better at generalization and could possible perform better over different speakers and datasets.

Although meta-learning has been used in many applications, its use in SER has not been well explored. Papers [31] and [30] use two different variations of MAML algorithm to achieve high emotion classification accuracy over multiple languages. However, they do not perform DG. Rather, they concentrate on domain adaptation, which is a technique where a small amount of data from test domain is used to converge the model quickly to predict rest of the data from the test domain.

2.3.2.2 Contrastive learning

Contrastive learning is a DL technique that learns from the similarity between similar and dissimilar pairs of data in batches. The similarity is usually calculated using distance measures such as Euclidean distance or cosine similarity between data points. For this, the data points have to be embedded in a lower dimensional latent space. Embeddings are created using base encoders from input data pairs and embedded in latent space using projection modules in such a way that agreement between similar data pair is maximized by minimizing a contrastive loss formula. This contrastive loss formula usually depends on similarity such as cosine similarity [34] as mentioned before. Using a downstream classification task, one can classify the embeddings of input data as required. Hence, contrastive learning models are generally self-supervised, while the adequate formation of data pairs is essential. A depiction of contrastive learning has been presented in 2.5.

Data augmentations could be used to create even more similar and dissimilar pairs for contrastive learning. Instead of using two different input data of the same class to create a pair, a single data can be augmented in two different ways to create a pair. Heavy data augmentation is required for proper training as it creates more diverse pairs for training batches. More diverse pairs reduce the requirement of input data thus reducing the size of required dataset. As there is a lack of well-defined and large open datasets for emotions available, this reduced requirement is helpful. However, increasing training data significantly improves performance as said in [34].

Contrastive learning for audio is mostly used in the field of audio representation which could be extended to SER. In [36], data pairs are created for different types of data, namely augmented images of log-Mel spectrograms and their respective augmented waveforms. This model has achieved higher accuracy scores compared to other audio representation models. Author of [35] used Siamese networks for performing SER, while they tested their approach on two distinct contrastive loss functions, each utilizing cross entropy loss and cosine similarity. In both approaches, a CNN-based architecture, such as ResNet, was utilized as encoder. It is possible that other deep-learning architectures, such as LSTMs and Bi-directional LSTMs (B-LSTMs) could be employed as encoders to further improve their results. Finally, [39] utilized an encoder to create latent space representations of utterances out of which pairs are formed for contrastive learning.

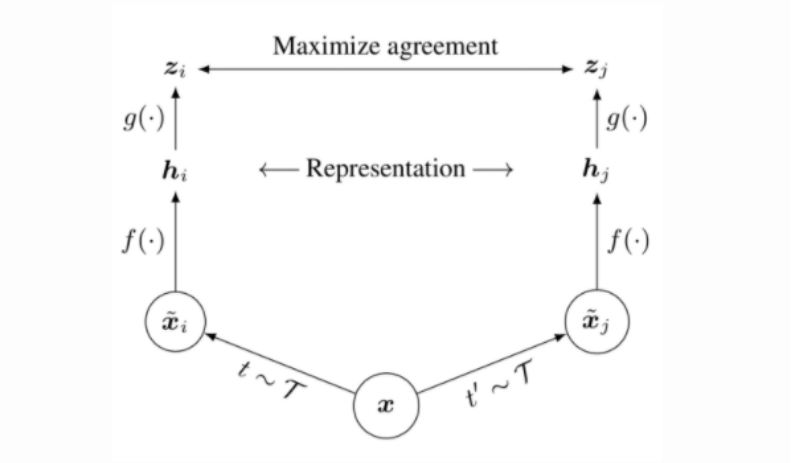


Figure 2.5: Depiction of contrastive learning, taken from [34]. f(.) denotes the encoder, while g(.) denotes the projection heads. z_i and z_j are the projections of the x_i and x_j, which are augmented outputs of input x.

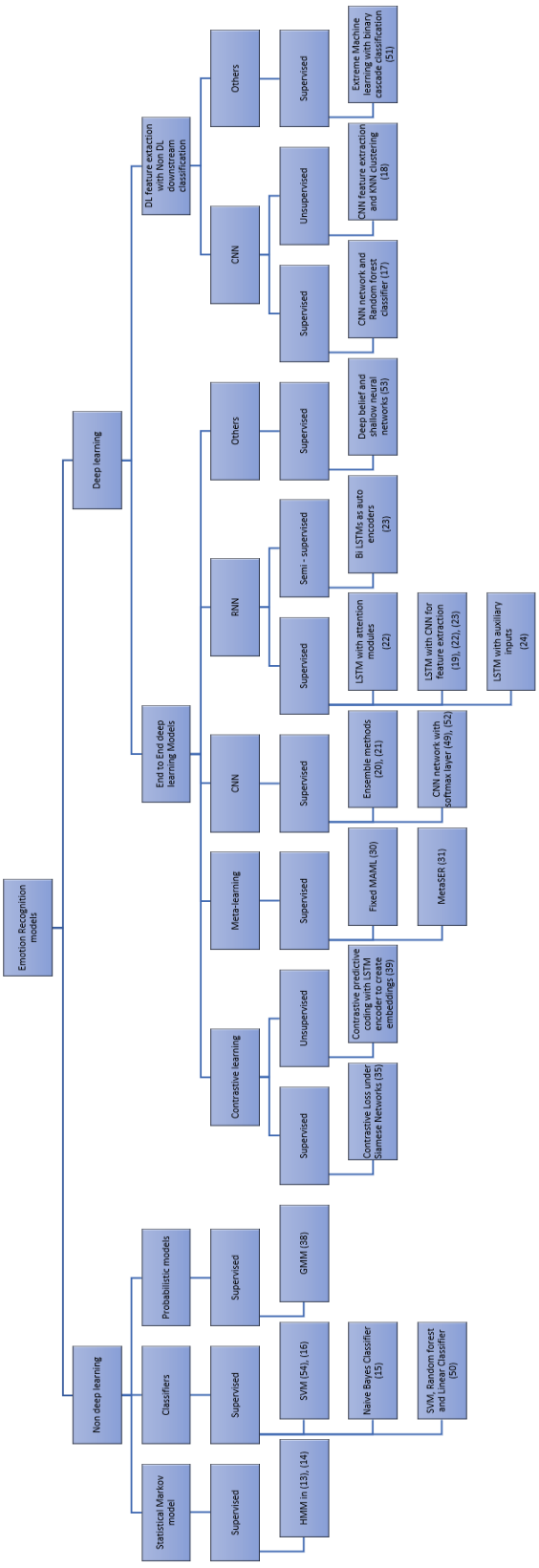


Figure 2.4: Papers categorized into different types of models

2.4 Domain generalization

In deep learning, models usually tend to overfit on their training data domain, especially models such as Convolution Neural Networks (CNN), which are extremely powerful. They try to learn most of the learnable features of training dataset, even if it does not contribute a lot while predicting data from test dataset. Although every bit of learned features help, it may not help in predicting data from a different data domain, which is similar to the original data domain. This new data domain, which is different, may have some features in common with the original data domain. An example of this situation is training a SER on training data from a group of speakers. This may allow the SER to predict a test set from that same group of speakers with a good accuracy score. However, it may fall short when it tries to predict a test set from an entirely different group of speakers. It could be said that the SER has become speaker dependent.

This is where domain generalization is beneficial. Domain generalization is performed when a DL model tries to learn the semantics of the source domain, which could be generalized to a target domain, if there are semantics to be learnt. The DL tries to learn a generalized predictive function which could then be used to predict Out-of-distribution (OOD) data (data different from the training data distribution).

There have also been work done on domain generalization in the field of SER. The paper [73] introduces a technique dependent on mining semi-hard triplets to perform alignment using triplet loss in order to learn domain invariant features for SER. Authors of [74] introduced a model called Adversarial Discriminative Domain Generalization (ADDoG), with critic network and feature extractor network in which one is trained when the other is frozen, so that model learns to distinguish invariant features by challenging itself. Authors of [72] used a combination of CNN and RNN architecture (called CNNRNNATT) to run tests on DG with SER.

2.4.1 Applications of domain generalization

Domain generalization is used when obtaining training data is expensive or difficult. It may be easier to make use of data from different domains and then generalize to the required domain. It also tries to make the model more robust in the face of domain shifts present in real world scenarios. DG could be used in many applications, some of which are listed below:

- **Character recognition**: DG could be utilized to predict written characters from alphabets of a language, written in a different style from the training data. This could be used to recognize sentences from different handwriting.
- **Speech recognition**: DG could be used to generalize DL speech recognition networks over different speaker, sometimes even over different languages.
- **Medical imaging**: DG could be utilized to generalize over domain shifts caused by varying factors such as difference in imaging equipment.
- **Face recognition**: DG helps in generalizing and recognizing the face of a person even when a domain shift is present due to factors such as viewing angle, viewing distance and noise.
- **Object recognition**: Most of the time, objects present in daily life, such as vases, TVs etc, vary a lot in appearance, but still have some similar features. A DG model tries to learn these features of the objects to be recognized. It could then be used to recognize different types of the same objects. For example, a DG model could use the PACS dataset to learn domain agnostic features of houses from paintings, sketches and cartoons of houses, and then try to recognize photos of houses in the real world.

2.4.2 Taxonomy of domain generalization techniques

The paper [58] presents a taxonomy of domain generalization methods. According to the authors of [58], domain generalization can be performed by manipulating the data, having different learning

strategies or by learning specific representations. Data manipulation strategies involve using data augmentation such as Domain Randomization and Pyramid Consistency (DRPC) [63], which tries to enforce pyramid consistency on augmented images as well as real images from different domains. Learning specific representations is another path that could be taken and involves learning specific domain invariant features by using DL networks that recognize them. An example of learning specific domain invariant features is seen in variation of Domain-Adversarial Training of Neural Networks (DANN) [65], which uses a generator to generate features to confuse the discriminator and the discriminator tries to distinguish domains. The third category includes different learning strategies, which modify the way of learning for backbone networks to create a robust model which performs well against domain shifts. Learning strategies include techniques such as ensemble learning (an ensemble of multiple models performing tasks on different domains), Meta-learning and Self-supervised learning.

This thesis uses the third category, i.e it uses two different learning strategies from Meta-learning (MLDG) and Self-supervised learning (SelfReg). The methods this thesis uses are marked in green in figure 2.6, which shows the taxonomy of domain generalization as seen in [58].

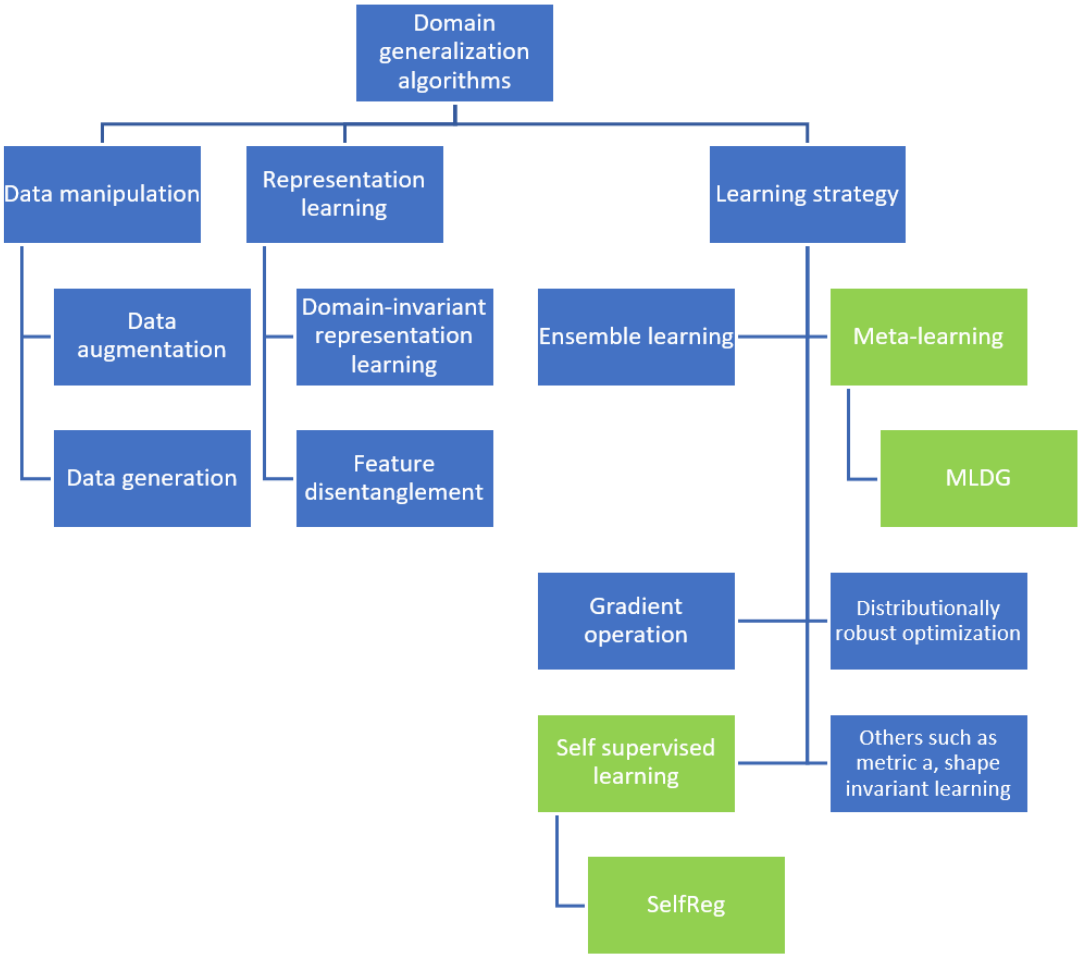


Figure 2.6: Taxonomy of domain generalization, taken from [58]

2.5 Speech emotion datasets

Emotion datasets can be divided into three main types, namely acted, simulated and natural [37]. Emotion datasets usually provide information on categorical emotions. Some datasets also provide information on Arousal, Valence and dominance values. From the available datasets, IEMOCAP is the only dataset which contains both acted and evoked audio samples. Table 2.1 contains detailed information for each dataset in particular. However, it is important to note that not all of presented datasets are publicly available.

Table 2.1: Emotion datasets

No.	Dataset	Description	Open dataset	Type
1.	IEMOCAP [9]	Collected by the Speech Analysis and Interpretation Laboratory (SAIL) at the University of Southern California (USC), IEMOCAP is an audio-visual database consisting of over 12 hours of data. 10 actors with markers in face, head and hand perform conversations in English displaying five types of emotions (happiness, anger, sadness, frustration, and neutral state).	No (restricted access)	Acted and Evoked
2.	EmoDB [10]	This open dataset consists of 10 actors (5 male and 5 female) hold conversations in German displaying seven types of emotions neutral (neutral), anger (Ärger), fear (Angst), joy (Freude), sadness (Trauer), disgust (Ekel) and boredom (Langeweile).	Yes	Evoked
3.	RAVDESS [11]	The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) is an open dataset and contains 7356 files (approx. 24.8 gb) of audio statements made by 24 actors (12 male and 12 female) in North American accent. Emotions depicted includes calm, happy, sad, angry, fearful, surprise, disgust (all with 2 levels of intensity – normal and strong), and neutral expression.	Yes	Acted
4.	CHEAVD [25]	CASIA Chinese Natural Emotional Audio–Visual Database is an audio-visual database consisting of over 140 mins of data. The data is extracted from films and talk shows. It consists of data from 238 speakers of varying age groups labelled into 26 non-prototypical emotional states labelled by four native speakers.	No	Natural
5.	NNIME[26]	The NTHU-NTUA Chinese Interactive Multimodal Emotion Corpus consists of over 11 hours of audio data acted by 22 females and 20 males. It is labelled as angry, happy, sad, neutral, surprise, and frustration by 49 annotators.	No	Acted
6.	EmoFilm–Zenodo [27]	EmoFilm dataset consists of English and Italian audio data from 43 films and 207 speakers. It consists of 1115 files classified into anger, contempt, happiness, fear, and sadness.	No (restricted access)	Natural
7.	TESS [40]	The Toronto emotional speech set consists of 200 target English words displaying each of seven emotions (anger, disgust, fear, happiness, pleasant surprise, sadness, and neutral). This is performed by two actresses.	Yes	Acted

8.	SAVEE [41]	The Surrey Audio-Visual Expressed Emotion (SAVEE) consists of 480 British English utterances by seven male actors. It consists of 3 common, 2 emotion-specific and 10 generic sentences which are phonetically balanced and can be classified into anger, disgust, fear, happiness, sadness and surprise.	Yes	Acted
9.	EMOVO [42]	EMOVO is an Italian database consisting of utterances of 6 actors who speak 14 sentences simulating 6 emotional states (disgust, fear, anger, joy, surprise, sadness) and the neutral state. They are validated by two different groups of 24 listeners	Yes	Acted
10.	ShEMO [43]	The Sharif Emotional Speech Database consists of 3000 utterances Extracted from online radio and has up to 3 hours and 25 minutes of audio data. It covers 87 speakers simulating anger, fear, happiness, sadness and surprise, as well as neutral state. They are validated by two different groups of 24 listeners	Yes	Natural
11.	URDU [44]	URDU database consists of 400 utterances by 38 speakers (27 male and 11 female), gathered from Urdu talk shows. They simulate Angry, Happy, Neutral, and Sad emotions.	Yes	Natural
12.	CAFE [59]	The Canadian French Emotional (CaFE) dataset consists of utterances from 6 male and 6 female actors, who pronounce 6 different sentences. This dataset depicts six basic emotions (sadness, happiness, anger, fear, disgust and surprise) and a separate neutral emotion	Yes	Acted
13.	AESDD [60]	The Acted Emotional Speech Dynamic Database (AESDD) consists of audio data files recorded by professional actors in Greek language. It depicts five emotions: anger, disgust, fear, happiness, and sadness	Yes	Acted
14.	CREMA-D [61]	The Crowd Sourced Emotional Multimodal Actors Dataset consists of 7,442 original clips of English language from 91 actors who spoke from a set of 12 sentences. They simulate Anger, Disgust, Fear, Happy, Neutral and Sad emotions.	Yes	Acted

This thesis uses nine datasets from the above table. They are CREMA-D, AESDD, CAFE, EMODB, RAVDESS, TESS, SAVEE, EMOVO and SHEMO. These datasets were selected due to ease of availability and for the reason that they are mostly open datasets. Also only five emotions i.e. anger, disgust, fear, happiness, and sadness are considered as they are common among all the datasets and can be used for domain generalization.

Chapter 3

Preliminaries

In this chapter, the concepts of domain and Domain Generalization (DG) are explained in detail, together with their mathematics definitions, to provide the necessary background for the reader to understand the developed approaches discussed in Chapter 4. In addition to the domain generalization optimization problem, the architecture of the chosen DG algorithms, namely MLDG [29] and SelfReg[48], are presented alongside with their objectives.

3.1 Domain generalization

In this work, the underlining problem is how to predict emotions from speech data that originate a domain different from the one the predictive model has initially being trained. However, before mathematically expressing the domain generalisation problem, the term "*domain*" has to be defined.

Definition of a domain: Let $\mathcal{X}$ denote a nonempty input space and $\mathcal{Y}$ an output space. A "*domain*" is composed of data that are sampled from a distribution. We denote it as $\mathcal{S}= (x_i, y_i)_{i=1}^{n} \tilde{P}_{XY}$, where $x \in \mathcal{X} \subset \mathbb{R}^d, y \in \mathcal{Y} \subset \mathbb{R}$ denotes the label, and P_{XY} denotes the joint distribution of the input sample and output label. Here, X and Y corresponds to random variables.

Given the above definition, a "*domain*" could be considered as a collection of data points, which are collected from the same distribution. Given the aforementioned definition of "*domain*", the concept of DG, in [58], is as follows:

Domain generalization (DG) : In domain generalization, we are provided with M train (source) "*domain*" $\mathcal{S}_{train} = \{\mathcal{S}_i \mid i = 1, \ldots, M\}$, where $\mathcal{S}_i = (x_i^j, y_i^j)_{j=1}^{n_i}$ denotes the i^{th} "*domain*". Furthermore, for each pair of "*domain*", the joint distributions differs, hence $P_{XY}^i \neq P_{XY}^j$, $1 \leq i \neq j \leq M$. The goal of domain generalization is to learn a robust and generalizable predictive function $h : \mathcal{X} \rightarrow \mathcal{Y}$ from the M train "*domain*" to achieve a minimum prediction error on an unseen test "*domain*" $\mathcal{S}_{test}$. Therefore, the domain generalization problem is formulated as:

$$\min_h \mathbb{E}_{x,y \in \mathcal{S}_{test}} [\, l(h(x), y)] \ , \tag{3.1}$$

, where $\mathbb{E}$ is the expectation and $l(\cdot,\cdot)$ is the loss function.

A intuitive understanding of the domain generalization problem is provided in Figure 3.1.

3.1.1 Meta-Learning Domain Generalization (MLDG)

Meta-learning aims to learn from its own backbone network's experience on performing different tasks (learning to learn). Although a considerable amount of meta-learning algorithms concentrate

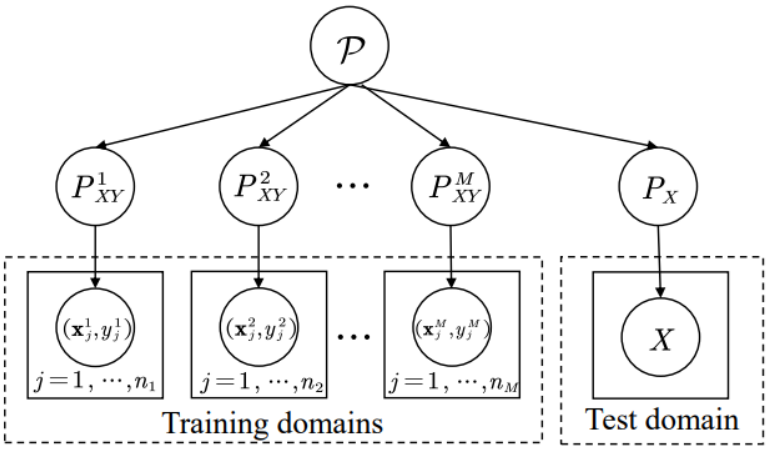


Figure 3.1: Illustration of Domain Generalization. Adapted from [58]

on solving problems related to few shot learning, where a limited number of examples with supervised information for a target task are provided, these algorithms could be extended to perform domain generalization as well. The MLDG algorithm [29], inspired by the MAML meta-learning algorithm, performs domain generalization by adjusting the MAML algorithm's objective. Additionally, MLDG is model-agnostic, thus it could work in conjunction with any backbone model architecture.

A meta-learning algorithm can be said to have two parts – Episodes construction and Meta-representation [62]. Similar to MAML, MLDG samples tasks from randomly selected datasets of the source domain. Assume that the backbone network is parameterized as Θ. During the episodes construction step, the original source domain $\mathcal{S}$ is split at random into two disjoint sets, the $\mathcal{S}-V$ meta train domains denoted by $\bar{\mathcal{S}}$ and V meta test domains denoted by $\breve{\mathcal{S}}$. The Episodes construction step represents the multiple possible domain shifts within the source domain from which the backbone network could learn. During the Meta-representation step, the goal is to choose which layers of the backbone network should be meta-learned and update them. Hence, MLDG learns and optimizes the entire backbone network rather than specific layers. When training has been completed, the acquired model is then tested on the test domain, which is entirely disjoint from the source domain.

MLDG has two objectives implemented in the form of nested training loops. The first objective aims to utilize the meta train domain, $\bar{\mathcal{S}}$, to minimize the following loss function:

$$\mathcal{F}(.) = \frac{1}{S-V} \sum_{S-V}^{i=1} \frac{1}{N_i} \sum_{N_i}^{j=1} l_\Theta(\hat{y}_j^{(i)}, y_j^{(i)}) \ , \tag{3.2}$$

, where $\hat{y}$ denotes the predicted label, y corresponds to true label, $l(\cdot)$ refers to cross entropy loss, i is the sampled domains, N_i is the number of samples of the i^{th} meta-test domain, and j denotes data points.

With a backpropagation step to update the backbone network based on the performance on the meta train set, the effect of this update will be observed on the meta test domain. The second objective of MLDG is to check whether optimizing the network based on the meta train set has a beneficial effect on the meta test set. Therefore, the second objective mimics situations in which the model is used to test data from domains different from the train domain, i.e it simulates domain shifts. Mathematically, this objective can be formulated as follows:

$$\mathcal{G}(.) = \frac{1}{V} \sum_{V}^{i=1} \frac{1}{N_i} \sum_{N_i}^{j=1} l_{\Theta'}(\hat{y}_j^{(i)}, y_j^{(i)}) \ , \tag{3.3}$$

where $\hat{y}$ denotes predicted label, y indicates the true label, $l(.)$ refers to cross entropy loss, i denotes sampled domains, N_i is the number of samples of the i^{th} meta-test domain, and j denotes data points.

Considering the the aforementioned objectives, MLDG aims to simultaneously minimize both losses defined in Equations 3.2−3.3. Therefore, the overall optimization function of MLDG is the following:

$$\arg\min_{\Theta} \mathcal{F}(\Theta) + \beta\mathcal{G}(\Theta - \alpha\mathcal{F}'(\Theta)) \tag{3.4}$$

, where $\mathcal{F}'(\Theta) = \nabla_{\Theta}$ i.e the gradient obtained from meta train step, α is the meta-train step size and β is the meta-test step size.

3.1.2 Self-supervised Contrastive Regularization (SelfReg)

A widely used approach for learning domain invariant features is domain alignment. When invariant features among source domains are learnt, they are expected to withstand domain shifts in test domains. Such features can be learnt by minimizing the difference between similar distributions of domains and aligning similar features. This can be done by minimizing moments, KL divergence, contrastive loss etc. SelfReg [48] utilizes contrastive loss to perform alignment between domains, thus aims to learn domain-invariant representations. As in the case of MLDG, SelfReg is also model-agnostic, and could be used with a wide range of model architectures.

SelfReg aims to identify positive pairs and minimize their distance. This is in contrast with other contrastive learning algorithms, whose objective is to mine hard or semi-hard negative pairs and increase the difference between them. As identifying proper negative pairs from the input data is considered a hard task as mentioned in [48], the positive pair technique is primarily exploited in domain generalization. Additionally, data augmentation could be utilized to further increase the number of positive pairs. An intuitive understanding of SelfReg is presented in Figure 3.2.

Similar to MLDG, SelfReg aims to minimize two objectives during each training step. The first objective is to minimize a pre-defined loss, called *Individualized In-batch Dissimilarity Loss* ($\mathcal{L}_{ind}$). Minimizing $\mathcal{L}_{ind}$ aligns different representations belonging to the same class. This is done by minimizing $\mathcal{L}_{ind}$ loss between the original representations and a shuffled copy of the original representations. However, before aligning the pairs, it is made sure that the copies are passed through a Multi-Layer Perceptron (MLP) layer, termed Class-specific Domain Perturbation Layer (CDPL). In this way, the model is prevented from learning a collapsed representation (i.e. directly learning a representation, which results in zero loss). This was also observed in [66], where authors identified this behavior when using augmented data. Therefore, the MLP layer, whose weights are not updated by the gradients of the backbone network, perturbs the copies of features. Mathematically the first objective can be formulated as follows:

$$\mathcal{L}_{ind}(\mathbf{z}) = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{z}_i^c - f_{CDPL}(\mathbf{z}_{j\in[1,N]}^c)\|_2^2 \ , \tag{3.5}$$

where $\mathbf{z}_i^c$ represents the i^{th} latent representation out of N representations belonging to a class c of the source domain. Minimizing this loss forces the model to learn domain invariant features from different representations belonging to the same domain. The same objective can also be extended to logits. Here, the logits vector replace the latent representation of Equation 3.5.

The second objective is to minimize *Heterogeneous In-batch Dissimilarity Loss*. Before calculating this loss, two distinct shuffled copies of representations belonging to the same class, $\mathbf{z}_i^c$ and $\mathbf{z}_j^c$, are combined using Mixup [79]. Assuming the output of the CDPL layer i.e $f_{CDPL}(\mathbf{z}_i)$ is $\mathbf{u}_i^c$, Mixup of the two representations can be defined as follows:

$$\bar{\mathbf{u}}_i^c = \gamma\mathbf{u}_i^c + (1-\gamma)\mathbf{u}_{j\in[1,N]}^c \ , \tag{3.6}$$

, where $\gamma \sim Beta(\alpha, \beta)$ (Beta distribution) and $\gamma \in [0, 1]$.

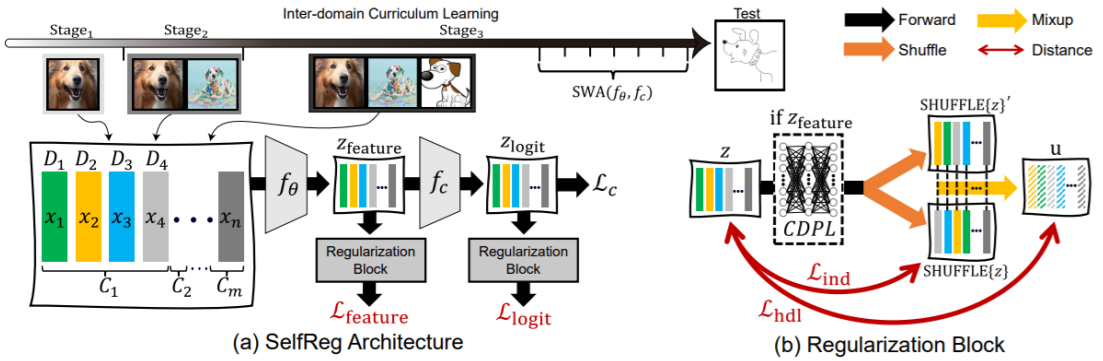


Figure 3.2: Illustration of SelfReg architecture. SelfReg utilizes Inter-domain learning curriculum and stochastic weight average to optimize gradients "*in conflict direction*". Self-supervised contrastive losses are used to regularize and learn domain-invariant representations. Adapted from [48].

Using Equation 3.6, the Heterogeneous In-batch Dissimilarity Loss is defined as follows:

$$\mathcal{L}_{hdl}(\mathbf{z}) = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{z}_i^c - \bar{\mathbf{u}}_i^c\|_2^2 \ , \tag{3.7}$$

, where $\bar{\mathbf{u}}_i^c$ is the output of Mixup as defined in Equation 3.6. Minimizing this loss further aligns the data pairs and applies pressure on the model to learn even more domain invariant features from different domains involved in Mixup. This loss is also be applied to logits, with logits replacing representations in the loss formula.

In SelfReg, these two aforementioned objectives are scaled to control their impact during the training procedure. Additionally, SelfReg also considers the standard Categorical Cross-Entropy (CCE) loss, which considers the difference between the predicted and original class labels. The resulting optimization function of SelfReg is as follows:

$$\mathcal{L} = \mathcal{L}_c + C_{scale}(\lambda_{logits}(\mathcal{L}_{hdl\_logits} + \mathcal{L}_{ind\_logits}) + \lambda_{feature}(\mathcal{L}_{hdl\_feature} + \mathcal{L}_{ind\_feature})) \tag{3.8}$$

, where $C_{scale}, \lambda_{logits}$ and $\lambda_{feature}$ are all scaling factors and $\mathcal{L}_c$ is the classification loss.

To average network's weights over a specified period of time, SelfReg utilizes Stochastic Weights Averaging (SWA). This weight averaging techniques could be Incorporated to any loss function to improve generalization, as a flatter minima is present over a range of local minima [48].

3.1.3 Beyond meta-learning and contrastive learning

Apart from meta-learning and contrastive learning, other state-of-the-art readily-available approaches for cross-domain classification tasks have been considered in this work. For the convenience of the reader, a concise overview is presented here.

3.1.3.1 Representation self-challenging

Representation Self-challenging [69] (RSC) has achieved significant performance when comes to domain generalization in image datasets, such as PACS [78], VLCS [86], Office-Home [87] and ImageNet-Sketch [88]. Specifically, RSC discards most dominant features of input data, and aims to learn any other representations relevant to the input data. Thus, RSC forces the backbone network to learn less dominant features. This is done by identifying the top $(100-p)^{th}$ percentile (where p is a constant) of the network's parameters by calculating the gradients of the upper

layers. Then, these parameters are muted by setting the respective values to zero, before updating the entire network. Intuitively, the network iteratively forgets the most dominant representations, and guides the model to learn less important features could mostly be domain invariant. As an example consider learning from images of cats and dogs. In this case, dominant features, such as ears, could easily be distinguished between the two animals. However, other features, such as whiskers, are much harder to classify correctly. Thus, discarding dominant features in this case could result in a much more powerful network. In this work, we consider whether RSC learning scheme could help improve cross-domain speech emotion recognition.

3.1.3.2 Feature-based classification

Recently, the extraction and utilization of embeddings from raw audio data using a large-scale pre-trained model has received a considerable attention [77]. In [77], the embeddings extracted from the 12th of a 600 million parameter model has achieved state-of-the-art performance across a number of audio classification tasks. Such models are often pre-trained on million hours of audio data obtained from different environments in different conditions, such as YT-U [89]. To mitigate the need for loading such large-scale model to extract embeddings from raw audio signals, recent works [77] have proposed the construction of significantly smaller model with similar representational power via knowledge distillation. In this direction, TRILLson [77], a model distilled from CAP12 [77], achieved a significant reduction in the number of parameters, while maintaining over 90% of CAP12's performance. In this work, we explore whether the embeddings extracted using TRILLson could help improve cross-domain speech emotion recognition.

Chapter 4

Methodology

This thesis is about training and testing a robust SER model which could be used for domain generalization utilizing two different learning schemes, namely MLDG and SelfReg. This section describes the algorithms of MLDG and SelfReg, along with the pre-processing pipeline used. In these models, ResNet18 was utilized as a backbone network.

4.1 Problem formulation

According to the definition of domain generalization in Section 3.1, M training (source) domains are required for learning invariant features amongst them, those features which could be extended to predict data from a separate, but similar test domain. The two learning techniques MLDG and SelfReg were originally intended for domain generalization in the image domain [29, 48]. Therefore, in this work, an adaption of these approaches has been made to perform cross-corpus domain generalization on speech emotion data.

Our goal is to learn a predictive function $h : \mathcal{X} \rightarrow \mathcal{Y}$ (where $\mathcal{X}$ denote a non-empty input space and $\mathcal{Y}$ an output space in all the domains used), through means of training a deep learning network. This predictive function should achieve minimum prediction error, when used to predict emotions from speech data from a test domain, while it was trained in domain, originated from a different distribution. In essence, the test loss obtained during the test phase between the original labels and the labels predicted using function $h : \mathcal{X} \rightarrow \mathcal{Y}$ must be minimum. This can be represented as:

$$\min_{h} \mathbb{E}_{x,y \in \mathcal{S}_{test}} [\, l(h(x), y)]; \tag{4.1}$$

where $\mathbb{E}$ is the expectation and $l(\cdot, \cdot)$ is the loss function. The loss function used, as the problem is a multi-class classification problem (there are five emotion classes), should be Categorical Cross-Entropy loss (CCE). As the performance of a model is more easily understood with accuracy and as the recognition of emotions does not have heavy penalties for false positives, accuracy is chosen as the main objective to improve in this work. Hence, the thesis monitors and presents the results as sparse categorical accuracy (as the problem is a multi-classification problem).

4.2 MLDG

The objective of Meta-Learning Domain Generalization (MLDG) is depicted in Equation 4.2. The goal is to update the parameters of the backbone network by minimizing the following:

$$\arg\min_{\Theta} \mathcal{F}(\Theta) + \beta \mathcal{G}(\Theta - \alpha \mathcal{F}'(\Theta)) \tag{4.2}$$

The implemented algorithm is described in the following sequential steps:

- **Input data**: Different input data domains are combined to form a single source domain $\mathcal{S}$. All the input data batches are pooled together, and random batch indices, which are uniformly generated, are selected to randomly pick training batches. The remaining batches form the test batches. This random selection simulates the selection of different random tasks per training step, although there is only one task present in this case, which is the multi-classification problem involving five emotion classes. Before the training loop begins, the current parameters of the models are saved as a checkpoint.

- **Meta-train** The first objective is implemented as a custom inner training loop to minimize $\mathcal{F}$, explained in 3.1.1. This loss is calculated from the labels predicted by the backbone network, as a result of a single training episode on passing the randomly selected training batches. During meta-train step, the calculation of the first order gradients of loss $\mathcal{F}$ occurs in an inner training loop within a gradient tape (a TensorFlow API that keeps track of variables and calculates gradients). This gradient tape is nested within another gradient tape present in the outer training loop, which calculates the second order gradients of loss $\mathcal{F}$ for the final Meta-optimization.

- **Update parameters**: The gradient ∇_Θ of the parameters (Θ) of the backbone network with respect to the loss function, which is defined in Equation 3.2, is calculated and used to update the parameters of the backbone network according to the rule $\Theta' = \Theta - \alpha\nabla_\Theta$. The first order gradients of $\mathcal{F}$ from the inner training loop is passed to an optimizer (assumed to be optimizer A) within the inner training loop which updates the parameters Θ by a single step to Θ'.

- **Meta-test**: After a single training step, the next step is to simulate a testing phase with the rest of the unselected batches. This is done to imitate practical situations where the model is required to test data from unseen domains. The test loss obtained from this updated model is the loss $\mathcal{B}$ indicated in Equation 4.2. This loss is dependent on the gradients of the loss $\mathcal{F}$ and the test batches. The gradient of this loss is calculated in the outer training loop along with the second order gradients of loss $\mathcal{F}$, which is required for the final meta-optimization step

- **Rollback**: The parameter θ' is restored to θ by rolling back to the checkpoint saved before the meta-train step.

- **Meta-optimization**: From the Equation **??**, it is clear that to update parameters permanently, the second derivative of loss $\mathcal{F}$ is needed. The second order gradients of loss $\mathcal{F}$ (calculated in the meta-test step), along with gradients of loss $\mathcal{G}$ are fed to another optimizer (assumed to be optimizer B) to reduce the final objective shown in Equation 4.2.

The algorithm of MLDG is provided in Algorithm 1. Since the original MLDG implementation was given in PyTorch, a adaptation to TensorFlow was performed in this work. Once the training procedure is complete, the final backbone network is tested then tested in an unseen domain, which is entirely different from the source domain.

Algorithm 1: MLDG algorithm

Data: From source domains, collectively denoted as $\mathcal{S}$
Initialization: ResNet18 with model parameters Θ, hyperparameters α, β and γ
for *epoch* ***in*** *epochs* **do**
  Random split: $\bar{\mathcal{S}}$ and $\check{\mathcal{S}} \leftarrow \mathcal{S}$
  Meta-train: $\nabla_\Theta = \mathcal{F}'(\bar{\mathcal{S}}, \Theta)$
  Update parameters: $\Theta' = \Theta - \alpha \nabla_\Theta$
  Meta-test: Loss $\mathcal{G}(\check{\mathcal{S}}, \Theta)$
  Rollback: Rollback Θ' obtained from **Meta-train** back to Θ
  Meta-optimization: Update Θ

$$\Theta = \Theta - \gamma \frac{\partial(\mathcal{F}(\bar{\mathcal{S}}; \Theta) + \beta \mathcal{G}(\check{\mathcal{S}}; \Theta - \alpha \mathcal{F}'(\Theta)))}{\partial \Theta}$$

end

4.3 SelfReg

Self-supervised Contrastive Regularization (SelfReg) is a learning technique to create robust models for domain generalization using contrastive learning, a well established field in deep learning. SelfReg defines multiple losses that helps align different pairs of similar data from multiple source domains, so as to learn domain invariant features. Unlike contrastive learning algorithms that mine negative data pairs to maximize difference between the pairs, SelfReg relies on reducing the difference between positive data pairs using predefined losses. An advantage of this technique is that unlike hard negative mining, it is easier to recognize positive pairs. Identifying negative pairs from different domains usually requires expensive computation and complex algorithms. It may be easier in the image domain, but it is not so when it comes to audio where there are limited common invariant features to learn. This thesis doesn't use SWA as it was seen that it provided negligible improvement in preliminary experiments. An algorithm of SelfReg as well as an explanation of the algorithm has been added below.

- **Input data**: Data from three different domains are used as input. In some scenarios, only two datasets are used as input. This affects the IDCL, as it means one less interval to add another dataset has to be calculated.
- **Inter Domain Curriculum Learning (IDCL)**: SelfReg uses IDCL, which introduces new data domains as input in a staggered fashion. Use of IDCL was shown to perform better empirically via experiments in [48]. The calculation of the epochs in which data is to be added is shown in the algorithm. This could be extended to a larger number of input domains.
- **Sort input batches**: The data from input batches are sorted according to their classes to cluster similar data together and determine positive pairs accordingly.
- **Perform forward pass**: After a forward pass with input data batches sorted (clustered) according to classes, logits are obtained as the output of the dense layer of the backbone network.
- **Obtaining features**: An auxiliary model with the same layers as the backbone network, except the topmost dense layer, is initialized using the backbone network's current parameters. The same sorted input batches used in previous steps are passed into the auxiliary model to obtain features as output from the global average pooling layer.
- **Create a copy and shuffle**: Two copies of the logits and features are made. The clusters in both the copies are shuffled. One of the copy takes part in Mix-up.
- **Pass features through** f_{CDPL}: The copies are fed through an MLP layer called Class-specific Domain Perturbation Layer. This is to prevent the backbone network from learning a collapsed representation.

Algorithm 2: SelfReg algorithm

Data: From three source domains, collectively denoted as $\mathcal{S}$
Initialization: ResNet18 with model parameters Θ, hyperparameters $C_{Scale}, \lambda_{logits}$ and $\lambda_{features}$
for *epoch* ***in*** *epochs* **do**
  if *epoch* $\leq$ *(epoch/3)/2* **then**
    Input: Batches from Domain 1;
  end
  if *(epoch/3)/2* $\leq$ *epoch* $\leq$ *epoch/3* **then**
    Input: Batches from Domain 1 and 2 combined;
  end
  if *(epoch/3)* $\leq$ *epoch* $\leq$ *epochs* **then**
    Input: Batches from Domain 1, 2 and 3 combined;
  end
  Sort input batches: Sort according to class;
  Perform forward pass: Train using batches from input;
  Obtain features and logits: Features from second last layer;
  Create a copy and shuffle: Copy of logits and f_{CDPL} output. Shuffle amongst classes;
  Pass copied features through f_{CDPL};
  Calculate $\mathcal{L}_{ind}$: For both logits and features. shuffled and original.

$$\mathcal{L}_{ind}(\mathbf{z}) = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{z}_i^c - f_{CDPL}(\mathbf{z}_{j\in[\,1,N]}^c)\|_2^2$$

  Perform mix-up: On shuffled copies of features and logits;

$$\bar{\mathbf{u}}_i^c = \gamma \mathbf{u}_i^c + (1-\gamma)\mathbf{u}_{j\in[\,1,N]}^c$$

  Calculate $\mathcal{L}_{hdl}$: For both logits and features. shuffled, augmented and original;

$$\mathcal{L}_{hdl}(\mathbf{z}) = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{z}_i^c - \bar{\mathbf{u}}_i^c\|_2^2$$

  Calculate final loss: From classification loss, and $\mathcal{L}_{hdl}$ and $\mathcal{L}_{ind}$ after scaling;

$$\mathcal{L} = \mathcal{L}_c + C_{scale}(\lambda_{logits}(\mathcal{L}_{hdl\_logits} + \mathcal{L}_{ind\_logits}) + \lambda_{feature}(\mathcal{L}_{hdl\_feature} + \mathcal{L}_{ind\_feature}))$$

  Calculate gradients and update parameters: Update Θ to minimize $\mathcal{L}$.;
end

- **Calculate** $\mathcal{L}_{ind}$: The MSE loss between original features and the features passed through the f_{CDPL} is calculated. The same is done for logits. The loss is described in detail in the preliminaries 3 section.

- **Perform mix-up**: One copy of features and logits undergo mix-up with the other copy.

- **Calculate** $\mathcal{L}_{hdl}$: The MSE loss between original features and the features after Mix-up is calculated. The same is done for logits. The loss is described in detail in the preliminaries at 3.1.2 section.

- **Calculate Final loss**: These losses coupled with the classification loss forms the main loss as depicted in Equation 3.8.

- **Update parameters**: The gradients are calculated for this loss and fed to an optimizer to optimize the parameters of the model.

The algorithm of SelfReg is depicted in Algorithm 2. It can be seen from the algorithm that in order to implement $\mathcal{L}_{hdl}$ and $\mathcal{L}_{ind}$, one has to use Mean Square Error (MSE) loss, as described in the formula in Equations 3.5 and 3.7.

Figure 4.1: Illustration of Mixup and Cutmix from [70]. On the left, Mixup is applied using two images belonging to two different classes, where a clear overlay is observed. On the right, Cutmix has been applied on the same images, patching only a specific region from one image to the other.

4.4 Model Architecture and Optimization

We utilized the ResNet18 architecture as backbone network in this work due to the fact that it is a lightweight model with sufficient capacity for learning powerful generalizable features. In addition, this CNN-based model avoids capturing fine-grained details, which can undermine the model's domain generalizability and result in overfitting on the training domain. The backbone network is coded to provide logits as output to allow for more flexibility with any developed algorithms. As an activation function, we used ReLu non-linear activation function, a well-established choice for these models. The range of hyper-parameters used by the backbone network in the experiments performed in Chapter 5 is shown in Table 4.1

Across all experiments we use the same pre-processing pipeline. Firstly, the input audio data is converted into log-Mel spectrogram, which is then used as input to the backbone network. The reason for selecting log-Mel spectrograms as input has to do with the selected algorithms, as they are meant for images. The pre-processing pipeline can be seen in Figure 4.2.

Table 4.1: Details on backbone network, hyper-parameter range and input used in experiments

Parameters	Values
Backbone network	ResNet18
Activation function	ReLu
Optimizers	Adam, SGD, Adagrad
Learning rate	0.0001 - 0.1
Epochs	8 - 150
Batch size	32 - 128
Input	Log Mel spectrogram
Mel bins per window	128
Sample rate	16 kHz

SelfReg uses techniques Mixup [79] to mix two features from the same class. To further improve the algorithm performance, we consider Cutmix [70] as an augmentation strategy. Cutmix is build upon Mixup [79], which combines two images together in different random ratios so as to add noise to the original image. However, it tries to retain some characteristics of the original image by only replacing a part of the image. An example of Cutmix and Mixup from [70] can be seen in Figure 4.1.

Another optimization by changing the loss function of SelfReg was attempted. It is inherently difficult to identify learnable features from Log mel spectrograms. A new loss function introduced in [71] is the focal loss. [71] introduces a modulating factor to regular cross entropy loss and then tries to recognize outlier features rather than dominant features, which is rather helpful for DG.

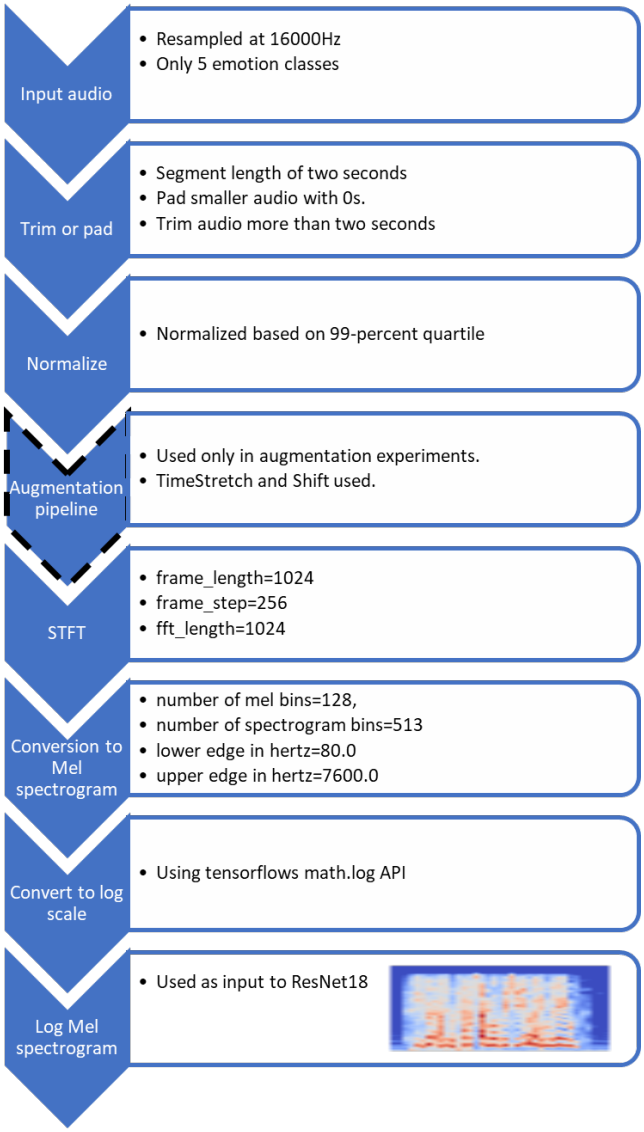


Figure 4.2: Our pre-processing pipeline

To further improve the performance, experiments related to augmentations are also performed. The library called Audiomentations [75] which is compatible with TensorFlow GPU and is suitable for performing large scale audio augmentation. This is done to induce similarity between audio from different languages.

4.5 Model pre-training strategy

While MLDG and SelfReg do not require pre-trained weights for initialization, their performance can significantly improve when appropriate pre-trained weights are used. In the image recognition domain, both MLDG and SelfReg used weights from [82], pre-trained on ImageNet dataset, for initialization. Furthermore, in the field of audio classification, pre-training not only improves training speed and probability of convergence, but can enable learning domain invariant features ([68]). Thus, this necessitates pre-training model to achieve better results in this thesis. For this purpose, VoxCeleb was used to pre-train ResNet18. VoxCeleb [80] is a large-scale audio dataset,

Table 4.2: Details on backbone network, hyper-parameters and inputs used during pre-training

Pre-training Parameters	Parameters
Backbone network	ResNet18
Optimizers	Adam
Learning rate	0.0005 with decay step of 75
Batch size	256
Epochs	75
Loss	CCE
Input	Log Mel spectrogram
Mel bins per window	128
Sample rate	16 kHz

which consists of over 150,000 samples obtained from 1,251 speakers. The hyperparameters used in pre-training are shown in Table 4.2, where an early stopping mechanism was employed to avoid overfitting and improve models generalizability. In Chapter 5, when referring to pre-trained model, we implicitly indicate a ResNet18 model pre-trained on VoxCeleb.

Chapter 5

Experiments

In this chapter, all experiments conducted to answer research questions are outlined. These experiments evaluate the performance of MLDG and SelfReg. The results and their inference are also presented in Chapter 5. The detailed results of experiments has been added to Appendix A.

5.1 Datasets

During experimentation nine distinct datasets were considered, namely CREMA-D, AESDD, CAFE, EMODB, RAVDESS, TESS, SAVEE, EMOVO and SHEMO, the details of whom are presented in Section 2.5. These datasets were selected due to their availability and ability to capture a wide range of speakers' characteristics. In addition to the versatility of speakers' origins, these datasets also introduce multi-linguistic problems to the emotion recognition task. Therefore, each dataset is considered a separate domain during experimentation. The datasets utilized during training are called train domains, while those used for performing final tests are named test domains. In domain generalization, the task at hand will be to learn common invariant features from each of train datasets, which have their own unique distribution of data.

The audio files of each dataset were re-sampled at 16kHz using the FFmpeg library [76], a standard approach to reduce the computational complexity without removing vital information from the original audio signals. Afterwards, the re-sampled audio signals were either trimmed or padded to two-seconds and normalized. Following the normalization, we computed STFT with a fixed frame length of 1024 to construct spectrograms and convert them from the linear to log-Mel scale with 128 bins, resulting in the final log-Mel spectrograms. It is important to note that, Log Mel spectrograms of speech audio data as input were used throughout experimentation, unless mentioned otherwise.

5.2 Evaluation strategy

The thesis also follows a set of evaluation strategies while performing multiple experiments. These strategies are outlined below:

- **Uniform data pre-processing**: Models explored throughout experimentation utilize identical pre-processing pipelines and use 128-bins log-Mel spectrogram, as explained in Section 4.4.
- **Consistent data partitioning**: The same train/test split was used during evaluation. Each datasets predefined split was choosen to enable reproducibility. In cases where no standard split exists, such as for AESDD and EMOVO dataset, splits were populated ensuring that no speaker overlap is present between the train and test set.
- **Eliminating randomness**: A minimum of 5 iterations per experiment was performed to reduce any randomness due to weights or kernels initialization. Furthermore, a seed was

Table 5.1: Emotion datasets used in the six domain generalization scenarios. Here, ○ represent train domains, while ● indicate the test domain.

Scenario	CREMA-D	SAVEE	TESS	RAVDESS	AESDD	CaFE	EMOVO	ShEMO	EmoDB
SC1	○	○	○	●					
SC2	○	○	●	○					
SC3		○		○		●			○
SC4					●		○	○	○
SC5	○	○		○				●	
SC6			○					●	○

provided in all functions regarding data partitioning and shuffling, allowing the control over the training process.

- **Consistent set of emotions**: A set of five emotions, i.e. happiness, sadness, anger, disgust and fear, were selected during experimentation. This choice was dictated from the need to combine distinct datasets (regarded as domains) to perform domain generalization, thus only the common emotions across all datasets could be utilized.

- **Rigorous evaluation metrics**: As the main performance metric, Sparse Categorical Accuracy (noted as accuracy) was selected in the experiments, while the confusion matrix was computed in various cases to draw further conclusions. Moreover, the Categorical Cross-Entropy (CCE) loss was utilized during training (unless mentioned otherwise), as it is standard for most classification tasks.

5.3 Scenarios

Domain generalization experiments were executed on all or a subset of six different scenarios with distinct training and test domains. To evaluate the perform of the developed approaches in domain generalization, six distinct scenarios were considered, each introducing a unique challenge. These scenarios were constructed by combining various datasets into a single one, which had different domains (i.e., the individual datasets). The datasets utilized as train or test domains for the six different scenarios are shown in Table 5.1.

The selection of these specific combinations was based on the following:

- **Single language cross-corpus DG**: Scenario 1 and 2 use only English speech emotion datasets as training and test domains. Scenario 2 was added because of its high sparse categorical accuracy score, as presented in sections 5.5 and 5.6.

- **Cross-lingual cross-corpus DG (multiple similar languages)**: Scenario 3 uses two English and one German speech emotion datasets for its training, and a French speech emotion dataset for its test data. This scenario uses speech emotion datasets from languages, which share lexical similarity with English.

- **Cross-lingual cross-corpus DG (multiple dissimilar languages)**: Scenario 4 uses three different languages (i.e., German, Persian and Italian) for training and Greek language for testing. These languages differ greatly from each other.

- **Cross-lingual cross-corpus DG (two dissimilar languages)**: Scenario 5 uses only English speech emotion datasets for training and Persian (a language significantly different from English) for testing. This is an extreme case, as the train domain does not learn domain invariant features of emotions from different languages. Rather it tries to learn domain invariant features of emotions only from English. There may not be many learnable common features between the training and test datasets.

- **Cross-lingual cross-corpus DG (corpora with high accuracy results)**: Scenario 6 uses speech emotion datasets that have a good accuracy score when trained and tested individually using the backbone network ResNet18, as presented in Section 5.4.1.

5.4 Baseline experiments

Before running experiments, it is necessary to verify if the chosen backbone network and the chosen algorithms work properly and what are their performances. Thus, we perform two preliminary experiments to determine the baselines.

5.4.1 Backbone model performance

From these experiments we aim to observe whether the backbone network has an acceptable performance when it comes to recognising emotions in a single speech emotion dataset. Thus, we performed experiments with all datasets listed in Table 5.1. A detailed list of the hyper-parameters used in these experiments is shown in Table 5.2, where a cosine learning rate schedule with decay steps of 150, and early stopping mechanism were employed. The results in Figure 5.1 are an average of sparse categorical accuracies obtained over four experiments. The error bars depict standard deviation of sparse categorical accuracies over four independent runs.

From Figure 5.1, we observe that the performance our choosen model is lower when compared to state of the art architectures in various datasets. However, during our experimentation only

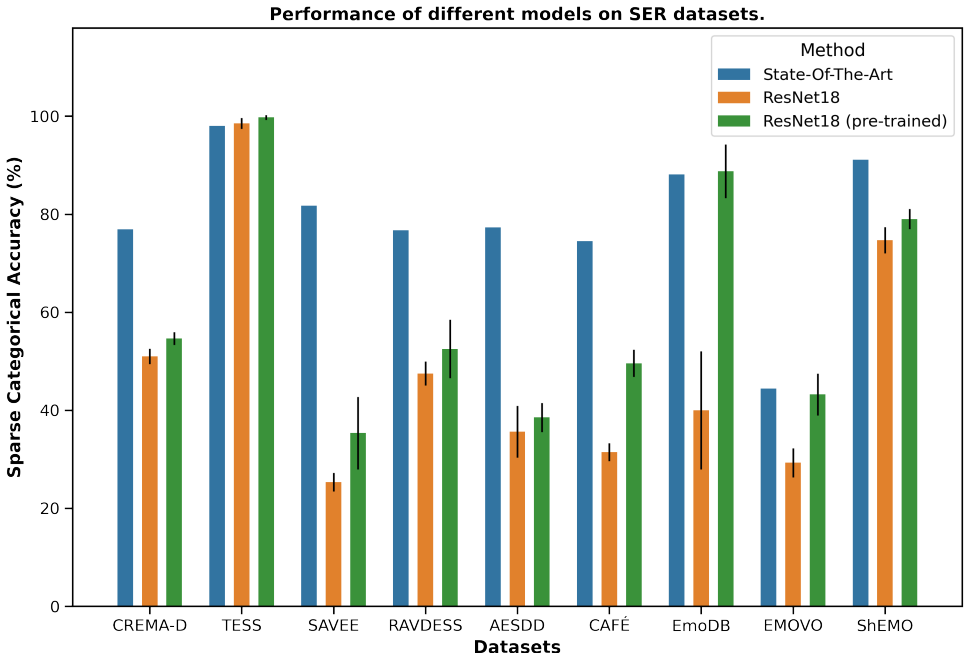


Figure 5.1: Baseline results on speech emotion dataset. The state-of-the-art accuracies are from [83] except TESS, whose accuracy is from [84].

Table 5.2: Hyper-parameters of preliminary experiments with individual speech emotion datasets.

Model	Optimizer	Learning rate	Epochs	Batch Size
ResNet18	SGD	0.005	150	64
ResNet18 (pre-trained)	Adam	0.005	150	64

five emotions were utilized, which resulted in a considerable reduction of available training data. Therefore, a direct comparison with other baselines is of little significance. Additionally, datasets, such as SAVEE and EMOVO, have very few training data, even without the reduction of emotion classes, further incommoding to the task's difficulty. The poor performance of these datasets could also be attributed to the choice of emotions that are utilized. Disgust is much hard to determine and decode when compared to the neutral state which was discarded, since it was not present across all datasets. [67].

5.4.2 MLDG and SelfReg performance on image domain

Additionally, experiments were conducted to verify whether the developed algorithms are improving models' generalizability over the vision domain, from which they were originated. For this, we conducted experiments with PACS dataset, which is widely-used to validate domain generalizability over the vision domain. The PACS dataset, introduced in [78], has a total of four different domains - Art paintings, Photos, Sketches and Cartoons. These domains each contains pictures classified into seven different classes - Dog, Elephant, Giraffe, Guitar, Horse, House and Person. Domain generalization can be performed on different combination of these four domains used as training and test domains. At first, the PACS dataset was trained and tested on ResNet18, the backbone network, both with weights pre-trained with ImageNet obtained from [82], and without using pre-trained weights. Then, the PACS dataset was trained and tested on MLDG and SelfReg, both pre-trained with ImageNet, in the same way it was done in their respective papers [29, 48]. The hyper-parameters of these experiments are shown in Table 5.4, while the results are shown in Table 5.3. These results are the average of 5 iterations.

From the results, it is clear that MLDG and SelfReg do improve cross-corpus DG results when applied to the backbone network (ResNet18). SelfReg results depend on the augmentation provided to the PACS dataset. We only used horizontal flip and random saturation. The use of all augmentations performed in the original PyTorch implementation destroyed the data and reduced accuracy scores drastically. As the use of augmentation varies in the TensorFlow reproduction performed in this thesis, the accuracy of the TensorFlow's SelfReg algorithm drops below the accuracy of its Pytorch implementation. It was also observed that IDCL provides significant improvement to SelfReg when tested on the PACS dataset.

Table 5.3: Preliminary experiments on PACS dataset

Model	Accuracy
Supervised	22.31
Supervised (pre-trained)	59.9
MLDG	70.70
SelfReg	64.01

Table 5.4: Hyper-parameters of preliminary experiments with PACS datasets. In SelfReg, Piecewise Constant Decay was utilized with a decay step of 290.

Method	Optimizer	Learning rate	Epochs	Batch Size
Supervised	SGD	0.01	100	128
Supervised (pre-trained)	SGD	0.01	100	128
MLDG (pre-trained)	SGD/SGD	0.005 / 0.005	10	128
SelfReg (pre-trained)	Adagrad	0.007 - 1E-8	18	128

5.4.3 Supervised baseline performance on all scenarios

After the preliminary experiments, the next step was to determine the backbone network's performance while considering cross-corpus DG, with which the performance of MLDG and SelfReg could be compared with. ResNet18 was used directly, without any learning techniques, to perform domain generalization on the following six scenarios as described before. The hyper-parameters of these experiments are shown in Table 5.5, while the results are shown in Table 5.7. The learning schedule Cosine Decay with decay steps = 100 and alpha = 0.05, and early stopping callback was used. These results are the average of 5 iterations. The hyper-parameters and optimizers were kept constant for all scenarios for further comparisons. The optimizers were chosen in such a way that they provided the best results for all scenarios. It can be seen that Adam optimizer performed best. If a baseline result has high standard deviation values, then it means that it could be fine-tuned further to get slightly better results.

From the results Table 5.7, it can be seen that pre-training generally provides better results, as mentioned in many other literature, such as [68]. These results shows that pre-training improves generalization. However, in case of Scenario 5, which is an extreme situation, a reduction in accuracy was observed. It could be hypothesized that pre-trained weights may cause the model's parameters to converge much more easily towards a predictive function much more suitable for English language (which form the training set). This would make it difficult for the model to predict Persian language, as it is much different than English. On the other hand, using random initialization could slow down this convergence, preventing the model from overfitting towards English. Table 5.6 shows the results for two different optimizers, keeping the hyper-parameters same. These results are the average of 5 iterations. The accuracy pattern of Scenario 5 remains the same even when optimizers were changed, as it can be seen in Table 5.6. This shows that the advantage of pre-training remains even though optimizers are changed. It is also seen that SGD performs better than Adam for scenario 5.

Table 5.5: Hyper-parameters of ResNet18 on baseline cross-corpus DG experiments of Table 5.7.

Method	Optimizer	Learning rate	Epochs	Batch Size
Supervised	Adam	0.01	100	64
Supervised (pre-trained)	Adam	0.002	100	64

Table 5.6: Baseline cross-corpus DG experiments on Scenario 5 with different optimizers

Scenario	ResNet18-SGD	ResNet18-Adam	ResNet18-SGD (pre-trained)	ResNet18-Adam (pre-trained)
SC 5	43.7 ± 2.99	40.38 ± 4.20	25.37 ± 11.81	24.88 ± 7.21

Table 5.7: Baseline cross-corpus DG performance using ResNet18

Scenario	Supervised	Supervised (Pre-trained)
SC 1	27.51 ± 2.89	31.57 ± 2.24
SC 2	33.26 ± 4.10	49.96 ± 8.17
SC 3	25.51 ± 0.89	41.69 ± 3.48
SC 4	26.92 ± 1.70	29.36 ± 3.34
SC 5	40.38 ± 4.20	21.22 ± 4.41
SC 6	20.83 ± 3.89	24.81 ± 6.15

5.5 Performance of MLDG

The goal of this section is to obtain the best sparse categorical accuracy score possible to determine the performance for all the scenarios explained in the evaluation strategy using the meta-learning algorithm (MLDG). The scenarios cover different cross-corpus DG possibilities over a single language corpora (English) or over multiple language corpora as required by sub-research questions 1 and 2.

5.5.1 Experiments

The first experiment was to train and test MLDG both with and without ResNet18's pre-trained weights. The results of this experiment are shown in Figure 5.2. It can be seen that MLDG actually shows improvement on all scenarios compared to the baseline results shown in Table 5.7, with the exception of Scenario 5 (which is an extreme case scenario). The hyper-parameters and optimizers used are noted in Table 5.8. Adam optimizer was chosen for these experiments as it performed better during baseline experiments. The detailed results has been added to the Appendix as A.2.

The second experiment was to check the influence of optimizers on the performance of MLDG. Optimizers A and B of the MLDG model are varied for all six scenarios. It was found that Adam optimizer for both optimizer A and B performed the best across all scenarios, except for scenario 5 and 6. As a result, the experiment on scenario 5 and 6 was re-run by varying optimizers separately, to select the best optimizer combination. For scenario 5, Adam as optimizer A and SGD as optimizer B had the best accuracies. For scenario 6, SGD as optimizer A and B had the best accuracies. These results are shown in Tables 5.9 and 5.10. Table 5.10 does not include pre-trained scores as the pre-trained model did not converge properly for scenario 6 when using SGD as both optimizer A and B. The hyper-parameters were the same as the previous step (see Table 5.8) and only the optimizers were changed.

5.5.2 Observations on performance of MLDG

From the above results, we conclude that the MLDG model was sensitive to the optimizer used. Using a different optimizer prevented the model from converging. On the other hand, utilizing a proper optimizer helped in achieving higher accuracies as well as converging the model to a certain extent. We observe that the accuracies still have high standard deviation. The effects of choosing the wrong optimizer can be clearly seen in Figure 5.3 which showcases accuracies of MLDG in different scenarios using two different optimizers. These results are the average of 5 iterations.

Another point that could be observed is that pre-training the backbone network with a dataset containing an extensive collection of speech audio, such as VoxCeleb, provides a significant improvement in accuracies. This is in line with the standard results in literature, such as [68], which state that pre-training on relevant datasets help create a robust model suitable to perform domain generalization. An exception is the extreme case of scenario 5, where using pre-trained weights and/or applying the algorithm does not improve the performance more than the baseline accuracy score. Instead of increasing, the accuracy decreased. This could be attributed to the lack of similarity in *vocals* between English and Persian languages. This could significantly decrease the number of common invariant features that could be learnt to distinguish emotions, which results in a large number of false predictions. This is visible in the confusion matrices of scenario 5, when

Table 5.8: Hyper-parameters of MLDG while performing domain generalization experiments

Method	Optimizer	Learning rate	Epochs	Batch Size
MLDG	Adam / Adam	0.001 / 0.0004	15	64
MLDG (pre-trained)	Adam / Adam	0.003 / 0.0008	10	64

Table 5.9: MLDG accuracies varying both optimizers for Scenario 5

Scenario	MLDG (Adam/Adam)	MLDG (Adam/SGD)	MLDG (Pre-trained) (Adam/Adam)	MLDG (Pre-trained) (Adam/SGD)
SC 5	22.97 ± 4.14	28.65 ± 12.60	23.77 ± 4.66	38.41 ± 5.07

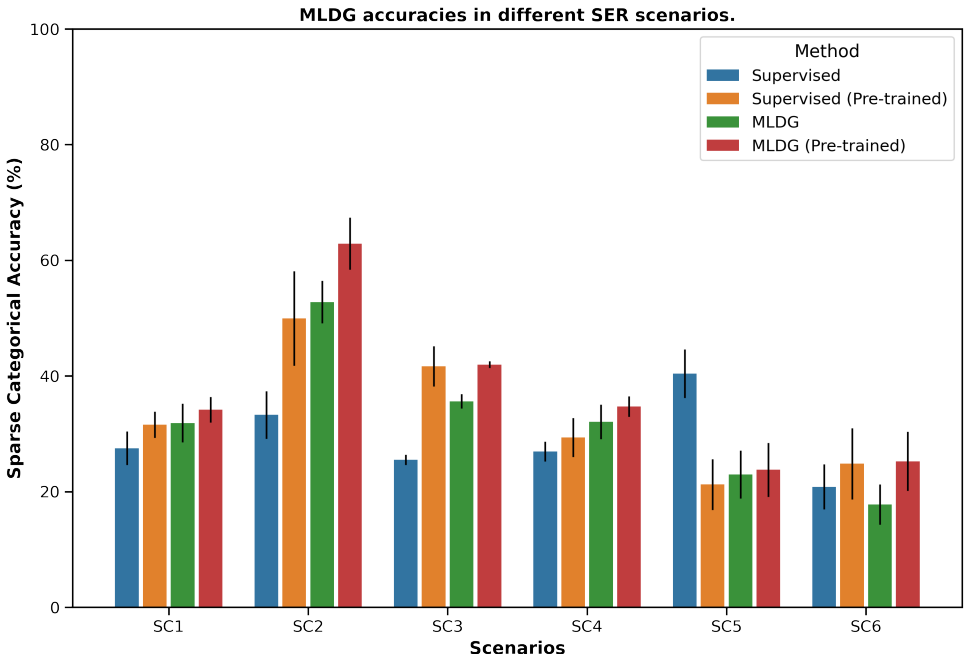


Figure 5.2: Sparse categorical accuracies of MLDG for different scenarios

compared with scenario 2 which has the highest accuracy, as seen in Figure 5.4. This misinterpretation of a certain emotion does seem to be dependent on the emotion itself, as removing the over-predicted emotion simply shifts the problem to another. These over-predicted emotions are usually those that are easier to predict (i.e., emotions with high arousal values), such as fear and anger. According to [67], emotions such as anger and fear are often easier to predict due to their easily distinguishable features such as high intensity. On the other hand, emotions such as disgust is often harder to identify due to lack of easily distinguishable features. This points to a problem in log-mel spectrogram, i.e the lack of proper representation for every emotions.

This is evidence in Figures 5.5a and 5.5b, where scenario 1 was considered with and without the fear emotion, respectively. While including fear class, we observe that the model tends to over-predict fear across all samples. However, upon the removal of the fear emotion, we note in Figure 5.5b that anger now is over-predicted.

To illustrate this problem clearly, t-SNE embeddings for Scenarios 1,2 and 6 are shown in Figure 5.6. Scenarios 1,2 and 6 were chosen as Scenario 2 has the best result, Scenario 1 has an intermediate result and Scenario 6 has the poorest result. Specifically, in Figure 5.6a and 5.6b, although proper clusters are formed, some emotions are not predicted at all. As for scenario 6 in Figure 5.6c, there is a high overlap between all t-SNE embeddings clusters due to model's poor performance.

Another observation is that, ignoring baseline results of Scenario 5, MLDG did improve pre-trained accuracy of Scenario 5, as it can be seen in Figure 5.2. This is due to the inherent learning process which uses two different losses. This slows down the pre-trained backbone network from overfitting towards one single language and thus improves accuracy when using pre-trained weights. This is unlike normal ResNet18 behaviour, which tends to overfit quickly towards a language when

Table 5.10: MLDG accuracies varying both optimizers for Scenario 6

Scenario	MLDG Adam/Adam	MLDG SGD/SGD
SC 6	17.78 ± 4.14	27.16 ± 2.61

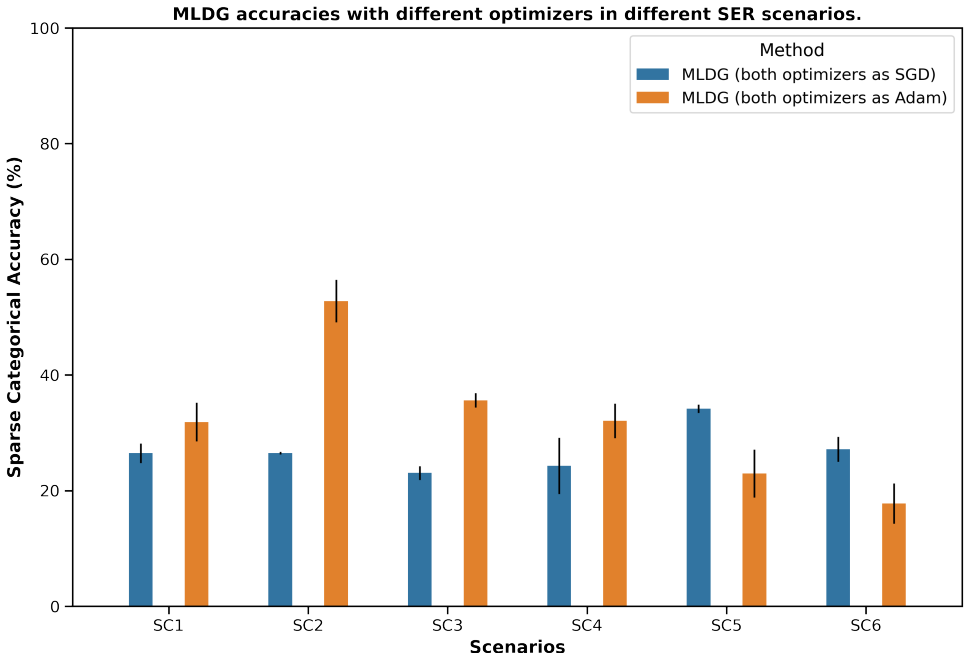
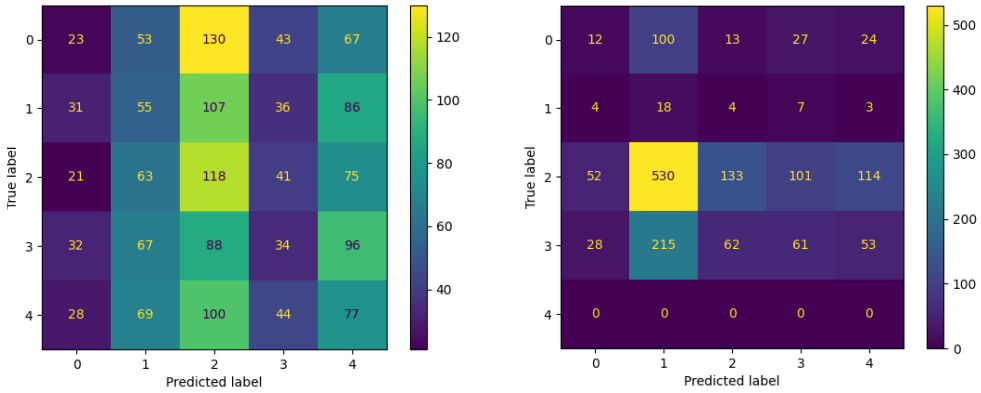


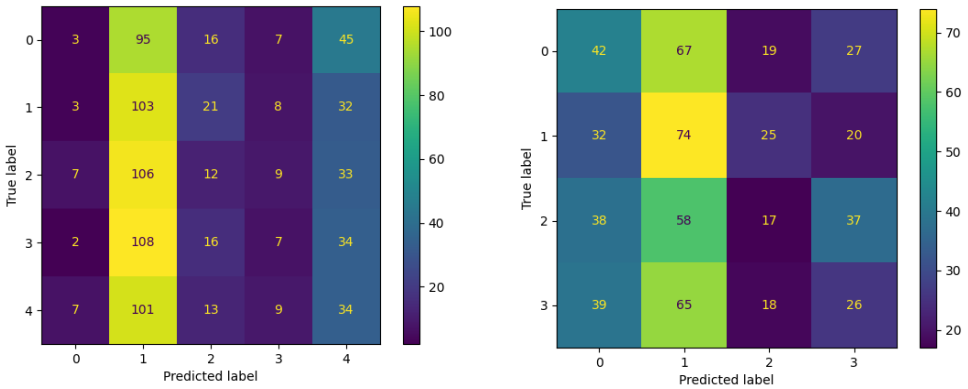
Figure 5.3: Sparse categorical accuracies of MLDG without pre-trained weights for different scenarios with varying optimizers

trained on a network initialized with pre-trained weights. One may note that the accuracy score of Scenario 2, which is very high. This could be attributed to the ease of prediction of the test dataset which is the TESS dataset. TESS has very high baseline scores, as seen in Figure 5.1. However,



(a) Scenario 2 (b) Scenario 5

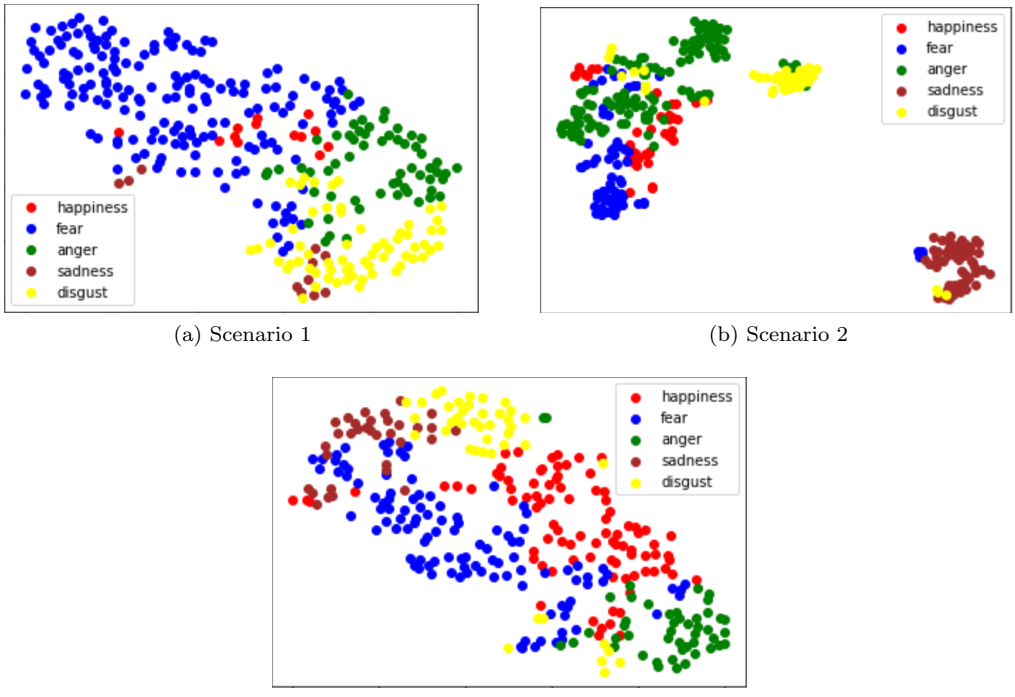
Figure 5.4: MLDG confusion matrices for Scenarios 2 and 5. Values 0, 1, 2, 3 and 4 correspond to happiness, fear, anger, sadness and disgust, respectively.



(a) Scenario 1 (b) Scenario 1 without fear emotion.

Figure 5.5: MLDG confusion matrices for Scenarios 1. Values 0, 1, 2, 3 and 4 correspond to happiness, fear, anger, sadness and disgust, respectively. In confusion matrix of scenario 1 "without fear" emotion, values 0, 1, 2 and 3 correspond to happiness, anger, sadness and disgust, respectively.

even with the improvement in sparse categorical accuracies, the accuracies are not good enough to be practical. Scenario 1,3 and 6 have low accuracies. This could be attributed to the presence of small sized datasets such as SAVEE, TESS in the train domain. Hence, the performance of MLDG is still not good enough to be practical in the field of domain generalization on SER, even with an improvement to baseline accuracy.



(a) Scenario 1 (b) Scenario 2

(c) Scenario 6

Figure 5.6: MLDG t-SNE embeddings for different scenarios

5.6 Performance of SelfReg

The goal of this section is to obtain the best sparse categorical accuracy score possible to determine the performance for all the scenarios explained in the evaluation strategy using the contrastive learning algorithm (SelfReg). The scenarios cover different cross-corpus DG possibilities over a single language corpora (English) or over multiple language corpora as required by sub-research questions 1 and 2. The following subsection explains the experiments performed.

5.6.1 Experiments

The first experiment was to train and test SelfReg both with and without ResNet18's pre-trained weights according the the six scenarios described earlier. The results of this experiment are shown in Figure 5.7. It can be seen that SelfReg only improves the performance on Scenario 1 and 4 compared to the baseline accuracies as seen in Figure 5.7. The hyper-parameters and optimizers used are noted in Table 5.11. Preliminary experiments were conducted to investigate which optimizer is most suited for SelfReg. Adagrad, introduced in [85], seems to result in the best model performance. We omit the results of this hyper-parameter search, since they are of no significant value.

The next experiment replaces the Categorical Cross Entropy loss with a state of the art loss called Focal loss [71]. Additionally, the Mix-up augmentation was replaced with Cut-mix [70], since it was shown to increase the generalizability of trained models [70]. Those modifications were applied in the original SelfReg architecture, after which the aforementioned experiments where repeated, thus determining the performance improvement over the original results shown

Table 5.11: Hyper-parameters of SelfReg while performing cross-corpus DG experiments

Method	Optimizer	Learning rate	Epochs	Batch Size
SelfReg	Adagrad	0.01	32	64
SelfReg (pre-trained)	Adagrad	0.005	36	64

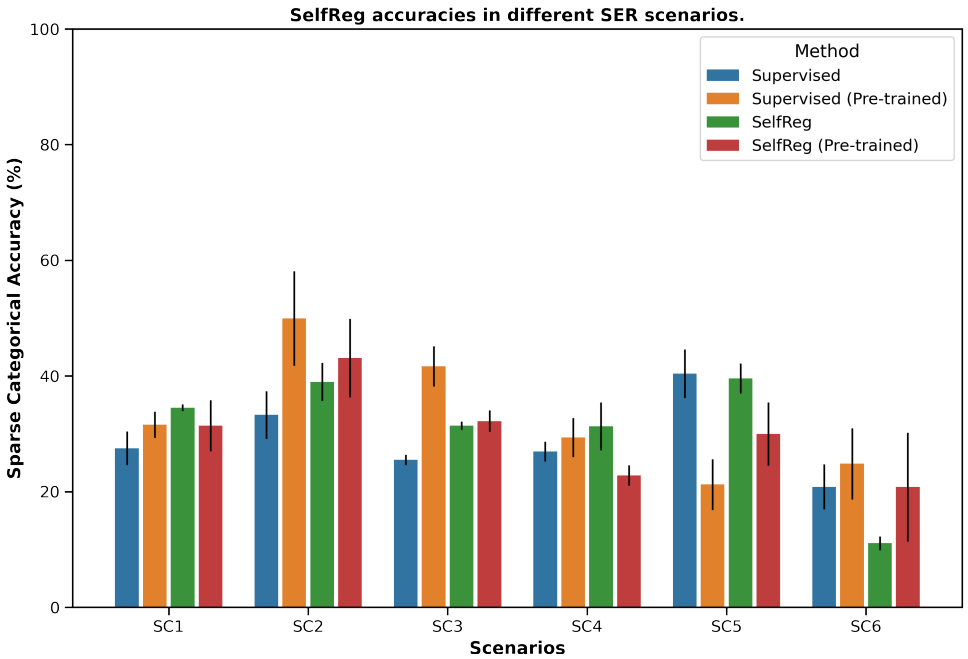


Figure 5.7: Sparse categorical accuracies of SelfReg for different scenarios

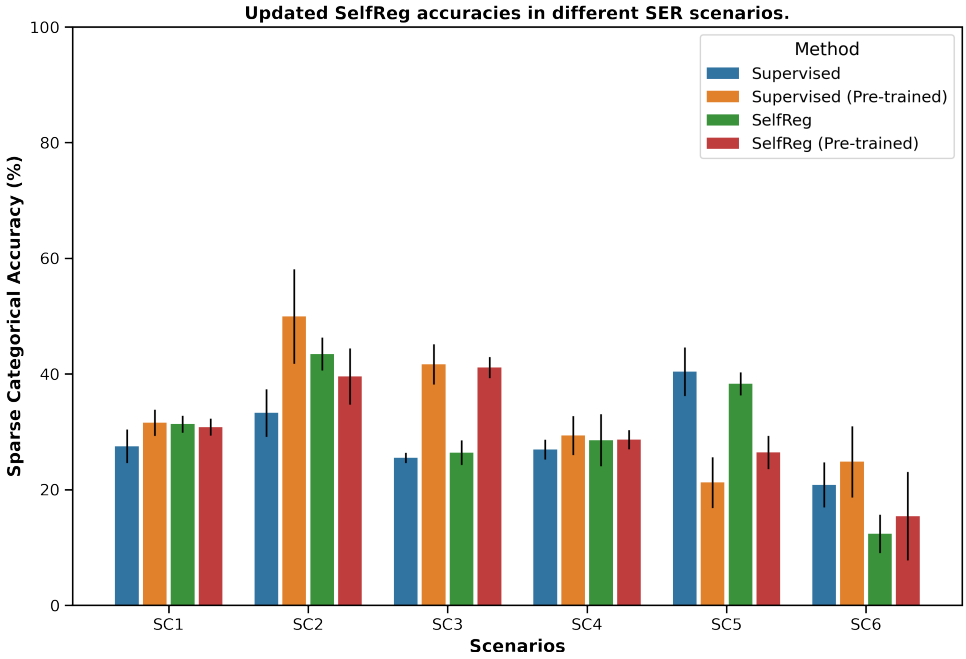


Figure 5.8: Sparse categorical accuracies of modified SelfReg for different scenarios

in Figure 5.7. For a rigorous evaluation, the same hyper-parameters and optimizer as listed in Table 5.11 were used. It is seen that the results in Figure 5.8 did not change a lot compared to Figure 5.7. Even then, the accuracy in most scenarios is worse than the baseline accuracies.

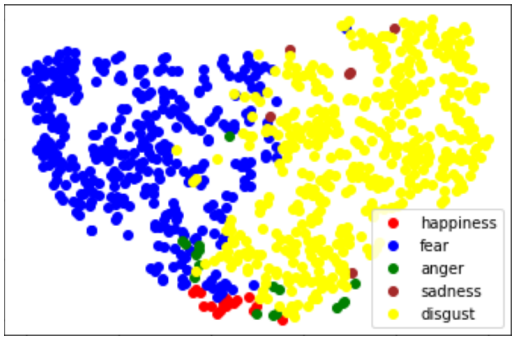
5.6.2 Observations on performance of SelfReg

The low accuracy of the SelfReg experiments could be attributed to the working concept of SelfReg model. SelfReg depends on aligning positive pairs of data together by optimizing two different losses called Individualized In-batch Dissimilarity Loss and Heterogeneous In-batch Dissimilarity Loss. To decrease these losses, the model learns invariant common features belonging to the domain. However, if the input data does not have many common invariant features, then the positive pairs do not align properly, increasing the difficulty to decrease the losses. These results show the disadvantage of using positive pairs, as the model fails to distinguish between emotions and over-predicts some emotions, a result of insufficient learnt representations. This impedes the formation of proper cluster of similar features (as the objective of SelfReg is to bring similar data and features together). To illustrate this problem clearly, t-SNE embeddings for Scenarios 1,2 and 6 are shown in Figure 5.9. Scenarios 1,2 and 6 were chosen as Scenario 2 has the best result, Scenario 1 has an intermediate result and Scenario 6 has the poorest result. It can be seen that a few emotions such as anger and fear have been over-predicted, similar to MLDG, as it can be seen in Figure 5.6.

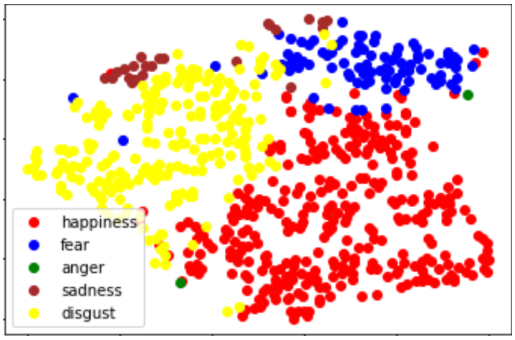
Another point to be noted is that in some scenarios, pre-training seems to reduce the accuracy scores drastically. Initializing using pre-trained weights could possibly interfere with the alignment of similar data from different classes and thus, the formation of proper cluster of similar features.

It is also seen that the results of the SelfReg after introducing focal loss and Cutmix shown in Figure 5.8 were worse than the original results presented in Figure 5.7. In the literature presented in [71], although the focal loss helps identify outlier features, it was also mentioned that it helps better identify negative pairs. This is the opposite of SelfReg's objective, which is to align positive pairs. So, this could be a reason for its bad performance.

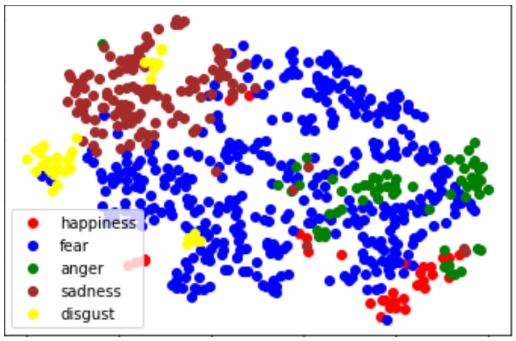
The sparse categorical accuracies of different scenarios show do not show improvement from the baseline (except scenario 1 and 4). Using pre-trained weights tends to decrease accuracy in



(a) Scenario 1



(b) Scenario 2



(c) Scenario 6

Figure 5.9: SelfReg t-SNE embeddings for different scenarios

scenarios (except Scenario 2, 3 and 6) The models have low accuracies due to lack of common representations between the datasets of the training and test domains. Perhaps it would be better if the SER network learns to distinguish emotions by mining hard negative pairs with distinguishable features, or by using a discriminator and a generator (even though they require complex computations and increased resources).

The accuracies are not good enough to be practical. This could be attributed to the presence of small sized datasets such as SAVEE, TESS in the train domain, as contrastive learning algorithms heavily depend on the large amounts of data. Scenario 2 has a decent accuracy. However, the test domain of Scenario 2 is TESS, which has very high standard recognition rates even when trained as an individual dataset, as seen in Figure 5.1, thus it is easy to identify.

5.7 Identifying the cause of low performance

As seen in the previous results, the accuracy scores of both algorithms are not good enough to be practical. This could be due to a problem with the learning technique, the backbone network, or the data used as input. In the next few experiments, we try to determine if changing the learning technique or the backbone network could improve the current results. Finding a better learning technique or model out of scope of this thesis, however. The experiments explained in this section tries to answer the sub-research question 3.

5.7.1 Experiments

The first experiment is to change the learning technique to something other than MLDG or SelfReg. This new learning technique, called RSC [69], was fine-tuned and uses different hyper-parameters.

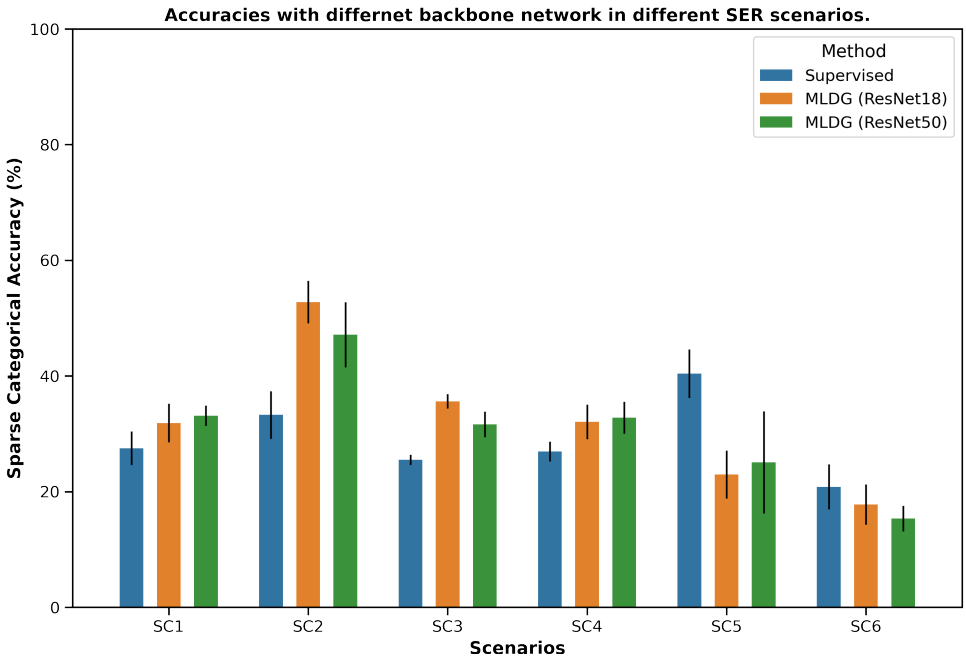


Figure 5.10: Sparse categorical accuracies for different scenarios by varying backbone networks

The optimizer used by RSC is Adam, with a learning rate of 0.0004. The model is trained for 50 epochs with a training batch size of 64. The results of this experiment are shown in Table 5.12.

Table 5.12: Performance comparison between RSC, MLDG and SelfReg

Scenario	MLDG	SelfReg	RSC
SC 1	31.87	34.51	40.3
SC 3	35.62	31.38	43.1
SC 4	32.06	31.3	38.5

The accuracies, which were used for comparison, are the results of the previous MLDG and SelfReg experiments presented in Tables A.2 and A.3. Although the original RSC paper uses ResNet50, this experiment changes the backbone of the network to ResNet18, which is used in other models.

The next experiment is to observe whether replacing ResNet18 with a more powerful model leads to better results. To proceed with this experiment, the backbone network of MLDG was changed from ResNet18 to ResNet50. The hyper-parameters remain the same as the ones used for experiments in Section 5.5, the ones specified in Table 5.8. We only use MLDG model to vary the backbone network. This is because unlike SelfReg, MLDG had shown a positive improvement in accuracy scores from the baseline for all scenarios except Scenario 5. The results are shown in Table A.7 and in Figure 5.10.

5.7.2 Observations on varying DG technique and backbone network

It can be seen from the results that there is a significant improvement for all scenarios when RSC is used. On the other hand, changing the backbone network to a better CNN network does not seem to significantly change the results (the results show small improvements in some scenarios and small decrements in some scenarios), as it can be seen in Figure 5.10. A point to note is that using a more powerful backbone network does not necessarily improve the results, as it tends

to overfit towards the training set by learning many features, most of which tend not be domain invariant.

Another point to be noted is that techniques which help in learning outlier features and tend to control the amount of dominant features learnt seem to perform better. This can also be seen in many other articles such as Adversarial Discriminative Domain Generalization (ADDoG) [74], which uses techniques categorized as representation learning to perform cross-corpus DG in audio seem to perform better, as they control the features to be learnt to some extent. In ADDoG, the model tries to train encoder and critic separately to control what representations are learnt (by the encoder) and to distinguish between representations (by the critic).

The last point to be noted is that even on replacing older methods (backbone networks and DG techniques) with superior methods in the experiments, the achieved improvements are not significant. This could mean that the low accuracy scores may be the result of a problem present in the input form of data (the log-mel spectrogram), rather than a problem with the model. The log-mel spectrogram may not have enough domain invariant features that could properly distinguish between emotions, even between same languages. This could be due to the difference in the expression of emotions in different languages. Some words which express anger in a language may sound similar, and/or have similar features with words that express happiness in a different language (because both anger and happiness have high arousal values). Consequently, the model may have a hard time distinguishing between emotions learnt from different training domains. Therefore, there is a need to discover a different type of representation that can clearly represent the features of different emotions without much overlap.

5.8 Performance evaluation with varying input

As noted in the previous sections, the accuracy scores are still not good even if changes to the learning techniques provided some improvements. Another avenue for improvement is varying the type of input. In the next few experiments, we try to determine if modifying the input, such as augmenting the input, improves the current results from Section 5.5. A point to be noted is that these experiments are to determine which path could lead to a higher improvement. Finding a better form of input or best augmentation technique is out of scope of this thesis, however. The experiments explained in this subsection tries to answer the the sub-research question 4.

5.8.1 Experiments

The first experiment is to create a copy of the original input speech audio data used in the six scenarios and augment it before introducing it to the pre-processing stage. The augmented audio input is added along with the original audio input. As a result, the number of inputs is doubled. These inputs are used to perform experiments on MLDG and SelfReg models, to determine their performance in six scenarios. The augmentation are obtained from the Audiomentations library [75] and are presented in Table 5.13. Other parameters of the experiment remains the same as the experiments performed in Section 5.5 and 5.6. This experiment is executed using both for MLDG and SelfReg models. However, in the case of SelfReg, this experiment was performed only on the model which was not pre-trained with VoxCeleb. This is because SelfReg shows better performance in most scenarios without pre-training. The results of this experiment is shown in Figure 5.11 and 5.12.

The next experiment is similar to the previous one, except that only the augmented data is used as input. The augmentation used from the Audiomentations library are presented in Table 5.13 (same as in the previous experiment). Other parameters of the experiment remain the same as in the experiments performed in Section 5.5 and 5.6. This experiment is used to verify whether the improvement in accuracies of MLDG and SelfReg models are due to augmentation or due to the increase of data samples. However, in the case of SelfReg, this experiment was performed only on the model which was not pre-trained with VoxCeleb. The results of the experiment are shown in Figure 5.11 and 5.12.

Table 5.13: Augmentations used from the Audiomentations library

Augmentation	Explanation	Parameters
Time Stretch	Stretches the signal along the time domain without changing the pitch	min_rate = 0.8 max_rate = 1.25 probability = 0.7
Time Shift	Shifts the signal along the time domain	min_fraction =- 0.5 max_fraction = 0.5 probability = 0.8

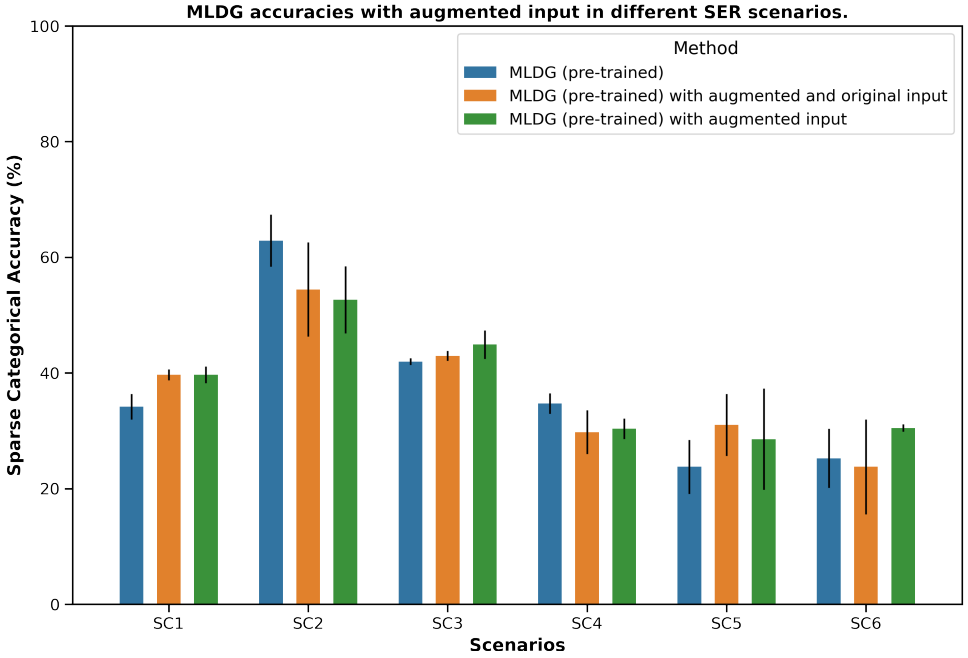


Figure 5.11: Sparse categorical accuracies of MLDG for different scenarios with augmented data

The final experiment is different from all other experiments of this thesis. The goal of the experiment is to perform domain generalization on embeddings of different dataset combinations (according to the six scenarios) extracted from the heavily pre-trained model released by Google, called TRILLson [77]. The MLDGs backbone network is replaced by a simple Multi-Layer Perceptron (MLP) network, as the TRILLson model has already been pre-trained and does not require further training. Hence, only the upper dense layer is trained by MLDG. The MLPs details are

Table 5.14: Hyper-parameters of MLDG when using TRILLson [77] embeddings

MLDG Parameters	Values
Backbone Network	3-layer MLP
Kernel regularizer	L2 with rate 1e-5
Optimizer	Adam / Adam
Learning rate	0.02 / 0.008
Epochs	30
Batch size	64
Input shape	(1024,1)

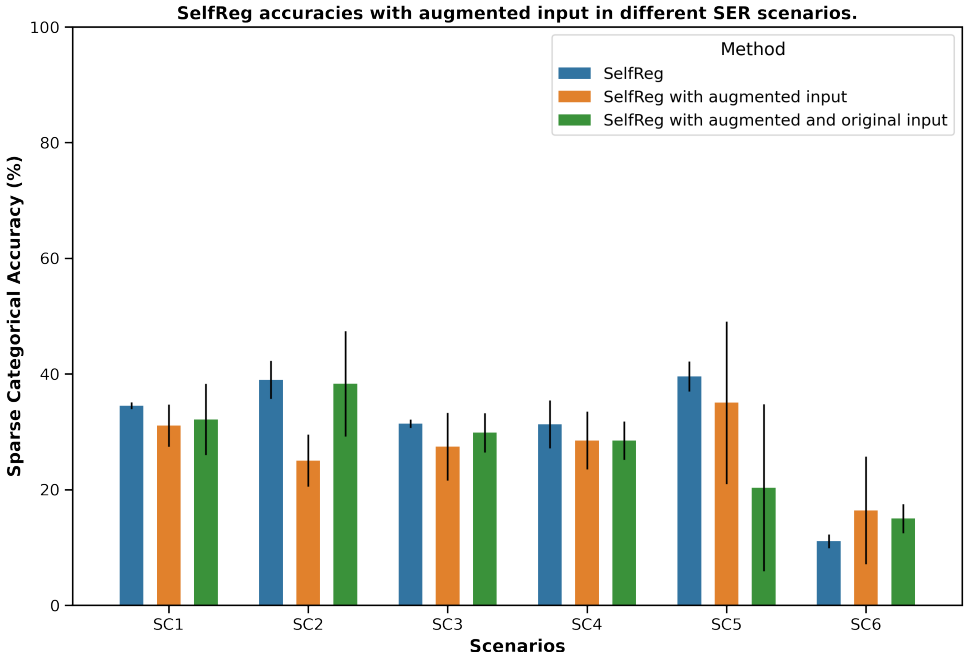


Figure 5.12: Sparse categorical accuracies of SelfReg for different scenarios with augmented data.

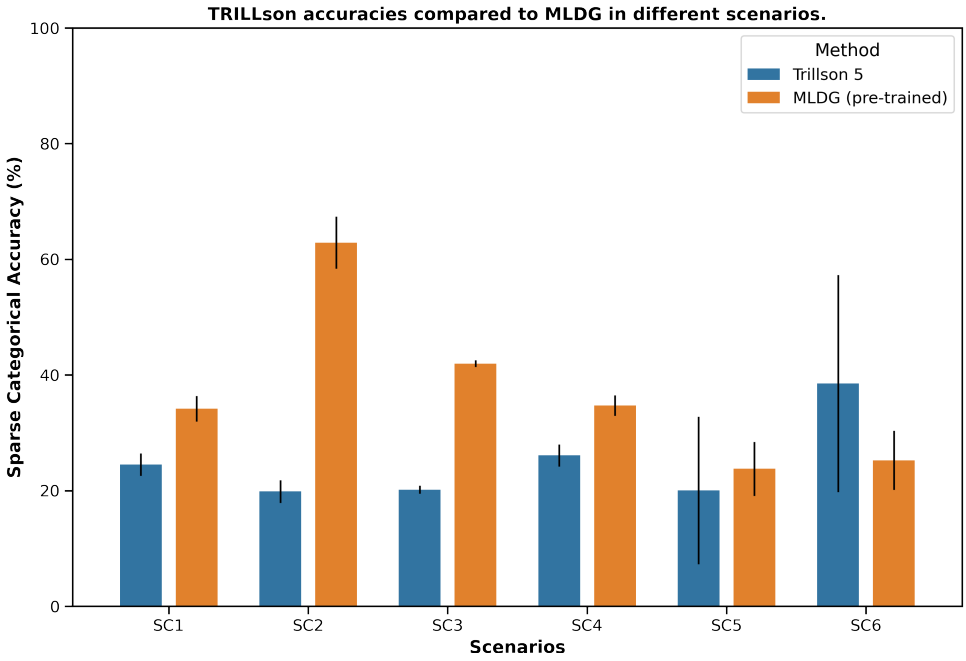


Figure 5.13: Accuracy of MLDG using TRILLson's [77] embeddings as input to perform cross-corpus DG for all scenarios.

provided in Table 5.14 and the results are shown in Figure 5.13.

5.8.2 Observations on varying inputs

From the presented results, it can be seen that augmentations show improvement in most Scenarios (i.e., 4 scenarios out of 6) for MLDG. It shows good improvement in the last 2 scenarios which use

ShEMO dataset as the test set. This could be due to the augmented speech emotion data from train domain showing some similarity to the Persian speech emotion data. The lower accuracy of Scenario 2 could be due to the erasure of common features due to augmentation. The results of these experiments show that it is necessary to determine the effects of augmentation on common invariant features before applying them on speech emotion data.

Another point to be noted is that the use of augmented data and the use of both augmented and original data do not seem to have much difference in accuracy in all scenarios in MLDG. However, SelfReg is more sensitive to augmented data as it depends on augmentations to provide an increase in the number of positive pairs for alignment. This leads to a wider variation in SelfReg's results in some scenarios.

The final point to note is that even though the TRILLson model does not perform well in scenarios 1 to 5, it performs much better in Scenario 6 compared to MLDG and SelfReg results from Table A.2 and A.3, even though the model did not converge very well. This could indicate the presence of common embeddings between the train and test domain extracted by TRILLson.

Therefore, it can be seen that using augmentation and embeddings from an extensively trained model have good potential to improve accuracy scores, as seen by the significant improvement in some scenarios, even though they may seem irregular.

Chapter 6

Conclusions

Domain generalization in the field of audio, especially emotion recognition, is still under development. It is an area undergoing heavy research to create a model whose performance is good enough to be practical. The main goal of this thesis is to find out whether meta-learning and contrastive learning can help improve performance of emotion recognition when performing domain generalization in the field of audio. This goal was divided into 4 sub-research questions outlined in Chapter 1. To answer these questions, one meta-learning technique (i.e., MLDG) and one contrastive learning technique (i.e., SelfReg) were selected, and extensive experiments were performed. The meta-learning algorithm provided an improved accuracy over the baseline in most scenarios. However, These accuracies were not high enough to be practical. In the case of SelfReg, the model showed improvement in far less scenarios (only two scenarios). However, this does not mean that contrastive learning is worse than meta-learning when it comes to domain generalization in audio. SelfReg is a model sensitive to augmented data and is dependent on common invariant features that could be learnt. Unlike MLDG, which teaches the backbone network how to learn by simulating multiple test scenarios using a part of the train domain, SelfReg depends on the presence of common features in positive pairs of data. MLDG learns from meta-data, while SelfReg learns directly from data and has multiple objectives that tries to control the features which could be learnt. However, if most of the features of the training set are not in common with those of the test set, or if there is not much common invariant features amongst the input datasets to be learnt, then the SelfReg model is affected drastically. The reason for low accuracy of both models can be seen by performing experiments by varying the learning strategy and the backbone network, using the state of the art domain generalization techniques and CNN networks.

In performing experiments using RSC learning technique, although one could observe a good increase in accuracy, the performance was still not good enough. Using a different backbone network did not seem to solve the problem either. There also seems to be an improvement while performing augmentation, as augmentation bridges the difference between languages to a certain extent. One may conclude from these results that performance can be improved by using a better technique that can recognize and distinguish common invariant features of the domain, or use a different form of data representation other than mel-spectrogram as input, such as embeddings from a heavily pre-trained model that represent most invariant features. Improvement can also be obtained on simulating different testing scenarios similar to MLDG.

6.1 Future work

In this thesis, it was seen that using algorithms that learn domain invariant features by testing models on multiple domains to simulate domain shifts (such as MLDG), and control the learning process of dominant features that are learnt by models (such as RSC) show good improvement over baseline accuracies. Using augmentations or different inputs such as embeddings from pre-trained models could also improve accuracy scores while performing cross-corpus DG. Cross-corpus

Domain generalization using Generative Adversarial Network (GANs) such as one proposed in [74], or using embeddings from models such as TRILLson could be further developed to address the current shortcomings of domain generalization in SER. This technology could be useful in many fields and applications, such as in the field of medical and psychological healthcare, customer care and home automation.

An exciting path of future work would be to develop a proper input for speech emotions with plentiful distinguishable representations to be learned, to perform DG for SER. Right now, there exist models, which have been trained on millions of hours of audio data such as CAP12 [77]. A distilled version of CAP12, the so called TRILLson, has been shown to create embeddings that are robust to domain shifts in terms of speech audio, on in its paper [77]. If similar models could be trained on speech emotion or related datasets, it could help create robust embeddings that could perform domain generalization on SER extremely well. However, before that, it is necessary to compile huge amounts of sound emotion data for such an idea to become reality.

Bibliography

[1] Quintero, Olga & Bustamante, Paola & López, Natalia & Pérez, Maeria. (2015). Recognition and regionalization of emotions in the arousal-valence plane. 2015. 10.1109/EMBC.2015.7319769.

[2] L. R. Rabiner and R. W. Schafer. “Introduction to Digital SpeechProcessing”. Foundations and Trends in Signal Processing Vol. 1,Nos. 1–2, 2007

[3] Sahidullah, Md.; Saha, Goutam (May 2012). ”Design, analysis and experimental evaluation of block based transformation in MFCC computation for speaker recognition”. Speech Communication. 54 (4): 543–565. doi:10.1016/j.specom.2011.11.004

[4] H. Purwins, B. Li, T. Virtanen, J. Schlüter, S. Chang and T. Sainath, ”Deep Learning for Audio Signal Processing,” in IEEE Journal of Selected Topics in Signal Processing, vol. 13, no. 2, pp. 206-219, May 2019, doi: 10.1109/JSTSP.2019.2908700.

[5] Mitrovic, Dalibor & Zeppelzauer, Matthias & Breiteneder, Christian. (2010). Features for Content-Based Audio Retrieval.. Advances in Computers. 78. 71-150.

[6] Eyben, Florian & Wöllmer, Martin & Schuller, Björn. (2010). openSMILE – The Munich Versatile and Fast Open-Source Audio Feature Extractor. MM’10 - Proceedings of the ACM Multimedia 2010 International Conference. 1459-1462. 10.1145/1873951.1874246.

[7] J. -H. Hsu, M. -H. Su, C. -H. Wu and Y. -H. Chen, ”Speech Emotion Recognition Considering Nonverbal Vocalization in Affective Conversations,” in IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 29, pp. 1675-1686, 2021, doi: 10.1109/TASLP.2021.3076364.

[8] Y. Ü. Sonmez and A. Varol, ”New Trends in Speech Emotion Recognition,” 2019 7th International Symposium on Digital Forensics and Security (ISDFS), 2019, pp. 1-7, doi: 10.1109/ISDFS.2019.8757528.

[9] Busso, C., M. Bulut, Chi-Chun Lee, Ebrahim Kazemzadeh, E. Provost, Samuel Kim, J. N. Chang, Sungbok Lee and Shrikanth S. Narayanan. “IEMOCAP: interactive emotional dyadic motion capture database.” Language Resources and Evaluation 42 (2008): 335-359.

[10] Burkhardt, Felix & Paeschke, Astrid & Rolfes, M. & Sendlmeier, Walter & Weiss, Benjamin. (2005). A database of German emotional speech. 9th European Conference on Speech Communication and Technology. 5. 1517-1520. 10.21437/Interspeech.2005-446.

[11] Livingstone SR, Russo FA (2018) The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English. PLoS ONE 13(5): e0196391. https://doi.org/10.1371/journal.pone.0196391.

[12] J. A. Russell. A circumplex model of affect. Journal of personality and social psychology, 39(6), 1161, 1980.

[13] I. S. Hong, Y. J. Ko, H. S. Shin and and Y. J. Kim, "Emotion Recognition from Korean Language using MFCC, HMM, and Speech Speed", The 12th International Conference on Multimedia Information Technology and Applications(MITA2016), pp.12-15, 2016

[14] Nogueiras, Albino, Asunción Moreno, Antonio Bonafonte, and José B. Mariño. "Speech emotion recognition using hidden Markov models." In Seventh European conference on speech communication and technology. 2001.

[15] A. Khan and U. K. Roy, "Emotion recognition using prosodie and spectral features of speech and Naïve Bayes Classifier," 2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), 2017, pp. 1017-1021, doi: 10.1109/WiSPNET.2017.8299916.

[16] R. A. A., M. Nasrun and C. Setianingsih, "Human Emotion Detection with Speech Recognition Using Mel-frequency Cepstral Coefficient and Support Vector Machine," 2021 International Conference on Artificial Intelligence and Mechatronics Systems (AIMS), 2021, pp. 1-6, doi: 10.1109/AIMS52415.2021.9466077.

[17] L. Zheng, Q. Li, H. Ban and S. Liu, "Speech emotion recognition based on convolution neural network combined with random forest," 2018 Chinese Control And Decision Conference (CCDC), 2018, pp. 4143-4147, doi: 10.1109/CCDC.2018.8407844

[18] UmaMaheswari, J. and A. Akila. "An Enhanced Human Speech Emotion Recognition Using Hybrid of PRNN and KNN." 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon) (2019): 177-183.

[19] Zhao, Jianfeng et al. "Speech emotion recognition using deep 1D & 2D CNN LSTM networks." Biomed. Signal Process. Control. 47 (2019): 312-323.

[20] K. Noh, J. Lim, S. Chung, G. Kim and H. Jeong, "Ensemble Classifier based on Decision-Fusion of Multiple Models for Speech Emotion Recognition," 2018 International Conference on Information and Communication Technology Convergence (ICTC), 2018, pp. 1246-1248, doi: 10.1109/ICTC.2018.8539502.

[21] Dias Issa, M. Fatih Demirci, Adnan Yazici, Speech emotion recognition with deep convolutional neural networks, Biomedical Signal Processing and Control, Volume 59, 2020, 101894, ISSN 1746-8094, https://doi.org/10.1016/j.bspc.2020.101894.

[22] J. -H. Hsu, M. -H. Su, C. -H. Wu and Y. -H. Chen, "Speech Emotion Recognition Considering Nonverbal Vocalization in Affective Conversations," in IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 29, pp. 1675-1686, 2021, doi: 10.1109/TASLP.2021.3076364.

[23] C. Zhang and L. Xue, "Two-stream Emotion-embedded Autoencoder for Speech Emotion Recognition," 2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), 2021, pp. 1-6, doi: 10.1109/IEMTRONICS52119.2021.9422602

[24] Ando, A., Mori, T., Kobashikawa, S., & Toda, T. (2021). Speech emotion recognition based on listener-dependent emotion perception models. APSIPA Transactions on Signal and Information Processing, 10, E6. doi:10.1017/ATSIP.2021.7

[25] Li, Y., Tao, J., Chao, L. et al. CHEAVD: a Chinese natural emotional audio–visual database. J Ambient Intell Human Comput 8, 913–924 (2017). https://doi.org/10.1007/s12652-016-0406-z

[26] Chou, Huang-Cheng & Lin, Wei-Cheng & Chang, Lien-Chiang & Li, Chyi-Chang & Ma, Hsi-Pin & Lee, Chi-Chun. (2017). NNIME: The NTHU-NTUA Chinese interactive multimodal emotion corpus. 292-298. 10.1109/ACII.2017.8273615.

[27] Emilia Parada-Cabaleiro, Giovanni Costantini, Anton Batliner, Alice Baird, & Bjoern Schuller. (2018). EmoFilm - A multilingual emotional speech corpus [Data set]. Interspeech, Hyderabad, India. Zenodo. https://doi.org/10.5281/zenodo.1326428

[28] Finn, Chelsea, P. Abbeel and Sergey Levine. "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks." ICML (2017).

[29] Li, Da, Yongxin Yang, Yi-Zhe Song and Timothy M. Hospedales. "Learning to Generalize: Meta-Learning for Domain Generalization." ArXiv abs/1710.03463 (2018)

[30] Naman, Anugunj & Mancini, Liliana. (2021). Fixed-MAML for Few Shot Classification in Multilingual Speech Emotion Recognition.arXiv:2101.01356 [cs.SD]

[31] S. Chopra, P. Mathur, R. Sawhney and R. R. Shah, "Meta-Learning for Low-Resource Speech Emotion Recognition," ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2021, pp. 6259-6263, doi: 10.1109/ICASSP39728.2021.9414373.

[32] Y. Ü. Sonmez and A. Varol, "New Trends in Speech Emotion Recognition," 2019 7th International Symposium on Digital Forensics and Security (ISDFS), 2019, pp. 1-7, doi: 10.1109/ISDFS.2019.8757528.

[33] Popova, Anastasiya & Rassadin, Alexandr & Ponomarenko, Alexander. (2018). Emotion Recognition in Sound. Studies in Computational Intelligence. 736. 117-124. 10.1007/978-3-319-66604-4_18.

[34] Chen, Ting, Simon Kornblith, Mohammad Norouzi and Geoffrey E. Hinton. "A Simple Framework for Contrastive Learning of Visual Representations." ArXiv abs/2002.05709 (2020)

[35] Lian, Zheng & Li, Ya & Tao, Jianhua & Huang, Jian. (2018). Speech Emotion Recognition via Contrastive Loss under Siamese Networks. 21-26. 10.1145/3267935.3267946.

[36] Wang, L., & Oord, A.V. (2021). Multi-Format Contrastive Learning of Audio Representations. ArXiv, abs/2103.06508.

[37] Ververidis, Dimitrios and Constantine Kotropoulos. "Emotional speech recognition: Resources, features, and methods." Speech Commun. 48 (2006): 1162-1181

[38] Vondra M., Vích R. (2009) Recognition of Emotions in German Speech Using Gaussian Mixture Models. In: Esposito A., Hussain A., Marinaro M., Martone R. (eds) Multimodal Signals: Cognitive and Algorithmic Issues. Lecture Notes in Computer Science, vol 5398. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-00525-1_26

[39] M. Li et al., "Contrastive Unsupervised Learning for Speech Emotion Recognition," ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2021, pp. 6329-6333, doi: 10.1109/ICASSP39728.2021.9413910.

[40] M. K. Pichora and Kate Dupuis, "Toronto emotional speech set (TESS)," 2020, Scholars Portal Dataverse.

[41] Jackson, Philip & ul haq, Sana. (2011). Surrey Audio-Visual Expressed Emotion (SAVEE) database.

[42] Costantini, Giovanni & Iaderola, Iacopo & Paoloni,& Todisco, Massimiliano. (2014). EMOVO Corpus: an Italian Emotional Speech Database.

[43] Mohamad Nezami, Omid & Lou, Paria & Karami, Mansooreh. (2019). ShEMO: a large-scale validated database for Persian speech emotion detection. Language Resources and Evaluation. 53. 10.1007/s10579-018-9427-x.

[44] S. Latif, A. Qayyum, M. Usman and J. Qadir, "Cross Lingual Speech Emotion Recognition: Urdu vs. Western Languages," 2018 International Conference on Frontiers of Information Technology (FIT), 2018, pp. 88-93, doi: 10.1109/FIT.2018.00023.

[45] Shi-wook Lee, "The generalization effect for multilingual speech emotion recognition across heterogeneous languages," in ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2019, pp. 5881–5885.

[46] Shivali Goel and Homayoon Beigi, "Cross lingual cross corpus speech emotion recognition," arXiv preprint arXiv:2003.07996, 2020.

[47] I. Shahin, A. B. Nassif and S. Hamsa, "Emotion Recognition Using Hybrid Gaussian Mixture Model and Deep Neural Network," in IEEE Access, vol. 7, pp. 26777-26787, 2019, doi: 10.1109/ACCESS.2019.2901352.

[48] Kim, D., Park, S., Kim, J., & Lee, J. (2021). SelfReg: Self-supervised Contrastive Regularization for Domain Generalization. ArXiv, abs/2104.09841.

[49] S. Suganya and E. Y. A. Charles, "Speech Emotion Recognition Using Deep Learning on audio recordings," 2019 19th International Conference on Advances in ICT for Emerging Regions (ICTer), 2019, pp. 1-6, doi: 10.1109/ICTer48817.2019.9023737.

[50] F. Abri, L. F. Gutiérrez, A. Siami Namin, D. R. W. Sears and K. S. Jones, "Predicting Emotions Perceived from Sounds," 2020 IEEE International Conference on Big Data (Big Data), 2020, pp. 2057-2064, doi: 10.1109/BigData50022.2020.9377842.

[51] S. Deb and S. Dandapat, "Emotion Classification Using Segmentation of Vowel-Like and Non-Vowel-Like Regions," in IEEE Transactions on Affective Computing, vol. 10, no. 3, pp. 360-373, 1 July-Sept. 2019, doi: 10.1109/TAFFC.2017.2730187.

[52] R. Rajak and R. Mall, "Emotion recognition from audio, dimensional and discrete categorization using CNNs," TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON), 2019, pp. 301-305, doi: 10.1109/TENCON.2019.8929459.

[53] J. Wang and Z. Han, "Research on Speech Emotion Recognition Technology based on Deep and Shallow Neural Network," 2019 Chinese Control Conference (CCC), 2019, pp. 3555-3558, doi: 10.23919/ChiCC.2019.8866568.

[54] T. Seehapoch and S. Wongthanavasu, "Speech emotion recognition using Support Vector Machines," 2013 5th International Conference on Knowledge and Smart Technology (KST), 2013, pp. 86-91, doi: 10.1109/KST.2013.6512793.

[55] Dou, Qi & Coelho de Castro, Daniel & Kamnitsas, Konstantinos& Glocker, Ben. (2019). Domain Generalization via ModelAgnostic Learning of Semantic Features.

[56] Zhou, Pan & Feng, Jiashi & Ma, Chao & Xiong, Caiming & HOI, Steven & Ee, Weinan. (2020). Towards Theoretically Understanding Why SGD Generalizes Better Than ADAM in Deep Learning.

[57] Y. Susanto, A. G. Livingstone, B. C. Ng and E. Cambria, "The Hourglass Model Revisited," in IEEE Intelligent Systems, vol. 35, no. 5, pp. 96-102, 1 Sept.-Oct. 2020, doi: 10.1109/MIS.2020.2992799.

[58] Wang, Jindong & Lan, Cuiling & Liu, Chang & Ouyang, Yidong & Qin, Tao. (2021). Generalizing to Unseen Domains: A Survey on Domain Generalization. 4627-4635. 10.24963/ijcai.2021/628.

[59] Gournay, Philippe & Lahaie, Olivier & Lefebvre, Roch. (2018). A Canadian french emotional speech dataset. 399-402. 10.1145/3204949.3208121.

[60] Vryzas, N., Kotsakis, R., Liatsou, A., Dimoulas, C. A., & Kalliris, G. (2018). Speech emotion recognition for performance interaction. Journal of the Audio Engineering Society, 66(6), 457-467.

[61] H. Cao, D. G. Cooper, M. K. Keutmann, R. C. Gur, A. Nenkova and R. Verma, "CREMA-D: Crowd-Sourced Emotional Multimodal Actors Dataset," in IEEE Transactions on Affective Computing, vol. 5, no. 4, pp. 377-390, 1 Oct.-Dec. 2014, doi: 10.1109/TAFFC.2014.2336244.

[62] Zhou, Kaiyang & Liu, Ziwei & Qiao, Yu & Xiang, Tao & Loy, Chen Change. (2021). Domain Generalization: A Survey. arXiv:2103.02503

[63] X. Yue, Y. Zhang, S. Zhao, A. Sangiovanni-Vincentelli, K. Keutzer, and B. Gong, "Domain randomization and pyramid consistency: Simulation-to-real generalization without accessing target domain data," in ICCV, 2019, pp. 2100–2110.

[64] Z. Huang, H. Wang, E. P. Xing, and D. Huang, "Self-challenging improves cross-domain generalization," in ECCV, vol. 2, 2020.

[65] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in ICML, 2015, pp. 1180–1189.

[66] Grill, Jean-Bastien, et al. "Bootstrap your own latent: A new approach to self-supervised learning." arXiv preprint arXiv:2006.07733 (2020).

[67] Banse, R., & Scherer, K. R. (1996). Acoustic profiles in vocal emotion expression. Journal of Personality and Social Psychology, 70(3), 614–636. https://doi.org/10.1037/0022-3514.70.3.614

[68] Hendrycks, Dan & Lee, Kimin & Mazeika, Mantas. (2019). Using Pre-Training Can Improve Model Robustness and Uncertainty.

[69] Huang, Zeyi & Wang, Haohan & Xing, Eric & Huang, Dong. (2020). Self-challenging Improves Cross-Domain Generalization. 10.1007/978-3-030-58536-5_8.

[70] Yun, Sangdoo & Han, Dongyoon & Chun, Sanghyuk & Oh, Seong Joon & Yoo, Youngjoon & Choe, Junsuk. (2019). CutMix: Regularization Strategy to Train Strong Classifiers With Localizable Features. 6022-6031. 10.1109/ICCV.2019.00612.

[71] Lin, Tsung-Yi & Goyal, Priya & Girshick, Ross & He, Kaiming & Dollar, Piotr. (2017). Focal Loss for Dense Object Detection. 2999-3007. 10.1109/ICCV.2017.324.

[72] Braunschweiler, Norbert, Rama Doddipatla, Simon Keizer and Svetlana Stoyanchev. "A Study on Cross-Corpus Speech Emotion Recognition and Data Augmentation." 2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU) (2021): 24-30.

[73] Lee, Shi-wook. "Domain Generalization with Triplet Network for Cross-Corpus Speech Emotion Recognition." 2021 IEEE Spoken Language Technology Workshop (SLT) (2021): 389-396.

[74] Gideon, John, Melvin G. McInnis and Emily Mower Provost. "Improving Cross-Corpus Speech Emotion Recognition with Adversarial Discriminative Domain Generalization (ADDoG)." IEEE Transactions on Affective Computing 12 (2021): 1055-1068.

[75] Iver Jordal, Araik Tamazian, Emmanouil Theofanis Chourdakis, Céline Angonin, askskro, Nikolay Karpov, Omer Sarioglu, kvilouras, Enis Berk Çoban, Florian Mirus, Jeong-Yoon Lee, Kwanghee Choi, MarvinLvn, SolomidHero, & Tanel Alumäe. (2022). iver56/audiomentations: v0.25.0 (v0.25.0). Zenodo. https://doi.org/10.5281/zenodo.6594177

[76] Tomar, S. (2006). Converting video formats with FFmpeg. Linux Journal, 2006(146), 10.

[77] Shor, Joel & Venugopalan, Subhashini. (2022). TRILLsson: Distilled Universal Paralinguistic Speech Representations.

[78] Yu, Samuel & Wu, Peter & Liang, Paul & Salakhutdinov, Ruslan & Morency, Louis-Philippe. (2022). PACS: A Dataset for Physical Audiovisual CommonSense Reasoning.

[79] Zhang, Hongyi & Cisse, Moustapha & Dauphin, Yann & Lopez-Paz, David. (2017). Mixup: Beyond Empirical Risk Minimization.

[80] A. Nagrani, J. S. Chung, A. Zisserman VoxCeleb: a large-scale speaker identification dataset, INTERSPEECH, 2017.

[81] Lugovic, Sergej & Dunđer, Ivan & Horvat, Marko. (2016). Techniques and Applications of Emotion Recognition in Speech. 10.1109/MIPRO.2016.7522336.

[82] GitHub repository by Xiao, Richard. TensorFlow-ResNets. https://github.com/RichardXiao13/TensorFlow-ResNets/releases/download/v0.3.0/resnet18_imagenet_notop.h5

[83] Scheidwasser-Clow, Neil & Kegler, Mikolaj & Beckmann, Pierre & Cerňak, Miloš. (2021). SERAB: A multi-lingual benchmark for speech emotion recognition.

[84] Ahmed, Md & Islam, Salekul & D, Ph & Islam, A.K.M. & Shatabda, Swakkhar. (2021). An Ensemble 1D-CNN-LSTM-GRU Model with Data Augmentation for Speech Emotion Recognition.

[85] Lydia, Agnes & Francis, Sagayaraj. (2019). Adagrad - An Optimizer for Stochastic Gradient Descent. Volume 6. 566-568.

[86] Fang, Chen & Xu, Ye & Rockmore, Daniel. (2013). Unbiased Metric Learning: On the Utilization of Multiple Datasets and Web Images for Softening Bias. 1657-1664. 10.1109/ICCV.2013.208.

[87] Ramakrishnan, Raghavendran & Nagabandi, Bhadrinath & Eusebio, Jose & Chakraborty, Shayok & Venkateswara, Hemanth & Panchanathan, Sethuraman. (2020). Deep Hashing Network for Unsupervised Domain Adaptation. 10.1007/978-3-030-45529-3_4.

[88] Haohan Wang, Songwei Ge, Eric P. Xing, and Zachary C. Lipton. "Learning Robust Global Representations by Penalizing Local Predictive Power

[89] Zhang, Yu, Daniel S. Park, Wei Han, James Qin, Anmol Gulati, Joel Shor, Aren Jansen, Yuanzhong Xu, Yanping Huang, Shibo Wang, Zongwei Zhou, Bo Li, Min Ma, William Chan, Jiahui Yu, Yongqiang Wang, Liangliang Cao, Khe Chai Sim, Bhuvana Ramabhadran, Tara N. Sainath, Franccoise Beaufays, Zhifeng Chen, Quoc V. Le, Chung-Cheng Chiu, Ruoming Pang and Yonghui Wu. "BigSSL: Exploring the Frontier of Large-Scale Semi-Supervised Learning for Automatic Speech Recognition." ArXiv abs/2109.13226 (2022)

Appendices

Appendix A

Detailed results

	ResNet18	ResNet18 (pre-trained)
CREMA-D	51.01	54.65
TESS	98.55	99.75
SAVEE	25.33	35.33
RAVDESS	47.5	52.5
AESDD	35.62	38.52
CAFÉ	31.45	49.58
EmoDB	40.01	88.75
EMOVO	29.29	43.21
ShEMO	74.7	79.02

Table A.1: Detailed results of preliminary experiment involving ResNet18 on Individual emotion audio datasets.

	ResNet18 accuracy	ResNet18 accuracy (pre-trained)	MLDG accuracy	MLDG accuracy (pre-trained)
Scenario 1	27.51± 2.89	31.57±2.24	31.87± 3.33	34.17± 2.19
Scenario 2	33.26± 4.10	49.96± 8.17	52.78± 3.68	62.87± 4.50
Scenario 3	25.51± 0.89	41.69± 3.48	35.62± 1.25	41.95± 0.58
Scenario 4	26.92± 1.70	29.36± 3.34	32.06± 2.97	34.71± 1.79
Scenario 5	40.38± 4.20	21.22± 4.41	22.97± 4.14	23.77± 4.66
Scenario 6	20.83± 3.89	24.81± 6.15	17.78± 3.49	25.23± 5.11

Table A.2: Detailed MLDG Sparse categorical accuracy results

	ResNet18 accuracy	ResNet18 accuracy (pre-trained)	SelfReg accuracy	SelfReg accuracy (pre-trained)
Scenario 1	27.51± 2.89	31.57± 2.24	31.33± 1.47	30.81± 1.45
Scenario 2	33.26± 4.10	49.96± 8.17	43.45± 2.83	39.58± 4.87
Scenario 3	25.51± 0.89	41.69± 3.48	26.4± 2.12	41.14± 1.82
Scenario 4	26.92± 1.70	29.36± 3.34	28.55± 4.51	28.65± 1.67
Scenario 5	40.38± 4.20	21.22± 4.41	38.28± 1.99	26.42± 2.86
Scenario 6	20.83± 3.89	37.75± 6.15	12.34± 3.31	15.42± 7.65

Table A.4: Detailed updated SelfReg Sparse categorical accuracy results

	ResNet18 accuracy	ResNet18 accuracy (pre-trained)	SelfReg accuracy	SelfReg accuracy (Pre-trained)
Scenario 1	27.51± 2.89	31.57±2.24	34.51± 0.57	31.41±4.43
Scenario 2	33.26± 4.10	49.96± 8.17	38.98± 3.29	43.09±6.79
Scenario 3	25.51± 0.89	41.69± 3.48	31.38± 0.72	32.19±1.85
Scenario 4	26.92± 1.70	29.36± 3.34	31.3± 4.15	22.8±1.75
Scenario 5	40.38± 4.20	21.22± 4.41	39.55± 2.59	29.95±5.47
Scenario 6	20.83± 3.89	24.81± 6.15	11.07± 1.18	20.78±9.40

Table A.3: Detailed SelfReg Sparse categorical accuracy results

	SelfReg accuracy	SelfReg accuracy (augmented input)	SelfReg accuracy (augmented and original input)
Scenario 1	34.51± 0.57	31.05± 3.64	32.13± 6.15
Scenario 2	38.98± 3.29	25.02± 4.51	38.30± 9.09
Scenario 3	31.38± 0.72	27.44± 5.84	29.85± 3.40
Scenario 4	31.3± 4.15	28.50± 5.01	28.50± 3.31
Scenario 5	39.55± 2.59	35.02± 14.05	20.33± 14.45
Scenario 6	11.07± 1.18	16.40± 9.30	14.99± 2.53

Table A.5: Detailed SelfReg Sparse categorical accuracy results with augmented and/or normal input

	MLDG accuracy	MLDG accuracy (augmented input)	MLDG accuracy (augmented and original input)
Scenario 1	34.17±2.19	39.66± 0.94	39.68± 1.45
Scenario 2	62.87± 4.50	54.43± 8.15	52.65± 5.77
Scenario 3	41.95± 0.58	42.96± 0.84	44.91± 2.45
Scenario 4	34.71± 1.79	29.77± 3.80	30.34± 1.78
Scenario 5	23.77± 4.66	31.02± 5.36	28.54± 8.74
Scenario 6	25.23± 5.11	23.77± 8.20	30.49± 0.63

Table A.6: Detailed MLDG (pre-trained) Sparse categorical accuracy results with augmented and/or normal input

	ResNet18 accuracy	MLDG accuracy (ResNet18)	MLDG accuracy (ResNet50)
Scenario 1	27.51±2.89	31.87± 3.33	33.12±1.74
Scenario 2	33.26± 4.10	52.78± 3.68	47.12± 5.62
Scenario 3	25.51± 0.89	35.62± 1.25	31.62± 2.22
Scenario 4	26.92± 1.70	32.06± 2.97	32.77± 2.77
Scenario 5	40.38± 4.20	22.97± 4.14	25.06± 8.83
Scenario 6	20.83± 3.89	17.78± 3.49	15.34± 2.19

Table A.7: Detailed Results of MLDG with ResNet50 as backbone in comparison with other results

Scenarios	TRILLson accuracy using MLDG
Scenario 1	24.49±1.93
Scenario 2	19.84±1.96
Scenario 3	20.16±0.69
Scenario 4	26.08±1.92
Scenario 5	20.02±12.76
Scenario 6	38.53±18.77

Table A.8: Detailed Results of MLDG using TRILLSON's embeddings as input to perform DG in all scenarios