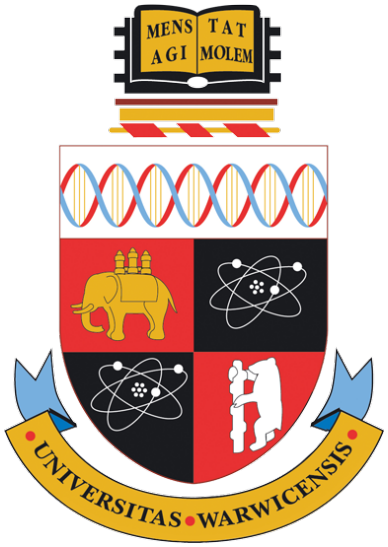
WARWICK
THE UNIVERSITY OF WARWICK



Graph Neural Network for Audio Representation Learning

by

Amir Shirian

A thesis submitted in partial fulfilment of the requirements for the degree of

Doctor of Philosophy in Computer Science

University of Warwick, Department of Computer Science

September 2022

Contents

List of Tables

List of Figures

Acknowledgments

I want to start by sincerely thanking my supervisors, Drs. Tanaya Guha and Victor Sanchez. Their patience and encouragement throughout my PhD studies made this thesis possible. They made my research experience so enjoyable and fulfilling by welcoming every novel idea with enthusiasm and an open mind, for which I am incredibly grateful.

I want to express my gratitude to my friends for making my experience at The University of Warwick so wonderful. I'm also grateful to my research colleagues, Dr Subarna Tripathi, Intel AI lab, and Dr Krishna Somandepalli, Google Research, for helping and guiding me with PhD projects.

Finally, I would like to thank my dear wife, Mona Ahmadian, and my parents, Fariba Nobakht and Ahmadreza Shirian. I owe my achievements to their love and support.

Declarations

1 Publications

I hereby declare that this dissertation is original work and has not been submitted for a degree or diploma, or other qualification at any other University. Parts of this thesis have been previously published by the author in the following:

[1] Amir Shirian and Tanaya Guha. Compact graph architecture for speech emotion recognition. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6284–6288, 2021.

[2] Amir Shirian, Subarna Tripathi, and Tanaya Guha. Dynamic emotion modeling with learnable graphs and graph inception network. *IEEE Transactions on Multimedia*, 2021.

[3] Amir Shirian, Krishna Somandepalli, and Tanaya Guha. Self-supervised graphs for audio representation learning with limited labeled data. *IEEE Journal of Selected Topics in Signal Processing*, pages 1–11, 2022.

[4] Amir Shirian, Krishna Somandepalli, Victor Sanchez, and Tanaya Guha. Visually-aware acoustic event detection using heterogeneous graphs. In *INTERSPEECH 2022*, pages 641–645. International Speech Communication Association, 2022.

2 Sponsorships and Grants

All research works contained within are funded by the department of computer science, University of Warwick.

Abstract

Learning audio representations is an important task with many potential applications. Whether it takes the shape of speech, music, or ambient sounds, audio is a common form of data that may communicate rich information. Audio representation learning is also a fundamental ingredient of deep learning. However, learning a good representation is a challenging task. Audio representation learning can also enable more accurate downstream tasks both in audio and video, such as emotion recognition. For audio representation learning, such a representation should contain the information needed to understand the input sound and make discriminative patterns. This necessitates a sizable volume of carefully annotated data, which requires a considerable amount of labour.

In this thesis, we propose a set of models for audio representation learning. We address the discriminative patterns by proposing graph structure and graph neural network to further process it. Our work is the first to consider the graph structure for audio data. In contrast to existing methods that use approximation, our first model proposes a manual graph structure and uses a graph convolution layer with accurate graph convolution operation. In the second model, By integrating a graph inception network, we expand the manually created graph structure and simultaneously learn it with the primary objective in our model. In the third model, we addressed the dearth of annotated data by including a semi-supervised graph technique that represents audio corpora as nodes in a graph and connects them depending on label information in smaller subgraphs. We brought up the issue of leveraging multimodal data to improve audio representation learning in addition to earlier works. To accommodate multimodal input data, we included heterogeneous graph data to our fourth model. Additionally, we created a new graph architecture to handle multimodal data.

Acronyms

AEC acoustic event classification.

AP average precision.

CE cross-entropy.

CNN convolution neural network.

DL deep learning.

GCN graph convolution network.

HGNN heterogeneous graph neural network.

HMM hidden Markov model.

KNN K-nearest neighbour.

L-GRIN Learnable graph Inception network.

LLD low-level descriptor.

LPCC linear prediction cepstral coefficient.

LSTM Long ShortTerm Memory.

mAP mean average precision.

MFCC Mel frequency Cepstral coefficient.

MI mutual information.

ML machine learning.

MLP multi-layer Perceptron.

MSE mean squared error.

RNN Recurrent neural network.

SER speech emotion recognition.

SOTA state-of-the-art.

SSL self-supervised learning.

SVM support vector machine.

Symbols

$\mathbf{x}$	The node feature
$\mathbf{w}$	The graph convolution kernel
$\mathbf{X}$	The graph node features
$\mathbf{A}$	The adjacency matrix
$\mathbf{D}$	The diagonal matrix
$\mathbf{L}$	The Laplacian matrix
$\mathcal{G}$	A graph
$\mathcal{V}$	The set of nodes
$\mathcal{E}$	The set of edges
y	A label
$\mathbf{u}$	An eigen vector
$H^{(k+1)}$	The graph layer's output
$e_{(ij)}$	The graph edge connecting node i to j
$\mathbf{A}_{ij}$	The adjacency element corresponding to $e_{(ij)}$
$\|\cdot\|_F$	Frobenious norm
$\mathbf{A} \odot \mathbf{B}$	An element-wise multiplication between two $\mathbf{A}$ and $\mathbf{B}$ matrices
$\|\cdot\|_2{}^2$	The L2 norm
$\mathcal{L}$	The loss function
$\|\mathcal{V}\|$	The number of nodes in a graph
$\mathcal{V}_l$	The labeled nodes
$\mathcal{V}_u$	The unlabeled nodes
Θ	The learnable parameters
O	The set of node types
R	The set of edge types
$\mathcal{E}_{vv}$	The set of edges between video-only nodes
$\mathcal{E}_{aa}$	The set of edges between audio-only nodes
$\mathcal{E}_{va}$	The set of edges between audio-video nodes
$\mathbf{x}_K^a$	The audio node features in layer K
$\mathbf{x}_K^v$	The video node features in layer K
$\mathbf{h}_G$	The graph-level representation

Chapter 1

Introduction

1.1 Motivation

Audio is a ubiquitous form of data that can convey rich information, whether in the form of speech, music, or surrounding environment sounds. We are always surrounded by dynamic auditory events, some of which are nice, such as singing birds, human narrating a well-articulated story, and babbling brooks, and others which are less pleasant, such as the sound of a lawnmower on a Saturday morning! Humans have the ability to recognise, filter, and understand a wide range of existing sounds from an early age, allowing them to focus on crucial features in the audio while filtering out a vast number of distractions. Machines must be able to analyse and understand the auditory environment with great precision in the era of artificial intelligence (AI).

Representation learning is a fundamental ingredient of deep learning. However, learning a good representation is a challenging task. For audio representation learning, such a representation should contain the information needed to understand the input sound and make discriminative patterns. Two crucial components of a good representation are robustness and interpretability. A robust representation should also be reusable in fine-tuning tasks as well. Hence it should capture the structure of the data. Interpretability is another desired characteristic which sheds light on the underlying mechanisms in deep networks.

Audio representation learning has been used in a wide variety of applications and still gaining popularity, such as remote health monitoring [5], self-driving cars [6], acoustic surveillance [7], investigating the dynamics of animal behaviour [8], as well as the use of automatic speech recognition (ASR) and emotion recognition systems for human-computer interaction [9–11].

Even though audio representation learning has been the focus of considerable research, there are still problems. Following the developments in computer vision (CV) and natural language processing (NLP), many studies are applying

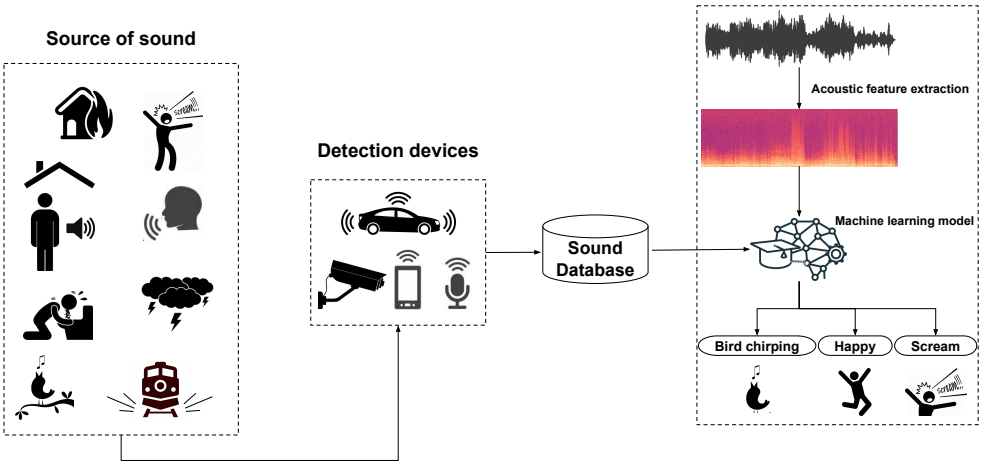


Figure 1.1: There are various sounds all around us every day. Many tools pick them up and let us keep and use them later. We can then use the stored audio database and ML systems to process and classify the input audio.

those techniques to the learning of audio representations without any modification that causes conceptual mistakes. For instance, a lot of research on audio data has been conducted without considering the key distinction between audio and other types of data by applying convolution neural networks (CNN), a very well-liked and potent method for CV, directly to raw or preprocessed audio data (Spectrogram).

Audio is in fact temporal data, and then learning the statistical regularities of environmental events is a powerful tool for enhancing performance. In addition, expectations about the temporal occurrence of events (when) are often tied with the expectations about certain event-related properties (what and where) happening at these time steps. While many works have been ignored, audio is a temporal data type whose temporality must be taken into account both during preprocessing and while creating the architecture for processing. However, it remains unclear whether this often implicit type of modelling can be proactively enhanced by explicit knowledge about temporal regularities. To further explore this hypothesis, in this work, we offer graph modelling of audio data to enable explicit modelling of temporal occurrence of events. In addition, we strive to learn an appropriate deep representation for audio recognition using graph neural networks (GNNs) that addresses the aforementioned issues and apply them to two speech emotion recognition (SER) and acoustic event classification tasks (AEC).

Concurrent with the progress of AI, the deployment of machine learning solutions on mobile (edge) devices for consumer applications is gaining more and more attention [12–14]. Thus, ML systems with stable performance and great accuracy in real-world situations are required for all of these applications. However, the limited storage and computation resources provided by these

platforms are an obstacle that is being addressed by many researchers working to reduce their complexity [15–17]. In this thesis, our proposed graph-based model is a compact, efficient and scalable way to represent data results in a lightweight graph-based network with a much fewer number of parameters enabling them to perform swiftly on edge devices without losing performance.

Finally, audio perception by humans is inherently multimodal in nature. It involves processing both aural and visual cues. Visual cues are important not only for audio source localization [18], but also for improving audio perception [19]. Perceptual studies have also revealed that visual cues can even change how sound is heard [20]. In this thesis, motivated by the success of graph-based methods in multimodal problems, we propose a heterogeneous graph-based approach to learn visually-aware audio representations.

1.2 Challenges

Despite recent significant advancements in audio representation learning, a number of technological obstacles remain. We identify the following main challenges as follows:

Audio feature extraction. Despite recent substantial advances in the field of ML, existing computer audition systems are still unable to do audio interpretation and analysis with the precision of a human in situations when humans are capable of doing so. Furthermore, traditional ML approaches were limited in their ability to process the raw input data. For many years, developing ML systems required precise engineering and fundamental domain knowledge to develop a feature extractor (e.g., OPENSMILE [8, 9]) that mapped raw audio characteristics like amplitude and frequency into meaningful feature vectors from which the machine learner could recognise patterns. Due to the huge amount of task-specific consideration, this manual feature engineering method is arduous and time-consuming.

Such features are also task-dependent, requiring human expertise to pick and design discriminative information for a specific task [10]. Task-specific feature sets, on the other hand, frequently outperform more general feature vectors in terms of performance. Because these features are fine-tuned for a specific domain, such as acoustic features for affective computing, emotion classification [11–13], and music genre classification [14], they may not perform well for tasks that differ slightly, causing the ML system's performance to suffer [15].

Lack of explicit audio temporal modelling. Although attention to audio representation learning has grown, the field suffers from a lack of looking into data insights. Semantics can be delivered from the whole audio sample, but the details are much more important for fine-grained analysis. For each

segment in audio, the appropriate features are extracted and will be used to analyse each frame separately. In addition, semantic relations between frames are important as each frame's feature. Then finding and discovering these relations are a tedious and labour-intensive process. Some earlier methods used dynamic modelling approaches to address the lack of relations in a single audio clip. But dynamic processes prompted greater problems which mostly resulted in much more computation and slightly better results.

Lack of large labelled datasets In audio representation learning, the lack of large annotated datasets and a wide diversity of audio applications impedes many developments. The availability of large high-quality labelled datasets has the power to push forward a research area. For example, the ImageNet dataset played a vital role in the breakthrough of Convolutional Neural Networks (CNNs) [21] for image classification.

Audio modality may provide incomplete information. Although one modality conveys a meaning not borne by the other modalities, but the same meaning is conveyed at the same time via several modalities which might be missed in the target modality in case of noise or other problems related to acquiring, storing, and reading data. In addition, the fusion of multiple sensors can facilitate the capture of complementary information or trends that may not be captured by individual modalities. Moreover, multiple sensors observing the same data can make more robust predictions because detecting changes in it may only be possible when multiple modalities are present.

1.3 Contributions

This thesis proposes novel graph-based deep models for representation learning with a new graph-based approach. Our approaches are more effective (modelling spatio-temporal details in the audio data), light (low number of parameters), and precise (achieved state-of-the-art performance) and even have the ability to process heterogeneous data. The main contributions of this thesis are as follows:

- **Audio as graph (Chapter 3)**
  - *Transforming raw audio to graphs.* For the first time in the audio community, we provide two ways for constructing graphs in order to represent the input audio data in a new domain and preserve both spatial and temporal information in a single data format.
  - *New graph convolution formulation.* Following the previous contribution, we propose a generic graph convolution network leveraging the concepts in the graph signal processing field.

- *Application to SER.* Speech emotion recognition is a long-lasting speech application that is a diverse and challenging task to try. After developing a graph-based solution to learn audio representation, we apply it to well-known speech emotion recognition datasets: IEMOCAP, MSP-IMPROV.

- **Graph structure learning (Chapter 4)**
  - *Dynamic graph structure learning.* Since the graph structure is not naturally defined, we propose to learn a (sub)optimal structure. Graph structure learning is trained jointly with the classification losses.
  - *Graph inception network.* To capture information in different graph depths, we propose a new graph architecture.
  - *Application to SER generates SOTA results.* The effectiveness of the proposed model is tested on different datasets in speech emotion recognition task.

- **Graph self-supervised learning (Chapter 5)**
  - *Subgraph-based auxiliary learning.* We develop a subgraph-based auxiliary learning framework for audio representation learning with limited labelled data.
  - *Graph self-supervision tasks.* We introduce a set of new graph tasks that can be applied to any type of graph.
  - *Application to SER and Acoustic event classification.* We demonstrate the superior performance of our proposed method for two tasks: acoustic event classification, and speech emotion classification on three large benchmark audio databases.

- **Graph multi-modal learning (Chapter 6)**
  - *Transforming audiovisual input to heteregeneous graphs.* We develop a graph construction method for converting multimodal data to a heterogeneous graph.
  - *Heterogeneous graph network.* We introduce a novel heterogeneous graph neural network that can capture modality-specific information as well as complementary information between modalities.

1.4 Thesis Outline

The rest of the thesis is organised as follows:

- **Chapter 2: Literature Survey**
  This chapter reviews the history and current directions in four key areas: audio representation learning, Graph neural networks, Self-supervised learning, and Multi-modal representation learning. In the audio representation learning section, the works in speech emotion recognition and acoustic event classification have been reviewed. In addition, graph literature is divided into two spectral and spatial subfields, and the related works are reviewed. As follows, self-supervised works in graph literature are surveyed. Finally, multi-modal representation learning approaches are explored in the audio-visual context.

- **Chapter 3: Audio as graph**
  In this chapter, we cast audio representation learning as a graph classification problem. We model a speech signal as a simple graph, where each node corresponds to a short windowed segment of the signal. Owing to this particular graph structure, we take advantage of certain results in graph signal processing to perform accurate graph convolution.

- **Chapter 4: Graph learning**
  This chapter makes the point that manually constructing a graph is not the best solution and that the underlying graph structure must be learned during the training phase. Then, in order to simultaneously capture information from various graph depths and the structure of the graph itself, we also create a novel graph architecture.

- **Chapter 5: Self-supervised graphs for audio representation learning**
  In this chapter, we propose a graph self-supervised learning approach to learn effective audio representations from a limited amount of labelled data. Considering each audio sample as a graph node, we propose a subgraph-based learning framework with new self-supervision tasks.

- **Chapter 6: Visually-aware graph-based audio representation learning**
  In this chapter, we propose a visually-aware audio representation learning approach based on heterogeneous graphs in the context of acoustic event classification. Our heterogeneous graph model creates a shared space for audio and visual modalities that takes advantage of their spatial and temporal relationships explicitly.

- **Chapter 7: Conclusion and future work**
  In the final chapter, we summarise the contributions of this thesis. We then discuss applications of our work and new research directions made possible by the presented contributions.

Chapter 2

Related Works

In this chapter, we survey works related to the contributions of this thesis. The chapter is organised as follows: Section 2.1 presents a taxonomy of tasks and methods for audio representation learning and summarises key works in the area. Section 2.2 gives an in-depth overview of graph neural networks. We review the history and current state-of-the-art solutions for the task. Section 2.3 summaries self-supervised learning in graphs. Finally, Section 2.4 surveys developments in methods that leverage other modalities to further boost audio representation learning.

2.1 Audio Representation Learning

Research on audio or speech processing has traditionally considered the task of designing hand-engineered acoustic features as a separate distinct problem from the task of designing efficient machine learning (ML) models to make prediction and classification decisions. This methodology has two primary drawbacks: first, feature engineering is manual, which is laborious and requires human expertise; second, the created features might not be the most appropriate for the objective at hand. This has motivated the adoption of a recent trend in the audio community towards the utilisation of representation learning techniques, which can learn an intermediate representation of the input signal automatically that better suits the task at hand and hence lead to improved performance. With advancements in deep learning (DL), where the representations are more advantageous and less reliant on human knowledge, it is now more important than ever to train representations for tasks like classification, prediction, etc.

Despite the fact that there is a large body of literature on this topic, we only survey two sub-fields: speech emotion recognition and acoustic event classification that we studied them in this thesis as applications.

2.1.1 Speech Emotion Recognition

Speech Emotion Recognition (SER) has been an essential part of Human-Computer Interaction (HCI) and other complex speech processing systems over the past ten years. An SER system typically identifies the presence of various emotions in the speaker by extracting and categorising the salient features from a preprocessed speech signal. However, there are significant quantitative and qualitative differences between how humans and machines recognise and correlate emotional aspects of speech signals, which makes it extremely difficult to combine knowledge from interdisciplinary fields, particularly speech emotion recognition, applied psychology, and human-computer interface.

There are many methods and algorithms used for the task of emotion recognition from speech. Each one of these methods is trying to solve the problem from a specific angle and has advantages and shortcomings. Many speech emotion recognition systems still rely on low-level acoustic, prosodic and lexical features that are then fed to deep models for classification. Several features like rhythm and intonation that human beings can recognize are known as prosodic features or para-linguistic features as these features manage the components of speech that are properties of massive units as in sentences, words, syllables, and expressions and sentences [22]. Prosodic characteristics are long-term features since they are extracted from massive units. These features are the ones passing on unique properties of emotional substance for speech emotion recognition. Commonly used prosodic qualities are based on energy, duration, and fundamental frequency properties. Mel Frequency Cepstral Coefficient is another of the most popular low-level acoustic features in automatic speech recognition (MFCC). The vocal tract's form is represented by the short-time power spectrum envelope, which is represented by MFCCs. Before translating the utterances into the frequency domain with a short-time discrete Fourier transform to produce MFCC, the utterances are divided into several segments. Numerous sub-band energies are calculated using the Mel filter bank. After that, the logarithm of respective sub-bands is computed. Lastly, MFCC is determined by applying the inverse Fourier transform [23].

SER can be carried out in two ways: (1) traditional models, and (2) deep learning models.

Traditional Methods. In earlier efforts to recognize emotions from the speech signal, almost all the implementations used hand-crafted features such as low-level descriptors (LLD), MFCCs, and linear prediction cepstral coefficients (LPCC). They were implementing frameworks based on machine learning algorithms that needed extensive feature engineering and a deep understanding of the subject matter to be able to infer the features helping the most to bring them into the calculations. Since the 1960s, Hidden Markov models

(HMM) have been used extensively in voice recognition. HMM is a statistical Markov model in which the system is assumed to be a Markov process with an unobserved state. They are designed to produce a flexible model that complies with the temporal characteristics of speech, classifying minute variations in an audio input. Naturally, HMMs were one of the first choices to try in SER [24]. Support Vector Machines (SVM), as one of the well-known methods, is used as a classifier for emotion recognition [25, 26]. The classifier is generally described for linearly separable patterns by splitting the hyperplane. SVM makes use of the kernel trick to model nonlinear decision boundaries. The SVM classifier aims to detect that hyperplane having a maximum margin between two classes' data points. The original data points are mapped to a new space if the given patterns are not linearly separable by utilizing a kernel function. K-nearest neighbour (KNN) [27], decision tree [28], naive Bayes classifier [29], artificial neural network (ANN) [27] (input, output, and only one hidden layer) are other methods that have been used in this literature.

Deep Learning Methods. The introduction of AlexNet by Krizhevsky et al. [21] in 2012, a multi-layer convolutional neural network trained on ImageNet in 2010 to recognise 1000 different classes and obtain great results, is one of the most well-known instances that sparked the widespread application of deep learning. Following that, numerous deep architectures emerged. A method based on a deep neural network with fully connected and convolutional pooling layers has been proposed by Harar et al. [30]. They have restricted their classes to angry, neutral, and sad compared to earlier research. They have eliminated silence from their signals in their system before segmenting the files into 20 ms segments with no overlap. They have six layers of convolution in their network before any feature selection, followed by dropout layers with p values of 0.1, a lattice of two parallel feature selectors, and then a sequence of fully connected layers. Zhang et al. [31] have developed an Emotion recognition system based on a deep convolutional neural network designed for the ImageNet LSVRC-2010 contest. This network, AlexNet, is also pretrained with a dataset of 1.2 million images, then fine-tuned using samples that they had from EMO-DB. Using this system, they can recognize three classes of emotions (angry, sad, and happy) plus a neutral category. Moreover, they have demonstrated that their system can have accuracies over 80% on EMO-DB, about 20% more than the baseline SVM standard. They have also applied their method to 3 other databases ([32], eNterface05 [33], and Baum-1s [34]) and were able to get results higher than the baseline method. They concentrated on how autonomous feature selection in deep convolutional neural networks can outperform feature selection in shallow convolutional networks and statistical model-based techniques like HMM. Deep convolutional neural networks are effective at simulating even the smallest transient signal characteristics. However, this flexibility comes

at the expense of tuning exponentially more variables, which necessitates the use of additional training samples. Millions of samples are used to train these networks when used for picture applications. However, the number of samples in SER is typically restricted to thousands. Additionally, this increases the likelihood of overfitting in deep convolutional network-based solutions.

Recurrent neural networks (RNN) can learn and react to the temporal event without changing the slowly shaped weights thanks to their feedback connection, forming short-term activations for recent events. This feature can be beneficial in the case of applications where time is an essential feature, like speech processing, music composition, and video description. The magnitude of the weights will determine whether the error signals flow backwards in time, grow or disappear when they are trained through backpropagation through time. As a result, the network will either produce fluctuating weights or train and converge slowly [35]. To be able to incorporate the short-term adaptation of RNNs and avoid the problems above, a new architecture called Long ShortTerm Memory (LSTM) has been introduced [35]. In recent years, LSTM networks have been becoming the centre of attention for many applications involving time-series events. Speech Processing and especially speech emotion recognition are two of these applications [36, 37]. In an early proposal for using LSTM networks [38], a multimodal LSTM-based classification network has been proposed for exploiting acoustic, linguistic, and visual information. In their study, they compared both unidirectional and bidirectional LSTM networks. They also have compared their proposed results with the AVEC 2011 Audio/Visual Emotion Challenge [39]. In this research, they extract 1941 audio features composed of Prosodic, Spectral, and Voice quality features, linguistic word-level content, and all the video features extracted by applying the Viola-Jones method, segmented optical flow, and head tilt. Then all the features are fed to a unidirectional and a bidirectional LSTM network. Later in another work [40], a context-aware system was proposed for end-to-end recognition of emotions in speech using CNNs followed by LSTM networks. The big difference in their method versus other deep learning algorithms is that they do not preselect features before training the network. They introduce raw input to the system and let the black box choose the best representations for the features. In another recent research, a system based on two layers of modified LSTMs with 512 and 256 hidden units is proposed, followed by a layer of attention weighting on both time dimension and feature dimension and two fully connected layers at the end. In their research, they have stated that humans' attention on the whole stimuli is not balanced, and it has been shown incorporating this concept creates excellent results in image processing. Therefore, they have proposed a self-attention mechanism to the forgetting gate of an LSTM layer, which results in the same performance while reducing the computations.

After introducing the attention concept and transformer architecture [41] in natural language processing, This model has been widely used in other applications. Recently, the Transformer has also been adapted for audio processing but is typically used in combination with a CNN [42–44]. A Transformer and a CNN are combined in each model block in [44], while in [42, 43], the authors stack a Transformer on top of a CNN. Other initiatives [45–47] combine CNNs with less complex attention modules. In contrast to these studies, another recent work [48] proposes an audio Spectrogram Transformer (AST) that relies solely on attention mechanisms and is convolution-free and achieves state-of-the-art performance on popular speech benchmarks.

2.1.2 Acoustic Event Classification

Audible sounds are those that are detectable and audible to humans. They span the 20 Hz to 20 kHz range. Speech, music, and environmental sound are additional categories for audible sounds (Fig. 2.1). There are sounds that are naturally formed, such as thunder, rain, and lighting, as well as sounds made by animals like dog barking and other bird sounds. Similar to this, certain man-made noises fall into this group such as environmental noise in the area of the factory floor, jet cockpit, babble noise, highway, and the subway. Furthermore, some sounds such as car horns, siren, whistle, alarm, footsteps, the sound of the clock, the sound of air conditioner, and glass breaking come under artificial sounds.

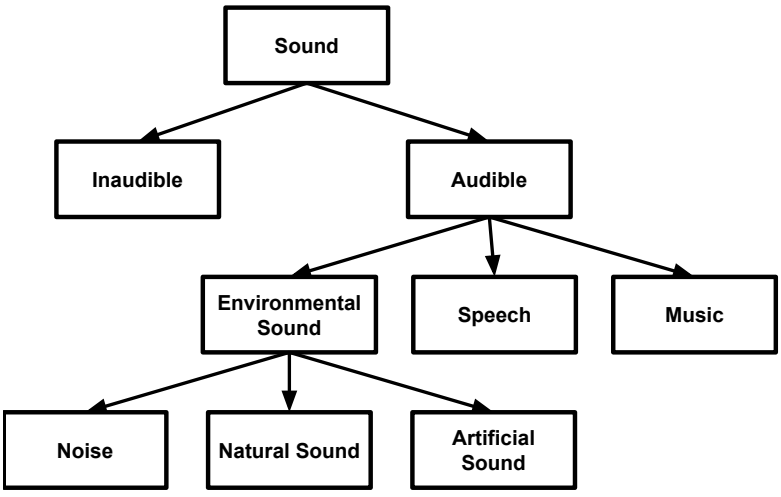


Figure 2.1: Classification of different types of sounds

The classification of natural sound is crucial for tracking environmental activity [49]. This category includes animal sounds as well. The forest department benefits from the classification of various animals using animal sound as an input in a number of ways [50]. Evaluation of biodiversity and environmental change can be done by classifying the variety of birds in a given area

[51, 52]. This kind of sound classification has recently become very popular and has proven to be extremely helpful in medical research [53]. Artificial sound categorization is mostly employed in home automation [54–56], urban sound analysis [57, 58], remote surveillance [59, 60], and other applications.

Similar to SER, traditional methods in this field only rely on hand-crafted features in the time domain such as ZCR, power of the signal, and entropy of energy and frequency domain such as spectral centroid, spectral space, the entropy of spectral, flux, the spectral roll of, MFCC that are then fed to Gaussian mixture model [61], HMM [62], ANN [63], KNN [63], and SVM [63] for classification.

After shining deep learning models, they have been used extensively for acoustic sound classification [64, 65]. An early work [64] utilised a two-layer convolution network to classify environment and urban sounds. In this study, overlapping segments were generated by resampling the raw recordings. A log-scaled Mel-spectrogram was then extracted from them after that. The short input audio segments are then classified using a CNN model after applying augmentation techniques to compensate for the lack of labelled data. The outcomes demonstrated that the suggested strategy outperformed the traditional methods by a significant margin. RNNs and their combinations with CNNs have also been widely used in this application as well [66–68].

Unfortunately, there are still very few publically available databases of environmental recordings, both in terms of quantity and size. The lack of human-annotated data is crucial because the size of the learning dataset has a significant impact on the performance of supervised deep models. Therefore, semi/self-supervised and unsupervised [69] methods are proposed to mitigate this effect. Among them, self-supervised learning [70] attracts a lot of interest due to its success in other domains. In a study [71], a triplet loss-based SSL has been utilised for audio classification. In a recent attempt [72], the author proposed three different techniques, Word2Vec, Audio2Vec and temporal gap, to perform SSL for sound classification. In another line of work [73], an AudioBERT is presented to learn prototypical representations from sound using SSL.

2.2 Graph Neural Networks

Deep learning on graph data has emerged as a major topic in the past few years. This is because graphs provide a natural and convenient way to deal with large data. Unlike other neural networks which perform on structured data, graph neural networks are a class of deep learning methods designed to perform inference on unstructured data described by graphs. GNNs are neural networks that can be directly applied to graphs, and provide an easy

way to do node-level, edge-level, and graph-level prediction tasks. Among the different graph neural networks, graph convolution networks (GCN) have received the most attention [74–76] owing to the popularity of traditional CNNs. GCNs have wide applications in computer vision, natural language processing, social network analysis, and even in problems arising in physics, chemistry and biology [77]. In computer vision, GCNs have been successfully applied to action recognition [78, 79], face clustering [80], object detection [81], visual reasoning [82], and in image generation generation using scene graphs [83, 84]. The GCNs can be broadly classified into two categories: *spatial* and *spectral*. Below, we explore each of them in detail.

2.2.1 Spatial Graph Neural Networks

In spatial-based approaches, they tried to imitate the convolution operation as aggregating nodes' attributes from neighbours [74]. This approach has many limitations because graph structures have unordered nodes and a different number of neighbours around each node. To go beyond these limitations, Niepert et al. [85] proposed a batch-training algorithm in a part of a graph to improve scalability on large-scale graphs. They also used fixed-sized neighbourhoods for each graph to deal with a different number of neighbours around each node. Hamilton et al. convert graph structure data into grid structure data with fixed size formation. They also proposed a ranking method based on nodes' features to use standard CNN for generating node level outputs [86]. In another work, Xu et al. [76] showed some shortcomings of previous GCNs in distinguishing certain simple graph structures. Based on the Weisfeiler-Lehman graph isomorphism test, they developed their structure to go beyond previous problems. They stated instead of a linear projection after going throw the spectral domain, they used a Multi-layer Perceptron (MLP) network and achieved state-of-the-art performance on graph classification tasks. In this paper, we extend the idea of [76] and build our model.

2.2.2 Spectral Graph Neural Networks

Spectral-based approaches formulate convolution operation in the perspective of graph signal processing [87] which defines convolution as a filter in the graph domain. Bruna et al. proposed the first spectral convolution neural network [88]. They proposed the filter in the spectral domain is a set of learnable parameters. Due to high computation cost which in order or N^3 where N is dimension of input data, Defferrad et al. [89] proposed ChebNet which redefines filter g as Chebyshev polynomials of the diagonal matrix of eigenvalues, i.e, $g_\theta = \sum_{i=0}^{K-1} \theta_i T_k(\hat{\Lambda})$ where $\hat{\Lambda} = 2\Lambda/\lambda_{max} - I_N$ and K is number of polynomials.

The Chebyshev polynomials were computed recursively, and no need to have eigen decomposition. Thus the computation complexity is reduced from $O(N^3)$ to $O(K|E|)$ where $|E|$ is the number of edges in the graph. In follow-up work, Kipf and Welling [75] introduced a first-order approximation of Chebyshev polynomials to simplify it. Based on their assumptions, the approximation becomes:

$$x *_G g = \theta_0 x - \theta_1 D^{\frac{1}{2}} A D^{\frac{-1}{2}} x \tag{2.1}$$

To reduce the number of parameters and avoid overfitting, they assumed $\theta = \theta_0 = -\theta_1$. Thus, the 2.1 becomes:

$$x *_G g = \theta(I_n - \theta_1 D^{\frac{1}{2}} A D^{\frac{-1}{2}})x = \theta \hat{A} x \tag{2.2}$$

They also showed by using a kind of renormalization trick, they achieved a better result than ChebNet. Xu et al. [90] proposed that wavelets are localized in vertex domain and sparse and used graph wavelets instead of eigenvectors of graph Laplacian. Although they just achieved edge improvements, they showed in one case, the number of non-zero elements in wavelet transform has dramatically decreased.

Pooling operation is one of the important layers in convolution networks which helps to reduce dimension, rotational and position variance and aggregate low-level features in the neighbourhood to obtain high-level ones. Henaff et al. [91] showed using a simple max/mean pooling is the beginning of the network helps to reduce the dimensionality of the graph and mitigate the cost of the graph Fourier transform computation. Xu et al. [76]. In ChebNet [89], they proposed input graphs are processed by the coarsening process. After that, the vertices from the coarsened and original graph are used to form a balanced binary tree. Through this process, they could use regular 1D pooling on the vertex domain. Different from ChebNet, Zhang et al. [92] introduced SortPooling, which performs pooling by rearranging vertices in meaningful and consistent order according to their structural information. They also solve the problem of different sizes of graphs by truncating or extending vertices to a specific number. This approach enhances the accuracy in many baseline data sets and also allows end-to-end gradient-based training.

Unlike previous papers which tried to cluster nodes in vertex domain and perform pooling, DIFFPOOL [93] proposed a differentiated graph pooling module that can generate hierarchical representations of graphs and also can be used with different GCNs. In this process, they learn an assignment matrix $S^{(l)}$ and embedding matrix $Z^{(l)}$:

$$
\begin{aligned}
Z^{(l)} &= \mathrm{GNN}_{l,embed}(A^{(l)}, X^{(l)}) \\
S^{(l)} &= \mathrm{softmax}\big(\mathrm{GNN}_{l,pool}(A^{(l)}, X^{(l)})\big)
\end{aligned}
\tag{2.3}
$$

where $A^{(l)}$ is coarsened adjacency matrix and $X^{(l)}$ is the input cluster node features. Given the assignment and embedding matrices, they proposed the new representation in each layer as:

$$
\begin{aligned}
X^{(l+1)} &= {S^{(l)}}^T Z^{(l)} \\
A^{(l+1)} &= {S^{(l)}}^T A^{(l)} S^{(l)}
\end{aligned}
\tag{2.4}
$$

The formulation used in Equation 2.3 is general and can be combined with any GNN not only to enhance the performance but also to speed up the convolution operation.

2.3 Self-supervised Learning

Supervised learning has been a prominent paradigm requiring accurate annotation of datasets, commonly completed manually through painstaking human effort. Due to the massive quantities of data necessary to train state-of-the-art machine learning models effectively, several alternative means of supervision have been proposed. Pre-training neural network models on the large Imagenet dataset [94], before fine-tuning on a target dataset, has become the de-facto standard in settings where annotated data is limited. Alternative means of building large annotated datasets for pre-training, such as mining social media websites [95] have also been proposed. This section reviews advancements in two methods used for reducing the burden of human labour in data annotation related to our contributions: self-supervised learning in *audio* and *graph* data.

2.3.1 Self-supervised Learning in Audio Data

SSL has gained considerable popularity since its introduction in natural language processing [96], and computer vision [97–99] owing to its ability to learn effective data representations without requiring manual annotations. Acquiring detailed manual annotations is arguably more difficult (and often expensive) in many audio and speech processing tasks, which makes SSL an increasingly popular paradigm in audio analysis. Contrast predictive coding (CPC) was the earliest work on SSL in the audio domain [100]. This work demonstrated the applicability of contrastive SSL to audio by predicting latent audio features at

a future time instant. The popular model, Wav2Vec, further refined this CPC approach [101] to produce state-of-the-art audio presentations.

The majority of SSL works in the audio domain rely on extracting audio descriptors and then utilising deep models to reconstruct a perturbed version of those descriptions as SSL tasks. For example, a self-supervised neural voice synthesizer was used [102] to reconstruct the input as an SSL (pretext) task. The Problem Agnostic Speech Encoder (PASE) [103] is another recent work that seeks to learn multitask speech representations from raw audio by predicting a number of handcrafted attributes like MFCCs and prosody features. Teacher-student models have also been investigated, where the trained model from the previous epoch serves as the teacher for the current epoch [104]. Several recent papers in audio analysis report that SSL can improve over fully supervised models [72, 105, 106] while others use crossmodal self-supervision from visual domain [107, 108].

2.3.2 Graphs and SSL

Following the immense success of SSL on computer vision and natural language processing, very recently, there has been increasing interest in applying SSL to graph-structured data. However, it is non-trivial to transfer the pretext tasks designed for image/text for graph data analytics. The main challenge is that graphs are in irregular non-Euclidean data space. Compared to the 2D/1D regular-grid Euclidean spaces where image/language data reside in, non-Euclidean spaces are more general but more complex. Therefore, some pretext tasks for grid-structure data cannot be mapped to graph data directly. Furthermore, the data examples (nodes) in graph data are correlated with the topological structure naturally, while the examples in image and text are often independent. Hence, how to deal with such dependency in graph SSL becomes a challenge for pretext task designs.

Fig. 2.2 illustrates some examples of SSL in different research areas. The history of graph SSL goes back to at least the early studies on unsupervised graph embedding [109, 110]. These methods learn node representations by maximizing the agreement between contextual nodes within truncated random walks. A classical unsupervised learning model, graph autoencoder (GAE) [111], can also be regarded as a graph SSL method that learns to rebuild the graph structure. Graph SSL can be divided into three types: (1) reconstruction-based, (2) auxiliary property-based, and (3) contrastive-based methods. The categorizations of these methods are briefly discussed below.

Reconstruction-based Methods (Fig. 2.3) change the graph from two angles: the graph structure and its features (both nodes and edges) and generate a perturbed version [111, 112]. Additionally, they contain two sub-networks,

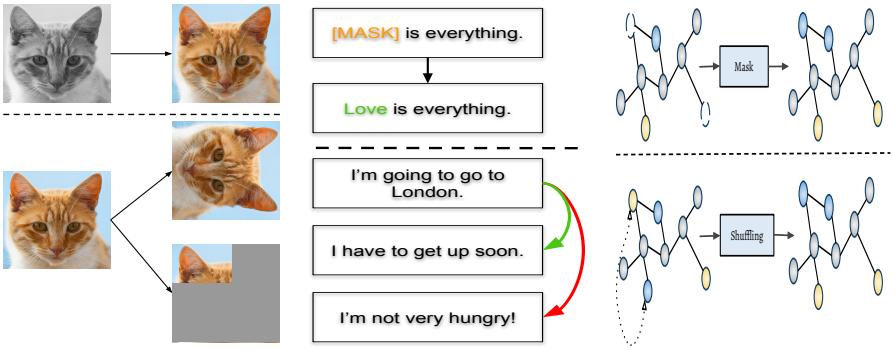


Figure 2.2: SSL examples of SSL pretext tasks in image, text, and graph analytics. (a) image colonizing (upper) and image contrastive learning (bottom). (b) mask language modeling (upper) and next sentence prediction (bottom). (c) graph mask (upper) and graph shuffling (bottom)

one for extracting graph representation and the other for reconstructing the graph by decoding. The perturbed graph is then fed to the model to get the reconstructed graph. The models were eventually made to learn the underlying information of the data using a reconstruction loss. Under this framework, graph SSL can be formulated as:

$$\theta^\star, \phi^\star = \arg\min_{\theta,\phi} \mathcal{L}_{ssl}\Big(g_\phi\big(f_\theta(\tilde{\mathcal{G}})\big), \mathcal{G}\Big) \tag{2.5}$$

where $f, g, \mathcal{G}, \tilde{\mathcal{G}}$ are graph encoder, pretext decoder, input graph, and the perturbed graph data. In addition, θ and ϕ are the learnable parameters that will be trained with reconstruction loss.

Auxiliary Property-based Methods (Fig. 2.4) enrich the supervision signals by computing the graph attributes from the graph embedding representation and compare it with the actual value [112–114]. The approaches are therefore divided into regression and classification tasks according to their output value type. The SSL task can be formulated as below:

$$\theta^\star, \phi^\star = \arg\min_{\theta,\phi} \mathcal{L}_{ssl}\Big(g_\phi\big(f_\theta(\tilde{\mathcal{G}})\big), c\Big) \tag{2.6}$$

where c denotes the specific graph auxiliary property value. For regression-based approaches, c can be localized, or global graph properties, such as the node degree or distance to clusters within $\mathcal{G}$ [112]. For classification-based methods, on the other hand, the auxiliary properties are typically constructed as pseudo labels, such as the graph partition or cluster indices [114]. Regarding the objective function, $\mathcal{L}_{ssl}$ can be a mean squared error (MSE) for regression-based and cross-entropy (CE) loss for classification-based methods.

Contrastive-based Methods (Fig. 2.5) are usually developed based on the

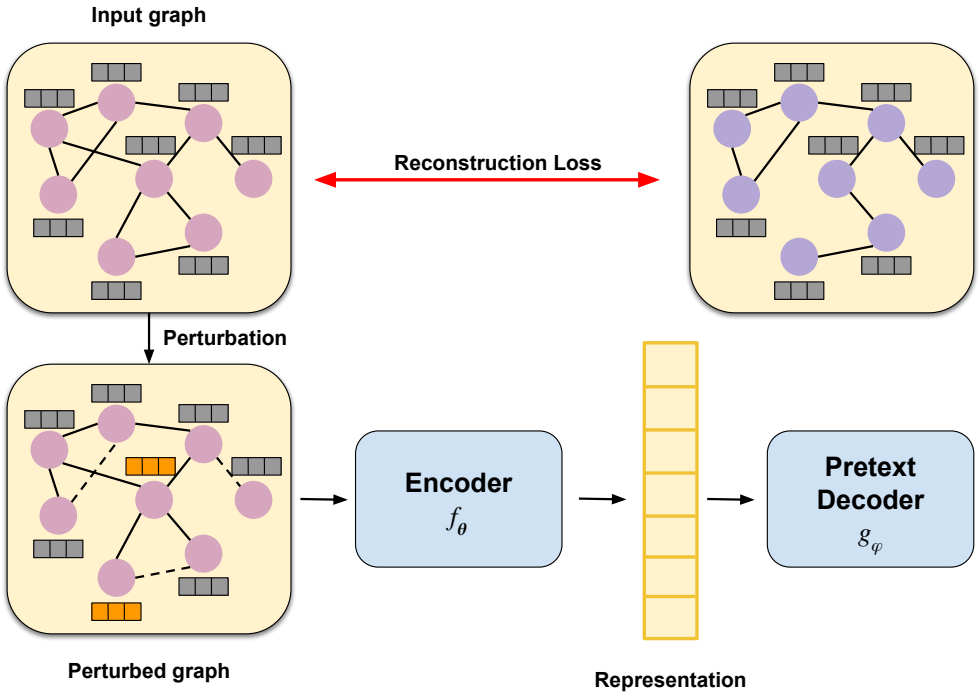


Figure 2.3: Reconstruction-based graph SSL methods. The model input is generated by an (optional) graph perturbation. In the pretext task, a generative decoder tries to recover the original graph from the middle representation, with a loss function aiming to minimize the difference between the reconstructed and original graphs

concept of mutual information (MI) maximization, where the estimated MI between augmented instances of the same object (e.g., node, subgraph, and graph) is maximized [115–117]. For a contrastive-based graph, SSL can be formulated as below:

$$\theta^\star, \phi^\star = \arg\min_{\theta,\phi} \mathcal{L}_{ssl}\Big(g_\phi\big(f_\theta(\tilde{\mathcal{G}}_1), f_\theta(\tilde{\mathcal{G}}_2)\big)\Big) \tag{2.7}$$

where $\tilde{\mathcal{G}}_1$ and $\tilde{\mathcal{G}}_2$ are two differently augmented instances of $\mathcal{G}$. In these methods, the pretext tasks aim to estimate and maximize the MI between positive pairs (e.g., augmented instances of the same object) and minimize the MI between negative samples.

2.4 Multi-modal Representation learning

Multimodal representation learning, which aims to narrow the heterogeneity gap among different modalities, plays an indispensable role in utilising ubiquitous multimodal data. Due to the powerful representation ability with multiple levels of abstraction, deep learning-based multimodal representation learning has attracted much attention in recent years.

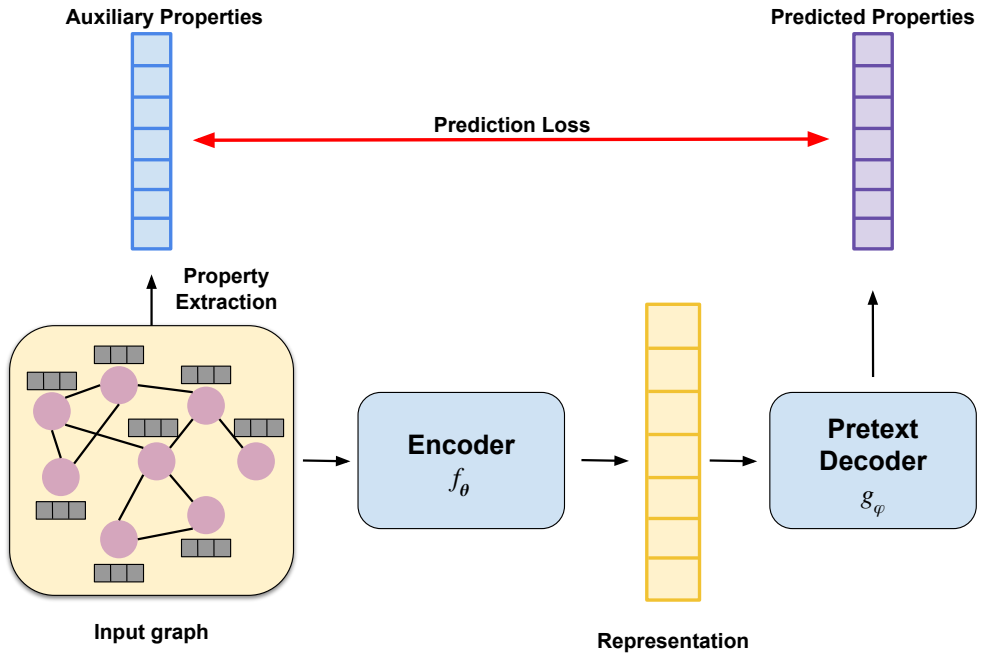


Figure 2.4: Auxiliary property-based graph SSL methods. The auxiliary properties are extracted from graphs. A classification- or regression- based decoder aims to predict the extracted properties under the training of CE/MSE loss.

Different cognitive signals describing distinct features of the same object are captured in multiple types of media, such as text, image, video, sound, and graph, to convey comprehensive knowledge about objects in the environment. The term "modality" in the context of representation learning refers to a certain method or technique of information encoding. As a result, the various media types indicated above also refer to modalities, and tasks for learning representations via many modalities will be referred to as multimodal. Multimodal data are more informative than unimodal data because they show an object from various perspectives, which are typically complimentary or supplemental in content. Early studies on speech recognition, for instance, demonstrated that the visual modality gives information about lip motions and mouth articulations, such as opening and closing, and can therefore aid in improving speech recognition performance [20]. Utilising the extensive semantics that various mediums provide is consequently beneficial.

However, even though it is simple for humans to perceive the environment using extensive data from various sensory organs [20], it is still unclear how to give machines analogue cognitive capacities. The heterogeneity gap in multimodal data is one of the difficulties we face. Fig. 2.6 illustrates how the vector representations associated with comparable semantics might be entirely different because the feature vectors from various modalities were initially

positioned in uneven subspaces. This problem, known as the heterogeneity gap, would prevent the following machine learning modules from fully utilising the multimodal data [118]. Projecting the heterogeneous features into a common subspace, where the multimodal data with comparable semantics will be represented by similar vectors, is a popular approach for solving this issue [119]. Therefore, the main goal of multimodal representation learning is to reduce the distribution gap in a shared semantic subspace while maintaining modality-specific semantics.

In the past few decades, numerous studies using a variety of methodologies have been utilized to decrease the heterogeneity gap. As a result, several applications have benefited from the development of multimodal representation learning. For instance, by utilising fused features from several media, cross-media analysis tasks, including video classification [120], event detection [121, 122], and sentiment analysis [123, 124], can perform better. Cross-modal retrieval is the process of retrieving images using text as input or the opposite. It is made feasible by using cross-modal similarity or cross-modal correlation [125]. Cross-modal translation [126], a novel class of multimodal application, has most recently attracted significant interest in the computer vision community. It aims to transform one modality into another, as the name suggests. Examples of applications in this area are text-to-image synthesis [127], video explanation [128], and image caption [129].

As this thesis solely focuses on audio representation learning, we will only discuss works that support audio learning while attending to video data in the following. The majority of existing works on learning audiovisual representations rely on maintaining a tight temporal synchrony between the visual and audio modalities [130–132]. Consider a scene of a bike moving away from the camera. The revving sound of the bike fades as it moves away. While an audio-only-based model may not be capable of detecting the fading sound as 'bike', taking into account the bike as a visual cue, it is possible to identify the event as 'motorbike running'. Computer vision-inspired models are common [133–135], where two augmented views of a given audio/audiovisual sample are fed to a shared 'backbone', followed by optimizing a contrastive loss [136–140], distillation [140, 141], quantization [130] or information maximization [142, 143]. However, the vision-inspired audio representation learning methods do not take full advantage of the temporal information available in video data or the complementary knowledge between modalities. Another difficult aspect of such approaches is that data augmentation functions, being vision-inspired, are not often well-suited to a multimodal input.

Heterogeneous graphs are a compact, efficient, and scalable way to represent data involving multiple different entities and their relations [144, 145]. Modelling the interaction of entities (including modalities) with heterogeneous

graphs is a relatively new paradigm. Multimodal heterogeneous graphs have been successfully used to address various problems in computer vision and natural language processing, such as visual-question answering [146], multi-media recommendation [144, 147], audio-visual sentiment analysis [148], and cross-modal retrieval [145]. Multimodal heterogeneous graphs lead to a closer coupling between concepts in multiple modalities, resulting in a significant performance improvement over previous methods [144, 146–148].

2.5 Summary

This chapter presented an overview of existing research related to our contributions to audio representation learning, graph neural networks, graph self-supervised learning, and multi-modal representation learning. Firstly, we reviewed audio representation learning in two subfields: speech emotion recognition and acoustic event classification. The aforementioned fields are growing steadily thanks to advancements in deep learning models, along with more extensive and more diverse datasets. We highlighted the most important concerns in these areas and investigated the suggested solutions.

Secondly, we discussed literature on graph neural networks. By grouping works into spectral and spatial approaches, we see a notable paucity of spatial approaches compared to spectral works. We first concentrated on spatial methods and studied a few techniques that were built around solely taking geometry into account. On the other hand, we studied spectral techniques in more detail and made connections to graph signal processing materials. Thus, we will focus on this difference in Chapter 3 and evaluate which can be more effective for our application. We also summarised graph-based solutions for other applications and realized that our work in Chapter 3 is the first attempt to apply graphs to audio data.

Thirdly, we review the literature on self-supervised learning related to our contributions: audio representing learning and graphs structural learning. These methods paved the way for proposing our contribution, in particular, semi-supervised learning of audio data with graph self-supervised learning proposed in Chapter 5.

Finally, we review literature proposing multi-modal learning. We specifically focus more on cases with only audio with the supervision of video data. There are a number of studies in both graph multi-modal processing and multi-modal representation learning that we can find, but none combine the two. In Chapter 6, we propose a heterogeneous graph and architecture for stratifying the audio representation with visual supervision.

In the next chapters, we will explain how the proposed graph modelling can be applied to audio data in different applications.

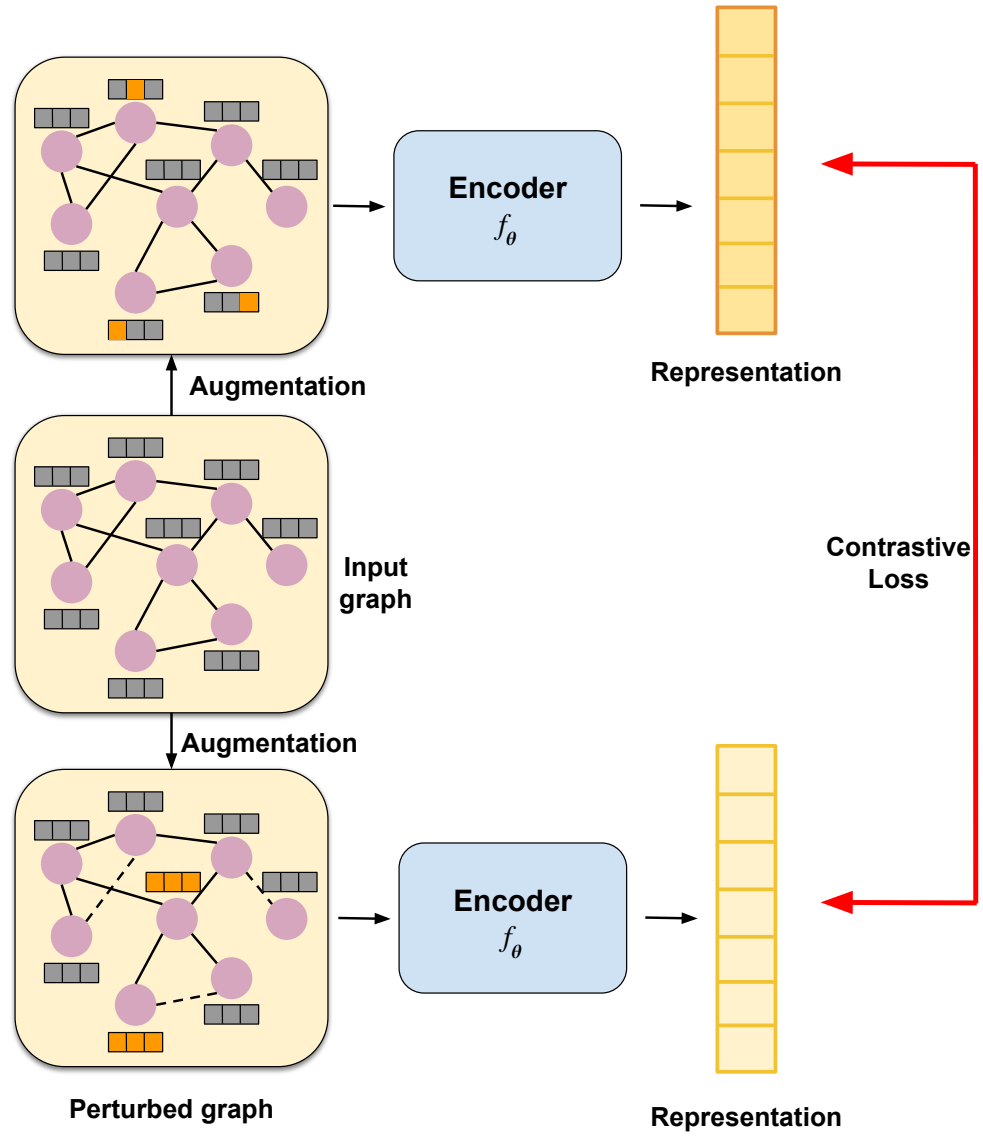


Figure 2.5: Contrast-based graph SSL methods. Two different augmented views are constructed from the original graph. Then, the model is trained by maximizing the MI between two views.

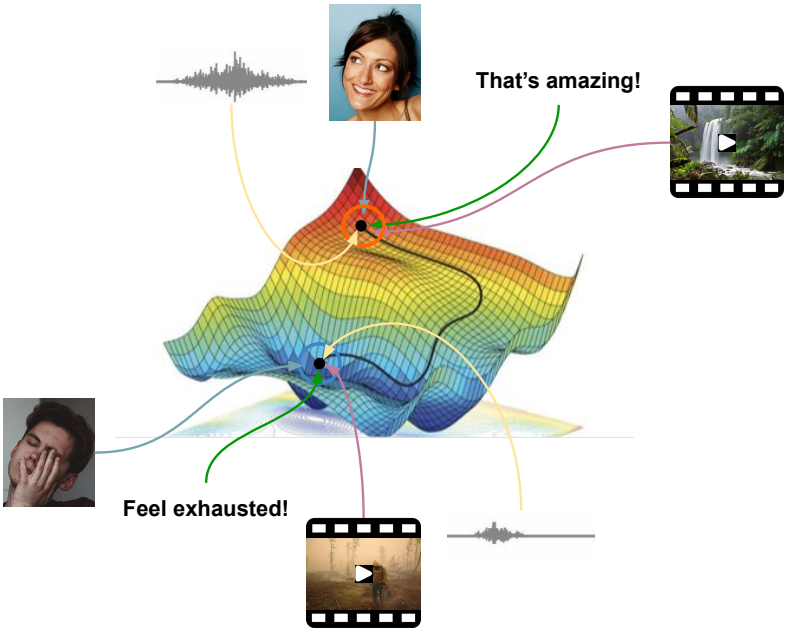


Figure 2.6: Schematic of the common subspace learning, which aims to project the heterogeneous data of different modalities into a common subspace. This image shows that the different modalities of a phenomenon share common knowledge in an ideal embedding space.

Chapter 3

Audio as Graph

3.1 Introduction

Machine recognition of emotional content in speech is crucial in many human-centric systems, such as behavioural health monitoring and empathetic conversational systems. SER, in general is a challenging task due to the huge variability in emotion expression and perception across speakers, languages and cultures.

Many SER approaches follow a two-stage framework, where a set of LLDs are first extracted from raw audio. The LLDs are then input to a deep learning model to generate discrete (or continuous) emotion labels [36, 149–151]. While using hand-crafted acoustic features is still common in SER, lexical features [25, 152] and log Mel spectrograms are also used as inputs [153]. Spectrograms are often used with CNNs [153] that do not explicitly model the speech dynamics. Explicit modelling of the temporal dynamics is important in SER as it reflects the changes in emotion dynamics [154]. To capture the temporal dynamics of emotion, recurrent models, especially the LSTMs, are popular [36, 150, 151]. The recurrent models, though predominant in SER, often lead to complex architecture with millions of trainable parameters.

A compact, efficient and scalable way to represent data is in the form of graphs. GCNs [75] have been successfully used to address various problems in computer vision and natural language processing, such as action recognition [78], object tracking [155] and text classification [156]. In the context of audio analysis, we are aware of only one recent work that proposed an attention-based graph neural network architecture for few-shot audio classification [157].

Motivated by the recent success of GCNs, we propose to adopt a deep graph approach to SER. We base our work on *spectral* GCNs, which have a strong foundation in graph signal processing [158]. Spectral GCNs perform convolution operation on the spectrum of the graph Laplacian, considering the convolution kernel (a diagonal matrix) to be learnable [88]. This involves

eigen decomposition of the graph Laplacian matrix, which is computationally expensive. To reduce the computational cost, ChebNet approximates the convolution operation (including the learnable convolution kernel) in terms of Chebyshev polynomials [89]. The most popular form of GCN uses a first-order approximation of the Chebyshev polynomial to further simplify the convolution operation to a linear projection [75]. Such GCN models are simple to implement and have been successfully used for various node classification tasks in social media networks and citation networks [75].

In this chapter, we cast SER as a graph classification problem. We model a speech signal as a simple graph, where each node corresponds to a short windowed segment of the signal. Each node is connected to only two adjacent nodes, thus transforming the signal to a *line graph* or a *cycle graph*. Owing to this particular graph structure, we take advantage of certain results in graph signal processing [159] to perform accurate graph convolution (in contrast to the approximations used in popular GCNs). This leads to a light-weight GCN architecture with superior emotion recognition performance on the IEMOCAP [160] and the MSP-IMPROV [161] databases.

To summarize, the contributions of this chapter are as follows: (i) To the best of our knowledge, this is the first work that takes a graph classification approach to SER. (ii) Leveraging theories from graph signal processing, we propose a GCN-based graph classification approach that can efficiently perform accurate graph convolution. (iii) Our model has significantly fewer trainable parameters ($\sim$30K only). Despite its smaller size, our model achieves superior performance on both IEMOCAP and MSP-IMPROV databases outperforming relevant and competitive baselines.

3.2 Proposed Graph Approach

In this section, we describe our graph classification approach to SER. First, we construct a graph from each speech sample. Next, we develop a new GCN architecture that assigns a discrete emotion label to each (speech sample transformed to) graph. Fig. 3.2 gives an overview of our approach. Below, we describe each component in detail.

3.2.1 Graph Construction

Given a speech signal (utterance), the first step is to construct a corresponding graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} \in \{v_i\}_{i=1}^{M}$ is the set of M nodes, and $\mathcal{E}$ is the set of all edges between the nodes. The adjacency matrix of G is denoted by $\mathbf{A} \in \mathbb{R}^{M \times M}$ where an element $(\mathbf{A})_{ij}$ denotes the edge weight connecting v_i and v_j.

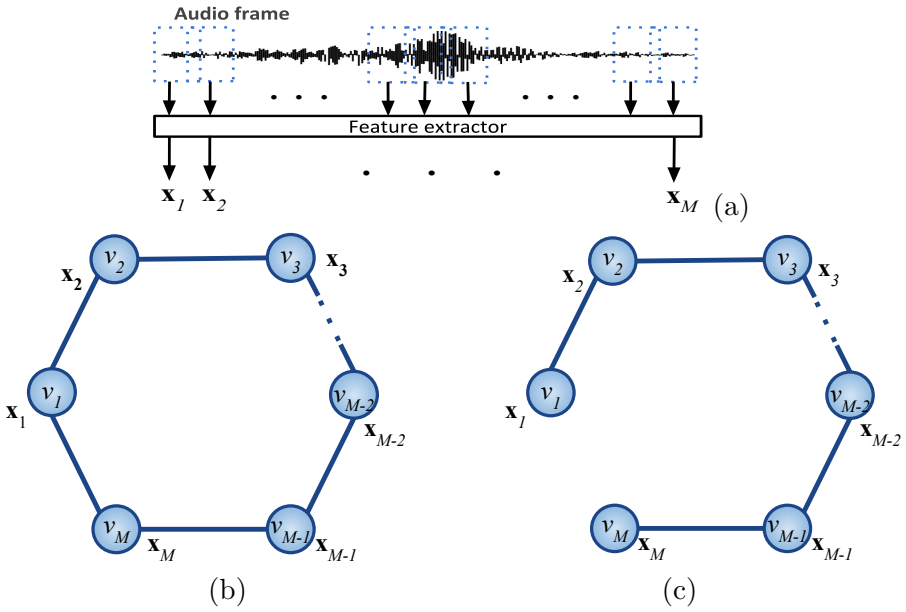


Figure 3.1: Graph construction from the speech input. (a) LLDs are extracted as node features $\mathbf{x}_i$ from raw speech segments; (b) cycle graph and (c) chain graph.

Our graph construction strategy follows a simple frame-to-node transformation, where M frames (short, overlapping segments) of the speech signal form the M nodes in G (see Fig. 3.1(a)). Since the graph structure is not naturally defined here, we investigate two simple undirected graph structures: (i) a *cycle graph* defined by the adjacency matrix $\mathbf{A}_c$, and (ii) a *line graph* defined by adjacency $\mathbf{A}_l$. The two graph structures are shown in Fig. 3.1(b)-(c).

$$\mathbf{A}_c = \begin{bmatrix} 0 & 1 & 0 & \cdots & 1 \\ 1 & 0 & 1 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & \cdots & 1 & 0 \end{bmatrix} \quad \mathbf{A}_l = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 1 & 0 & 1 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}$$

These two graph structures are important because of the special structures of their graph Laplacians, which significantly simplifies spectral GCN operations. This is discussed in the following section in more detail.

Each node v_i is also associated with a *node feature* vector $\mathbf{x}_i \in \mathbb{R}^P$. A node feature vector contains LLDs extracted from the corresponding speech segment. A feature matrix $\mathbf{X} \in \mathbb{R}^{M \times P}$ containing all node feature vectors is defined as $\mathbf{X} = [\mathbf{x}_1, \cdots \mathbf{x}_M]$.

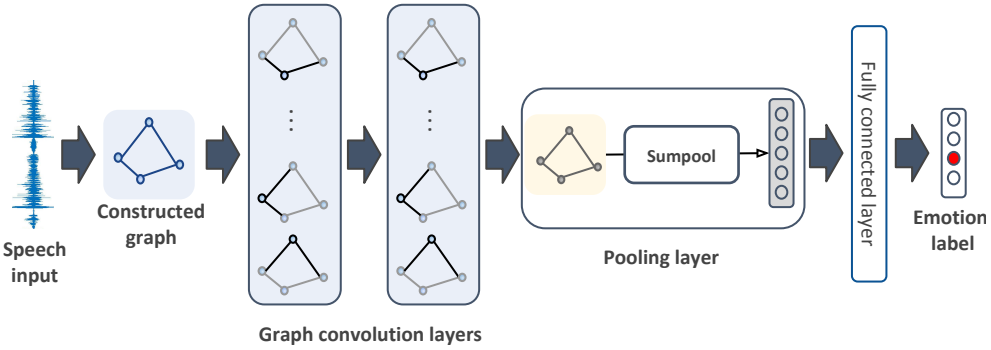


Figure 3.2: Our proposed graph-based architecture for SER consists of two graph convolution layers and a pooling layer to learn graph embedding from node embeddings to facilitate emotion classification.

3.2.2 Graph Classification

Given a set of (utterances transformed to) graphs $\{G_1, ..., G_N\}$ and their true labels $\{\mathbf{y}_1, ..., \mathbf{y}_N\}$ represented as one-hot vectors, our task is to develop a GCN architecture that is able to recognize the emotional content in the utterances. Our architecture comprises two graph convolution layers, a pooling layer that yields a graph-level embedding vector, followed by a fully connected layer that produces the discrete emotion labels (Fig.3.2).

Graph convolution layer. We base our model on a spectral GCN, which performs graph convolution in the spectral domain. Following the theory of graph signal processing [158], graph convolution in time domain is defined as

$$\mathbf{h} = \mathbf{x}_i * \mathbf{w} \tag{3.1}$$

where $\mathbf{w}$ is the graph convolution kernel (learnable) and $\mathbf{x}_i$ is the input node features. This is equivalent to a product in the graph spectral domain.

$$\hat{\mathbf{h}} = \hat{\mathbf{x}}_i \odot \hat{\mathbf{w}} \tag{3.2}$$

where $\hat{\mathbf{h}}$, $\hat{\mathbf{x}}_i$, and $\hat{\mathbf{g}}$ denote the output, node features and the convolution filter in the graph spectral domain i.e., their graph Fourier transforms (GFT). Considering the node feature matrix and adopting a matrix notation, we get

$$\hat{\mathbf{H}} = \hat{\mathbf{X}}\hat{\mathbf{W}} \tag{3.3}$$

In order to have $\hat{\mathbf{X}}$ and $\hat{\mathbf{W}}$, we usually compute the normalized graph Laplacian matrix

$$\mathcal{L} = \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}} \tag{3.4}$$

where $\mathbf{D}$ is the degree matrix, $\mathbf{L} = \mathbf{D} - \mathbf{A}$ where $\mathbf{A}$ is the adjacency matrix of

the graph. The eigen decomposition of $\mathcal{L}$ can be written as

$$\mathcal{L} = \mathbf{U}\Lambda\mathbf{U}^T = \sum_{i=1}^{M} \lambda_i \mathbf{u}_i {\mathbf{u}_i}^T \tag{3.5}$$

where λ_i is the i^{th} eigen value of $\mathcal{L}$ corresponding to the eigen vector $\mathbf{u}_i$, $\Lambda = \text{diag}(\lambda_i)$ and $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2 \cdots \mathbf{u}_N]$. The exact graph convolution operation is thus defined as

$$\hat{\mathbf{H}} = (\mathbf{U}^T\mathbf{X})(\mathbf{U}^T\mathbf{W})$$
$$\mathbf{H} = \mathbf{U}\hat{\mathbf{H}}$$

The graph convolution propagation at k^{th} layer thus becomes

$$\mathbf{H}^{(k+1)} = \mathbf{U}\Big((\mathbf{U}^T\mathbf{H}^{(k)})(\mathbf{U}^T\mathbf{W}^{(k)})\Big) \tag{3.6}$$

where $\mathbf{H}^{(0)} = \mathbf{X}$ and $\mathbf{W}$ is learnable. Note that for $\mathbf{A} = \mathbf{A}_c$ (cycle graph), $\mathbf{L}$ takes the following form

$$\mathbf{L} = \begin{bmatrix} 2 & -1 & 0 & \cdots & -1 \\ -1 & 2 & -1 & \cdots & 0 \\ 0 & -1 & 2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -1 & 0 & \cdots & -1 & 2 \end{bmatrix}$$

The $\mathbf{L}$ is circulant and GFT is equivalent to the Discrete Fourier Transform (DFT) [159]. Similarly, for $\mathbf{A} = \mathbf{A}_l$ (line graph), GFT is equivalent to Discrete Cosine Transform (DCT). This makes the convolution operation convenient and computationally efficient as we can avoid eigen decomposition that can be computationally expensive for arbitrary graph structures.

Following a recent work on GCN [76], we propose to learn the convolution kernel in Eq. (3.6) in terms of a Multi-Layer Perceptron (MLP). Finally, our convolution operation takes the following form

$$\mathbf{H}^{(k+1)} = \mathbf{U}\Big(\text{MLP}\big(\mathbf{U}^T\mathbf{H}^{(k)}\big)\Big), \tag{3.7}$$

where only the MLP parameters are learnable.

Pooling layer. Our objective is to classify entire graphs (as opposed to the more popular task of graph node classification). Hence, we need a function to attain a *graph-level* representation $\mathbf{h}_G \in \mathbb{R}^Q$ from the node-level embeddings. This can be obtained by pooling the node-level embeddings $\mathbf{H}^{(k)}$ at the final layer before passing them onto the classification layer. Common choices for

pooling functions in the graph domain are mean, max, and sum pooling [75, 85]. Max and mean pooling often fail to preserve the underlying information about the graph structure, while sum pooling has shown to be a better alternative [76]. We use sum pooling to obtain the graph-level representation:

$$\mathbf{h}_G = \text{sumpool}(\mathbf{H}^{(K)}) = \sum_{i=1}^{M} \mathbf{h}_i{}^{(K)} \tag{3.8}$$

The pooling layer is followed by one fully-connected layer, which produces the classification labels. Our GCN model is trained with the cross-entropy loss $= -\sum_n \mathbf{y}_n \log \tilde{\mathbf{y}}_n$.

3.3 Evaluation

In this section, we present experimental results and analysis to evaluate the performance of the proposed GCN architecture with application on speech emotion recognition.

3.3.1 Dataset

We evaluate our model on two most popular SER databases: IEMOCAP [160] and MSP-IMPROV [162]. For both databases, a single utterance may have multiple labels owing to different annotators. We consider only the label which has majority agreement.

The **IEMOCAP** database contains a total of 12 hours of data collected over 5 dyadic sessions with 10 subjects. To be consistent with previous studies, we used four emotion classes :*anger*, *joy*, *neutral*, and *sadness*. The final dataset contains a total of 4490 utterances including 1103 *anger*, 595 *joy*, 1708 *neutral* and 1084 *sad*.

The **MSP-IMPROV** database contains utterances from 12 speakers collected across six sessions. The dataset contains a total of 7798 utterances including 792 samples for *anger*, 3477 for *neutral*, 885 for *sad* and 2644 samples for *joy*.

3.3.2 Node Features

We extract a set of low-level descriptors (LLDs) from the speech utterances as proposed for Interspeech2009 emotion challenge [163] using the OpenSMILE toolkit [164]. The feature set includes Mel-frequency cepstral coefficients (MFCCs), zero-crossing rate, voice probability, fundamental frequency (F0) and frame energy. For each sample, we use a sliding window of length 25ms (with a stride length of 10ms) to extract the LLDs locally. Each feature is then

smoothed using a moving average filter, and the smoothed version is used to compute their respective first order delta coefficients. In addition, we also add spontaneity as a binary feature for IEMOCAP as this information is known to help SER [165]. The spontaneity information comes with the database. Altogether this yields node feature vectors of dimension $P = 35$ for IEMOCAP and $P = 34$ for MSP-IMPROV (no spontaneity feature).

3.3.3 Implementation Details

Each speech sample produces a graph of $M = 120$ nodes, where each node corresponds to an (overlapping) speech segment of length 25ms. Padding is used to make the samples of equal length. The dimension of the graph embedding is set to $Q = 64$. We perform a 5-fold cross-validation and report both average weighted accuracy (WA) and unweighted accuracy (UA). All parameters and validation remain the same for both databases. Our network weights are initialized following the Xavier initialization [166]. We used Adam optimizer with a learning rate of 0.01 and a decay rate of 0.5 after each 50 epoch for all experiments. We used Pytorch for our experiments on an NVIDIA RTX-2080Ti GPU.

3.3.4 Results and Analysis

Comparison with graph-based models. We compare our model against three state-of-the-art deep graph models using the same node features and a cycle graph structure.

GCN [75]. A natural baseline to compare with our model is a spectral GCN in its standard form. The original network [75] is designed for node classification and only yields node-level embeddings. To obtain a graph-level embedding, we used the sum pooling function.

PATCHY-SAN [85]. A recent architecture that learns CNNs for arbitrary graphs. This architecture is originally developed for graph classification.

PATCHY-Diff [93]. A recent work on hierarchical GCN proposes to use a differentiable pooling layer between graph convolution layers. We used this pooling layer with PATCHY-SAN as in the original paper.

Table 3.1 and Table 3.3 compare our model against these existing graph models (Graph baselines) in terms of SER accuracy. All these models use the same node features. Clearly, our model outperforms all the graph baselines by a significant margin. Compared to the popular GCN [75], our model improves the recognition accuracy by more than 9% and 3% on IEMOCAP and MSP-IMPROV, respectively. This result indicates that accurate convolution in the graph domain improves the accuracy significantly.

Comparison with SER state-of-the-art. In addition to the graph baselines,

Table 3.1: Results and comparison on the IEMOCAP databases in terms of weighted accuracy (WA) and unweighted accuracy (UA).

Model	**WA (%)**	**UA (%)**
Graph baselines		
GCN [75]	56.14	52.36
PATCHY-SAN [85]	60.34	56.27
PATCHY-Diff [93]	63.23	58.71
SER models		
Attn-BLSTM 2016 [151]	59.33	49.96
BLR 2017 [167]	62.54	57.85
RNN 2017 [36]	63.50	58.80
CRNN 2018 [168]	63.98	60.35
SegCNN 2019 [153]	64.53	**62.34**
LSTM 2019 [37]	58.72	-
CNN-LSTM 2019 [37]	59.23	-
Ours (cycle)	**65.29**	**62.27**
Ours (line)	64.69	61.14
Ours (cycle w/o MLP)	64.19	60.31

we compare our model with a number of recent SER models. These include a Bayesian model [167], CNN models ([37], SegCNN [153]), RNN architectures ([36], RCNN [168]), LSTM models ([37], Attn-BLSTM [151]) and a CNN-LSTM model [37]. The majority of the above models (except the models by Latif et al. [37]) use LLDs as input.

Tables 3.1 and 3.3 show that our model outperforms (i) all graph baselines despite a simpler architecture and (ii) all LSTM-based architectures (a class of models most commonly used in SER) on both databases by a significant margin. Our model (with the cycle graph structure) achieves the highest WA on IEMOCAP and is comparable to the state-of-the-art WA on MSP-IMPROV. In terms of UA, our model's performance is comparable to SegCNN on IEMOCAP.

Nevertheless, our model has significantly fewer parameters: 30K learnable parameters vs. 8.8M in SegCNN and 0.8M in LSTM.

Network size. Table 3.2 compares the number of learnable network parameters

Table 3.2: Model size comparison in terms of learnable parameters

GCN	PTCHY-SAN	PTCHY-Diff	BLSTM	**Ours**
∼76K	∼60K	∼68K	∼0.8M	∼**30K**

Table 3.3: Results and comparison on the MSP-IMPROV databases in terms of weighted accuracy (WA) and unweighted accuracy (UA).

Model	**WA (%)**	**UA (%)**
Graph baselines		
GCN [75]	54.71	51.42
PATCHY-SAN [85]	55.47	52.33
PATCHY-Diff [93]	56.18	53.12
SER models		
ProgNet 2017 [161]	**58.40**	-
CNN 2019 [37]	50.84	-
LSTM 2019 [37]	51.21	-
CNN-LSTM 2019 [37]	52.36	-
Ours (cycle)	**57.82**	**55.42**
Ours (line)	57.08	54.75
Ours (cycle w/o MLP)	56.82	53.22

for various models with ours. All graph networks are smaller (an order of magnitude smaller) than LSTM architectures yet highly accurate. Our model has the highest accuracy with half the parameters of other graph-based networks. This is owing to the lightweight convolution operation and because of the choice of our graph structure. In our approach, graph structure remains the same for all samples, which requires us to compute the eigen-decomposition only once. This operation can even be replaced by directly using DFT or DCT kernels, as mentioned earlier.

Ablation: Between the two graph structures we investigated, higher SER accuracy is achieved using the cyclic structure on both the databases. With the line graph, the model accuracy is slightly lower. We also compare our model's performance for different pooling strategies (used to compute graph-level representation from node embeddings) in Table 3.4. Aligned with observations made in past works (e.g., [76]), sumpool shows improvement over meanpool

Table 3.4: Comparing different pooling strategies for our model on the IEMOCAP database.

Pooling	Maxpool	Meanpool	Sumpool
WA (%)	61.68	62.45	**65.29**

and maxpool by 2.8% and 3.6% on IEMOCAP. When using graph convolution without MLP (see Eq. 3.6) performance drops by 1% (see Table 3.1 and 3.3). These results confirm that each proposed component in our network (MLP convolution, sumpool) contributes positively towards its performance.

3.4 Conclusion

We proposed a compact and efficient GCN architecture (with only 30K learnable parameters) for recognizing emotion content in speech. To the best of our knowledge, this is the first graph-based approach to SER. We transformed speech utterances to graphs with simple structures that largely simplify the convolution operation on graphs. Also, the graph structure we defined remains the same for all samples as our edges are not weighted. This leads to a lightweight GCN architecture which outperforms LSTMs, standard GCNs and several other recent graph models in SER. Our model produces higher or comparable performance to the state-of-the-art on two benchmark databases with significantly fewer learnable parameters.

In the next chapter, we will continue exploring graph modelling on audio graphs by learning graph structure.

Chapter 4

Audio Graph Learning

4.1 Introduction

Human emotion is expressed, perceived and captured using a variety of dynamic data modalities, such as speech (verbal), videos (facial expressions) and motion capture (body gestures). Modelling and analysis of these cues are critical for many human-centric systems with applications ranging from driver's safety to mental healthcare to human-robot conversational systems. In recent years, significant progress has been made towards the recognition and analysis of emotion using dynamic facial expressions [169, 170], speech [154, 171] and body gestures [172]. Since human emotion is inherently multimodal, research efforts that combine information from multiple modalities are also on the rise [173]. Besides expressed emotion, work has also been done to analyze emotion evoked by natural images [174], videos [175] and music [176].

In the literature of dynamic emotion recognition, recurrent models, such as LSTM, are common [154, 177]. These networks often have a complex architecture with millions of trainable parameters requiring large amounts of training data. This makes many emotion recognition models incompatible for use in resource-constrained devices. A compact, efficient and scalable way to represent data is in the form of graphs. We thus adopt a graph approach to building a compact model for dynamic emotion recognition. Furthermore, existing emotion recognition models assume a prior knowledge of the input modality. Since emotion can be sensed through a variety of modalities, a generalized model that can handle disparate modalities efficiently is important. We show that our *modality-agnostic* graph approach is able to achieve state-of-the-art accuracy across various modalities with significantly fewer trainable parameters.

Traditionally, sequences are modelled using RNNs. However, recent literature has successfully used the idea of defining a sequence over a graph [78, 178, 179]. Considering a video frame sequence as a 'structured' graph,

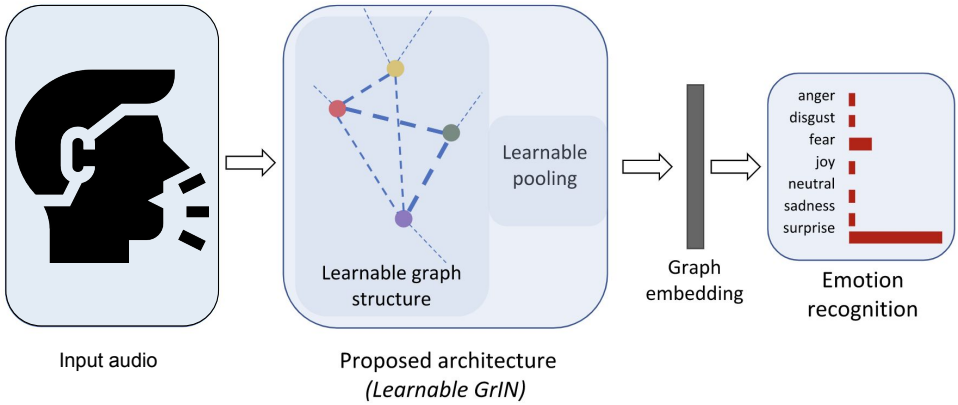


Figure 4.1: A graph approach to modelling speech. Data samples are transformed to a *learnable* graph structure, where each node corresponds to a short temporal segment. A novel graph architecture (L-GRIN) produces an embedding for the entire graph, facilitating speech emotion recognition.

Mao et al. showed that graph models can outperform RNNs [178]. Motivated by these recent successes and in the pursuit of a compact model, we propose to adopt a graph approach to model emotion dynamics. Subsequently, we cast emotion recognition as a joint graph learning and classification problem (see Fig. 4.1 for an overview). In our approach, each dynamic data sample is represented as a graph, where each node corresponds to a short temporal segment in the data. Each node is associated with the features extracted from the short temporal segment (frame) as its node attributes. This frame-to-node graph construction approach focuses on modelling the temporal dynamics in data; note that spatio-temporal structure (e.g., facial keypoints structure) within the graph resists the idea of a generic, modality-agnostic model and also increases model size significantly. Our graph structure (and hence the model) does not change with the choice of modality or node attributes. Modelling as a graph offers compactness and convenience to handle missing data (particularly common in mocap).

The graph structure, i.e., the edge weights connecting the nodes, is not naturally defined here. When a graph structure is not known apriori, a common practice is to manually construct the graph. This, however, results in sub-optimal graphs. We thus propose to learn the graph structure itself during the training stage. This is a generalized formulation, where the temporal dependencies between the nodes are automatically discovered. The only assumption we make is that the graph structure remains the same for all samples in a given database. To this end, we propose a novel GCN architecture, the *Learnable Graph Inception Network* (L-GrIN), with several novel components: a new definition of graph convolution that uses a non-linear layer-wise projection

technique, introduction of an inception module in graph domain, learnable graph structure and a learnable graph-to-vector pooling function. Our architecture produces superior results on the IEMOCAP, a popular speech emotion recognition database. In summary, the main contributions of this chapter are as follows:

- A generalized, modality-agnostic graph approach to classifying dynamic signals that combines graph learning with graph classification.
- A novel graph architecture, termed **L-GrIN**, with a new graph convolution layer, a graph inception module, learnable graph structure and learnable graph-to-vector pooling.
- State-of-the-art performance on speech emotion recognition task.

4.2 Proposed Graph Learning Method

In this section, we describe our deep graph approach to emotion recognition. First, we construct a graph from dynamic input data following a generalized frame-to-node approach. Next, we propose a novel architecture that jointly performs graph learning and graph classification. This is achieved by optimizing over a new loss function that combines classification loss and a graph structure loss. The proposed architecture, L-GRIN, is illustrated in Fig. 4.3. Below, we describe each component of this network in detail.

4.2.1 Graph Construction

Given a dynamic input sequence, our first task is to construct an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ that can efficiently capture the emotion-related dynamics in the data, where $\mathcal{V}$ is the set of nodes with cardinality $|\mathcal{V}| = M$ and $\mathcal{E}$ is the set of all edges between the connected nodes. A representative description of $\mathcal{G}$ is typically given by an adjacency matrix $\mathbf{A} \in \mathbb{R}^{M \times M}$ which is symmetric for an undirected graph.

Our graph construction approach follows a *frame-to-node* transformation, where M frames in the data form the M graph nodes $\{v_i\}_{i=1}^{M} \in \mathcal{V}$ (see Fig. 4.2). A frame refers to a small temporal segment of the data, e.g., an audio segment of length 40ms. In order to encode the temporal information, a frame (node) should be connected with weights to a series of past and future nodes. An element $(\mathbf{A})_{ij} \in \mathbf{A}$ contains the weight corresponding to the edge $e_{ij} \in \mathcal{E}$ connecting v_i and v_j. Note that the graph structure is not naturally defined here, i.e., the elements in $\mathbf{A}$ are unknown. A common way to define the elements in $\mathbf{A}$ is through constructing a distance function manually [78]. However, this may result in a sub-optimal graph representation. Hence, we propose to learn

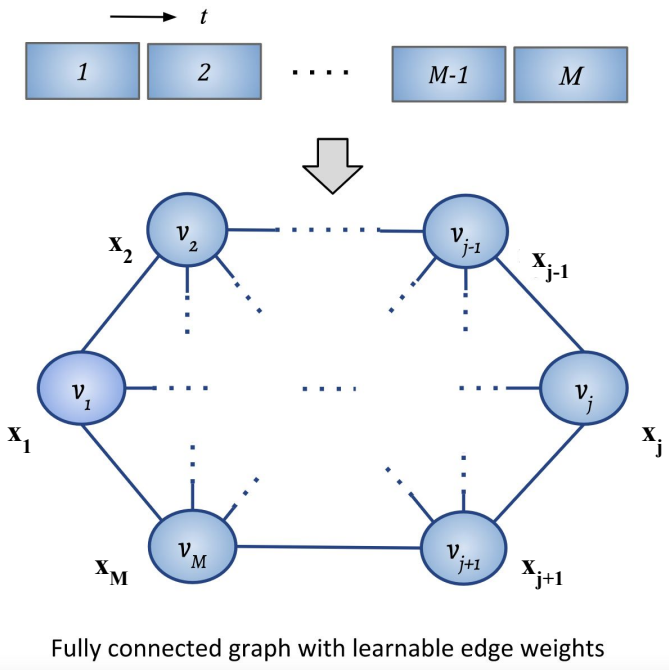


Figure 4.2: Graph construction: Given a dynamic input sequence of M segments, a fully-connected graph with M nodes is constructed without making any assumption. The edge weights are learned during the training phase. Each node is associated with a node attribute vector $\mathbf{x}_i$.

the elements in $\mathbf{A}$ by jointly optimizing a structural loss combined with a classification loss. This loss function will be discussed in Section 4.2.2.

In order to capture the emotion content at the node level, we rely on modality-specific features or even raw data. Each node v_i is thus associated with a *node feature* vector $\mathbf{x}_i \in \mathbb{R}^P$. A feature matrix $\mathbf{X} \in \mathbb{R}^{M \times P}$ consisting all the node feature vectors is defined as $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_M]^T$. Features for individual modalities are discussed in Section A.

4.2.2 Learnable Graph Inception Network

Given a set of (dynamic inputs transformed to) graphs $\{G_1, ..., G_N\}$ and their true labels $\{\mathbf{y}_1, ..., \mathbf{y}_N\}$, our task is to develop a deep graph architecture that is able to recognize the emotional content in the data. Since the graph structure is not naturally defined here, we also learn an optimal $\mathbf{A}$ from the training data with the underlying assumption that each graph has different node features but the same edge weights. We formulate this as a joint graph learning and graph classification problem.

A common GCN architecture takes the node feature matrix $\mathbf{X} \in \mathbb{R}^{M \times P}$ and the graph adjacency matrix $\mathbf{A}$ as inputs and produces a *node-level* representation matrix $\mathbf{Z} \in \mathbb{R}^{M \times Q}$, where Q is the dimension of the output feature vector

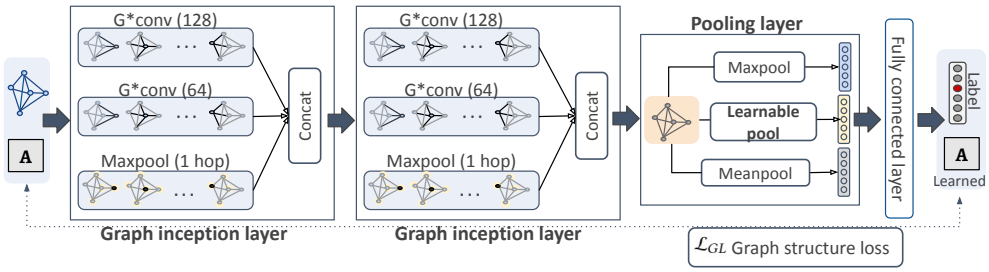


Figure 4.3: Our proposed architecture, L-GrIN, consists of two graph inception layers (with a new spectral graph convolution layer) and a pooling layer (two fixed pooling layers and a learnable pooling layer). The inception layers produce node-level representations that are pooled to obtain a graph-level representation by the pooling layer. L-GRIN also learns the underlying graph structure (adjacency matrix) by jointly optimizing a classification loss and a graph structure loss.

at each node. A GCN layer $\mathbf{H}^{(k+1)}$ can be defined as a non-linear function of its previous layer as follows

$$\mathbf{H}^{(k+1)} = \sigma(\mathbf{A}\mathbf{H}^{(k)}\mathbf{W}^{(k)}) \tag{4.1}$$

where $\mathbf{W}^{(k)}$ is the weight matrix for the k^{th} layer of the neural network, σ is a non-linear activation function, such as a ReLU, and k is the layer number $(k = 0, \cdots K)$. Note that $\mathbf{H}^{(0)} = \mathbf{X}$ and $\mathbf{H}^{(K)} = \mathbf{Z}$. An effective improvement on this propagation rule has been recently proposed [75].

$$\mathbf{H}^{(k+1)} = \sigma\big(\mathbf{D}^{-\frac{1}{2}}(\mathbf{A}+\mathbf{I})\mathbf{D}^{-\frac{1}{2}}\mathbf{H}^{(k)}\mathbf{W}^{(k)}\big) \tag{4.2}$$

where $\mathbf{D}$ is the degree matrix of $\mathbf{A}$, and $\mathbf{I}$ is an $M \times M$ identity matrix. Note that the term within the parenthesis in Eq. (4.2) is simply a linear projection and can be re-written as

$$\mathbf{H}^{(k+1)} = \sigma(\hat{\mathbf{A}}\mathbf{H}^{(k)}\mathbf{W}^{(k)}) \tag{4.3}$$

where $\hat{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{A}+\mathbf{I})\mathbf{D}^{-\frac{1}{2}}$.

We present a new GCN architecture, called L-GrIN (see Fig. 4.3), for joint graph learning and classification. It has the following four new components:

• **Non-linear spectral graph convolution ($\mathcal{G}^*$conv).** Motivated by a recent work on spatial graph neural network [76], we replace the linear projection in (4.3) by a multi-layer perceptron (MLP) layer, and replace $\hat{\mathbf{A}}$ by a learnable $\mathbf{A}$. Thus, instead of the linear layer in (4.3), we define a new spectral graph convolution layer $\mathcal{G}^*(\cdot)$ as follows:

$$\mathcal{G}^*(\mathbf{H}^{(k)}) = \sigma\Big(\mathrm{MLP}^{(k)}\big(\mathrm{ReLU}(\mathbf{A})\mathbf{H}^{(k)}\big)\Big) \tag{4.4}$$

where MLP(.) has two hidden layers with η neurons each, $\mathbf{A}$ is the learnable adjacency matrix and σ is a nonlinear activation function. $\mathbf{A}$ is learned through a joint optimization process described later in this section. The ReLU$(\cdot)$ in Eq. (4.4) ensures the non-negativity of $\mathbf{A}$. We refer to the convolution operation defined above as $\mathcal{G}^*$**conv** in the rest of the chapter.

• **Graph inception.** We extend the idea of inception layer in traditional CNNs [180] to the graph domain, and introduce a *graph inception* module in our architecture (see Fig. 4.3). Our graph inception layer consists of two graph convolution layers and one maxpool layer operating on directly connected (1-hop) neighbours only.

Given an input $\mathbf{H}^{(k)}$, the proposed graph inception layer is defined as follows:

$$\mathbf{H}^{(k+1)} = \Big[\mathcal{G}_1^*(\mathbf{H}^{(k)}) \,|\, \mathcal{G}_2^*(\mathbf{H}^{(k)}) \,|\, \text{maxpool}(\mathbf{H}^{(k)})\Big] \tag{4.5}$$

where $|$ denotes concatenation of the node features, and $\mathcal{G}_1^*$ and $\mathcal{G}_2^*$ are two $\mathcal{G}^*$**conv** layers (see Eq. (4.4)) with different size of their MLP layers ($\eta = 128$ for $\mathcal{G}_1^*$ and $\eta = 64$ for $\mathcal{G}_2^*$). Hence, for an input of $\mathbf{H}^{(k)} \in \mathbb{R}^{M \times P}$, the inception layer produces an output $\mathbf{H}^{(k+1)} \in \mathbb{R}^{M \times (128+64+P)}$.

The motivation behind the inception layer is to be able to capture the emotion dynamics at multiple temporal scales. The two $\mathcal{G}^*$**conv** layers that yield embeddings of different dimensions can be interpreted as a *multiscale analysis* on graphs in the spectral domain. Like a traditional inception layer in CNN, our graph inception layer also combines features from multiple scales allowing the network to have both width and depth. Our graph inception layer has fewer parameters (compared to inception networks in CNNs), enabling us to feed the input directly to the inception layer.

The maxpool function in Eq. (4.5) operates on every node separately. For each node v_i, we only consider its directly connected neighbours (1-hop) and maxpool over the embeddings along the feature dimension. Note that as we start with a fully-connected graph, initially, this operation is the same as maxpooling over all nodes, but this changes quickly as we start learning the graph structure.

• **Learnable pooling for graph-level representation.** Our objective is to classify entire graphs, as opposed to the more common task of classifying each node. Hence, we seek a *graph-level* representation $\mathbf{h}_G \in \mathbb{R}^Q$ as the output of our network. This can be obtained by pooling the node-level representations $\mathbf{H}^{(k)}$ at the K-th layer before passing them to the classification layer (see Fig.4.3). Common choices for pooling functions in the graph domain are mean, max and sum pooling. Max and mean pooling often can not preserve the underlying information about the graph structure, while sum pooling is shown to be a better alternative [76]. However, all these pooling functions treat

every neighbouring node with equal importance, which may not be optimal. To this end, we propose to *learn* a pooling function Ψ that combines the node embeddings from the K-th layer to produce an embedding for the entire graph. Additionally, we also use maxpool and meanpool and combine all the graph-level embeddings together. The pooling layer is thus defined as follows:

$$\mathbf{h}_G = \left[\text{maxpool}(\mathbf{H}^{(K)}) \,|\, \Psi(\mathbf{H}^{(K)}) \,|\, \text{meanpool}(\mathbf{H}^{(K)})\right] \tag{4.6}$$

$$\Psi(\mathbf{H}^{(K)}) = \mathbf{H}^{(K)}\mathbf{p}$$

where $\mathbf{p}$ has learnable weights to combine node-level embeddings to obtain a graph-level embedding.

• **Learnable adjacency** ($\mathbf{A}$). Recall that in our task, the graph structure is not known. Although we can define such a structure manually, the results are sub-optimal. An effective approach would be to learn the graph structure (adjacency matrix) itself by jointly optimizing over a classification loss and graph learning loss. We assume that all videos have the same underlying graph structure containing the same number of nodes and edges. This largely simplifies our task of graph structure learning. The overall loss $\mathcal{L}$ for joint graph learning and classification is composed of two components: (i) $\mathcal{L}_{GC}$: a graph classification loss, and (ii) $\mathcal{L}_{GL}$: a graph learning loss. The classification loss $\mathcal{L}_{GC}$ is defined as the cross-entropy loss:

$$\mathcal{L}_{GC} = -\sum_{n=1}^{N} \mathbf{y}_n \log \hat{\mathbf{y}}_n \tag{4.7}$$

where $\hat{\mathbf{y}}_n$ is the predicted label for the n^{th} sample. The graph learning loss, $\mathcal{L}_{GL}$, is designed to facilitate learning the pooling vector $\mathbf{p}$ and the adjacency matrix $\mathbf{A}$. This is defined as follows:

$$\mathcal{L}_{GL} = \underbrace{\lambda_1 \mathbf{e}^T(\mathbf{A}_d \odot \mathbf{A})\mathbf{e} + \lambda_2 \|\mathbf{A}\|_F^2}_{\text{graph structure loss}} + \underbrace{\lambda_3 \|\mathbf{p}\|_2^2}_{\text{learnable pooling}} \tag{4.8}$$

where $\odot$ denotes element-wise multiplication, $\mathbf{e}$ is an all-ones vector, $\|\cdot\|_F$ denotes Frobenious norm, λ_1, λ_2, and λ_3 control the relative weights of the three terms, and $\mathbf{A}_d$ is a structure matrix defined as follows:

$$(\mathbf{A}_d)_{ij} = (i-j)^2 \tag{4.9}$$

The structure matrix $\mathbf{A}_d$ forces the nodes that are temporally close to each other to have stronger relationships, i.e. higher weights in the $\mathbf{A}$. The larger the squared distance between two nodes v_i and v_j (frames), the smaller will be the weights in $(\mathbf{A})_{ij}$. The ReLU operation (see Eq. (4.4)) ensures the non-negativity

of the elements in $\mathbf{A}$. The overall optimization is thus as follows:

$$\min_{\mathbf{A},\mathbf{p},\boldsymbol{\Theta}^{(1:k)}} \mathcal{L} = \min_{\mathbf{A},\mathbf{p},\boldsymbol{\Theta}^{(1:k)}} \left[\mathcal{L}_{GC} + \mathcal{L}_{GL}\right]$$

where Θ denotes all other learnable network parameters across all graph convolution layers, including its constituent MLP layers. Every term in the overall loss function $\mathcal{L}$ is differentiable, thereby allowing an end-to-end optimization.

4.3 Evaluation

In this section, we evaluated our proposed model on speech emotion recognition.

4.3.1 Databases

We use the popular IEMOCAP database [160] for evaluating the performance of our model on speech emotion recognition. This database contains a total of 12 hours of data recorded in 5 sessions, where each session contains utterances from two speakers. The final database contains a total of 5531 utterances: 1103 *angry*, 1708 *neutral*, 1636 *happy* and 1084 *sad*.

4.3.2 Node Features

We extract a set of LLDs from the raw speech utterances as proposed for Interspeech2009 emotion challenge [163] using the OpenSMILE toolkit [181]. The feature set includes MFCCs, zero-crossing rate, voice probability, fundamental frequency (F0) and frame energy. For each sample, we use a sliding window of length 25ms with a stride length of 10ms to extract the LLDs locally. Each feature is then smoothed using a moving average filter, and the smoothed version is used to compute their respective first-order delta coefficients. In addition, motivated by a recent work on speech emotion recognition [182], we also add *spontaneity* as a binary feature. The spontaneity information comes with the database. Altogether this produces node feature vectors of dimension $P = 35$.

4.3.3 Implementation Details

Each audio sample produces a graph of $M = 120$ nodes, where each node corresponds to an (overlapping) speech segment of length 25ms. Cyclic padding is used to make the samples of equal length as before. We perform a 5-fold cross-validation and report the average unweighted accuracy in Table 4.1. The unweighted accuracy, a standard evaluation strategy for IEMOCAP, does not take into account the class imbalances. It simply computes the total number of correct classifications across all classes. All other parameters and settings

Table 4.1: Speech emotion recognition results on IEMOCAP database.

Model	**Accuracy (%)**	**Params**
*BLSTM (baseline)	58.04	$\sim$ 0.8M
*GCN (baseline)	56.14	$\sim$ 78K
*PATCHY-SAN [85]	60.34	$\sim$ 60K
*PATCHY-Diff [93]	63.23	$\sim$ 68K
CNN [37]	58.52	$\sim$ 0.45M
CNN-LSTM [37]	59.23	$\sim$ 0.6M
Rep learning [183]	50.40	-
LSTM-CTC [154]	64.20	$\sim$ 0.4M
***L-GrIN**	**65.50**	$\sim$ 92K

* use same node features

remain the same as before to show the generalizability of our model. We set $\lambda_1 = \lambda_2 = 0.1$ and $\lambda_3 = 1 \times 10^{-4}$ (see Eq. 4.8). We used Adam optimizer with a learning rate of 0.01 and decay rate of 0.5 after each 50 epoch for all experiments. To initialize the learnable adjacency matrix $\mathbf{A}$, we generate a random matrix whose elements are drawn from a Normal distribution with zero mean and unit variance. We used Pytorch for implementing our model and the baselines and an NVIDIA RTX-2080Ti GPU for all experiments.

4.3.4 Baselines, State-of-the-art

Our model is compared with two baselines (BLSTM and GCN), two state-of-the-art graph-based architectures (PATCHY-SAN [85] and PATCHY-Diff [93]) as before. In addition, we also compare our model with four state-of-art methods in speech emotion recognition: CNN [37], CNN-LSTM [37], representation learning [183] and LSTM with Connectionist Temporal Modeling (LSTM-CTC) [154].

4.3.5 Results

Table 4.1 shows that our model performs better than the baselines and state-of-the-art methods on IEMOCAP. Our model's performance may seem only slightly better (1.3%) compared to LSTM-CTC, but it requires 4 times more parameters than ours. LSTM-CTC uses 238-dimensional feature vectors where our feature dimension is only 35. Although PATCHY-Diff yields a competitive accuracy with a smaller model size on IEMOCAP, it trails L-GrIN by a large margin on other databases. Note that PATCHY-SAN and PATCHY-Diff perform better than BLSTM and CNN-LSTM methods, indicating the

effectiveness of graph-based methods in general.

4.3.6 Network Analysis

In this section, we explore the effect of the components and our contributions in our method. In our method, we proposed a light L-GrIN to process graphs in different resolutions with the ability to learn the graph structure at the same time with the main objective. We also proposed a learnable pooling layer to attend to the node importance for achieving the graph representation. A more extensive analysis can be found in Appendix A.

Network Size Table 4.1 lists the number of learnable network parameters for the baselines, state-of-the-art graph-based architectures and the proposed L-GrIN. As mentioned earlier, a graph network largely reduces the number of learnable parameters as compared to the BLSTM or CNN architectures without compromising the recognition accuracy. Our model has more parameters than the baseline GCN due to the inception layers and other learnable parameters but also improves the recognition accuracy significantly. PATCHY-SAN and PATCHY-Diff have smaller network sizes compared to L-GRIN, but both trail L-GrIN in terms of performance on all databases.

Learnable vs. Fixed Pooling Recall that to obtain a graph-level embedding from node-level embeddings, L-GrIN learns a pooling function (see Fig. 4.3). To show if learnable pooling indeed improves the recognition performance, we compare its performance with various fixed pooling strategies: max pooling, mean pooling and sort pooling (sortpool) [92]. Table 4.2 presents the comparisons on the IEMOCAP database in terms of speech emotion recognition accuracy, which clearly shows the advantage of learnable pooling over fixed pooling strategies. A similar trend is observed for other databases.

Learnable vs. Manually Constructed Adjacency An adjacency matrix represents the pairwise relationship between the graph nodes. When this information is not available naturally, a common practice is to manually construct an adjacency matrix. We argued earlier that this might result in sub-optimal

Table 4.2: Comparison between learnable and fixed pooling strategies on the IEMOCAP database. All experiments in this table use the same (binary) adjacency matrix for a fair comparison.

Pooling	**Accuracy** (%)
Maxpool	64.32
Meanpool	63.16
Sortpool [92]	61.05
Learnable pool	**65.50**

Table 4.3: Comparison between learnable and manually constructed graph structures. For a fair comparison, all experiments use maxpool to convert node embeddings to graph embeddings.

	Accuracy (%)	**Params**
	IEMOCAP	IEMOCAP
Binary	61.4	78K
Weighted	54.3	78K
Learnable	**65.5**	92K

graph structures, which in turn affect the classification performance. We now compare the performance of learnable adjacency with two fixed adjacency matrices:

(i). *Binary adjacency:* a natural choice is a binary adjacency matrix as used for graph-based action recognition [78]. This is defined as $(\mathbf{A}_b)_{ij} = 1$ if $|i - j| = 1$ and 0 otherwise, i.e., a node (frame) is connected only to its subsequent and preceding node in the temporal direction.

(ii). *Weighted adjacency:* Another adjacency matrix is formed by using the squared ℓ_2 distance between two node attributes as their edge weight. This is defined as $(\mathbf{A}_w)_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|_2^2$.

Table 4.3 compares the performance of the proposed learnable adjacency with the two fixed adjacency matrices described above on the IEMOCAP database. For this set of experiments, we used only maxpooling to obtain the graph-level embeddings for a fair comparison. Clearly, the learnable adjacency matrix shows consistent improvement in accuracy across all databases for a relatively small increase in model complexity (only 6% additional parameters). The results show that a learnable adjacency has better at generalizing across databases and modalities.

Ablation Study We performed exhaustive ablation experiments to investigate the contribution of each component we proposed to build L-GrIN. Table 4.4 presents the ablation results on the IEMOCAP database. We observe that each new component brings significant improvement (row 2 to row 5) over the performance of standard GCN [75] which has 56.1% recognition accuracy (the top row in Table 4.4). The introduction of the graph inception layer increases the recognition rate by 5%; when combined with our new graph convolution layer $\mathcal{G}^*$conv (Eq. (4.4)), the accuracy increases to 62.7%. Adding the learnable graph structure (learned $\mathbf{A}$) and learnable pooling bring the accuracy up to 65.5% both contributing to the accuracy. Removing the learnable adjacency

and pooling reduces the accuracy by 2.3% and 2.6% respectively. The ablation results show that each of the proposed components in our architecture is important and contributes positively towards its superior performance.

Table 4.4: Ablation study on the IEMOCAP database. Each new component in L-GRIN contributes towards its performance.

$\mathcal{G}^*$**conv**	**Inception**	**Learned A**	**Learned p**	**Accuracy** (%)
-	-	-	-	56.1
✓	-	-	-	57.2
-	✓	-	-	61.5
-	-	✓	-	57.9
-	-	-	✓	59.0
-	-	✓	✓	60.4
✓	✓	-	-	62.7
✓	✓	✓	-	62.9
✓	✓	-	✓	63.2
✓	✓	✓	✓	**65.5**

Inception Layer Settings We also investigate the effects of the graph inception layer hyperparameters: (i) the parameter η corresponding to the size of the graph convolution filters $\mathcal{G}_1^*$ and $\mathcal{G}_2^*$ in Eq. (4.5), and (ii) the number of graph inception layers in L-GrIN. First, we vary the filter dimensions (which can be interpreted as scales) in the two inception layers and note how this corresponds to the model's performance. Results for the RML database are presented in Table 4.5; similar trends have been observed for other databases. Results in Table 4.5 show that we achieve the best performance for the combination of (64, 128), which is used in our model. Next, we vary the number of inception layers in the model, each with (64, 128) filter combination (see Table 4.5. We observe that reducing or increasing the number of inception

Table 4.5: Analyzing inception layer settings on the IEMOCAP database.

Effect of filter size (η)	
Size of the two filters	**Accuracy (%)**
$(16, 32)$	61.6
$(32, 64)$	64.3
$(\mathbf{64}, \mathbf{128})$	**65.5**
$(128, 256)$	65.1
Effect of number of inception layers	
Number of layers	**Accuracy (%)**
1	63.0
2	**65.5**
3	65.3

layers from 2 results in a drop in performance. We chose to use two inception layers in the proposed model. It is obvious that the model size increases significantly as we add more inception layers or increase filter sizes within the layers. We notice a small drop in performance with larger filter sizes and with a higher number of inception layers. This could be possibly due to over-smoothing and over-mixing of the node features. However, the over-smoothing effect is not as prominent as in many node classification tasks.

4.4 Conclusion

We proposed a novel, generalized graph architecture that can recognize emotion in a variety of dynamic input sequences. Our proposed architecture, L-GrIN, learns to detect emotion while jointly learning the underlying graph structure (adjacency matrix) and a pooling function to yield graph-level representation from node-level embeddings. We proposed a new spectral graph convolution operation and introduced the idea of inception in the graph domain. The advantage of our model lies in its state-of-the-art performance on speech emotion recognition task, with significantly fewer parameters compared to the CNNs and RNNs (to check the results for other modalities, see Appendix A). This indicates that our model is suitable for applications in resource-constrained devices, such as smartphones.

We used both modality-specific features and even raw data as node features in this chapter. Our approach is not tied to any particular feature. In fact, our model can be trained end-to-end by combining it with modality-specific networks (e.g., a CNN) for feature extraction. The architecture we developed, although focuses on emotion recognition, is fairly generic. It will be applicable to a variety of classification tasks involving dynamic data, such as pose estimation, action recognition and visual speech recognition. Since our model makes no assumption about the graph structure, this is also applicable to common unstructured graphs.

In the next chapter, we will check the validity of self-supervision on graphs and how we can apply it to audio graphs.

Chapter 5

Self-supervised Graphs for Audio Representation Learning

5.1 Introduction

Large databases with high-quality manual annotations are scarce in the audio domain. For tasks such as speech-based emotion analysis, manual annotations are often difficult to acquire due to the subjectivity involved in the perception and expression of emotion across speakers, language and culture. On the other hand, for tasks such as acoustics event classification, manually annotating a large volume of audio data is simply impractical. Thus a core challenge in the audio analysis is to learn from a limited amount of labelled data while taking advantage of a larger amount of unlabeled training samples.

Why graphs. SSL has emerged as an effective approach to learning from unlabeled data [96–99]. We propose an SSL approach on graphs to learn effective audio representations from a limited amount of labelled data. Considering each audio sample as a node in a graph, we cast audio classification as a node labelling task. The motivation behind adopting a graph approach is two-fold: (i) It leads to compact models as compared to commonly used recurrent speech models as noted in recent works [1, 184]; (ii) A graph structure, if properly constructed, can efficiently capture the relationship between the small number of available labelled nodes and a larger number of unlabeled nodes. Extensive experiments with standard benchmarks bring out the advantages of graph-based methods in terms of performance compared to the non-graph models.

Following the success of SSL on images, it has been extended to graph data for both fully supervised [113], and semi-supervised [114, 185, 186] tasks. Graph SSL tasks usually involve learning the local or global structure or the

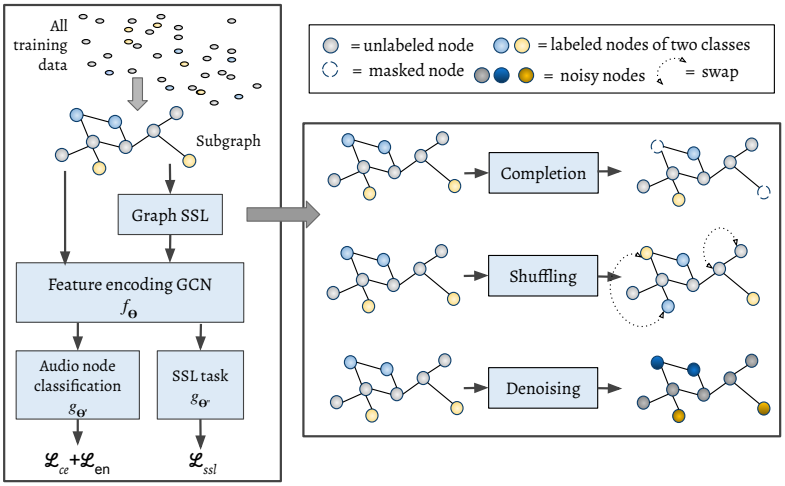


Figure 5.1: Our model: A subgraph-based audio representation learning framework with SSL task. Our subgraph construction technique is efficient, can handle class imbalance, and the SSL framework facilitates robust and effective learning from highly limited labelled data.

context information in the data [113, 114, 185]. Conventional graph tasks such as node clustering and graph partitioning have already been used as SSL tasks [185]. In the audio domain, SSL has also started gaining popularity. Several recent papers report that SSL can improve over fully supervised models [72] while others use crossmodal self-supervision from visual domain [107, 108]. However, works that use the graph approach to learning audio representation are limited. We are aware of only one recent work where an audio sequence has been considered as a line graph to exploit graph signal processing theory to achieve accurate spectral graph convolution [1].

In this chapter, we propose a graph SSL approach to learning effective audio representations from a limited amount of labelled data. Considering each audio sample as a graph node, we propose a subgraph-based learning framework with new self-supervision tasks. Our framework takes advantage of the entire pool of available data (labelled and unlabeled) during training, while during inference, our subgraphs are constructed using random edges with no overhead (e.g., nearest neighbour computation) of graph construction. In contrast to the more common SSL-then-finetune approach, we use an auxiliary learning paradigm where an SSL task and a node labelling task are performed jointly. Evaluation on large benchmark databases shows that our model achieves better results than fully supervised models outperforming state-of-the-art on several databases. To summarize, our contributions are as follows:

- We develop a subgraph-based auxiliary learning framework for audio representation learning with limited labelled data. To the best of our knowledge, ours is the first work on self-supervised semi-supervised audio

representation learning with graphs.

- We propose two new graph SSL tasks: graph denoising and graph shuffling, and show that they can improve the performance of any graph network for semi-supervised node classification.
- We demonstrate the superior performance of our model for two tasks (acoustic event detection and speech emotion classification) on three large benchmark audio databases. Our model, despite using limited labelled data, performs better or on par with fully supervised models and can produce representations that are robust to various types of audio noise.

5.2 Proposed Graph Self-supervised Tasks

In this section, we propose our self-supervised (sub)graph-based audio representation learning model. Our model consists of an audio feature encoder, a subgraph construction step and a multitask-SSL architecture with new pretext tasks and loss functions.

5.2.1 Audio Feature Encoder

Our model has a feature encoder $f : \mathcal{S} \rightarrow \mathbf{H}$ that takes raw audio $\mathcal{S}$ as input and returns embedding $\mathbf{H}$. These embeddings are used as node attributes in graph $\mathcal{G}$ that we construct using labelled and unlabeled training data (described below). Owing to the different types of audio data (speech and sound), we use two different feature encoders: low-level descriptors for speech data and log-spectrogram-based convolutional features for the generic (non-speech) audio. We chose simpler embeddings so as to demonstrate the effectiveness of our graph approach. Nevertheless, our model is not tied to any specific embedding, and rich audio embeddings such as *wav2vec* [101] may lead to better classification results. More details about f are provided in the Experiments section.

5.2.2 Graph Construction During Training

Given a collection of N (labelled and unlabeled) audio samples for training, we construct an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to capture the relationships among the samples, where $\mathcal{E}$ is the set of all edges between the connected nodes, and $\mathcal{V} = \mathcal{V}_l \cup \mathcal{V}_u$ has $|\mathcal{V}_l| = M$ labeled and $|\mathcal{V}_u| = (N - M)$ unlabeled nodes. To construct the graph $\mathcal{G}$, we first consider the labelled nodes. For a node $v_i \in \mathcal{V}_l$, we compute its k-nearest neighbors among *all* nodes in $\mathcal{V}_l$ based on their node attributes $\mathbf{z}_i$. We add an edge $e_{ij} \in \mathcal{E}$ with edge weight $a_{ij} = 1$ to the first Q nodes, if v_j is among the k-nearest neighbors of v_i and has the same label as v_i. We add another edge $e_{ip} \in \mathcal{E}$ with weight $a_{ip} = -1$ between v_i and its

farthest node $v_p \in \mathcal{V}_l$. This negative weight is expected to force the two nodes to be apart in the embedding space. Every unlabeled node in $\mathcal{V}_u$ is connected to its two nearest and one farthest neighbour with a respective edge weight of 1 and -1, where the neighbours can be any node in $\mathcal{V}$. We thus obtain a corresponding graph adjacency matrix $\mathbf{A} = \{a_{ij}\}_{i,j=1}^{N}$.

5.2.2.1 Subgraph Construction

Instead of constructing one large graph containing all training samples, we propose to construct and train on subgraphs. In our case, subgraph construction ensures that we do not end up with a *sparse graph*. This is also motivated by the observations made in several recent papers where subgraphs are able to learn local context more effectively (without oversmoothing or node dependence) while reducing computational load [187–190]. To construct a subgraph $\mathcal{G}_s$, we randomly select $N_s \in \mathcal{V}_l$ labeled nodes (equal number of samples from each class) and $M_s \in \mathcal{V}_u$ unlabeled training nodes, yielding a set of $\mathcal{V}_s$ nodes, $|\mathcal{V}_s| = N_s + M_s$. This procedure ensures the degrees of the nodes do not vary too much and class balance is maintained in each subgraph. Next, the edges in the subgraph are added the same way as for the full graph mentioned above. We also show that this approach produces better results than working with a single large graph.

5.2.3 Subgraph SSL Training with Limited Labels

We adopt an auxiliary learning paradigm to merge self-supervision into the main task of audio classification. This is done by jointly optimizing for node classification and an auxiliary graph SSL task. Our model (see Fig. 5.1) has a shared GCN module for learning the latent audio representations, which is followed by two branches: one for audio classification and the other for SSL. Our model is *inductive*, i.e., neither attributes nor edges of the test nodes are present during the training process. This is a more challenging scenario than transductive learning [191].

Our node (audio) classification sub-network uses supervision from only the true labels while producing *pseudolabels* for the unlabeled nodes. We train this sub-network using two loss functions:
(i) Cross-entropy loss $\mathcal{L}_{ce}$ computed for the *labeled* nodes.

$$\mathcal{L}_{ce} = -\sum_{v_p \in \mathcal{V}_l} \mathbf{y}_p \log(\hat{\mathbf{y}}_p) \tag{5.1}$$

(ii) For the *unlabeled* nodes, we propose to compute an **entropy regularization loss** $\mathcal{L}_{en}$. This can be considered as a measure of class overlap - the lower the

Algorithm 1: Subgraph-based SSL training

Input : Labeled nodes $\mathcal{V}_l$, unlabeled nodes $\mathcal{V}_u$, node embeddings $\mathbf{X}$
Output : Learned parameters $(\Theta, \Theta', \Theta'')$, pseudolabels $\hat{\mathbf{y}}_p$ for $\mathcal{V}_u$

for *each epoch* **do**
  $\mathcal{G}_s \leftarrow$ `subgraphConstruct`$(\mathcal{V}_l, \mathcal{V}_u, \mathbf{X})$
  $\hat{\mathcal{G}}_s \leftarrow$ `createGraphforSSL`$(\mathcal{G}_s)$
  $\hat{\mathbf{y}} \leftarrow g_{\Theta'}(f_{\Theta}(\mathcal{G}_s))$; $\tilde{\mathbf{X}} \leftarrow g_{\Theta''}(f_{\Theta}(\hat{\mathcal{G}}_s))$
  $\Theta, \Theta', \Theta'' \leftarrow \mathcal{L}_{ce} + \lambda_1 \mathcal{L}_{en} + \lambda_2 \mathcal{L}_{ssl}$
end
Function `subgraphConstruct` $(\mathcal{V}_l, \mathcal{V}_u, \mathbf{X})$
  $\mathcal{V}_s \leftarrow$ randomly select N_s nodes from $\mathcal{V}_l$ and M_s nodes from $\mathcal{V}_u$
  for $\forall v_s \in \mathcal{V}_s$ **do**
    $\mathcal{V}_{nn} \leftarrow$ nearest neighbors(v_s)
    if $v_s \in \mathcal{V}_l$ **then**
      $\mathcal{E}_s \overset{+}{\leftarrow}$ edge between v_s and 2 nearest nodes $v_i \in \mathcal{V}_{nn}$ with same labels with edge weight 1
    else
    end
    $\mathcal{E}_s \overset{+}{\leftarrow}$ edge between v_s and 2 nearest nodes $v_i \in \mathcal{V}_{nn}$ with edge weight 1
  end
  $\mathcal{E}_s \overset{+}{\leftarrow}$ edge between v_s and farthest $v_i \in \mathcal{V}_{nn}$ with weight (-1)
end
return $\mathcal{G}_s(\mathcal{V}_s, \mathcal{E}_s)$

entropy loss, the more distinguishable the predicted class labels are.

$$\mathcal{L}_{en} = -\sum_{v_p \in \mathcal{V}_u} P(\hat{\mathbf{y}}_p) \log(P(\hat{\mathbf{y}}_p)) \tag{5.2}$$

where $\mathbf{y}_p$ is the true label for node v_p, $\hat{\mathbf{y}}_p$ is its predicted label, $\mathcal{V}_l$ and $\mathcal{V}_u$ are the sets of labelled and unlabeled training samples. The SSL sub-network is trained using a graph SSL task optimizing over a loss function $\mathcal{L}_{ssl}$ (discussed below). Given a subgraph input $\mathcal{G}_s$, the overall optimization is given by:

$$\begin{aligned} \min_{\Theta, \Theta', \Theta''} \big[\mathcal{L}_{ce}(\Theta, \Theta', \mathcal{G}_s) + \lambda_1 \mathcal{L}_{en}(\Theta, \Theta', \mathcal{G}_s) + \\ \lambda_2 \mathcal{L}_{ssl}(\Theta, \Theta'', \hat{\mathcal{G}}_s) \big] \end{aligned} \tag{5.3}$$

where Θ, Θ', and Θ'' are the learnable parameters for the shared GCN, classification GCN and the SSL sub-networks, $\hat{\mathcal{G}}_s$ is the SSL variant of $\mathcal{G}_s$, and λ_1 and λ_2 control the relative weights of SSL loss and entropy regularization.

To this end, we experiment with three graph SSL tasks. These SSL tasks are *model-agnostic*, and can be used with any graph neural network. We propose two novel proxy tasks in the graph SSL domain: graph denoising and

graph shuffling. In addition, we also experiment with the recently introduced graph completion proxy task.

5.2.3.1 Graph Denoising (Proposed)

Motivated by the speech denoising autoencoder [192] that learns by reconstructing clean speech from noisy input, we propose the SSL task of graph denoising. Given a subgraph $\mathcal{G}_s$, we construct a noisy graph $\hat{\mathcal{G}}_s$, where Gaussian noise is added to every node feature vector $\hat{\mathbf{x}}_s(i) = \mathbf{x}_s(i) + \mathbf{z}(i)$, where $\mathbf{z}(i) \sim \mathcal{N}(0, \epsilon)$ by adding Gaussian noise with zero mean and ϵ variance to each node feature. The SSL regression task is to learn to reconstruct node feature matrix $\mathbf{X}_s$ from noisy $\hat{\mathbf{X}}_s$ by optimizing the following loss function:

$$\mathcal{L}_{ssl} = \frac{1}{|\mathcal{V}_s|} \|\tilde{\mathbf{X}}_s - \mathbf{X}_s\|_F^2 \text{ where } \tilde{\mathbf{X}}_s = g(\Theta, \Theta'', \hat{\mathbf{X}}_s) \tag{5.4}$$

It's worth mentioning that in a recent study [193], a denoising autoencoder has been used to perform self-supervision on graph data. This work is different in a number of ways. They used a self-supervised approach that jointly learns the graph structure and node features at the same time. While we apply the cleaning loss (Eq. 5.4) after a graph-based feature extractor, the referenced work proposed an auto-encoder architecture that receives noisy node features as well as the predicted graph structure as input. In addition, the loss function in the reference was binary cross-entropy while we tried to reconstruct the node features with L2 norm loss (MSE).

5.2.3.2 Graph Completion

This SSL task forces the network to learn to reconstruct missing information so as to learn local context. Following a recent work [185], we mask a random set of target nodes $\mathcal{V}_{sc} \subset \mathcal{V}_s$ by setting their node attributes to zero. The task is to recover this missing information for the target nodes. Given $\mathcal{G}_s$, denote the ground truth feature matrix corresponding to $\mathcal{V}_{sc}$ as $\mathbf{Z}_{sc}$ and its predicted version as $\tilde{\mathbf{Z}}_{sc}$, the SSL loss is then given by

$$\mathcal{L}_{ssl} = \frac{1}{|\mathcal{V}_{sc}|} \|\tilde{\mathbf{X}}_{sc} - \mathbf{X}_{sc}\|_F^2 \tag{5.5}$$

5.2.3.3 Graph Shuffling (Proposed)

We propose a novel SSL task that aims to determine whether or not a graph node is in its correct position. This task encourages the graph network to learn structural dependencies among nodes without using annotations. Given $\mathcal{G}_s$, we randomly sample a set of graph nodes $\mathcal{V}_{sh} \subset \mathcal{V}_s$ and shuffle their node attributes randomly with each other creating $\hat{\mathcal{G}}_s$. The SSL task is posed a

binary node classification task on $\hat{\mathcal{G}}_s$ where the model outputs 1 if is node is unchanged and 0 otherwise.

$$\mathcal{L}_{ssl} = -\frac{1}{|\mathcal{V}_{sh}|} \sum_{v_i \in \mathcal{V}_{sh}} y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i) \tag{5.6}$$

5.2.4 Subgraph Construction During Inference

After completing training, we obtain a large number of *pseudolabeled* samples. In order to create the subgraph during inference, we randomly sample $|\mathcal{V}_s|$ nodes (equal number of nodes from each class) from the entire set of training samples considering both true and psesuolabels. The edges between these nodes are constructed as was done during training. Each test node (audio sample) v_t has to be connected to this graph structure $\mathcal{G}_s$. Computing nearest neighbours during inference time may not always be practical; hence, we propose to connect v_t with T training nodes (labelled or pseudolabeled) *randomly* with edge weight 1.

5.2.4.1 Optimal Number of Random Edges

A natural question is what is the optimal value of T so as to ensure that we do not connect to only pseudolabeled nodes, which could be incorrect. Hence, we ask: *What is the minimum number of nodes a test node, v_t, should be connected with, such that there is at least one connection with a true-labelled node?*

Let $\mathcal{G}$ be a graph with $|\mathcal{V}|$ nodes, including known N true labelled and M pseudolabeled nodes. The probability of having all T edges from v_t to be connected with only the pseudolabeled nodes is given by $\frac{\binom{M-1}{T}}{\binom{N+M-1}{T}}$ using Hypergeometric distribution. Therefore, the probability of an v_t to be connected to at least one true labeled node is given by $P = 1 - \frac{\binom{M-1}{T}}{\binom{N+M-1}{T}}$. With known N and M, we set a high value of $P(= 0.9)$ to compute the value of T for our experiments. Once the graph is thus constructed, we use only the audio classification branch to determine the class labels for the test nodes.

5.3 Semi-supervised Acoustic Event Classification

This section presents extensive experimental results on the analysis of our model and demonstrates its effectiveness for audio classification tasks. For more analysis of the proposed self-supervised tasks, check Appendix B.

5.3.1 Datasets

We use a large scale weakly labeled database called the **AudioSet** [194] that contains audio segments from YouTube videos. We work with 33 class labels

that have a high rater confidence score (≥ 0.7) (see Fig. 5.2 for the names of those classes). This yields a training set of around 89,000 audio clips and a test set of more than 8,000 audio clips. We consider only 10% of the 89,000 training samples as labelled and the rest are used as unlabeled training data for our experiments.

5.3.2 Feature Encoder

To extract the node features, each audio clip is divided into non-overlapping 960 ms segments. For each segment, a log-mel spectrogram is computed by taking its short-time Fourier transform using a frame of length 25 ms with 10 ms overlap, 64 mel-spaced frequency bins and log-transforming the magnitude of each bin. This creates log-mel spectrograms of dimension 96×64 which are the input to the pre-trained VGGish network [195]. We use the 128-dimensional features extracted from the VGGish for each log-mel spectrogram and average over all segments to form the final vector representation of each audio clip. Note that although we use VGGish embeddings to be comparable with previous works, other generic audio embeddings will work as well.

5.3.3 Experimental Settings

To construct the subgraph during training, we sample data with $M_s = 5$, $N_s = 2 \times$ number of classes, and $T = 4$. This yields a subgraph size of 71 nodes. This process ensures the subgraph nodes are class-balanced every time. The subgraph construction process is repeated until all unlabeled nodes appear at least once, i.e., the subgraphs are constructed by sampling without replacement. Note that for constructing the edges, the nearest neighbours need to be computed only once for all the training nodes. For the test nodes, no nearest neighbour computation is needed due to our random edge construction strategy. For graph neural network, we select regular GCN [75] with 2 graph convolution layers and a hidden size of 256 for all layers. We used the same architecture for all experiments. We use the 80:10:10 train:validation:test split *only* for the semi-supervised framework, where we consider 10% of that 80% train data as labelled and the rest as unlabeled. All hyperparameters were chosen solely based on validation data, with test data accounting for no model or parameter selections. In order to obtain robust estimates of performance metrics, this process is repeated 5 times to report the average accuracy and standard deviation. We set $\lambda_1 = 0.01$ and $\lambda_2 = 0.1$ (see Eq. 5.3). Our model uses Xavier initialization and the Adam optimizer with a learning rate of 0.001 for all experiments. We use Pytorch for implementing our models and the baselines. All models were trained on a single NVIDIA RTX-2080Ti GPU.

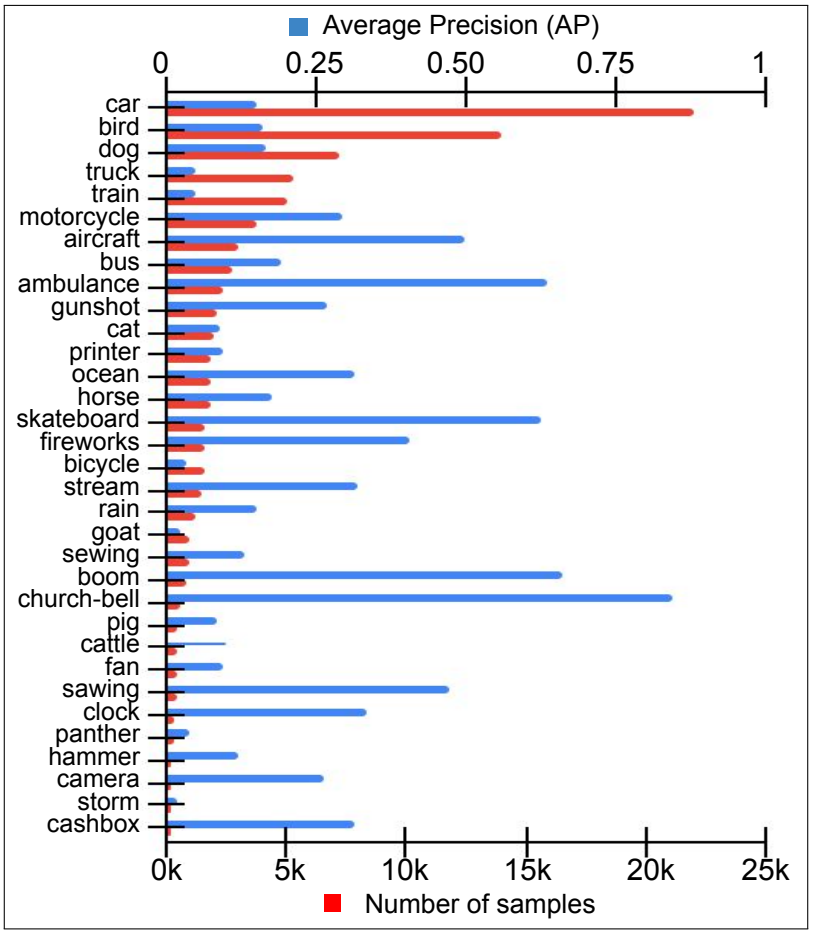


Figure 5.2: Results on **AudioSet**: Class distribution and average precision per class achieved using our model with graph completion task. Note that our model can handle the class-imbalance owing to our graph construction process, as high AP is achieved even for classes with fewer samples.

5.3.4 Baselines

We compare our method with a number of fully supervised models in Table 5.1. The Spectrogram-VGG model is the same as configuration A in [196] with only one change: the final layer being a softmax with 33 units. The feature for each audio input to the VGG model is a log-mel spectrogram of dimension 96×64 computed averaging across non-overlapping segments of length 960ms. We did not adjust the hyperparameters for baselines other than VGG and used the settings stated in the original papers. The fully supervised version of our model follows the same graph construction strategy as proposed with 80:10:10 (train:validation:test) split. The split process is done 5 times for our models, and average performance with standard deviation is reported. All other baseline implementations are done by the authors and follow the same evaluation protocol as the proposed methods.

Table 5.1: Acoustic event detection results on **AudioSet**: Our model, using only 10% labelled data, outperforms all semi-supervised models and several fully supervised models. The fully supervised version of our model without SSL shows comparable performance to the state-of-the-art, indicating the effectiveness of our subgraph-based learning strategy for audio classification. It's worth noting that the current amount of parameters for our model excludes the feature extractor, which may vary depending on the model.

Model	**mAP**	**Params**
Semi-supervised		
Ours w/o SSL	0.23 ± 0.01	218K
Ours w/ denoise	0.26 ± 0.00	260K
Ours w/ completion	0.27 ± 0.01	260K
Ours w/ shuffle	0.24 ± 0.00	219K
Ours w/ all three SSL	$\mathbf{0.28} \pm 0.02$	261K
Spectrogram-VGG	0.16 ± 0.05	6M
AST [48]	0.22 ± 0.01	88M
Fully supervised		
Ours w/o SSL	0.42 ± 0.02	218K
Spectrogram-VGG	0.26 ± 0.01	6M
DaiNet [197]	0.25 ± 0.07	1.8M
Wave-Logmel [45]	0.43 ± 0.04	81M
VATT [198]	0.39 ± 0.02	87M
AST [48]	$\mathbf{0.44} \pm 0.00$	88M

5.3.5 Results

Table 5.1 reports the average recognition accuracy (averaged over 5 runs) with standard deviation for each model and its variants. For the case of all three SSL, we applied all three self-supervised tasks in parallel with a shared feature encoder and add the corresponding loss functions to the main classification loss. It compares the performance of our model with different SSL tasks with that of fully supervised models in terms of mean Average Precision (mAP). The graph models with SSL outperform the plain graph model without SSL. When compared with the fully supervised models, our graph SSL models (denoise and completion in particular) outperform Spectrogram-VGG, and DaliNet [197]. Our model also has significantly fewer learnable parameters. The fully supervised GCN model uses the same graph construction method as proposed and performs very close to the state-of-the-art. This demonstrates the effectiveness of our graph construction strategy. Fig. 5.2 shows the average precision (AP) per class for the AudioSet database. A high AP is achieved even for classes with fewer samples, e.g., *church-bell* has an AP of 0.843 even with only 627 samples. This suggests that our model is not highly affected by

a lower number of samples owing to how we construct the sub-graphs during the training process.

5.4 Evaluation

In this section, we evaluate our method on semi-supervised speech emotion recognition.

5.4.1 Datasets

For this task, we use the two most popular speech emotion datasets.
The **IEMOCAP** [160] dataset contains 12 hours of speech collected over 5 dyadic sessions with 10 subjects. It includes 4490 utterances with labels, 1103 *anger*, 595 *joy*, 1708 *neutral* and 1084 *sad*.
The **MSP-IMPROV** [162] contains 7798 utterances from 12 speakers collected across six sessions including 792 samples for *anger*, 3477 for *neutral*, 885 for *sad* and 2644 samples for *joy*. The train-test split is the same.

5.4.2 Feature Encoder

Following relevant past works on speech emotion analysis [163], we extract a set of LLDs from the speech utterances using the OpenSMILE library [164]. This feature set includes MFCCs, zero-crossing rate, voice probability, fundamental frequency (F0) and frame energy. For each audio sample, we use a sliding window of length 25ms (with a stride length of 10ms) to extract the LLDs. The local features are smoothed temporally using a moving average filter, and the smoothed version is used to compute their respective first-order delta coefficients. Then we compute the mean, max, standard deviation, skew and kurtosis of the extracted LLDs and their delta coefficients to compute one feature vector per speech sample. Altogether this yields node embeddings of dimension 165 for an audio clip.

5.4.3 Experimental Settings

We follow the same settings as in the acoustic event detection task with a change in the graph size. The input graph used for this task has 53 nodes, where $M_s = 5$, $N_s = (12 \times \text{number of classes})$, and $T = 4$. Further discussion on the effect of graph size is presented in the next section.

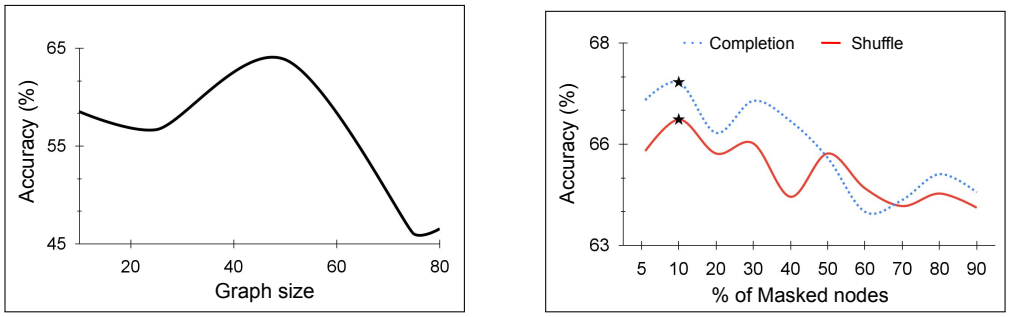
5.4.4 Results

Table 5.2 compares our model against the state-of-the-art supervised, non-graph semi-supervised, and non-graph self-supervised methods on the two speech

Table 5.2: Speech emotion recognition results: Classification (unweighted) accuracy (in %) on two benchmark databases are presented. Our model, using only 10% labelled data, outperforms semi-supervised and several fully supervised models on both databases. The fully supervised version of our model produces the highest accuracy even without SSL, indicating the effectiveness of our subgraph-based learning strategy. (* indicates audiovisual models).

Model	IEMOCAP	MSP-IMPROV	**Param**
Semi-supervised			
Ours w/o SSL	63.8 ± 2.2	58.6 ± 1.8	212K
Ours w/ denoise	68.0 ± 1.1	64.1 ± 1.0	271K
Ours w/ completion	66.4 ± 1.7	63.8 ± 1.5	271K
Ours w/ shuffle	65.9 ± 1.4	64.1 ± 1.3	213K
Ours w/ all three SSL	$\mathbf{68.6} \pm 1.2$	$\mathbf{65.2} \pm 1.8$	272K
LadderNet [199]	60.7	-	-
Transformer* [200]	61.2	-	-
SimCLR [201]	65.1	-	30M
Self-supervised			
SSAST [202]	59.6	-	89M
BYOL-S/CvT [203]	65.1	-	5M
wav2vec 2.0 [204]	65.6	-	317M
HuBERT [204]	67.6	-	316M
Fully supervised			
Ours w/o SSL	**70.5**	**66.7**	212K
SegCNN [153]	64.5	-	-
GA-GRU [205]	63.8	55.4	-
CNNattn [206]	66.7	-	-
WADAN [207]	64.5	-	-
SpeechGCN [1]	62.3	57.8	30K

databases. Clearly, our method (with denoise SSL in particular) outperforms others by a significant margin, using only 10% of the training data as labelled. Again note that the fully-supervised version of our model yields state-of-the-art result. Furthermore, as compared to non-graph self-supervised methods, our method provides comparable or higher results (especially with denoise and all three SSL) with a substantially reduced number of parameters. The results show that SSL with subgraph is an effective learning framework for speech classification even when labelled data is highly limited.



(a) (b)

Figure 5.3: (a) Impact of graph size on audio classification performance on the IEMOCAP database. (b) Effect of the fraction of masked nodes for the SSL tasks. The results shown are for the IEMOCAP database. The $\star$ indicates the best performance.

5.5 Model Analysis and Ablation Studies

5.5.1 Why Subgraph Instead of a Large Graph

To investigate whether subgraphs indeed produce superior performance, we experimented with a single big graph that includes all training samples simultaneously. When compared with the semi-supervised audio node classification task on IEMOCAP, the large graph achieves only a 49.44% recognition accuracy which is much lower than the accuracy (63.84%) achieved using the subgraphs-based learning framework, both without SSL. An intuitive explanation of the superior performance of subgraphs for audio classification: Since the graph structure is not given, constructing smaller graphs with balanced class samples introduces less error (as compared to a large graph) as subgraphs limit the number of unlabeled nodes per graph.

Fig. 5.3(a) shows how the classification performance varies with graph size (number of nodes). We observe that initially, the recognition accuracy improves as the graph size increases (up to size 50), but then starts decreasing. Overall, our observation is that subgraphs are more effective than using larger graphs in settings where we have limited labelled and a large amount of unlabeled training data.

5.5.2 Number of Masked Nodes in SSL Tasks

For the completion and the shuffle SSL tasks, we select a fraction of nodes (masked nodes) within a subgraph to apply the transformation. Fig. 5.3(b) shows how the fraction of masked nodes affects the classification performance. We observe that, in general, the recognition accuracy drops as masked nodes are increased beyond 10%. Following this observation, we mask 10% of the nodes in a subgraph during the SSL task.

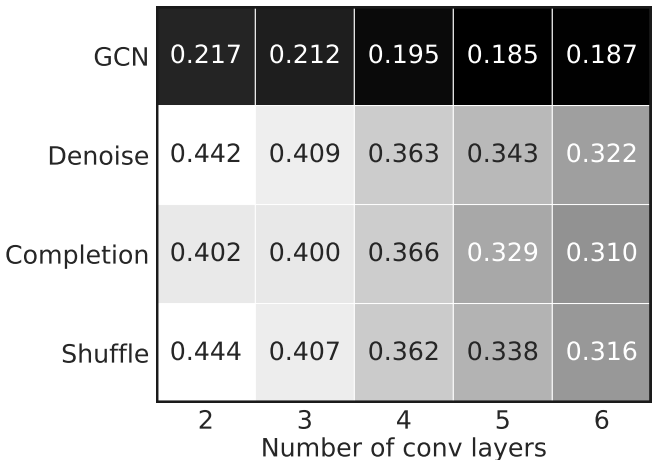


Figure 5.4: Oversmoothness analysis for IEMOCAP dataset: Measured in terms of mean average distance (MAD), is significantly smaller for our SSL-based models compared to GCN models. Darker colour indicates smaller MAD values, i.e., higher oversmoothing. We observed a similar trend for other datasets studied in our work.

5.5.3 Oversmoothness

GCN models are known to suffer from oversmoothness as the number of layers increases [208]. We investigate whether our model benefits from SSL in mitigating oversmoothness. A quantitative metric for measuring oversmoothness is Mean Average Distance (MAD). It measures the average distance from a node to all other nodes [208]. Using MAD, we experimented with a varying number of graph convolution layers on the IEMOCAP database. Fig. 5.4 shows that SSL-based models are more resilient to oversmoothness, producing more distinguishable node embeddings. This is clear from the average distances being much larger compared to the no SSL case. A similar trend is observed in other datasets.

Table 5.3: Model robustness against noise: Performance changes in recognition accuracy (in %) for noisy speech vs. clean speech. The results shown are for the IEMOCAP database where ↓ indicates a drop in performance.

	Speech noise types				
	Babble	Factory	White	Pink	Cockpit
Ours w/o SSL	2.8 ↓	1.6 ↓	3.1 ↓	2.0 ↓	1.1 ↓
Ours w/ denoise	**0.5** ↓	1.2 ↓	**0.5** ↓	**0.9** ↓	**0.8** ↓
Ours w/ completion	0.7 ↓	1.1 ↓	0.8 ↓	1.3 ↓	1.6 ↓
Ours w/ shuffle	0.8 ↓	**0.8** ↓	1.0 ↓	1.5 ↓	0.9 ↓

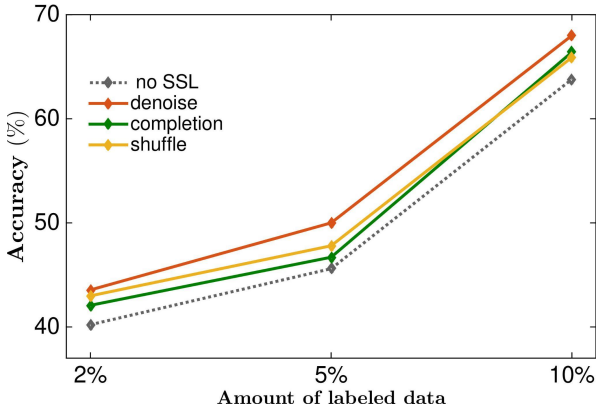


Figure 5.5: Performance using less than 10% labelled data: Our model performs reliably with SSL producing consistent improvement as labelled data becomes scarce; the denoise task performs the best (results shown for IEMOCAP).

5.5.4 Robustness Against Noise

To investigate the robustness of our model against noisy data, we experiment with six common noise types that may corrupt speech data: babble, factory, white, pink and cockpit noise [209]. Assuming additive noise, the noisy mixtures are obtained by adding the noise (one type at a time) to each speech sample first and then using the feature encoders as usual. Noise is added only to the test samples during inference. We compare the performance of our model on noisy test data with that with clean test input in Table 5.3 for the IEMOCAP dataset. Clearly, SSL provides significant robustness against noise as their drop in performance is consistently smaller compared to GCN semi-supervised results without SSL.

5.5.5 Reducing Labeled Data Further

We further investigate how the subgraph strategy holds up to even more scarce labelled data and whether or not the SSL tasks help. Fig. 5.5 shows the performance of our model with 2% and 5% labeled data in addition to 10% on the IEMOCAP database. We note that SSL tasks bring consistent improvement across all cases, where the denoise task performs the best.

5.6 Conclusion

Our work contributes to the understanding of semi-supervised audio representation learning - a relatively understudied topic in the acoustics and speech community. We developed a subgraph-based SSL framework for audio representation learning with limited labelled data. We make use of graphs to capture the underlying information in the unlabeled training samples and their relationship with the labelled samples. To this end, we proposed an

effective subgraph construction technique and two new graph SSL tasks. Our framework is generic, and can effectively handle both speech and non-speech audio data. Our model could achieve comparable or better performance than fully supervised models, despite using only 10% of the labelled data. Since the graph structure in our task has to be constructed first, our model is currently not end-to-end learnable. This could be addressed in future work where the graph structure itself is learned jointly with the embeddings. Furthermore, our current model relies on pre-trained embeddings, which gives the flexibility of choosing any suitable embeddings. Nevertheless, our model can be made end-to-end trainable which will be addressed in a future work.

In the next chapter, we step ahead, and multimodality will be explored to determine whether it is advantageous for learning audio representations.

Chapter 6

Visually-aware Graph-based Audio Representation Learning

6.1 Introduction

Audio perception by humans is inherently multimodal in nature. It involves processing both aural and visual cues. Visual cues are important not only for audio source localization [18], but also for improving audio perception [19]. Perceptual studies have also revealed that visual cues can even change how sound is heard [20].

The majority of existing works on learning audiovisual representations rely on maintaining a tight temporal synchrony between the visual and audio modalities [130–132]. Consider a scene of a bike moving away from the camera. The revving sound of the bike fades as it moves away. While an audio-only-based model may not be capable of detecting the fading sound as 'bike', taking into account the bike as a visual cue, it is possible to identify the event as 'motorbike running'. Computer vision-inspired models are common [133–135], where two augmented views of a given audio/audiovisual sample are fed to a shared 'backbone', followed by optimizing a contrastive loss [136–140], distillation [140, 141], quantization [130] or information maximization [142, 143]. However, the vision-inspired audio representation learning methods do not take full advantage of the temporal information available in video data or the complementary knowledge between modalities. Another difficult aspect of such approaches is that data augmentation functions, being vision-inspired, are not often well-suited to a multimodal input.

Heterogeneous graphs are a compact, efficient, and scalable way to represent data involving multiple different entities and their relations [144, 145]. Modelling the interaction of entities (including modalities) with heterogeneous graphs

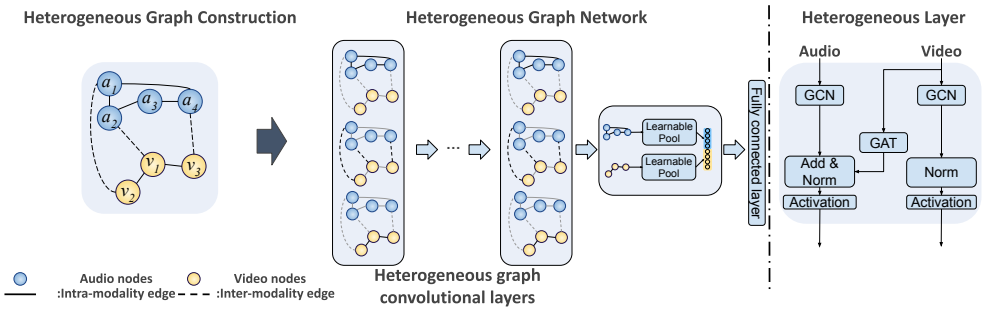


Figure 6.1: (*Left*) **Heterogeneous graph architecture**. We split the input video clip into Q and P overlapping segments and then construct the heterogeneous graph containing intra- and inter-modality edges between nodes. Each edge type is considered and processed by the corresponding GNN. For both audio and video modalities, heterogeneous graph convolution layers are utilised to extract the embedding for each node. Separate learnable pooling modules are then used to capture the overall graph representation. (*Right*) **Heterogeneous graph layer** has two independent audio and video flows taking into account intra-modality edges, as well as an attention layer connecting video nodes to audio nodes considering inter-modality edges.

is a relatively new paradigm. Multimodal heterogeneous graphs have been successfully used to address various problems in computer vision and natural language processing, such as visual-question answering [146], multimedia recommendation [144, 147], audio-visual sentiment analysis [148], and cross-modal retrieval [145]. Multimodal heterogeneous graphs lead to a closer coupling between concepts in multiple modalities, resulting in a significant performance improvement over previous methods [144, 146–148]. Motivated by the success of graph-based methods in multimodal problems, we propose a heterogeneous graph-based approach to learn visually-aware audio representations.

In this chapter, we propose a visually-aware audio representation learning approach based on heterogeneous graphs (see Fig.6.1 for an overview) in the context of acoustic event detection. Our heterogeneous graph model creates a shared space for audio and visual modalities that takes advantage of their spatial and temporal relationships explicitly. We first model the input audiovisual clip as a heterogeneous graph with two sub-graphs, one for each modality with edges capturing inter- and intra-modality relationships. We next develop a heterogeneous graph neural network which is able to capture rich audio representation incorporating complementary information from the visual information. Our contributions are as follows:

- We develop a graph construction method for converting an audiovisual clip to a multimodal heterogeneous graph.
- We propose a novel heterogeneous graph neural network (HGNN) that can capture modality-specific information as well as complementary

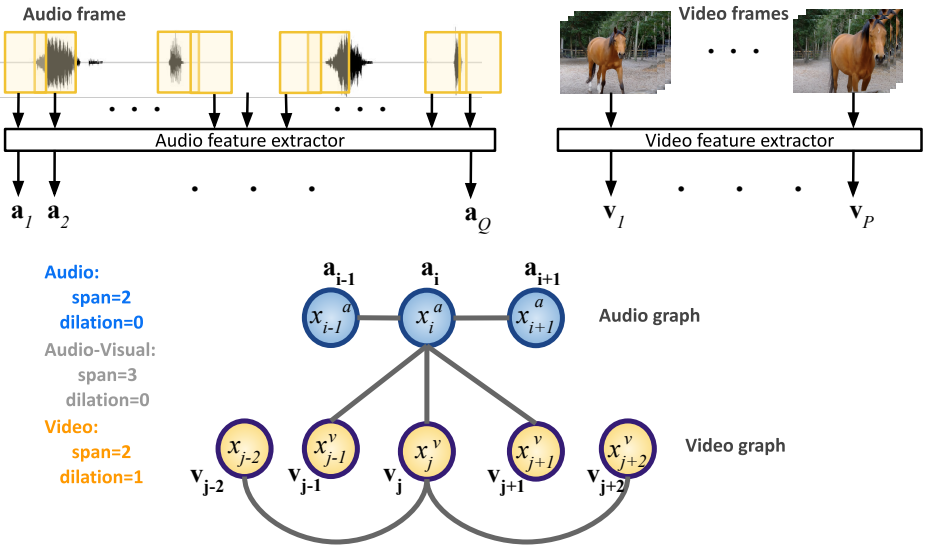


Figure 6.2: Heterogeneous graph construction process. For simplicity, the edges are only shown for v_i and v_j. Similar connections are added for each node.

information between modalities.

- We demonstrate improved performance by our model for the task of acoustic event classification on the large benchmark AudioSet dataset.

6.2 Proposed Method

This section describes our proposed approach for visually-aware audio representation learning. First, we construct heterogeneous graphs to represent the audiovisual data consisting of modality-specific subgraphs and inter-modality edges. Next, we propose a heterogeneous graph neural network (HGNN) architecture that performs graph classification in the context of acoustic event detection.

6.2.1 Heterogeneous Graph Construction

Our first task is to construct a heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, O, R)$, where $\mathcal{V}$ represents the set of nodes, $\mathcal{E}$ the set of edges, O is the set of node types (object/modality), and R is the set of edge types, where $|O| + |R| > 2$ Each node $v \in \mathcal{V}$ is associated with a node type and each edge $e \in \mathcal{E}$ is associated with an edge type.

Given an audiovisual input, we uniformly divide the video frames and the audio into P and Q segments (see Fig.6.2). The segments are used for feature extraction. Then, given the video and the audio segments, we construct a heterogeneous graph with node sets $\mathcal{V}^v = \{v_i\}_{i=1}^{P}$ and $\mathcal{V}^a = \{a_i\}_{i=1}^{Q}$, with edge sets $\mathcal{E} = \{\mathcal{E}_{vv}, \mathcal{E}_{aa}, \mathcal{E}_{va}\}$, which represent edges between video-only nodes, audio-

Figure 6.3: AudioSet sample video. An ambulance starts moving, sirening, and the camera tracks it.

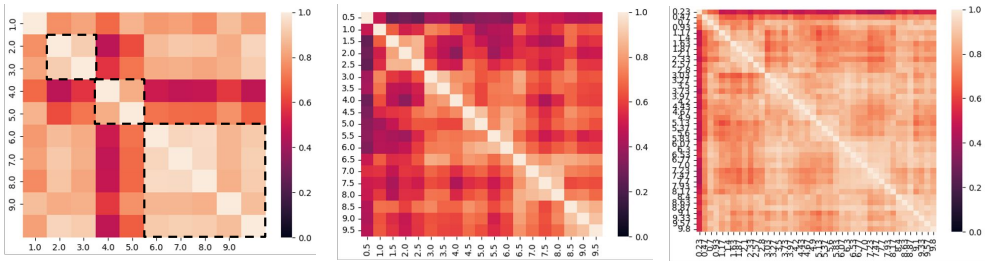


Figure 6.4: Correlation matrices. We compute the correlation matrices for the ambulance video data (Fig. 6.3) in three different time resolutions: 1, 0.5, and 0.25 second segments. Dashed lines show the pattern in (a).

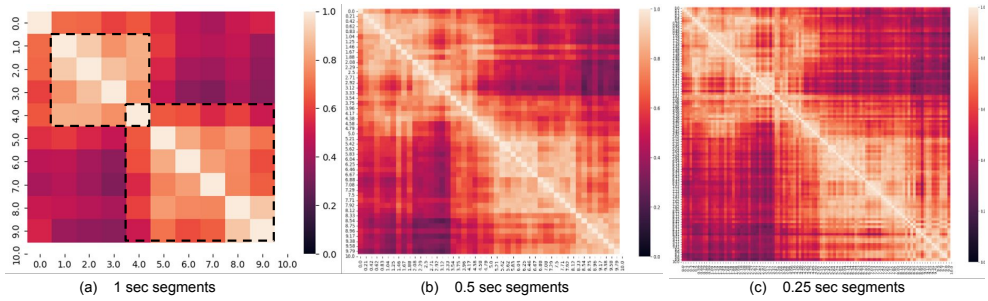


Figure 6.5: Correlation matrices. We compute the correlation matrices for the ambulance audio data (Fig. 6.3) in three different time resolutions: 1, 0.5, and 0.25 second segments. Dashed lines show the pattern in (a).

only nodes, and between audio-video nodes respectively. These corresponding adjacency matrices are denoted as $\mathbf{A}_v$, $\mathbf{A}_a$, and $\mathbf{A}_{va}$. Each node $v_i \in \mathcal{V}^v$ corresponds to a video segment and its associated feature vector is $\mathbf{x}_i^v \in \mathbb{R}^{d_v}$. Similarly, an audio node $a_i \in \mathcal{V}^a$ is associated with feature vector $\mathbf{x}_i^a \in \mathbb{R}^{d_a}$. Since the graph structure is not naturally defined here, we propose to add inter- and intra-modality edges (see Fig. 6.2). Additionally, Our graph has two parameters for each edge type, i.e, for $\mathcal{E}_{vv}, \mathcal{E}_{aa}, \mathcal{E}_{va}$: (i) *span across time* and (ii) *dilation*. The former denotes the number of nodes connected to each node in the temporal direction, whereas the latter denotes leaps between nodes. In total, we have six hyperparameters for graph construction.

Inspired by SlowFast [210], we further investigate our data by segmenting and visualizing the correlation matrices in various time resolutions. To do so,

Figure 6.6: AudioSet sample video. An agent switches on and off, and an alarm sounds.

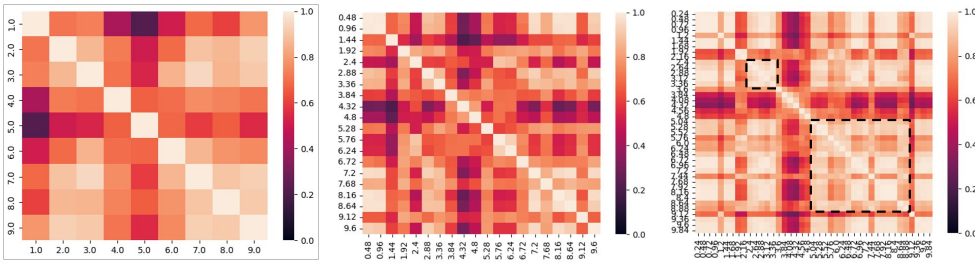


Figure 6.7: Correlation matrices. We compute the correlation matrices for the alarm video data (Fig. 6.6) in three different time resolutions: 1, 0.5, and 0.25 second segments. Dashed lines show the pattern in (c).

we picked two data samples from AudioSet. The first (Fig. 6.3) is a video clip showing a fast-moving ambulance with blue lights blinking and sirening. The second (Fig. 6.6) indicates a switch and an agent's hand turning it on or off; as a result, an alert sounds. The audio and video modalities are segmented with different time resolutions and the results are shown in Fig. 6.4, 6.5, and 6.7. As the patterns are indicated with dashed squares, we can observe that the fast actions will produce patterns in lower resolution in time (1 sec segments) both in the video (Fig. 6.4) and audio (6.5) modality.

Even if we specified two hyperparameters for each modality while building our graph, choosing them would be a heuristic process and not be ideal because we saw that each data requires a distinct set of connections in order to capture the proper representation. To mitigate this effect, we propose to choose these hyperparameters randomly from a fixed range. This solution has two implications. First, it enables the graph's nodes to have connections with varying ranges that can accommodate both long- and short-range dependencies. It also acts as a technique for augmentation. Because no graph will manifest over training epochs with the same connection, which is equivalent to the augmentation method is CV.

6.2.2 Heterogeneous Graph Neural Network (HGNN)

Given heterogeneous graphs $G_1, ..., G_N$ and their ground-truth labels $\mathbf{y}_1, ..., \mathbf{y}_N$, the task is to learn a d-dimensional graph representation $\mathbf{h}_{G_i} \in \mathbb{R}^d$ that captures rich structural and semantic information in G_i.

The key idea of most GNNs is to aggregate feature information from a node's neighbours and then update the node feature vector:

$$\mathbf{H}^{(k+1)} = \sigma\big(\mathbf{A}\mathbf{H}^{(k)}\mathbf{W}^{(k)}\big) \tag{6.1}$$

where $\mathbf{W}^{(k)}$ is the weight matrix for the k^{th} layer of the GNN, σ is a non-linear activation function, such as ReLU, and k is the layer number ($k = 0, \cdots K$). Because of the various node and edge types, this approach is not directly applicable to our heterogeneous graphs. Previous studies utilise meta-paths for processing heterogeneous graphs [211, 212], which has been shown to be inadequate to properly exploit the information provided by node and edge types [213]. To overcome this, we use separate GNNs for processing different edge types.

Our HGNN has three flows of information corresponding to the intra- and inter-modality edges, as shown in Fig. 6.1. Audio and video flow process the audio and video nodes by considering only intra-modality edges ($\mathcal{E}_{vv}, \mathcal{E}_{aa}$) between audio and video nodes, respectively. The third flow carries audio-related information from video nodes to audio nodes for the inter-modality edges ($\mathcal{E}_{va}$):

$$\begin{aligned} \mathbf{x}^a_{l+1} &= \mathrm{GNN}_{\theta_1}\big(\mathbf{x}^a_l, \mathbf{A}_a\big) + \mathrm{GNN}_{\theta_2}\big(\mathbf{x}^v_l, \mathbf{A}_{va}\big) \\ \mathbf{x}^m_{l+1} &= \mathrm{GNN}_{\theta_3}\big(\mathbf{x}^v_l, \mathbf{A}_v\big) \end{aligned} \tag{6.2}$$

where $\mathbf{x}^a_l$,$\mathbf{x}^v_l$ and GNN are audio and video node features in layer l, and GNN is a graph-based neural network such as GCN [75].

Our objective is to classify entire graphs, as opposed to the more common task of classifying each node. Hence, we seek a *graph-level* representation $\mathbf{h}_G \in \mathbb{R}^d$ as the output of our network. This can be obtained by pooling the node-level representations $\mathbf{x}^a_K$, $\mathbf{x}^v_K$ at the K-th layer before passing them to the classification layer (see Fig. 6.1). Common choices for pooling functions in the graph domain are `mean`, `max`, and `sum` pooling [75]. Max and mean pooling often fail to preserve the underlying information about the graph structure, while sum pooling has been shown to be a better alternative [76]. However, all these pooling functions treat adjacent nodes with equal importance, which may not be optimal. To this end and following [2], we propose to *learn* a pooling function Ψ that combines the node embeddings from the K-th layer to produce an embedding for the entire graph. The pooling layer for each modality is thus

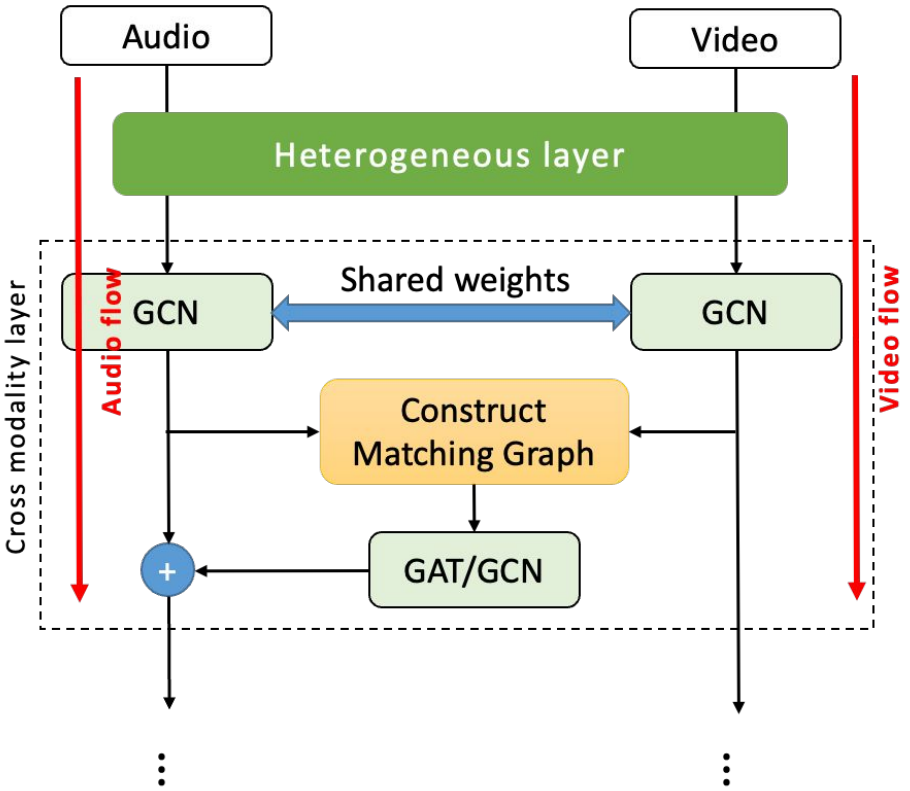


Figure 6.8: Cross-modality graph layer. This layer ensures that if the long-term dependencies are missing from the heterogeneous graph we constructed, they are taken into consideration in our model.

defined as follows:

$$\mathbf{h}_G = \left[\Psi_n(\mathbf{x}_K^a) \,|\, \Psi_m(\mathbf{x}_K^v)\right] = \mathbf{x}_K^a \mathbf{p}^a + \mathbf{x}_K^v \mathbf{p}^v \tag{6.3}$$

where $\mathbf{p}^a$ and $\mathbf{p}^v$ are learnable weights to combine node-level embeddings to obtain a graph-level embedding for audio and video nodes. The overall heterogeneous graph network is trained with a focal loss $\mathcal{L}$:

$$\mathcal{L} = -\sum_n \alpha(1-\mathbf{y}_n)^\gamma \log \tilde{\mathbf{y}}_n. \tag{6.4}$$

6.2.3 Graph Cross Modality Layer

We can further enhance the prominent modality in our graph by proposing a cross-modality layer. Our heterogeneous graph construction only relies on two hyperparameters which enables two modalities to connect to the neighbouring nodes in both space and time directions. But in real-world scenarios, we might have a relationship between modalities for a long distance in time. Thus, we propose a cross-modality layer in Fig 6.8. We use this layer to discover

prolonged relationships across modalities following a heterogeneous layer. All of the modalities are mapped in the shared-knowledge space using a shared GCN layer. Then, using the node features, a matched graph is built. The KNN graph, in which each node connects to the K-closest nodes from the opposite modalities, is a widely popular method. This graph is processed by a GNN network after graph construction. This assures that the prolonged information will be stored in the dominant modality by updating the processed node features from the dominant modality (in this case, audio).

6.3 Evaluation

In this section, we first discuss the dataset used for benchmarking and feature extraction details. We then present experimental results and analysis to evaluate the performance of the proposed HGNN architecture on the acoustic event classification problem.

6.3.1 Dataset

We use a large scale weakly labelled dataset **AudioSet** [194], which contains audio segments from YouTube videos. We work with 33 categories from the balanced set that have a high rater confidence score ($\{0.7, 1.0\}$). This yields a training set of 82,410 clips. For a fair comparison with baseline methods, we also use the original evaluation set, which has 85,487 test clips.

6.3.2 Feature Encoder

Audio Encoder. To extract the audio node features, each audio clip is divided into 960 ms segments with 764 ms overlap. For each segment, a log-mel spectrogram is computed by taking its short-time Fourier transform using a frame of 25 ms with 10ms overlap, 64 mel-spaced frequency bins, and log-transforming the magnitude of each bin. This creates log-mel spectrograms of dimensions 96×64, which are the input to the pre-trained VGGish network [195]. We use the 128-dimensional features extracted by the VGGish network for each log-mel spectrogram.

Video Encoder. Each video is segmented into non-overlapping 250 ms chunks to extract the video node features. The 1024-dimensional feature is then obtained by feeding each segment into an off-the-shelf 3D convolution network, S3D [214] (trained with self-supervision[215]). Note that our method is not limited to these pre-trained embeddings and can work with any generic embeddings for both audio and video.

6.3.3 Implementation Details

Each video clip produces a heterogeneous graph with $P = 40$ audio and $Q = 100$ video nodes, where each node corresponds to a 960 ms length audio or 250 ms length video segment. In all experiments, 4 heterogeneous layers are used, with a 512 embedding size for heterogeneous graph layers. We repeat our experiments 10 times with different seeds and report both mAP (mean average precision) and ROC-AUC (area under the ROC curve) values. Our network weights are initialized following the Xavier initialization. We used Adam optimizer with a learning rate of 0.005, a decay rate of 0.1 after 1500 iterations, and 1000 warm-up iterations for all experiments. We set $\alpha = 0.5$ and $\gamma = 2$ (see Eq. 6.4). The graph construction hyper-parameters are explored heuristically and set to *span audio* $= 6$, *dilation audio* $= 3$, *span video* $= 4$, *dilation video* $= 4$, *span audio-visual* $= 3$, and *dilation audio-visual* $= 1$ for all experiments. We use Pytorch on an NVIDIA RTX-2080Ti GPU.

6.3.4 Results and Analysis

6.3.4.1 Baselines

We compare our method with a number of fully and self-supervised models, as tabulated in Table 6.1. The Spectrogram-VGG model is the same as configuration A in [196], with only one change: the final layer is a softmax with 33 units. The feature for each audio input to the VGG model is a log-mel spectrogram of dimensions 96×64 computed by averaging across non-overlapping segments of length 960ms. We also compared our method with a graph-based work. Each node in this chapter represents an audio clip, and a KNN subgraph has been created, as well as a GNN that is trained using graph self-supervised proxy tasks [3]. We also use the two popular spatial and temporal network architectures, ResNet-1D [216] and LSTM, with pretrained embedding features for both audio and video as input, to further investigate the superiority of our graph modelling. All baseline hyper-parameters are set to the values published in the original papers. Note that we do not utilise any data augmentation, despite the fact that other methods used powerful data augmentations.

6.3.4.2 Results

Table 6.1 reports the mAP and ROC-AUC (averaged over 10 runs with different seeds) values with standard deviation for each model and their variants. It compares the performance of our model with different independent modalities and strong baselines with that of the heterogeneous model in terms of mean Average Precision. The heterogeneous graph model outperforms the homogeneous

Table 6.1: Acoustic event detection results on **AudioSet**

Model	**mAP**	**ROC-AUC**	**Params**
Ours audio only	0.42 ± 0.01	0.90 ± 0.00	1.4M
Ours video only	0.15 ± 0.02	0.75 ± 0.01	1.5M
Ours	$\mathbf{0.50} \pm 0.01$	0.93 ± 0.00	2.1M
Ours*	$\mathbf{0.51} \pm 0.02$	0.93 ± 0.03	2.5M
Baselines			
ResNet-1D	0.38 ± 0.03	0.89 ± 0.02	81.2M
ResNet-1D audio only	0.35 ± 0.01	0.90 ± 0.00	40.4M
LSTM audio only	0.40 ± 0.00	0.90 ± 0.00	0.8M
State-of-the-art			
DaiNet [197]	0.25 ± 0.07	-	1.8M
Spectrogram-VGG	0.26 ± 0.01	-	6M
VATT [198]	0.39 ± 0.02	-	87M
SSL graph [3]	0.42 ± 0.02	-	218K
Wave-Logmel [45]	0.43 ± 0.04	-	81M
AST [48]	0.44 ± 0.00	-	88M

graph and non-graph models. Our method leverages the pre-trained features as node attributes. Thus, to check the performance of our graph-based model, two strong baselines, ResNet-1D and LSTM, have been selected. Compared to these methods, our homogeneous graph sub-models achieve a superior mAP score that demonstrates the effectiveness of our graph-based modelling strategy. Furthermore, when compared to other baselines, our heterogeneous graph-based model achieves the greatest ROC-AUC score (0.93), implying more trustworthy predictions at various thresholds. When compared with the other supervised models, our heterogeneous graph model outperforms Spectrogram-VGG and DaliNet [197]. Our model also has significantly fewer learnable parameters compared with the recent transformer-based architectures, VATT and AST.

6.3.4.3 Ablation Experiments

We perform exhaustive ablation experiments to investigate the contribution of each component we propose to build our heterogeneous graph neural network. Table 6.2 presents the ablation results on the AudioSet dataset. We observe that each new component brings improvement. In all experiments, model performance is measured with mAP to quantify the recognition rate. The introduction of the heterogeneous graph increases the recognition rate by about 9%; when combined with our new graph attentional convolution layer between modalities (right half of Fig. 6.1), the performance increases to 0.49. Adding the learnable pooling brings up the mAP score to 0.50. Removing the learnable pooling, however, reduces the performance by about 3% and 1% for audio-only

and video-only models, respectively. The ablation results show that each of the proposed components in our architecture is important and contributes positively towards the overall model performance.

Table 6.2: Ablation experiments on the AudioSet dataset. Each new component in our heterogeneous network contributes towards its performance. Note that these results are from the models without utilising cross-modality layers.

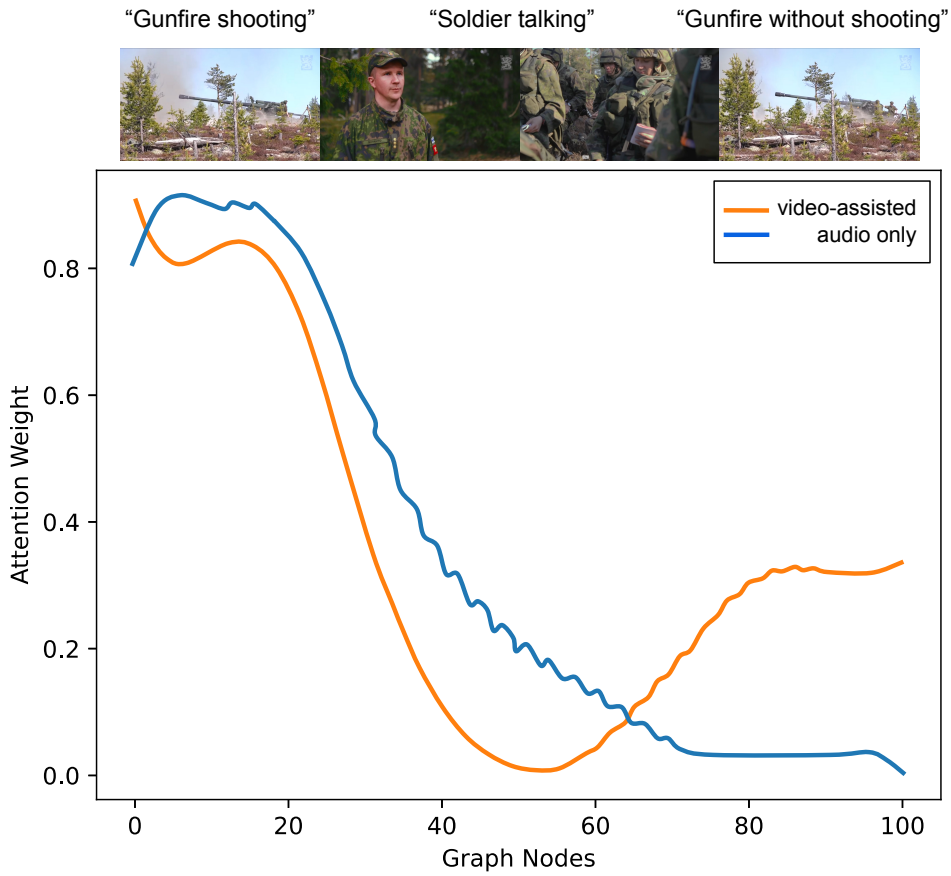
Audio	**Video**	**Attn**	**Learned p**	**mAP**
✓	-	-	-	0.38
✓	-	-	✓	0.41
-	✓	-	-	0.12
-	✓	-	✓	0.13
✓	✓	-	-	0.49
✓	✓	✓	-	0.49
✓	✓	✓	✓	**0.50**

6.3.4.4 Qualitative Results

We display how our model attends to different nodes to gain insights into its learning process. Because each video clip is divided into 100ms segments, each node represents a 100ms time window. In Fig. 6.9, 6.10, and 6.11, we show the attention weights corresponding to audio nodes in cases of with and without video supervision for three input videos from the test set with distinct acoustic classes. Then, for each video, we sample four frames and display them on top of each figure to provide more visual information. This gives rise to *salient* nodes for each input. The results show that the proposed model can extract visually complementary information to an audio event from heterogeneous graphs as input.

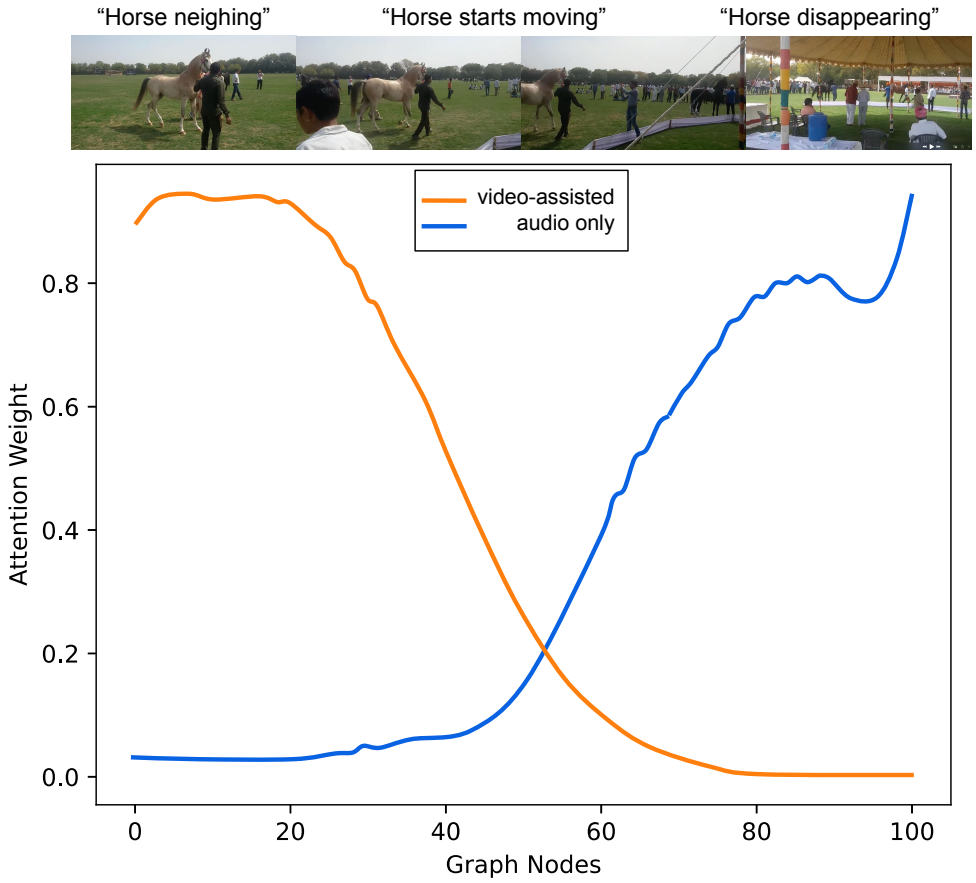
6.3.4.5 Add More Classes

We further push the complexity of the problem by increasing the number of classes gradually. Table 6.3 reports the performance of our model in a different number of classes in the training set. The performance decreased in mAP and ROC-AUC as we added more classes. We suggested using 33 classes with higher rater confidence scores which implies that those classes have lower annotation noise. One of the reasons our model can't maintain high accuracy could be that adding more classes allows the data to get noisier and mislead the model into learning inappropriate features. Another argument is that we only use the pre-trained features, which may not be the best way to provide our model with enough data to produce adequate discriminative power in the classification task.



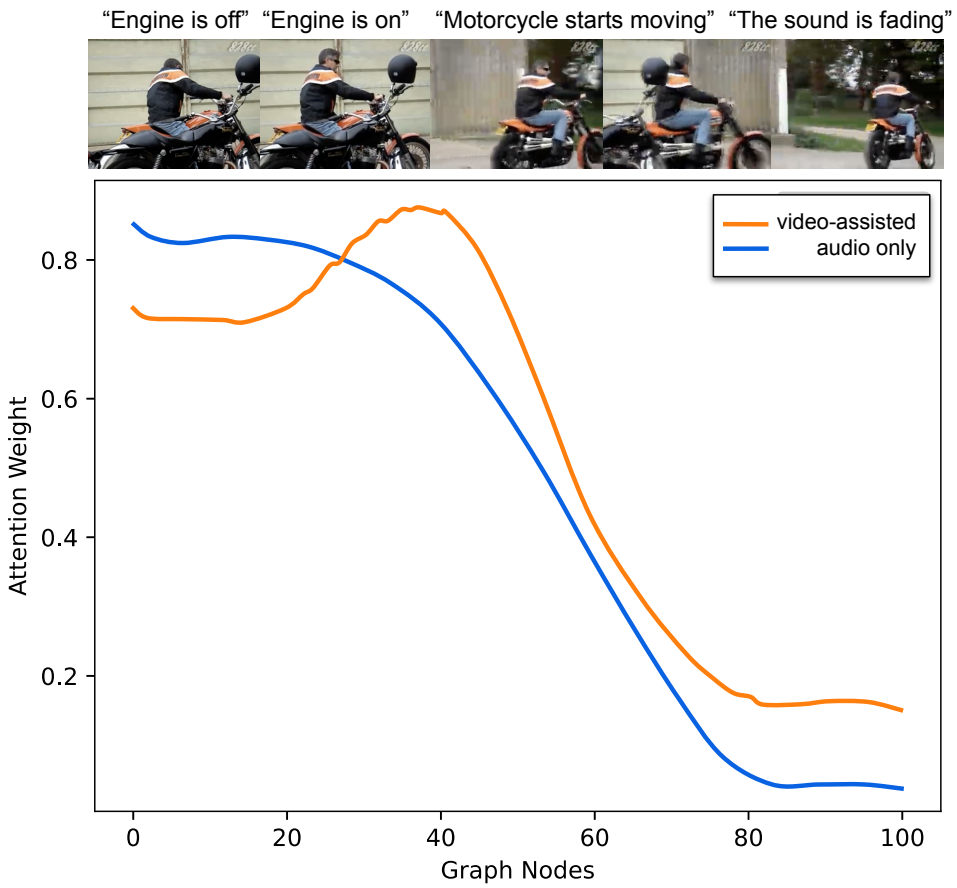
(a) Gunshot, gunfire

Figure 6.9: Qualitative results showing attention weights corresponding to the audio nodes for with (in blue) and without (in orange) video supervision. Each node represents a segment of 100-millisecond duration, and the ground-truth label for each video is provided above. Attention values were normalized and re-scaled to [0,1] range. This video begins with a strong machine firing sound. After that, a soldier is interrogated, followed by footage of troops. Finally, the machine appears again but is not fired. Even without the associated sound of shooting, the video-assisted audio nodes are able to recognise the firing machine towards the end by assigning higher attention weights to these moments.



(b) Horse neighing

Figure 6.10: Qualitative results. A horse begins neighing and moves away from the camera. As it moves, the sound fades. The horse is no longer visible or audible as time elapses. The audio-only model incorrectly detects these moments by assigning high attention values, while the video-assisted model correctly discards these moments.



(c) Motorcycle starting

Figure 6.11: Qualitative results. A video of a motorcycle moving. The video-assisted attention weights suggest that our model can capture additional meaningful patterns, such as the engine start. Furthermore, as the engine sound fades, the attention weights corresponding to the audio-only model decrease, and the video-assisted attention weights have relatively higher values indicating that the video information extracted by our model is complementary to the audio event information.

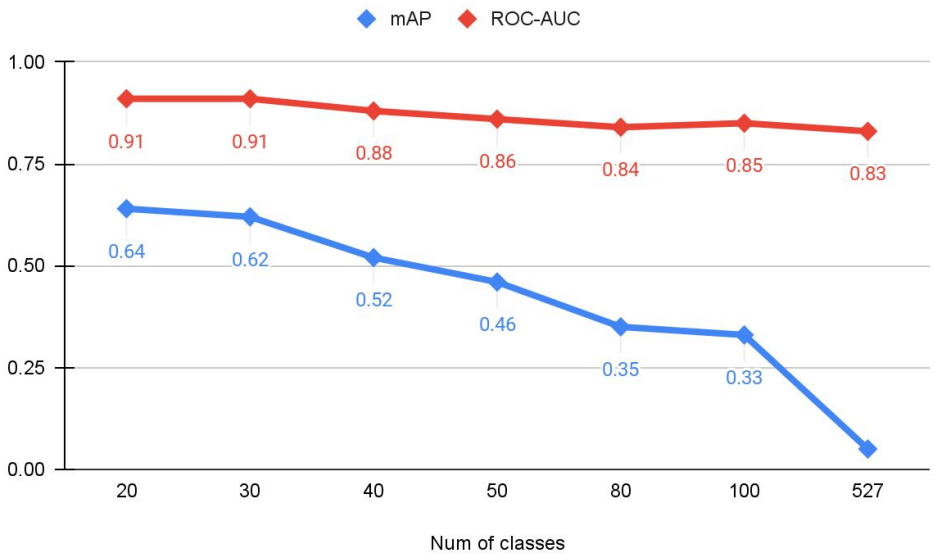


Figure 6.12: Increasing number of classes. Our model performs reliably good in a low number of classes, and as we increase the number of classes, the performance drops in both mAP and ROC-AUC

Table 6.3: Increasing number of classes. As we increased the number of classes in the training set, the number of trainable parameters and samples in the training set increased.

Number of classes	Parameters (M)	Number of samples (K)
20	2.19	1.08
30	2.2	1.79
40	2.21	2.31
50	2.22	2.85
80	2.25	5.56
100	2.27	6.76
527	2.71	39.83

6.3.4.6 Different Number of Multimodal Nodes

To further explore the power of the heterogeneous graph modelling for multimodal data, we experimented with acoustic event classification problem with the different number of nodes in both audio and video modality. In the graph construction step, we segmented both audio and video clips into smaller overlapped chunks and then used a pretrained network to extract corresponding features. In this section, we increase the number of audio and video nodes in our heterogeneous graph starting at 10 to observe how our model behaves. Note that we've tested it on a single modality case, and the results are in Table 6.5.

Table 6.4: Increasing number of classes. As we increased the number of classes in the training set, the number of trainable parameters and samples in the training set increased.

Modality	Number of nodes	mAP	ROC-AUC
Audio	10	0.42	0.91
	48	0.43	0.91
	100	0.45	0.92
	130	0.46	0.92
Video	10	0.09	0.70
	40	0.21	0.81
	100	0.22	0.81

6.3.4.7 Static v.s. Dynamic Edges

As we noted in Section 6.2.1, choosing the span over time and dilation hyperparameters for graph construction is not optimal, hence we suggested sampling random hyperparameters within a given range. The results of our investigation into this statement are presented in Table 6.5. Table 6.5 shows how our random hyperparameter selection takes into account various graph architectures to improve model performance in terms of both mAP and ROC-AUC metrics. Tuning hyperparameters used to be a tedious and long process and, in our case, poses a fixed graph structure for each multimodal graph. Table 6.5's three first rows are the output of manually choosing the hyperparameters. The dynamic range for choosing such hyperparameters is displayed in the rows below. It has been found that recommending this fixed range will increase performance with a significant margin and need significantly less parameter searching.

Table 6.5: Dynamic vs. static edges. We experiment with selecting graph construction hyperparameters within a fixed range to allow dynamic graph construction for each data point.

Dynamic	audio		video		inter-modality	mAP	ROC-AUC
edges	Span over time	Dilation	Span over time	Dilation	Span over time		
-	2	1	1	1	2	0.42	0.83
-	2	2	3	1	2	0.48	0.88
-	10	3	5	5	5	0.49	0.90
✓	[1,3)	1	[1,4)	1	[1,3)	0.38	0.86
✓	[1,7)	1	[1,7)	1	[1,3)	0.43	0.92
✓	[1,7)	[1,3)	[1,7)	[1,3)	[1,3)	0.47	0.92
✓	[1,7)	[1,4)	[1,3)	[1,4)	[1,3)	0.49	0.94
✓	[1,11)	[1,11)	[1,6)	[1,6)	[1,6)	0.54	0.94

6.4 Conclusion

In this chapter, we introduced the idea of heterogeneous graphs to model audio data with visual cues. We proposed a compact and efficient graph-based architecture that learns audio representations effectively in the context of acoustic event detection. We transformed an audiovisual input to a heterogeneous graph with different learnable hyper-parameters capturing intra and inter modalities connections in both spatial and temporal domains. Our heterogeneous graph model produces higher or comparable performance to the state-of-the-art on a popular benchmark dataset, the AudioSet. Our current model relies on pre-trained embeddings, which gives the flexibility of choosing any suitable embeddings. Nevertheless, our model can be made end-to-end trainable, which will be addressed as part of our future work.

Chapter 7

Conclusion and Future Work

In this thesis, we have proposed a set of algorithms for learning representation from audio data in the form of a graph. First, we explored how audio data could be modelled using a graph structure. In follows, we suggested learning the graph structure together with the major loss because the graph structure is not natural. As follow-up research, we created a graph self-supervised framework and presented three primary self-supervised tasks that can be used for any type of graph data or task while taking into account the low amount of validated data situations. Finally, we added heterogeneous graph modelling to improve the graph representation in one modality. Allowing us to include several modalities in a single graph structure. In order to process our multi-modal graph, we also suggested a new heterogeneous graph architecture. In this chapter, we summarise our contributions and consider future work.

7.1 Summary of Contributions

- We first presented a method for combining ideas from graph signal processing to audio data. An efficient, compact, and scalable graph-based model is introduced to process the constructed graph. This model achieved superior performance compared to the popular graph neural network by applying exact convolution operation in the graph domain.

- In Chapter 4, we further extended our idea and proposed a dynamic graph structure for audio data, which is learnt during training alongside the main objective. In this chapter, we propose a generalized graph approach that can take any time-varying (dynamic) data modality as input. To this end, we present the *Learnable Graph Inception Network* (L-GrIN) that jointly learns to recognize emotion and to identify the underlying graph structure in data. Our architecture comprises multiple novel components: a new graph convolution operation, a graph inception layer, learnable adjacency, and a learnable pooling function that yields a graph-level

embedding. We tested this method on audio data in this section, but we tried and extend it to other modalities of data in Appendix A.

- In Chapter 5, we explore a self-supervised graph approach to learning audio representations from highly limited labelled data. Considering each audio sample as a graph node, we propose a subgraph-based framework with novel self-supervision tasks to learn effective audio representations. During training, subgraphs are constructed by sampling the entire pool of available training data to exploit the relationship between the labelled and unlabeled audio samples. During inference, we use random edges to alleviate the overhead of graph construction. We evaluate our model on three benchmark audio datasets spanning two tasks: acoustic event classification and speech emotion recognition. We show that our semi-supervised model performs better or on par with fully supervised models and outperforms several competitive existing models. Our model is compact and can produce generalized audio representations robust to different types of signal noise.

- Finally, in Chapter 6, we employ heterogeneous graphs to explicitly capture the spatial and temporal relationships between the modalities and represent detailed information about the underlying signal. Using heterogeneous graph approaches to address the task of visually-aware acoustic event classification, which serves as a compact, efficient and scalable way to represent data in the form of graphs. Through heterogeneous graphs, we show efficiently modelling of intra- and inter-modality relationships both at spatial and temporal scales. Our model can easily be adapted to different scales of events through relevant hyperparameters. Experiments on *AudioSet*, a large benchmark, show that our model achieves state-of-the-art performance.

7.2 Future Work

Applying more insights into graph modelling. At first, we just thought of our proposed graph as a cycle and chain structure. We further explored it by trying to learn a dynamic structure for each graph in our data. However, prior knowledge can always strengthen our modelling easily. For instance, in our graph construction, we only take into consideration temporality as an a priori element, but there are additional task-based aspects that may be incorporated to transfer this knowledge into the graph structure and improve its realism.
Make an end-to-end model. While our current model relies on pre-trained embeddings, which gives the flexibility of choosing any suitable embeddings

given a task. Nevertheless, our model can be made end-to-end trainable which will be addressed in a future work.

Better self-supervised learning methods needed. In both our research and the GNN literature, we found that self-supervised learning can be useful. However, the performance improvement is not comparable to the CV and NLP. Studying this literature requires careful consideration of the possibility that other forms of data are lacking.

Make a multimodal graph. We've already demonstrated how using video modality can improve performance on tasks that are exclusive to audio. That's just a tiny hole that allows us to expand it to a more generic multimodal system. A variety of modalities would be included in our heterogeneous knowledge graph, and their connections would be represented by their edges. Then this graph can be used to do analysis in many tasks.

Generalized GNN. Following the most current DL methods, we trained every model in this thesis and for each application, to accept input graphs of the same size. GNNs can, however, process graphs of various sizes. The next step in this research would be to change the methodology for constructing input graphs to result in input graphs with various node sizes. In this manner, the GNN would learn to generalise on graphs of various lengths, which is equivalent to the input of varying lengths for the video or audio. This resolves the previously significant issue of input data of various sizes.

This thesis set out to expand on the state-of-the-art in audio representation learning. We have proposed graph-based models in different settings: supervised, semi-supervised, and self-supervised learning. All of the code and methods discussed in this thesis are available online at github.com/AmirSh15, which we hope will facilitate future research towards increasingly more sophisticated audio representation learning systems.

7.3 Broader Impact

Our work may be applied to classify speech and acoustics data. Therefore, it may be used to 'hear' and should be used carefully. We used public databases that are mostly balanced in terms of male and female subjects. However, they are not balanced considering factors such as ethnicity, language spoken and other demographic factors. The speech emotion analysis application uses only four archetypal expressions. We are aware that human emotion is far more complex and thus do not advocate the use of such systems for sensitive decision-making areas. We also note that automatic emotion classification from speech is an evolving area of research, and questions of fairness and universality remain to be explored.

Appendix A

First

To further explore our proposed method in Chapter 4, we now present extensive experimental results and analysis to evaluate the performance of the proposed method for facial and body emotion recognition.

A.1 Facial Emotion Recognition

A.1.1 Video Databases

We use three large video emotion recognition databases for our experiments. The databases are chosen based on their popularity in emotion recognition literature.

The **RML** database [32] contains 720 videos of 6 basic emotions: *anger, disgust, fear, joy, sadness*, *surprise* collected when the subjects speak. The subjects are from various ethnic groups and speak different languages.

The **eNTERFACE** [33] is contains 1170 videos of 42 subjects with six basic emotion classes as RML. These emotions are the reactions after listening to six different short stories, where each person reads out 5 phrases based on their emotional reaction.

The **RAVDESS** database [217] contains 4904 videos labeled with 8 classes: *anger, calmness, disgust, fear, joy, neutral, sadness* and *surprise*. This is the largest video emotion database currently available. Samples from each database are shown in Fig. A.2.

A.1.2 Node Features

The databases we use provide only raw video clips. We choose to use facial landmark points extracted from the video frames as node attributes. This is because landmark points are known to effectively capture the facial dynamics [218]. We extract 68 landmark points at every video frame using a state-of-the-art landmark detection method [219], resulting into node feature vectors

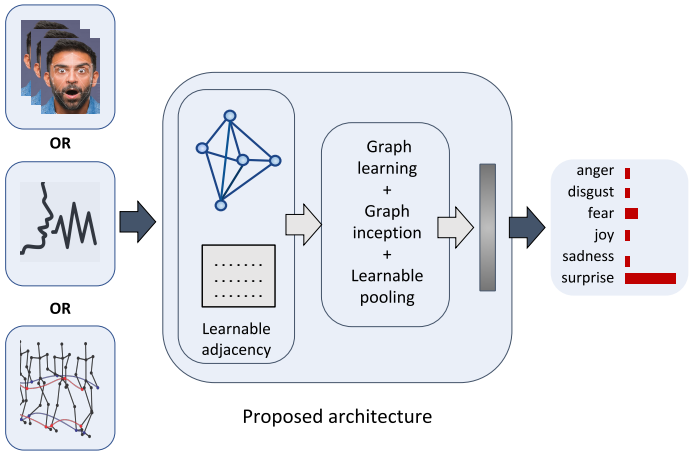


Figure A.1: A generalized graph approach to modelling emotion dynamics. Data samples are transformed to a *learnable* graph structure, where each node corresponds to a short temporal segment or frame. A novel graph architecture (L-GRIN) produces an embedding for the entire graph, facilitating emotion recognition.

of dimension $P = 136$.

A.1.3 Implementation Details

We use a 10-fold cross-validation for all three databases, and report the average recognition accuracy in Table A.1. We fix the length of each input video to 90 frames yielding a graph with $M = 90$ nodes. The shorter videos are simply padded by duplicating frames from the beginning of the video (cyclic padding). Our network weights are initialized following the Xavier initialization. We set $\lambda_1 = \lambda_2 = 0.1$ and $\lambda_3 = 1 \times 10^{-4}$ (see Eq. 4.8). We used Adam optimizer

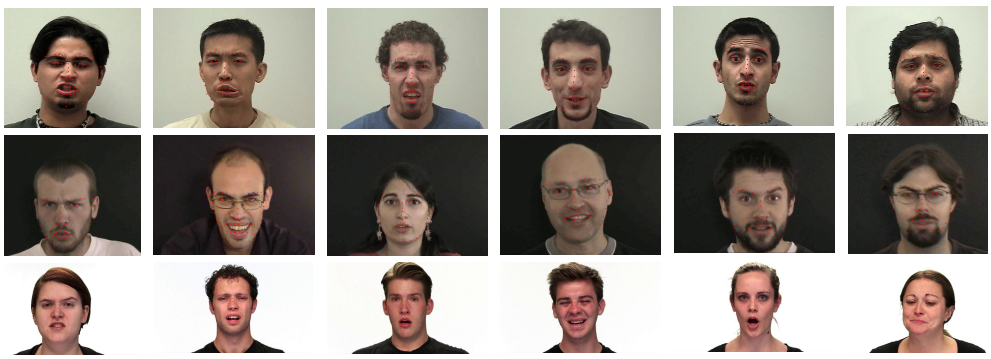


Figure A.2: Sample video frames from the three databases: RML (top row), eNTERFACE (middle row) and RAVDESS (bottom row). Each column shows one expression: (left to right) anger, disgust, fear, joy, surprise and sadness.

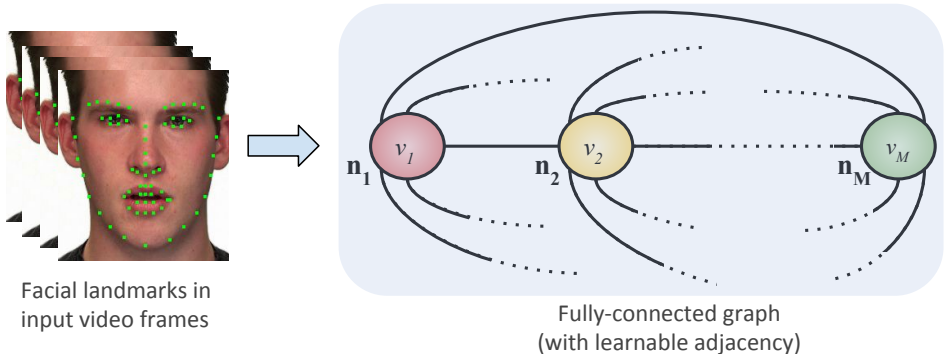


Figure A.3: *Graph construction:* Given a video of M frames, a fully-connected graph having M nodes is constructed. Each node corresponds to a video frame, and is associated with a node attribute vector consisting of the landmark point locations at the frame. The edge weights are learnable.

with a learning rate of 0.01 and decay rate of 0.5 after each 50 epochs for all experiments. To initialize the learnable adjacency matrix $\mathbf{A}$, we generate a random matrix whose elements are drawn from a Normal distribution with zero mean and unit variance. We used Pytorch for implementing our model and the baselines, and an NVIDIA RTX-2080Ti GPU for all experiments.

A.1.4 Baselines, State-of-the-art

We compare our model against two competitive and relevant baselines as follows:

BLSTM. The first baseline is a Bidirectional LSTM (BLSTM), an extension of the traditional LSTMs [220, 221]. LSTM and its variants have been successfully used in sentiment analysis in language and speech. This BLSTM comprises 1-layered bidirectional cells with embedding size 300 followed by a fully connected layer.

GCN [75]. A natural baseline to compare with our model is a spectral GCN in its standard form (as in Eq. (4.3)). The original network [75] is designed for node classification and only yields node-level embeddings. To obtain a graph-level embedding, we used max and mean pooling at the end of convolution layers. The GCN uses a binary adjacency matrix constructed following the method used in graph-based action recognition [78].

In addition to the baselines, we compare with two state-of-the-art graph classification architectures:

PATCHY-SAN [85] is a recent architecture that learns CNNs for arbitrary graphs. This architecture is originally developed for graph classification.

PATCHY-Diff [93] is referred to an architecture where PATCHY-SAN is used in combination with the differentiable pooling layer between graph convolution

Table A.1: Facial emotion recognition results on three video databases.

Model	**Accuracy (%)**			**Params**
	RML	eNTERFACE	RAVDESS	
*BLSTM	60.00	58.67	56.14	$\sim$ 1M
*GCN [75]	76.57	69.81	69.34	$\sim$ 102K
*PATCHY-SAN [85]	80.00	67.49	73.52	$\sim$ 52K
*PATCHY-Diff [93]	85.59	76.96	79.83	$\sim$ 71K
SENet [222]	71.20	79.22	71.06	$\sim$ 26M
AVEF [173]	82.48	85.69	-	-
KCFA [223]	82.22	76.00	-	-
OKL [224]	90.83	86.67	-	-
TJE [225]	-	-	72.30	-
***L-GrIN**	**94.11**	**87.49**	**85.65**	$\sim$ 120K

*use same node features

layers proposed recently [93].

SENet [222], Squeeze and Excitation net is a state-of-the-art CNN architecture recently proposed for facial emotion recognition in videos.

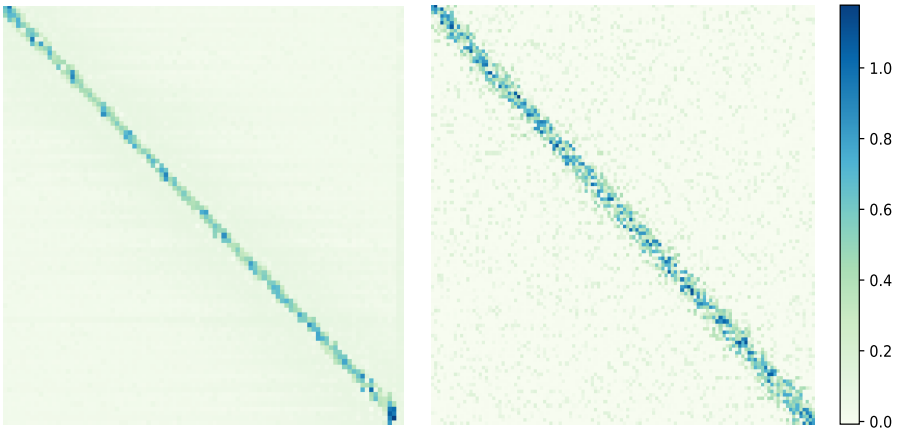
Comparisons are also made with other existing works on the respective databases: AudioVisual Emotion Fusion (AVEF) [173], Kernel Crossmodal Factor Analysis (KCFA) [223], Optimized Kernel-Laplacian (OKL) [224] and Temporal Joint Embeddings (TJE) [225].

A.1.5 Results

Table A.1 compares the performance of L-GrIN with all the methods mentioned above. Clearly, the proposed model outperforms all the existing methods by a significant margin, including the graph-based state-of-the-art architectures, such as PATCHY-SAN and PATCHY-Diff. Our model performs better than BLSTM - a class of models most commonly used in video-based emotion recognition. SENet is a very recent CNN architecture developed for emotion recognition, which also trails our model in terms of performance. When compared to the GCN baseline [75], L-GrIN improves the recognition accuracy by more than 10% on RML and eNTERFACE, and more than 5% on RAVDESS.

Also note that KCFA, OKL and TJE use both audio and visual information for recognition. Our model, even though uses only visual information, shows significant improvement over the audiovisual methods.

Fig. A.4 shows the learned adjacency matrix for the RAVDESS database. The learned graph structure shows higher values closer to the diagonal i.e., the weights shared among the neighboring nodes. This indicates higher temporal



RAVDESS (90×90) **MPI** (120×120)

Figure A.4: *Learned* adjacency matrices for facial and body emotion recognition showing strong temporal dependency between neighbouring segments. Darker values indicate higher weights.

dependencies locally and weaker dependency as we go further from a node.

A.2 Body Emotion Recognition

A.2.1 Databases

We use the MPI emotional body expression database [226] for our experiments. This database contains 1447 body motion samples of actors narrating coherent stories labeled with 11 emotions: *amusement, anger, disgust, fear, joy, neutral, pride, relief, sadness, shame,* and *surprise*. During their performance, a mocap system (device model: Xsens MVN) recorded the human motion using miniature inertial sensors. The system recorded dynamic 3D postures from 22 joints with a sampling rate of 120Hz.

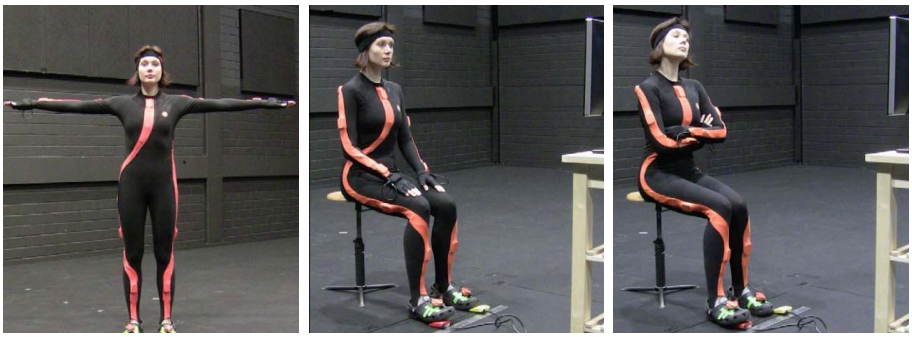


Figure A.5: Motion capture recording set-up for the MPI database showing an actor posing for (left to right) T pose (reference), neutral and pride pose.

A.2.2 Node Features

For this database, we use the raw information provided by the mocap system. Each node contains the 3D positions and orientations (measure in terms of the Euler angles, pitch, yaw and roll) at a given time-step. These measurements come with the database. The feature consists of Euler angles from 22 joints and additional location information of the reference point. We use all the information (without any preprocessing) as node features, resulting into a vector of dimension $P = 72$.

A.2.3 Implementation Details

Each input sample produces a graph of $M = 120$ nodes, where each node corresponds to a temporal segment of 120^{th} of a second. Cyclic padding is used as before. We perform a 5-fold cross-validation and report the average accuracy in Table A.2. All other network parameters remain the same as before.

A.2.4 Baselines, State-of-the-art

Our model is compared with the baselines (BLSTM and GCN), the state-of-the-art graph-based architectures (PATCHY-SAN and PATCHY-Diff), and a recent work on this database, i.e., trajectory learning [172]. The trajectory learning system [172] models neural motion and analyzes the spectral difference between an expressive motion and a neutral motion in order to recognize the body expressions.

A.2.5 Results

Table A.2 shows that L-GrIN outperforms the baselines and state-of-the-art methods on the MPI body expression database. Graph-based methods continue to perform well, indicating the effectiveness of graph-based methods for such tasks. Fig. A.4 shows the learned adjacency $\mathbf{A}$ for the MPI database. As before, the learned graph structure exhibit higher temporal dependencies among the neighboring nodes.

A.3 Network Analysis

A.3.1 Network Size

Tables A.1 and A.2 list the number of learnable network parameters for the baselines, state-of-the-art graph-based architectures and the proposed L-GrIN. As mentioned earlier, a graph network largely reduces the number of learnable parameters as compared to the BLSTM or CNN architectures such as SENet (see Table A.1) without compromising the recognition accuracy. Our model has

Table A.2: Body emotion recognition results on the MPI database.

Model	**Accuracy (%)**	**Parameters**
*BLSTM	45.52	$\sim$ 0.9M
*GCN	56.03	$\sim$ 92K
*PATCHY-SAN [85]	48.42	$\sim$ 80K
*PATCHY-Diff [93]	55.29	$\sim$ 71K
Trajectory learning [172]	50.00	-
***L-GrIN**	**58.59**	$\sim$ 110K

* use same node features

more parameters than the baseline GCN due to the inception layers and other learnable parameters, but also improves the recognition accuracy significantly. PATCHY-SAN and PATCHY-Diff have smaller network size compared to L-GRIN, but both trail L-GrIN in terms of performance on all databases. In case of facial emotion recognition, we discount the model size of the landmark detector in the comparison as it is common to all except SENet. For speech and body emotion recognition, no additional network was required as we used hand-crafted features and raw data.

A.3.2 Learnable vs. Fixed Pooling

Recall that to obtain a graph-level embedding from node-level embeddings, L-GrIN learns a pooling function (see Fig. 4.3). To show if learnable pooling indeed improves the recognition performance, we compare its performance with various fixed pooling strategies: max pooling, mean pooling and sort pooling (sortpool) [92]. Table A.3 presents the comparisons on the RML database in terms of facial emotion recognition accuracy, which clearly shows the advantage of learnable pooling over fixed pooling strategies. Similar trend is observed for other databases.

Table A.3: Comparison between learnable and fixed pooling strategies on the RML database. All experiments in this table use the same (binary) adjacency matrix for a fair comparison.

Pooling	**Accuracy** (%)
Maxpool	89.76
Meanpool	90.23
Sortpool [92]	83.66
Learnable pool	**91.50**

Table A.4: Comparison between learnable and manually constructed graph structures. For a fair comparison, all experiments use maxpool to convert node embeddings to graph embeddings.

	Accuracy (%)			**Params**		
	RML	IEMOCAP	MPI	RML	IEMOCAP	MPI
Binary	89.5	61.4	53.6	113K	78K	96K
Weighted	62.4	54.3	49.0	113K	78K	96K
Learnable	**91.5**	**65.5**	**58.9**	120K	92K	110K

A.3.3 Learnable vs. Manually Constructed Adjacency

An adjacency matrix represents the pairwise relationship between the graph nodes. When this information is not available naturally, a common practice is to manually construct an adjacency matrix. We argued earlier that this may result in sub-optimal graph structures, which in turn affects the classification performance. We now compare the performance of leranable adjacency with two fixed adjacency matrices:

(i) *Binary adjacency:* a natural choice is a binary adjacency matrix as used for graph-based action recognition [78]. This is defined as $(\mathbf{A}_b)_{ij} = 1$ if $|i - j| = 1$ and 0 otherwise, i.e., a node (frame) is connected only to its subsequent and preceding node in the temporal direction.

(ii) *Weighted adjacency:* Another adjacency matrix is formed by using the squared ℓ_2 distance between two node attributes as their edge weight. This is defined as $(\mathbf{A}_w)_{ij} = \|\mathbf{n}_i - \mathbf{n}_j\|_2^2$.

Table A.4 compares the performance of the proposed learnable adjacency with the two fixed adjacency matrices described above on the RML, IEMOCAP and the MPI databases. We chose one database from every modality. For this set of experiments, we used only maxpooling to obtain the graph-level embeddings for fair comparison. Clearly, the learnable adjacency matrix shows consistent improvement in accuracy across all databases for a relatively small increase in model complexity (only 6% additional parameters). The results show that a learnable adjacency has better at generalizing across databases and modalities.

A.3.4 Ablation Study

We performed exhaustive ablation experiments to investigate the contribution of each component we proposed to build L-GrIN. Table A.5 presents the ablation results on the RML database. We observe that each new component brings significant improvement (row 2 to row 5) over the performance of standard GCN

Table A.5: Ablation study on the RML database. Each new component in L-GRIN contributes towards its performance.

$\mathcal{G}^*$**conv**	**Inception**	**Learned A**	**Learned p**	**Accuracy** (%)
-	-	-	-	76.57
✓	-	-	-	80.12
-	✓	-	-	87.58
-	-	✓	-	79.78
-	-	-	✓	82.86
-	-	✓	✓	84.21
✓	✓	-	-	90.65
✓	✓	✓	-	91.50
✓	✓	-	✓	91.50
✓	✓	✓	✓	**94.11**

[75] which has 76.57% recognition accuracy (the top row in Table A.5). The introduction of the graph inception layer increases the recognition rate by 11%; when combined with our new graph convolution layer $\mathcal{G}^*$conv (Eq. (4.4)), the accuracy increases to 90.65%. Adding the learnable graph structure (learned $\mathbf{A}$) and learnable pooling bring the accuracy up to 94.11% both contributing to the accuracy. Removing either of the leanrable components reduces the accuracy by 2.61%. The ablation results show that each of the proposed components in our architecture is important, and contributes positively towards its superior performance. Similar ablation trend was observed for other databases.

A.3.5 Inception Layer Settings

We also investigate the effects of the graph inception layer hyperparameters: (i) the parameter η corresponding to the size of the graph convolution filters $\mathcal{G}_1^*$ and $\mathcal{G}_2^*$ in Eq. (4.5), and (ii) the number of graph inception layers in L-GrIN. First,

Table A.6: Analyzing inception layer settings on the RML database.

Effect of filter size (η)	
Size of the two filters	**Accuracy (%)**
$(16, 32)$	90.82
$(32, 64)$	92.47
$(\mathbf{64}, \mathbf{128})$	**94.11**
$(128, 256)$	93.13
Effect of number of inception layers	
Number of layers	**Accuracy (%)**
1	91.77
2	**94.11**
3	90.78

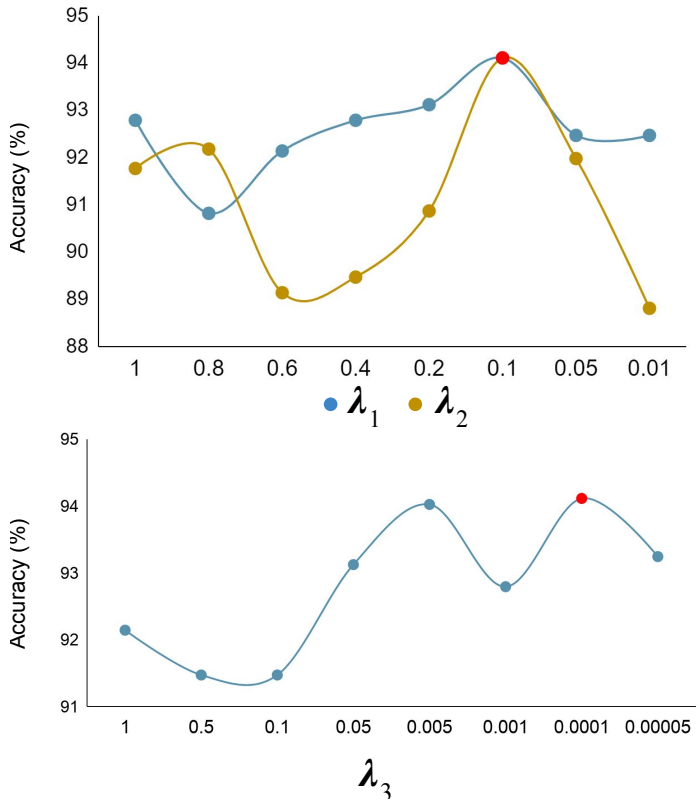


Figure A.6: Effect of the weight parameters in the loss function; experiments on the RML database.

we vary the filter dimensions (can be interpreted as scales) in the two inception layers and note how this correspond to the model's performance. Results for the RML database is presented in Table A.6; similar trends have been observed for other databases. Results in Table A.6 show that we achieve the best performance for the combination of (64, 128), which is used in our model. Next, we vary the number of inception layers in the model, each with (64, 128) filter combination (see Table A.6. We observe that reducing or increasing the number of inception layers from 2 results in a drop in performance. We chose to use two inception layers in the proposed model. It is obvious that the model size increases significantly as we add more inception layers or increase filter sizes within the layers. We notice a small drop in performance with larger filter sizes and with higher number of inception layers. This could be possibly due to over-smoothing and over-mixing of the node features. However, the over-smoothing effect is not as prominent as in many node classification tasks.

A.3.6 Analysis of the Control Weights

We also examine the impact of the weights controlling the various components of the loss function in Eq. (4.8), i.e., λ_1, λ_2 and λ_3. Fig. A.6 shows that highest performance is achieved for $\lambda_1 = \lambda_2 = 0.1$ and $\lambda_2 = 0.0001$ (marked red in the

Table A.7: Cross-corpus performance of our model (L-GrIN) for facial emotion recognition.

Trained on	**Evaluated on**	**Accuracy (%)**
RAVDESS	RML	81.94
	eNTERFACE	75.80
RML	RAVDESS	75.42
	eNTERFACE	61.71
eNTERFACE	RML	79.86
	RAVDESS	77.51

plots) on the RML database. We use these λ values in our experiments.

A.3.7 Cross-corpus Performance

Methods exhibiting superior performance on one corpus, often fall short when tested on another corpus having different statistical distributions. We investigated the ability of our model to generalize across databases by evaluating its cross-corpus performance. To this end, we trained L-GrIN on one database, followed by fine-tuning a fully-connected layer on the target database, without changing the graph structure (or other parameters) learned from the training database.

Results in Table A.7 shows that our model can generalize well producing consistent results under cross-corpus evaluation. Our cross-corpus results higher accuracy compared to the same-corpus GCN accuracy. Cross-corpus results are comparable with the same-corpus performance of PATCHY-SAN. This shows the strength of the proposed architecture. It is worth noticing that the RML database (when used for training) does not have *neutral* and *calmness* emotion classes, but our model still recognizes those emotions on RAVDESS with 67.2% and 73.4% accuracy.

A.3.8 Network Visualization

To get an insight into the learning process of our model, we visualized how it attends to different nodes. The video data are the most suitable for the visualization. We use our trained model, and then feed-forward each test video sample through the network, and identify the node (each node corresponds to a video frame) that responded most strongly towards the maxpooling layer. This yields a *salient* node corresponding to each input. We present the corresponding video frames - one example per emotion class for RML, eNTERFACE and RAVDESS databases in Fig. A.7. The results show that the proposed model is able to learn the salient information from the input graphs such that it is representative of each emotion.

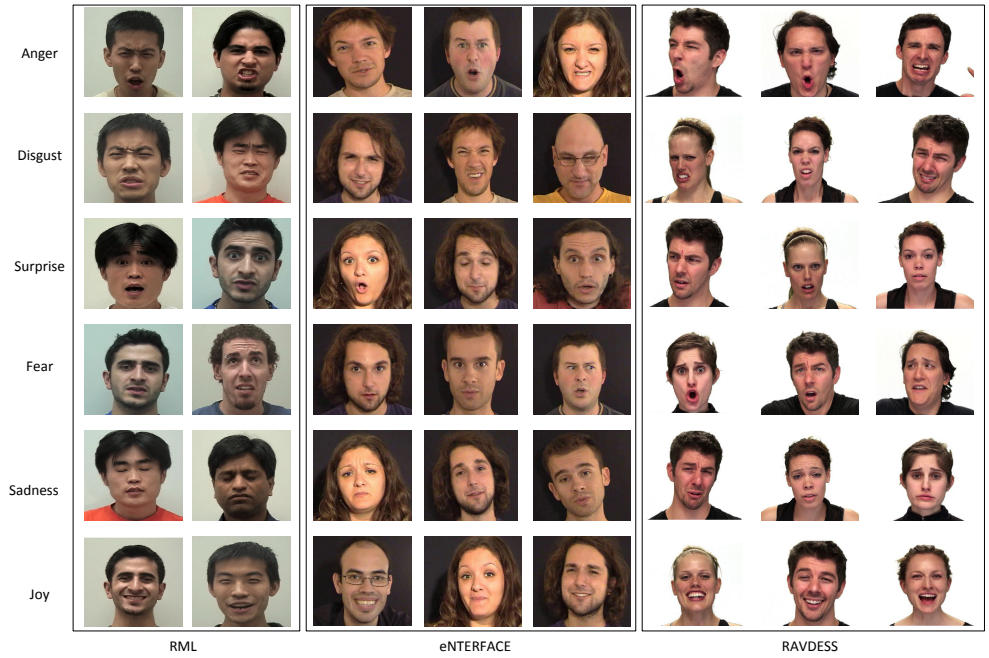


Figure A.7: Qualitative results showing the node (frame) for a graph input that generated the strongest response in our network. One result is displayed per class for the three databases. This shows that L-GRIN is able to learn the salient information for each emotion.

Appendix B

Second

Graph SSL Tasks Evaluation on Graph Data

We next attempt to understand the effectiveness of proposed self-supervised tasks in Chapter 5 on transductive semi-supervised node classification. This requires experimenting with databases where the graph structures are known so as to disentangle the effect of our proposed graph construction methodology. We use three graph citation databases **Cora** (2708 nodes, 7 classes) and **Citeseer** (3327 nodes, 6 classes) [227] and **Pubmed** (19717 nodes, 3 classes) [228] following the standard benchmarking framework [229].

To verify the universality of our SSL tasks, we conduct experiments on several state-of-the-art graph neural networks: (i) Standard GCN [75], (ii) Graph Attention Network (GAT) [230] - a powerful variant of GCN, (iii) Graph Isomorphism Network (GIN) [231] and (iv) GraphMix [232]. Based on the graph models, we use a joint learning framework where the SSL task as an auxiliary task and node classification is the primary task. Table **??** compares the performance of the above graph models when augmented with the three SSL tasks. Clearly, the SSL tasks (particularly denoising and completion) improve the accuracy of all models across all database cases (Table B.1). As mentioned before, the SSL tasks we consider are model-agnostic and thus can be used to enhance any graph model.

Table B.1: To show the effectiveness of our proposed framework when the graph structures are *known*, we present semi-supervised node classification results (% accuracy) on benchmark graph databases. This essentially disentangles the effect of our proposed graph construction methodology from the SSL-based semi supervised model. The results show that SSL tasks improve over basic models in almost all cases irrespective of the graph network used.

	SSL task	**Cora**	**Citeseer**	**Pubmed**
GCN	✗	80.9 ± 0.6	70.7 ± 0.6	79.1 ± 0.5
	denoise	81.1 ± 0.8	71.1 ± 0.8	78.4 ± 0.8
	completion	$\mathbf{81.6} \pm 0.8$	$\mathbf{71.6} \pm 0.6$	$\mathbf{79.2} \pm 0.7$
	shuffle	$\mathbf{81.6} \pm 0.6$	70.1 ± 1.1	78.4 ± 0.6
GAT	✗	83.1 ± 0.5	72.1 ± 0.6	77.5 ± 0.4
	denoise	$\mathbf{84.2} \pm 1.0$	$\mathbf{73.1} \pm 0.5$	$\mathbf{78.2} \pm 0.5$
	completion	84.2 ± 0.4	72.8 ± 1.1	78.0 ± 0.5
	shuffle	83.2 ± 0.9	72.6 ± 0.7	77.7 ± 0.4
GIN	✗	77.2 ± 0.5	68.1 ± 0.7	77.0 ± 0.4
	denoise	$\mathbf{78.9} \pm 0.8$	69.1 ± 1.4	77.2 ± 0.3
	completion	78.8 ± 0.6	$\mathbf{69.8} \pm 1.2$	$\mathbf{77.7} \pm 0.3$
	shuffle	78.6 ± 1.1	69.3 ± 0.8	77.3 ± 0.4
GraphMix	✗	83.8 ± 0.8	74.3 ± 0.7	80.6 ± 0.6
	denoise	$\mathbf{84.4} \pm 0.7$	$\mathbf{75.2} \pm 0.5$	81.4 ± 0.4
	completion	84.4 ± 0.7	74.3 ± 0.7	81.2 ± 0.3
	shuffle	84.2 ± 0.4	74.4 ± 0.6	$\mathbf{81.9} \pm 0.3$

Bibliography

[1] Amir Shirian and Tanaya Guha. Compact graph architecture for speech emotion recognition. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6284–6288, 2021.

[2] Amir Shirian, Subarna Tripathi, and Tanaya Guha. Dynamic emotion modeling with learnable graphs and graph inception network. *IEEE Transactions on Multimedia*, 2021.

[3] Amir Shirian, Krishna Somandepalli, and Tanaya Guha. Self-supervised graphs for audio representation learning with limited labeled data. *IEEE Journal of Selected Topics in Signal Processing*, pages 1–11, 2022.

[4] Amir Shirian, Krishna Somandepalli, Victor Sanchez, and Tanaya Guha. Visually-aware acoustic event detection using heterogeneous graphs. In *INTERSPEECH 2022*, pages 641–645. International Speech Communication Association, 2022.

[5] Joan Navarro, Ester Vidaña-Vila, Rosa Ma Alsina-Pagès, and Marcos Hervás. Real-time distributed architecture for remote acoustic elderly monitoring in residential-scale ambient assisted living scenarios. *Sensors*, 18(8):2492, 2018.

[6] Mahesh Kumar Nandwana and Taufiq Hasan. Towards smart-cars that can listen: Abnormal acoustic event detection on the road. In *INTERSPEECH*, pages 2968–2971, 2016.

[7] Yusuf Ozkan and Buket D Barkana. Forensic audio analysis and event recognition for smart surveillance systems. In *2019 IEEE International Symposium on Technologies for Homeland Security (HST)*, pages 1–6. IEEE, 2019.

[8] G Morfi. *Automatic detection and classi cation of bird sounds in low-resource wildlife audio datasets*. PhD thesis, Queen Mary University of London, 2019.

[9] Abeer Ali Alnuaim, Mohammed Zakariah, Aseel Alhadlaq, Chitra Shashidhar, Wesam Atef Hatamleh, Hussam Tarazi, Prashant Kumar Shukla, and Rajnish Ratna. Human-computer interaction with detection of speaker emotions using convolution neural networks. *Computational Intelligence and Neuroscience*, 2022, 2022.

[10] Alice Baird, Meishu Song, and Björn Schuller. Interaction with the soundscape: exploring emotional audio generation for improved individual wellbeing. In *International Conference on Human-Computer Interaction*, pages 229–242. Springer, 2020.

[11] Soumya Priyadarsini Panda. Automated speech recognition system in advancement of human-computer interaction. In *2017 International Conference on Computing Methodologies and Communication (ICCMC)*, pages 302–306. IEEE, 2017.

[12] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324, 2019.

[13] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[14] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[15] Yang He, Ping Liu, Ziwei Wang, Zhilan Hu, and Yi Yang. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4340–4349, 2019.

[16] Ashish Vaswani, Prajit Ramachandran, Aravind Srinivas, Niki Parmar, Blake Hechtman, and Jonathon Shlens. Scaling local self-attention for parameter efficient visual backbones. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12894–12904, 2021.

[17] Vivienne Sze, Yu-Hsin Chen, Tien-Ju Yang, and Joel S Emer. Efficient processing of deep neural networks. *Synthesis Lectures on Computer Architecture*, 15(2):1–341, 2020.

[18] Huriye Atilgan, Stephen M Town, Katherine C Wood, Gareth P Jones, Ross K Maddox, Adrian KC Lee, and Jennifer K Bizley. Integration of visual information in auditory cortex promotes auditory scene analysis through multisensory binding. *Neuron*, 97(3):640–655, 2018.

[19] Bowen Shi, Wei-Ning Hsu, and Abdelrahman Mohamed. Robust self-supervised audio-visual speech recognition. *arXiv preprint arXiv:2201.01763*, 2022.

[20] Harry McGurk and John MacDonald. Hearing lips and seeing voices. *Nature*, 264(5588):746–748, 1976.

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[22] Monorama Swain, Aurobinda Routray, and Prithviraj Kabisatpathy. Databases, features and classifiers for speech emotion recognition: a review. *International Journal of Speech Technology*, 21(1):93–120, 2018.

[23] Divya Gupta, Poonam Bansal, and Kavita Choudhary. The state of the art of feature extraction techniques in speech recognition. *Speech and language processing for human-machine communications*, pages 195–207, 2018.

[24] Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[25] Q Jin, C Li, S Chen, and H Wu. Speech emotion recognition with acoustic and lexical features. In *ICASSP*, pages 4749–4753. IEEE, 2015.

[26] Thapanee Seehapoch and Sartra Wongthanavasu. Speech emotion recognition using support vector machines. In *2013 5th international conference on Knowledge and smart technology (KST)*, pages 86–91. IEEE, 2013.

[27] S Renjith and KG Manju. Speech based emotion recognition in tamil and telugu using lpcc and hurst parameters—a comparitive study using knn and ann classifiers. In *2017 International conference on circuit, power and computing technologies (ICCPCT)*, pages 1–6. IEEE, 2017.

[28] Enes Yüncü, Hüseyin Hacihabiboglu, and Cem Bozsahin. Automatic speech emotion recognition using auditory models with binary decision tree and svm. In *2014 22nd international conference on pattern recognition*, pages 773–778. IEEE, 2014.

[29] Atreyee Khan and Uttam Kumar Roy. Emotion recognition using prosodie and spectral features of speech and naïve bayes classifier. In *2017 international conference on wireless communications, signal processing and networking (WiSPNET)*, pages 1017–1021. IEEE, 2017.

[30] Pavol Harár, Radim Burget, and Malay Kishore Dutta. Speech emotion recognition with deep learning. In *2017 4th International conference on signal processing and integrated networks (SPIN)*, pages 137–140. IEEE, 2017.

[31] Shiqing Zhang, Shiliang Zhang, Tiejun Huang, and Wen Gao. Speech emotion recognition using deep convolutional neural network and discriminant temporal pyramid matching. *IEEE Transactions on Multimedia*, 20(6):1576–1590, 2017.

[32] Y Wang and L Guan. Recognizing human emotional state from audiovisual signals. *IEEE Transactions on Multimedia*, 10(5):936–946, 2008.

[33] Olivier Martin, Irene Kotsia, Benoit Macq, and Ioannis Pitas. The enterface'05 audio-visual emotion database. In *International Conference on Data Engineering Workshops (ICDEW)*, pages 8–8, 2006.

[34] Sara Zhalehpour, Onur Onder, Zahid Akhtar, and Cigdem Eroglu Erdem. Baum-1: A spontaneous audio-visual face database of affective and mental states. *IEEE Transactions on Affective Computing*, 8(3):300–313, 2016.

[35] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[36] Seyedmahdad Mirsamadi, Emad Barsoum, and Cha Zhang. Automatic speech emotion recognition using recurrent neural networks with local attention. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2227–2231, 2017.

[37] Siddique Latif, Rajib Rana, Sara Khalifa, Raja Jurdak, and Julien Epps. Direct modelling of speech emotion from raw speech. *Proc. Interspeech 2019*, pages 3920–3924, 2019.

[38] Martin Wöllmer, Moritz Kaiser, Florian Eyben, Björn Schuller, and Gerhard Rigoll. Lstm-modeling of continuous emotions in an audiovisual affect recognition framework. *Image and Vision Computing*, 31(2):153–163, 2013.

[39] Milan Gnjatović and Dietmar Rösner. Inducing genuine emotions in simulated speech-based human-machine interaction: The nimitek corpus. *IEEE Transactions on Affective Computing*, 1(2):132–144, 2010.

[40] George Trigeorgis, Fabien Ringeval, Raymond Brueckner, Erik Marchi, Mihalis A Nicolaou, Björn Schuller, and Stefanos Zafeiriou. Adieu features? end-to-end speech emotion recognition using a deep convolutional recurrent network. In *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5200–5204. IEEE, 2016.

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[42] Koichi Miyazaki, Tatsuya Komatsu, Tomoki Hayashi, Shinji Watanabe, Tomoki Toda, and Kazuya Takeda. Convolution augmented transformer for semi-supervised sound event detection. In *Proc. Workshop Detection Classification Acoust. Scenes Events (DCASE)*, pages 100–104, 2020.

[43] Qiuqiang Kong, Yong Xu, Wenwu Wang, and Mark D Plumbley. Sound event detection of weakly labelled data with cnn-transformer and automatic threshold optimization. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2450–2460, 2020.

[44] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. Conformer: Convolution-augmented transformer for speech recognition. In Helen Meng, Bo Xu, and Thomas Fang Zheng, editors, *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association, Virtual Event, Shanghai, China, 25-29 October 2020*, pages 5036–5040. ISCA, 2020.

[45] Qiuqiang Kong, Yin Cao, Turab Iqbal, Yuxuan Wang, Wenwu Wang, and Mark D Plumbley. Panns: Large-scale pretrained audio neural networks for audio pattern recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2880–2894, 2020.

[46] Yuan Gong, Yu-An Chung, and James Glass. Psla: Improving audio tagging with pretraining, sampling, labeling, and aggregation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3292–3306, 2021.

[47] Oleg Rybakov, Natasha Kononenko, Niranjan Subrahmanya, Mirkó

Visontai, and Stella Laurenzo. Streaming keyword spotting on mobile devices. *arXiv preprint arXiv:2005.06720*, 2020.

[48] Yuan Gong, Yu-An Chung, and James Glass. AST: Audio spectrogram transformer. *arXiv preprint arXiv:2104.01778*, 2021.

[49] Stelios A Mitilineos, Stelios M Potirakis, Nicolas-Alexander Tatlas, and Maria Rangoussi. A two-level sound classification platform for environmental monitoring. *Journal of Sensors*, 2018, 2018.

[50] Emre Şaşmaz and F Boray Tek. Animal sound classification using a convolutional neural network. In *2018 3rd International Conference on Computer Science and Engineering (UBMK)*, pages 625–629. IEEE, 2018.

[51] Nabanita Das, Atreyee Mondal, Jyotismita Chaki, Neelamadhab Padhy, and Nilanjan Dey. Machine learning models for bird species recognition based on vocalization: A succinct review. *Information Technology and Intelligent Transportation Systems*, pages 117–124, 2020.

[52] Zachary J Ruff, Damon B Lesmeister, Cara L Appel, and Christopher M Sullivan. Workflow and convolutional neural network for automated identification of animal sounds. *Ecological Indicators*, 124:107419, 2021.

[53] Kimitake Ohkawa, Masaru Yamashita, and Shoichi Matsunaga. Classification between abnormal and normal respiration through observation rate of heart sounds within lung sounds. In *2018 26th European Signal Processing Conference (EUSIPCO)*, pages 1142–1146. IEEE, 2018.

[54] Jia-Ching Wang, Chang-Hong Lin, Bo-Wei Chen, and Min-Kang Tsai. Gabor-based nonuniform scale-frequency map for environmental sound classification in home automation. *IEEE Transactions on Automation Science and Engineering*, 11(2):607–613, 2013.

[55] Rafael Lima De Carvalho and Paulo Fernando Ferreira Rosa. Identification system for smart homes using footstep sounds. In *2010 IEEE international symposium on industrial electronics*, pages 1639–1644. IEEE, 2010.

[56] Sayed Khushal Shah, Zeenat Tariq, and Yugyung Lee. Audio iot analytics for home automation safety. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 5181–5186, 2018.

[57] Justin Salamon and Juan Pablo Bello. Feature learning with deep scattering for urban sound analysis. In *2015 23rd European signal processing conference (EUSIPCO)*, pages 724–728. IEEE, 2015.

[58] Sangeeta Srivastava, Dhrubojyoti Roy, Mark Cartwright, Juan P Bello, and Anish Arora. Specialized embedding approximation for edge intelligence: A case study in urban sound classification. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8378–8382. IEEE, 2021.

[59] Pasquale Foggia, Nicolai Petkov, Alessia Saggese, Nicola Strisciuglio, and Mario Vento. Reliable detection of audio events in highly noisy environments. *Pattern Recognition Letters*, 65:22–28, 2015.

[60] Muhammad Zohaib Anwar, Zeeshan Kaleem, and Abbas Jamalipour. Machine learning inspired sound-based amateur drone detection for public safety applications. *IEEE Transactions on Vehicular Technology*, 68(3):2526–2534, 2019.

[61] Annamaria Mesaros, Toni Heittola, Antti Eronen, and Tuomas Virtanen. Acoustic event detection in real life recordings. In *2010 18th European signal processing conference*, pages 1267–1271. IEEE, 2010.

[62] Toni Heittola, Annamaria Mesaros, Tuomas Virtanen, and Moncef Gabbouj. Supervised model training for overlapping sound events based on unsupervised source separation. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 8677–8681. IEEE, 2013.

[63] Xiaomeng Zhang, Hao Sun, Shuopeng Wang, and Jing Xu. Speech signal classification based on convolutional neural networks. In *International Conference on Cognitive Systems and Signal Processing*, pages 281–287. Springer, 2018.

[64] Karol J Piczak. Environmental sound classification with convolutional neural networks. In *2015 IEEE 25th international workshop on machine learning for signal processing (MLSP)*, pages 1–6. IEEE, 2015.

[65] Donghyeon Kim, Sangwook Park, David K Han, and Hanseok Ko. Multiband cnn architecture using adaptive frequency filter for acoustic event classification. *Applied Acoustics*, 172:107579, 2021.

[66] Sharnil Pandya and Hemant Ghayvat. Ambient acoustic event assistive framework for identification, detection, and recognition of unknown acoustic events of a residence. *Advanced Engineering Informatics*, 47:101238, 2021.

[67] Chieh-Chi Kao, Ming Sun, Weiran Wang, and Chao Wang. A comparison of pooling methods on lstm models for rare acoustic event classification. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 316–320. IEEE, 2020.

[68] Xugang Lu, Peng Shen, Sheng Li, Yu Tsao, and Hisashi Kawai. Temporal attentive pooling for acoustic event detection. In *Interspeech*, pages 1354–1357, 2018.

[69] Bálint Pál Tóth and Bálint Czeba. Convolutional neural networks for large-scale bird song classification in noisy environment. In *CLEF (Working Notes)*, pages 560–568, 2016.

[70] Achyut Mani Tripathi and Aakansha Mishra. Self-supervised learning for environmental sound classification. *Applied Acoustics*, 182:108183, 2021.

[71] Aren Jansen, Manoj Plakal, Ratheet Pandya, Daniel PW Ellis, Shawn Hershey, Jiayang Liu, R Channing Moore, and Rif A Saurous. Unsupervised learning of semantic audio representations. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 126–130. IEEE, 2018.

[72] Marco Tagliasacchi, Beat Gfeller, Félix de Chaumont Quitry, and Dominik Roblek. Pre-training audio representations with self-supervision. *IEEE Signal Processing Letters*, 27:600–604, 2020.

[73] Po-Han Chi, Pei-Hung Chung, Tsung-Han Wu, Chun-Cheng Hsieh, Yen-Hao Chen, Shang-Wen Li, and Hung-yi Lee. Audio albert: A lite bert for self-supervised learning of audio representation. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 344–350. IEEE, 2021.

[74] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International Conference on Machine Learning (ICML)*, pages 1263–1272, 2017.

[75] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[76] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations (ICLR)*, 2019.

[77] Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, and Maosong Sun. Graph neural networks: A review of methods and applications. *CoRR*, abs/1812.08434, 2018.

[78] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *AAAI Conference on Artificial Intelligence*, pages 7444–7452, 2018.

[79] Roei Herzig, Elad Levi, Huijuan Xu, Eli Brosh, Amir Globerson, and Trevor Darrell. Classifying collisions with spatio-temporal action graph networks. *ICCV Workshop*, 2019.

[80] Zhongdao Wang, Liang Zheng, Yali Li, and Shengjin Wang. Linkage based face clustering via graph convolution network. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1117–1125, 2019.

[81] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 652–660, 2017.

[82] Damien Teney, Lingqiao Liu, and Anton van den Hengel. Graph-structured representations for visual question answering. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3233–3241, 2017.

[83] Oron Ashual and Lior Wolf. Specifying object attributes and relations in interactive scene generation. In *International Conference on Computer Vision (ICCV)*, pages 4561–4569, 2019.

[84] Subarna Tripathi, Sharath Nittur Sridhar, Sairam Sundaresan, and Hanlin Tang. Compact scene graphs for layout composition and patch retrieval. In *Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019.

[85] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *International Conference on Machine Learning (ICML)*, pages 2014–2023, 2016.

[86] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pages 1024–1034, 2017.

[87] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *arXiv preprint arXiv:1211.0053*, 2012.

[88] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations (ICLR)*, 2014.

[89] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pages 3844–3852, 2016.

[90] Bingbing Xu, Huawei Shen, Qi Cao, Yunqi Qiu, and Xueqi Cheng. Graph wavelet neural network. In *International Conference on Learning Representations (ICLR)*, 2019.

[91] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*, 2015.

[92] Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. An end-to-end deep learning architecture for graph classification. In *AAAI Conference on Artificial Intelligence*, pages 4438–4445, 2018.

[93] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In *Advances in Neural Information Processing Systems*, pages 4800–4810, 2018.

[94] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[95] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European conference on computer vision (ECCV)*, pages 181–196, 2018.

[96] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proc. Conference of the North American Association for Computational Linguistics NAACL*, pages 4171–4186, 2019.

[97] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1422–1430, 2015.

[98] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E. Hinton. Big self-supervised models are strong semi-supervised learners. In *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[99] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10687–10698, 2020.

[100] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[101] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. wav2vec: Unsupervised pre-training for speech recognition. In Gernot Kubin and Zdravko Kacic, editors, *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pages 3465–3469. ISCA, 2019.

[102] Hyeong-Seok Choi, Juheon Lee, Wansoo Kim, Jie Lee, Hoon Heo, and Kyogu Lee. Neural analysis and synthesis: Reconstructing speech from self-supervised representations. *Advances in Neural Information Processing Systems(NeurIPS)*, 34, 2021.

[103] Santiago Pascual, Mirco Ravanelli, Joan Serrà, Antonio Bonafonte, and Yoshua Bengio. Learning problem-agnostic speech representations from multiple self-supervised tasks. In Gernot Kubin and Zdravko Kacic, editors, *Interspeech 2019*, pages 161–165, 2019.

[104] Anurag Kumar and Vamsi Krishna Ithapu. Secost: : Sequential co-supervision for large scale weakly labeled audio event detection. In *International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pages 666–670, 2020.

[105] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

[106] Daisuke Niizumi, Daiki Takeuchi, Yasunori Ohishi, Noboru Harada, and Kunio Kashino. Byol for audio: Self-supervised learning for general-purpose audio representation. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2021.

[107] Abhinav Shukla, Stavros Petridis, and Maja Pantic. Does visual self-supervision improve learning of speech representations for emotion recognition. *IEEE Transactions on Affective Computing*, 2021.

[108] Arsha Nagrani, Joon Son Chung, Samuel Albanie, and Andrew Zisserman. Disentangled speech embeddings using cross-modal self-supervision. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6829–6833, 2020.

[109] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

[110] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

[111] Thomas N. Kipf and Max Welling. Variational graph auto-encoders. *CoRR*, abs/1611.07308, 2016.

[112] Yuning You, Tianlong Chen, Zhangyang Wang, and Yang Shen. When does self-supervision help graph convolutional networks? In *International Conference on Machine Learning (ICML)*, pages 10871–10880, 2020.

[113] Anton Tsitsulin, Davide Mottin, Panagiotis Karras, Alex Bronstein, and Emmanuel Müller. SGR: Self-supervised spectral graph representation learning. *arXiv preprint arXiv:1811.06237*, 2018.

[114] Wei Jin, Tyler Derr, Haochen Liu, Yiqi Wang, Suhang Wang, Zitao Liu, and Jiliang Tang. Self-supervised learning on graphs: Deep insights and new direction. *arXiv preprint arXiv:2006.10141*, 2020.

[115] Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. *ICLR (Poster)*, 2(3):4, 2019.

[116] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. Gcc: Graph contrastive coding for graph neural network pre-training. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1150–1160, 2020.

[117] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, 33:5812–5823, 2020.

[118] Yuxin Peng and Jinwei Qi. Cm-gans: Cross-modal generative adversarial networks for common representation learning. *ACM Transactions on*

Multimedia Computing, Communications, and Applications (TOMM), 15(1):1–24, 2019.

[119] Nikhil Rasiwasia, Jose Costa Pereira, Emanuele Coviello, Gabriel Doyle, Gert RG Lanckriet, Roger Levy, and Nuno Vasconcelos. A new approach to cross-modal multimedia retrieval. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 251–260, 2010.

[120] Yanan Liu, Xiaoqing Feng, and Zhiguang Zhou. Multimodal video classification with stacked contractive autoencoders. *Signal Processing*, 120:761–766, 2016.

[121] Shuang Wu, Sravanthi Bondugula, Florian Luisier, Xiaodan Zhuang, and Pradeep Natarajan. Zero-shot event detection using multi-modal fusion of weakly supervised concepts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2665–2672, 2014.

[122] Amirhossein Habibian, Thomas Mensink, and Cees GM Snoek. Video2vec embeddings recognize events when examples are scarce. *IEEE transactions on pattern analysis and machine intelligence*, 39(10):2089–2103, 2016.

[123] Soujanya Poria, Erik Cambria, Newton Howard, Guang-Bin Huang, and Amir Hussain. Fusing audio, visual and textual clues for sentiment analysis from multimodal content. *Neurocomputing*, 174:50–59, 2016.

[124] Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. Tensor fusion network for multimodal sentiment analysis. *arXiv preprint arXiv:1707.07250*, 2017.

[125] Fangxiang Feng, Xiaojie Wang, and Ruifan Li. Cross-modal retrieval with correspondence autoencoder. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 7–16, 2014.

[126] Jinwei Qi and Yuxin Peng. Cross-modal bidirectional translation via reinforcement learning. In *IJCAI*, pages 2630–2636, 2018.

[127] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *International conference on machine learning*, pages 1060–1069. PMLR, 2016.

[128] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and

description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015.

[129] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057. PMLR, 2015.

[130] Humam Alwassel, Dhruv Mahajan, Bruno Korbar, Lorenzo Torresani, Bernard Ghanem, and Du Tran. Self-supervised learning by cross-modal audio-video clustering. *Advances in Neural Information Processing Systems*, 33:9758–9770, 2020.

[131] Yuki Asano, Mandela Patrick, Christian Rupprecht, and Andrea Vedaldi. Labelling unlabelled videos from scratch with multi-modal self-supervision. *Advances in Neural Information Processing Systems*, 33:4660–4671, 2020.

[132] Bruno Korbar, Du Tran, and Lorenzo Torresani. Cooperative learning of audio and video models from self-supervised synchronization. *Advances in Neural Information Processing Systems*, 31, 2018.

[133] Jean-Baptiste Alayrac, Adria Recasens, Rosalia Schneider, Relja Arandjelović, Jason Ramapuram, Jeffrey De Fauw, Lucas Smaira, Sander Dieleman, and Andrew Zisserman. Self-supervised multimodal versatile networks. *Advances in Neural Information Processing Systems*, 33:25–37, 2020.

[134] Rui Qian, Tianjian Meng, Boqing Gong, Ming-Hsuan Yang, Huisheng Wang, Serge Belongie, and Yin Cui. Spatiotemporal contrastive video representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6964–6974, 2021.

[135] Shuang Ma, Zhaoyang Zeng, Daniel J. McDuff, and Yale Song. Active contrastive learning of audio-visual video representations. In *9th International Conference on Learning Representations, ICLR*, 2021.

[136] Aaqib Saeed, David Grangier, and Neil Zeghidour. Contrastive learning of general-purpose audio representations. In *ICASSP*, pages 3875–3879. IEEE, 2021.

[137] Shuang Ma, Zhaoyang Zeng, Daniel McDuff, and Yale Song. Contrastive learning of global and local video representations. *Advances in Neural Information Processing Systems*, 34, 2021.

[138] Jianbo Jiao, Yifan Cai, Mohammad Alsharid, Lior Drukker, Aris T Papageorghiou, and J Alison Noble. Self-supervised contrastive video-speech representation learning for ultrasound. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 534–543. Springer, 2020.

[139] Simon Jenni and Hailin Jin. Time-equivariant contrastive video representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9970–9980, 2021.

[140] Yanbei Chen, Yongqin Xian, A Koepke, Ying Shan, and Zeynep Akata. Distilling audio-visual knowledge by compositional contrastive learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7016–7025, 2021.

[141] Miao Liu, Xin Chen, Yun Zhang, Yin Li, and James M. Rehg. Attention distillation for learning video representations. In *31st British Machine Vision Conference 2020, BMVC*, 2020.

[142] Abhinav Shukla, Stavros Petridis, and Maja Pantic. Learning speech representations from raw audio by joint audiovisual self-supervision. *arXiv preprint arXiv:2007.04134*, 2020.

[143] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pages 12310–12320. PMLR, 2021.

[144] Qifan Wang, Yinwei Wei, Jianhua Yin, Jianlong Wu, Xuemeng Song, Liqiang Nie, and Min Zhang. Dualgnn: Dual graph neural network for multimedia recommendation. *IEEE Transactions on Multimedia*, 2021.

[145] Shengsheng Qian, Dizhan Xue, Huaiwen Zhang, Quan Fang, and Changsheng Xu. Dual adversarial graph neural networks for multi-label cross-modal retrieval. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI*, pages 2440–2448, 2021.

[146] Raeid Saqur and Karthik Narasimhan. Multimodal graph networks for compositional generalization in visual question answering. *Advances in Neural Information Processing Systems*, 33:3070–3081, 2020.

[147] Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, Richang Hong, and Tat-Seng Chua. Mmgcn: Multi-modal graph convolution network for personalized recommendation of micro-video. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 1437–1445, 2019.

[148] Xiaocui Yang, Shi Feng, Yifei Zhang, and Daling Wang. Multimodal sentiment detection based on multi-channel graph neural networks. In *International Joint Conference on Natural Language Processing*, pages 328–339, 2021.

[149] D Tang, J Zeng, and M Li. An end-to-end deep learning framework for speech emotion recognition of atypical individuals. In *INTERSPEECH*, pages 162–166, 2018.

[150] Z Zhao, Y Zheng, Z Zhang, H Wang, Y Zhao, and C Li. Exploring spatio-temporal representations by integrating attention-based bidirectional-lstm-rnns and fcns for speech emotion recognition. In *INTERSPEECH*, pages 272–276, 2018.

[151] C-W Huang and S S Narayanan. Attention assisted discovery of sub-utterance structure in speech emotion recognition. In *INTERSPEECH*, pages 1387–1391, 2016.

[152] Z Aldeneh, S Khorram, D Dimitriadis, and E M Provost. Pooling acoustic and lexical features for the prediction of valence. In *Proc. International Conference on Multimodal Interaction*, pages 68–72, 2017.

[153] S Mao, PC Ching, and T Lee. Deep learning of segment-level feature representation with multiple instance learning for utterance-level speech emotion recognition. *INTERSPEECH*, pages 1686–1690, 2019.

[154] Wenjing Han, Huabin Ruan, Xiaomin Chen, Zhixiang Wang, Haifeng Li, and Björn W Schuller. Towards temporal modelling of categorical speech emotion recognition. In *Interspeech*, pages 932–936, 2018.

[155] J Gao, T Zhang, and C Xu. Graph convolutional tracking. In *CVPR*, pages 4649–4659, 2019.

[156] L Yao, C Mao, and Y Luo. Graph convolutional networks for text classification. In *AAAI*, volume 33, pages 7370–7377, 2019.

[157] S Zhang, Y Qin, K Sun, and Y Lin. Few-shot audio classification with attentional graph neural networks. *INTERSPEECH*, pages 3649–3653, 2019.

[158] D I Shuman, S K Narang, P Frossard, A Ortega, and P Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, May 2013.

[159] A Ortega, P Frossard, J Kovačević, J MF Moura, and P Vandergheynst. Graph signal processing: Overview, challenges, and applications. *Proceedings of the IEEE*, 106(5):808–828, 2018.

[160] Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N Chang, Sungbok Lee, and Shrikanth S Narayanan. Iemocap: Interactive emotional dyadic motion capture database. *Language resources and evaluation*, 42(4):335, 2008.

[161] J Gideon, S Khorram, Z Aldeneh, D Dimitriadis, and E M Provost. Progressive neural networks for transfer learning in emotion recognition. In *INTERSPEECH*, pages 1098–1102, 2017.

[162] C Busso, S Parthasarathy, A Burmania, M AbdelWahab, N Sadoughi, and E M Provost. MSP-IMPROV: An acted corpus of dyadic interactions to study emotion perception. *IEEE Transactions on Affective Computing*, 8(1):67–80, 2016.

[163] Björn Schuller, Stefan Steidl, and Anton Batliner. The interspeech 2009 emotion challenge. In *Tenth Annual Conference of the International Speech Communication Association*, 2009.

[164] F Eyben, F Weninger, F Gross, and B Schuller. Recent developments in opensmile, the munich open-source multimedia feature extractor. In *ACM international conference on Multimedia*, pages 835–838, 2013.

[165] K Mangalam and T Guha. Learning spontaneity to improve emotion recognition in speech. In *INTERSPEECH*, pages 946–950, 2018.

[166] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

[167] X Ma, Z Wu, J Jia, M Xu, H Meng, and L Cai. Speech emotion recognition with emotion-pair based framework considering emotion distribution information in dimensional emotion space. In *INTERSPEECH*, pages 1238–1242, 2017.

[168] D Luo, Y Zou, and D Huang. Investigation on joint representation learning for robust feature extraction in speech emotion recognition. In *INTERSPEECH*, pages 152–156, 2018.

[169] Huibin Li, Jian Sun, Zongben Xu, and Liming Chen. Multimodal 2d+ 3d facial expression recognition with deep fusion convolutional neural network. *IEEE Transactions on Multimedia*, 19(12):2816–2831.

[170] Xianzhang Pan, Guoliang Ying, Guodong Chen, Hongming Li, and Wenshu Li. A deep spatial and temporal aggregation framework for video-based facial expression recognition. *IEEE Access*, 7:48807–48815, 2019.

[171] Srinivas Parthasarathy and Carlos Busso. Semi-supervised speech emotion recognition with ladder networks. *IEEE/ACM transactions on audio, speech, and language processing*, 28:2697–2709, 2020.

[172] Arthur Crenn, Alexandre Meyer, Rizwan Ahmed Khan, Hubert Konik, and Saïda Bouakaz. Toward an efficient body expression recognition based on the synthesis of a neutral movement. In *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, pages 15–22, 2017.

[173] Yaxiong Ma, Yixue Hao, Min Chen, Jincai Chen, Ping Lu, and Andrej Košir. Audio-visual emotion fusion (avef): A deep efficient weighted approach. *Information Fusion*, 46:184–192, 2019.

[174] Wei Zhang, Xuanyu He, and Weizhi Lu. Exploring discriminative representations for image emotion recognition with cnns. *IEEE Transactions on Multimedia*, 22(2):515–523, 2019.

[175] Haimin Zhang and Min Xu. Recognition of emotions in user-generated videos with kernelized features. *IEEE Transactions on Multimedia*, 20(10):2824–2835, 2018.

[176] Gaurav Verma, Eeshan Gunesh Dhekane, and Tanaya Guha. Learning affective correspondence between music and image. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3975–3979, 2019.

[177] Min Kyu Lee, Dong Yoon Choi, Dae Ha Kim, and Byung Cheol Song. Visual scene-aware hybrid neural network architecture for video-based facial expression recognition. In *International Conference on Automatic Face & Gesture Recognition (FG)*, pages 1–8, 2019.

[178] Feng Mao, Xiang Wu, Hui Xue, and Rong Zhang. Hierarchical video frame sequence representation with deep convolutional graph network. In *European Conference on Computer Vision (ECCV)*, pages 262–270, 2018.

[179] Youngjoo Seo, Michaël Defferrard, Pierre Vandergheynst, and Xavier Bresson. Structured sequence modeling with graph convolutional recurrent networks. In *International Conference on Neural Information Processing (NeurIPS*, pages 362–373, 2018.

[180] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.

[181] Florian Eyben, Martin Wöllmer, and Björn Schuller. Opensmile—introducing the munich open-source emotion and affect recognition toolkit. In *2009 3rd international conference on affective computing and intelligent interaction and workshops*, pages 1–6. IEEE, 2009.

[182] Karttikeya Mangalam and Tanaya Guha. Learning spontaneity to improve emotion recognition in speech. *Proc. Interspeech 2018*, pages 946–950, 2018.

[183] Sayan Ghosh, Eugene Laksana, Louis-Philippe Morency, and Stefan Scherer. Representation learning for speech emotion recognition. In *Interspeech*, pages 3603–3607, 2016.

[184] Jiawang Liu and Haoxiang Wang. Graph isomorphism network for speech emotion recognition. In Hynek Hermansky, Honza Cernocký, Lukás Burget, Lori Lamel, Odette Scharenborg, and Petr Motlícek, editors, *Interspeech 2021, 22nd Annual Conference of the International Speech Communication Association, Brno, Czechia, 30 August - 3 September 2021*, pages 3405–3409. ISCA, 2021.

[185] Yuning You, Tianlong Chen, Zhangyang Wang, and Yang Shen. When does self-supervision help graph convolutional networks? In *International Conference on Machine Learning (ICML)*, pages 10871–10880, 2020.

[186] Qikui Zhu, Bo Du, and Pingkun Yan. Self-supervised training of graph convolutional networks. *arXiv preprint arXiv:2006.02380*, 2020.

[187] Emily Alsentzer, Samuel G. Finlayson, Michelle M. Li, and Marinka Zitnik. Subgraph neural networks. In *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[188] Weilin Cong, Rana Forsati, Mahmut Kandemir, and Mehrdad Mahdavi. Minimal variance sampling with provable guarantees for fast training of graph neural networks. In *International Conference on Knowledge Discovery & Data Mining (ICDM)*, pages 1393–1403, 2020.

[189] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. Graphsaint: Graph sampling based inductive learning method. In *International Conference on Learning Representations (ICLR)*, 2020.

[190] Kexin Huang and Marinka Zitnik. Graph meta learning via local subgraphs. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[191] William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 1024–1034, 2017.

[192] Xugang Lu, Yu Tsao, Shigeki Matsuda, and Chiori Hori. Speech enhancement based on deep denoising autoencoder. In *INTERSPEECH*, volume 2013, pages 436–440, 2013.

[193] Bahare Fatemi, Layla El Asri, and Seyed Mehran Kazemi. Slaps: Self-supervision improves structure learning for graph neural networks. *Advances in Neural Information Processing Systems*, 34, 2021.

[194] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 776–780, 2017.

[195] Shawn Hershey, Sourish Chaudhuri, Daniel PW Ellis, Jort F Gemmeke, Aren Jansen, R Channing Moore, Manoj Plakal, Devin Platt, Rif A Saurous, Bryan Seybold, et al. CNN architectures for large-scale audio classification. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 131–135, 2017.

[196] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[197] Wei Dai, Chia Dai, Shuhui Qu, Juncheng Li, and Samarjit Das. Very deep convolutional neural networks for raw waveforms. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 421–425. IEEE, 2017.

[198] Hassan Akbari, Linagzhe Yuan, Rui Qian, Wei-Hong Chuang, Shih-Fu Chang, Yin Cui, and Boqing Gong. VATT: Transformers for multimodal self-supervised learning from raw video, audio and text. *arXiv preprint arXiv:2104.11178*, 2021.

[199] Jian Huang, Ya Li, Jianhua Tao, Zhen Lian, et al. Speech emotion recognition from variable-length inputs with triplet loss function. In *Interspeech*, pages 3673–3677, 2018.

[200] Jingjun Liang, Ruichen Li, and Qin Jin. Semi-supervised multi-modal emotion recognition with cross-modal distribution matching. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 2852–2861, 2020.

[201] Dongwei Jiang, Wubo Li, Miao Cao, Ruixiong Zhang, Wei Zou, Kun Han, and Xiangang Li. Speech SIMCLR: Combining contrastive and reconstruction objective for self-supervised speech representation learning. *arXiv preprint arXiv:2010.13991*, 2020.

[202] Yuan Gong, Cheng-I Jeff Lai, Yu-An Chung, and James Glass. Ssast: Self-supervised audio spectrogram transformer. *arXiv preprint arXiv:2110.09784*, 2021.

[203] Neil Scheidwasser-Clow, Mikolaj Kegler, Pierre Beckmann, and Milos Cernak. Serab: A multi-lingual benchmark for speech emotion recognition. *arXiv preprint arXiv:2110.03414*, 2021.

[204] Shu-wen Yang, Po-Han Chi, Yung-Sung Chuang, Cheng-I Jeff Lai, Kushal Lakhotia, Yist Y Lin, Andy T Liu, Jiatong Shi, Xuankai Chang, Guan-Ting Lin, et al. Superb: Speech processing universal performance benchmark. *arXiv preprint arXiv:2105.01051*, 2021.

[205] Bo-Hao Su, Chun-Min Chang, Yun-Shao Lin, and Chi-Chun Lee. Improving speech emotion recognition using graph attentive bi-directional gated recurrent unit network. *INTERSPEECH*, pages 506–510, 2020.

[206] Shuiyang Mao, P. C. Ching, C.-C. Jay Kuo, and Tan Lee. Advancing multiple instance learning with attention modeling for categorical speech emotion recognition. In *INTERSPEECH*, 2020.

[207] Lu Yi and Man-Wai Mak. Improving speech emotion recognition with adversarial data augmentation network. *IEEE Trans on Neural Networks and Learning Systems*, 2020.

[208] Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *AAAI Conference on Artificial Intelligence*, volume 34, pages 3438–3445, 2020.

[209] Anurendra Kumar, Tanaya Guha, and Prasanta Kumar Ghosh. Dirichlet latent variable model: A dynamic model based on Dirichlet prior for audio

processing. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(5):919–931, 2019.

[210] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6202–6211, 2019.

[211] Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding. In *Proceedings of The Web Conference 2020*, pages 2331–2341, 2020.

[212] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous graph transformer. In *Proceedings of The Web Conference 2020*, pages 2704–2710, 2020.

[213] Qingsong Lv, Ming Ding, Qiang Liu, Yuxiang Chen, Wenzheng Feng, Siming He, Chang Zhou, Jianguo Jiang, Yuxiao Dong, and Jie Tang. Are we really making much progress? revisiting, benchmarking and refining heterogeneous graph neural networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1150–1160, 2021.

[214] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning for video understanding. *arXiv preprint arXiv:1712.04851*, 1(2):5, 2017.

[215] Tengda Han, Weidi Xie, and Andrew Zisserman. Self-supervised co-training for video representation learning. *Advances in Neural Information Processing Systems*, 33:5679–5690, 2020.

[216] Shenda Hong, Yanbo Xu, Alind Khare, Satria Priambada, Kevin Maher, Alaa Aljiffry, Jimeng Sun, and Alexey Tumanov. Holmes: health online model ensemble serving for deep learning models in intensive care units. In *ACM SIGKDD*, pages 1614–1624, 2020.

[217] Steven R Livingstone and Frank A Russo. 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, 2018.

[218] Jinwei Gu, Xiaodong Yang, Shalini De Mello, and Jan Kautz. Dynamic facial analysis: From bayesian filtering to recurrent neural network. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1548–1557, 2017.

[219] Adrian Bulat and Georgios Tzimiropoulos. How far are we from solving the 2d & 3d face alignment problem?(and a dataset of 230,000 3d facial landmarks). In *International Conference on Computer Vision (ICCV)*, pages 1021–1030, 2017.

[220] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5-6):602–610, 2005.

[221] Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. Lstm-based deep learning models for non-factoid answer selection. *International Conference on Learning Representations (ICLR) workshop*, 2016.

[222] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7132–7141, 2018.

[223] Yongjin Wang, Ling Guan, and Anastasios N Venetsanopoulos. Kernel cross-modal factor analysis for information fusion with application to bimodal emotion recognition. *IEEE Transactions on Multimedia*, 14(3):597–607, 2012.

[224] Kah Phooi Seng, Li-Minn Ang, and Chien Shing Ooi. A combined rule-based & machine learning audio-visual emotion recognition approach. *IEEE Transactions on Affective Computing*, 9(1):3–13, 2016.

[225] Esam Ghaleb, Mirela Popa, and Stylianos Asteriadis. Multimodal and temporal perception of audio-visual cues for emotion recognition. In *International Conference on Affective Computing & Intelligent Interaction (ACII)*, 2019.

[226] Ekaterina Volkova, Stephan De La Rosa, Heinrich H Bülthoff, and Betty Mohler. The mpi emotional body expressions database for narrative scenarios. *PloS one*, 9(12), 2014.

[227] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–93, 2008.

[228] Galileo Namata, Ben London, Lise Getoor, and Bert Huang. Query-driven active surveying for collective classification. In *International Workshop on Mining and Learning with Graphs*, volume 8, 2012.

[229] Vijay Prakash Dwivedi, Chaitanya K Joshi, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *arXiv preprint arXiv:2003.00982*, 2020.

[230] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[231] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations (ICLR)*, 2019.

[232] Vikas Verma, Meng Qu, Kenji Kawaguchi, Alex Lamb, Yoshua Bengio, Juho Kannala, and Jian Tang. Graphmix: Improved training of GNNs for semi-supervised learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10024–10032, 2021.