Deep Learning-based Audio Source Separation, Recognition, and Information Extraction for Multimedia Analysis

March 2020

Naoya Takahashi

Deep Learning-based Audio Source Separation, Recognition, and Information Extraction for Multimedia Analysis

Graduate School of Systems and Information Engineering

University of Tsukuba

March 2020

Naoya Takahashi

Abstract

As a vast number of multimedia become accessible, machine multimedia understanding becomes more and more important to retrieve information, summarize contents, or answer to a query. Currently, such tasks largely rely on label made by human which is prohibitively time consuming and can be in-accurate. To automatize such tasks, it is important to analyze multimedia contents itself by machines. Recent advance of deep learning greatly push the machine based multimedia analysis forward, yet the success is mostly limited to some domains that have large dataset and the analysis is often done in controlled or limited environment. For instance, most audio recognition works focus on speech of major languages that have large dataset with manual annotation, and the speech is often in controlled and non-overlapping environments. In this thesis, we focused on audio and video analysis, and address the variety of problems which is essential for realizing machine multimedia understanding in real-world, possibly not in a controlled environment. Specifically, we address the problems of handling overlapping sounds, low-resource dataset, and making use of non-speech sound for video analysis and propose a novel methods on audio source separation, automatic speech recognition for low-resource languages, audio event recognition, and audio visual video analysis including action recognition and highlight detection.

Acknowledgements

I thank my supervisor Prof. Shoji Makino for guiding my doctoral course. His concise advice and guidance helps me a lot to write this thesis. I also thank Prof. Takeshi Yamada for having nice discussion and giving me good advice at seminars which helps to organise this thesis. I appreciate helpful and thoughtful advice and comments from Prof. Kazuhiro Fukui, Prof. Keisuke Kameyama from Engineering, Information and Systems, and Prof. Nobutaka Ono from Tokyo Metropolitan University. I also would like to thank my colleagues at Sony, especially Mr. Yuki Mitsufuji who recommended and understood to go to the doctoral course while working. Last but not least, I thank my wife for her warm support by willingly doing housework and taking care my one year old sun. I could not finish thesis without her support and therefore, I dedicate this thesis to my family.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

As mobile device become omnipresent, vast number of multimedia becomes available on the internet. Multimedia can be seen as a tool or another dimension for extending human cognition beyond the limitation of time and space since one can know what happen there at that time through multimedia even the place and time the multimedia recorded is physically not accessible. However, it is prohibitively time consuming for human to go through vast number of videos and extract necessary, interesting information. Machine can help to extend our sense and enrich our activities by extracting only relevant information for individuals. Currently, multimedia search or recommendation systems largely relies on human annotation or statistics of viewing/listening history. Such information could be too abstract, take long time to collect or expensive to obtain. Moreover, the current search recommendation systems mostly propose item wise, i.e. in video search case, the system only propose video clips but do not tell which parts of the video are most related. If machine instead understands multimedia itself as humans do rather than relying on human annotations, it would greatly improve the accuracy of recommendation, search, summarization or prediction. Therefore, multimedia analysis such as scene recognition, concept classification, speech recognition, action recognition and highlight detection have become more and more important. This thesis aims at providing sets of audio and audio-visual technologies to realize the machine multimedia understanding that directly analyze multimedia data itself in real-world scenarios, rather than relying on human annotations. We focus on audio and video as multimedia contents since these are most widely shared, common multimedia beside text and image data.

1.2 Objectives

- what is the left Indian guy in this video saying?
- show me some videos in which two dogs are howling on the beach.
- create 1 min highlight video of my skate boarding from my archive.

To answer these questions or commands, the machine has to overcome many difficulties. For the first question, there might be several persons talking at the same time in the video and the machine has to selectively recognize the specified person's speech. The person may speak minor Indian local language, which is difficult to develop an automatic speech recognition (ASR) system due to the lack of human annotated training data. For answering the second command, the machine needs to recognize not only objects but also background and actions which emit sounds. The last command requests machine to understand an abstract context and finely time aligned scene analysis. Despite recent advance of deep learning based approach, it is still struggle to answer these simple questions and commands: application of deep learning is still limited to some domains such as controlled/limited environment, and/or domains that have large dataset. This thesis aim at

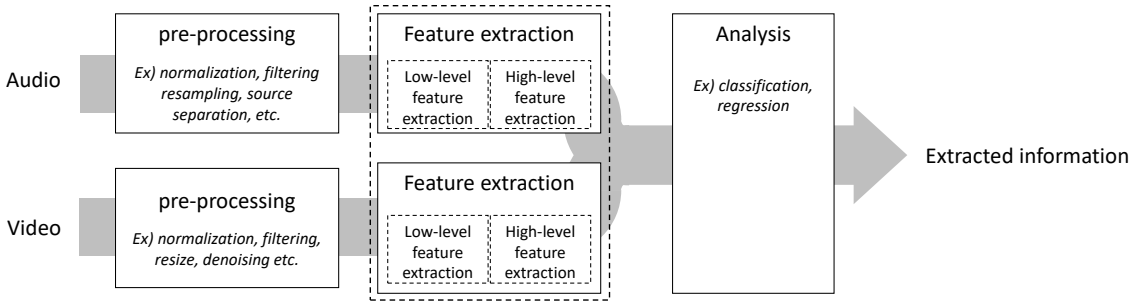

Figure 1.1: Pipeline of multimedia analysis. Here, we focused on audio and video. Feature extraction part can be processed separately, or merged.

machine multimedia understanding in such realistic and challenging scenarios and providing sets of techniques to address them.

1.3 Challenges

Figure 1.1 shows a general pipeline for the multimedia analysis, especially the cases of audio and video. The data is first preprosessed to fit to the following processes. The prepossessing could include normalization, filtering, resize/resampling and denoising. In addition to that, in audio case, audio source separation can be also included when multiple sources are overlapped and we wish to process each source individually, e.g. if two speakers speak at the same time,it is necessary to separate the voices to transcribe individually. The feature extraction part can be further split to low-level feature extraction and high level feature extraction. The low-level features are often designed manually using expert knowledge. Examples of the low-level features for audio could be the mel-frequency cepstrum coefficients (MFCC) or fbank features, while visual low-level features could be optical flow, HOG, SURF, SIFT. The high-level feature extraction will be designed to capture more abstract information which is more directly correlated with the final information to be recognized. Recently, deep neural networks (DNN) are commonly used for the high level feature extraction. It has become popular to combine low- and high-level feature extraction part and use single DNN to extract and analyze data in end-to-end manner in vision domain. On the other hand, in, audio domain, the low-level feature extraction is often incorporated, although approaches directly working on waveform are actively investigated nowadays. The analysis part extract the desired information from the features, e.g. transcribed text in the case of ASR, or highlight score in the case of the video highlight detection.

This thesis addresses problems of all three components and focuses on four major tasks:

1. **Audio source separation (SS)** In real world, many sound sources emit sounds simultaneously. In this case, it is necessary to separate desired source signal from others to avoid a confusion due to signals from other sources. For example, if multiple speakers speak simultaneously, the machine has to first separate voices to transcribe each speaker individually. Or if we wish to transcribe vocal melody line in the music robustly, separating the singing voice from other musical sound greatly helps to improve the accuracy. However, the audio source separation problem has been considered very challenging due to the nature of highly ill-posed problem. In this thesis, we consider different types of audio, namely speech, music, and noise, and propose methods that achieve state-of-the-art separation accuracy.

2. **Automatic speech recognition (ASR)** Speech is arguably one of the most informative and important signal for human. Automatic speech recognition (ASR) has been investigated since long time ago, as it is an essential component for a natural man-machine communication. Although recent advance of deep learning technique greatly improve the accuracy of ASR when the large corpus is available, ASR still struggles when a large corpus is not

available which is the case for many local languages and dialects. The main challenge is obtaining reliable pronunciation dictionary, which map words to sequences of sub-word units, from limited available resources. Manual preparation of such resources requires significant investment and expertise. Therefore, an automatic generation of pronunciation dictionary from the data is clearly required for many dialects and languages.

3. **Audio event recognition (AER)** Beside speech, non-speech sounds such as music or laughter provide important information as well. In most conversations no mention is made of the environment, like its location or people and objects present. Many application including ASR, video analysis and man-machine interaction could benefit from having such contextual knowledge. Furthermore, multi-media tasks such as video classification [1] and video summarization [2] have been shown to improve when including audio information. Compare to ASR, audio event recognition is less investigated, especially for DNN based method. It was partially due to the lack of large dataset. Few works directly applied ASR approach to AER, et, applying standard ASR approaches leads to inferior performance due to differences between speech and non-speech signals. Here, we introduce a novel approach for AER.

4. **Video analysis** Video analysis such as concept classification [3, 4, 5], action recognition [6, 7] and highlight detection [8] have become more and more important to retrieve [9, 10, 11] or summarize [12] videos for efficient browsing. Most DNN works on video analysis relies on visual cues only and audio is often not used at all. However, beside visual information, humans greatly rely on their hearing for scene understanding. Audio is clearly one of the key components for video analysis. Many works showed that audio and visual streams contain complementary information [5, 6], e.g. because audio is not limited to the line-of-sight.

   However, combining visual features and traditional audio features used for ASR such as MFCC typically only leads to marginal improvements. Instead, we propose a novel audio feature that extracted from AER model, which leads to significant performance improvements on action recognition and video highlight detection.

1.4 Contributions

In this thesis, challenges on machine multimedia understanding in realistic environment are broken down into aforementioned four technical domains and We provide sets of methods to overcome challenges in each domain. The summary of contributions in each domain is as follows:

1. **Audio source separation** We investigate a neural network architecture that consider several aspects of audio source separation and proposed novel neural network architectures that achieve state-of-the-art accuracy in music separation task. We also provide a novel method to recover target phase from mixture phase that is corrupted by other sources. We also investigate the problem of separating speech of multiple speakers and propose a novel method to separate speech of unknown number of speakers with single model.

2. **Speech recognition** We propose a data-driven pronunciation estimation and acoustic modeling method which only takes the orthographic transcription to jointly estimate a set of sub-word units and a reliable dictionary. This provide a way to develop ASR without manual preparation of a pronunciation dictionary which requires considerable investment and expertise.

3. **Audio event recognition** We introduce novel network architectures for AER with up to 9 layers and a large input field which allows the networks to directly model entire audio events and to be trained end-to-end. We also propose a data augmentation method which helps to prevent over-fitting. Moreover, We built an audio event dataset which contains a variety of sound events which may occur in consumer videos. The dataset was used to train the networks for generic audio feature extraction. We also make the pre-trained model available so that the research community can easily utilize our proposed features.

4. **Video analysis** We introduce AENet feature to leverage audio information more effectively and improve video analysis. We conducted experiments on different kinds of consumer video tasks, namely action recognition and video highlight detection, to show the superior performance and generality of the proposed features. To the best of our knowledge, this is the first work on consumer video highlight detection taking advantage of audio. On all tasks we outperform the state of the art results by leveraging the proposed audio features.

1.5 Related Works

1.5.1 Audio Source Separation (SS)

Audio source separation (SS) has recently been intensively studied. Various approaches have been introduced such as local Gaussian modeling [13, 14], non-negative factorization [15, 16, 17], kernel additive modeling [18] and their combinations [19, 20, 21]. Recently, deep neural network (DNN) based SS methods have shown a significant improvement over conventional methods. In [22, 23], a standard feedforward fully connected network (FNN) was used to estimate source spectra. A common way to exploit temporal contexts is to concatenate multiple frames as the input. However, the number of frames that can be used is limited in practice to avoid the explosion of the model size. In [24], long short-term memory (LSTM), a type of recurrent neural network (RNN), was used to model longer contexts. However, the model size tends to become excessively large and the training becomes slow owing to the full connection between the layers and the gate mechanism in an LSTM cell.

Although previous research on SS based on DNN mainly focuses on estimating the magnitude spectrum of target sources, there are several works on target phase recovering. One approach to phase estimation is to promote *consistency* [25, 26], where it modifies the mixture phase depending on the results of the estimated magnitude such that the modified phase satisfies *consistency*. Some recent works [27, 28, 29] attempted to combine Wiener filtering with consistency-based techniques. The extension of the above approach incorporating sinusoid models has shown promising results [30]. However, the consistency constrain itself is not directly designed to recover the target phase. There are few works that attempt to recover magnitude and phase concurrently. Williamson *et al.* proposed a twin-head DNN to infer both real and imaginary parts of the target spectrogram [31]. Several authors attempted to construct a fully complex-valued network by updating parameters based on complex back propagation [32, 33]. However, to achieve good performance, the network needs to be constrained by sparsity. Moreover, the currently available DNN frameworks such as PyTorch and Tensorflow do not support complex back propagation, thus preventing us from using the various modules that the framework supports.

There are also works on separating same type of sources, especially on speech separation. Previously, various approaches including spectral clustering [34] computational auditory scene analysis (CASA) [35], non-negative matrix factorization (NMF)[36, 37, 38, 39] were proposed to tackle this problem, yet showed limited success. Recent advances of deep learning based methods including deep clustering (DPCL)[40, 41], permutation invariant training (PIT) [42, 43, 44], deep attractor network (DANet) [45, 46] dramatically improved the accuracy of separation. However, most of these methods assume that the number of speakers is known in advance.

1.5.2 Automatic Speech Recognition (ASR)

Developing ASRs for dialects and under-resourced languages has attracted growing attention over the past few years [47, 48, 49]. A main challenge to develop ASR for under-resourced domains is to produce a reliable pronunciation dictionary from limited available resources. For major languages, however, a canonical pronunciation dictionary is usually already available. However, such dictionaries may be error-prone due to the fact that they are manually generated and in most cases do not cover pronunciation variants. There were several attempts to tackle these problems [50, 51, 52, 53].

Lu et al. [54] proposed a data-driven dictionary generator to include new pronunciations based on newly coming acoustic evidence. Goel et al. in [55] use a grapheme-to-phoneme approach to

guess the pronunciation and iteratively refine the acoustic model and the dictionary. However, these methods still require a high-quality initial pronunciation dictionary created by an expert.

In modern ASRs words are represented by smaller sub-word units such as phonemes and the pronunciation dictionary maps words to sequences of sub-word units. However, sub-word units do not essentially need to be linguistically motivated elements. In fact, given a set of acoustic samples, the linguistically defined units are most probably not the optimal ones for speech recognition [56]. For instance telephony speech, where high frequency components have been filtered out, requires a modified dictionary with slightly different set of fricatives than full-bandwidth speech.

Over the past few years, there have been several attempts to move beyond phoneme based sub-word units by jointly learn a set of sub-word units and their corresponding dictionary directly from the given data [57, 58, 54]. Bacchiani and Ostendorf [58] proposed an iterative acoustic segmentation and clustering approach to build sub-word units from speech signals and subsequently construct the dictionary based on the estimated sub-word units. Singh et al. [54] introduced a divide-and-conquer strategy to recursively update sub-word units and dictionary. The dictionary computation was done by means of an n-best type algorithm which is known to produce sub-optimal solutions. Although their approach demonstrates some promising results, the performance is still not comparable with a phoneme based ASR.

1.5.3 Audio Event Recognition (AER)

Traditional methods for AER apply techniques from ASR directly. For instance, Mel Frequency Cepstral Coefficients (MFCC) were modeled with Gaussian Mixture Models (GMM) or Support Vector Machines (SVM) [59, 60, 4, 61]. Yet, applying standard ASR approaches leads to inferior performance due to differences between speech and non-speech signals. Thus, more discriminative features were developed. Most were hand-crafted and derived from low-level descriptors such as MFCC [62, 63], filter banks [64, 65] or time-frequency descriptors [66]. These descriptors are frame-by-frame representations (typically frame length is in the order of tens of ms) and are usually modeled by GMMs to deal with the sounds of entire audio events that normally last seconds at least. Another common method to aggregate frame level descriptors is the Bag of Audio Words (BoAW) approach, followed by an SVM [63, 67, 68, 69]. These models discard the temporal order of the frame level features however, causing considerable information loss. Moreover, methods based on hand-crafted features optimize the feature extraction process and the classification process separately, rather than learning end-to-end.

Recently, DNN approaches have been shown to achieve superior performance over traditional methods. One advantage of DNNs is their capability to jointly learn feature representations and appropriate classifiers. In [70], a fully connected feed-forward DNN is built on top of MFCC features. Miquel *et al.* [71] utilize a Convolutional Neural Network (CNN) [72] to extract features from spectrograms. Recurrent Neural Networks are also used on top of low-level features such as MFCCs and fundamental frequency [73]. These networks are still relatively shallow (e.g. less than 3 layers). The recent success of deeper architectures in image analysis [74] and ASR [75] hinges on the availability of large amounts of training data. If a training dataset is small, it is difficult to train deep architectures from scratch in order not to over-fit the training set. Moreover, the networks take only a few frames as input and the complete acoustic events are modeled by Hidden Markov Models (HMM) or simply by calculating the mean of the network outputs, which is too simple to model complicated acoustic event structures.

Furthermore, these methods are task specific, i.e. the trained networks cannot be used for other tasks. We conclude that there was still a lack a generic way to represent audio signals. Such a generic representation would be very helpful for solving various audio analysis tasks in a unitary way.

1.5.4 Features for Video Analysis

Traditionally, visual video analysis relied on spatio-temporal interest points, described with low-level features such as SIFT, HOG, HOF, etc. [76, 77]. Given the current interest in learning deep representations through end-to-end training, several methods using convolutional neural networks

(CNN) have been proposed recently. Karpathy *et al.*introduced a large-scale dataset for sports classification in videos [78]. They investigated ways to improve single frame CNNs by fusing spatial features over multiple frames in time. Wang *et al.* [79] combine the trajectory pooling of [77] with CNN features. The best performance is achieved by combining RGB with motion information obtained through optical flow estimation [80, 81, 82], but this comes at a higher computational cost. A compromise between computational efficiency and performance is offered by C3D [83], which uses spatio-temporal 3D convolutions to encode appearance and motion information.

Many efforts have been dedicated to incorporate audio in video analysis by using audio only [4] or fusing audio and visual information [6]. In audio-based video analysis, feature extraction remains a fundamental problem. Many types of low level features such as short-time energy, zero crossing rate, pitch, frequency centroid, spectral flax, and Mel Frequency Cepstral Coefficients (MFCC) [3, 4, 7, 11] have been investigated. These features are very low level or not designed for video analysis, however. For instance, MFCC has originally been designed for automatic speech recognition (ASR), where it attempts to characterize phonemes which last tens to hundreds of milliseconds. While MFCC is often used as an audio feature for the aural detection of events, their audio characteristics differ from those of speech. Such sounds are not always stationary and some audio events could only be detected based on several seconds of sound. Thus, a more discriminative and generic feature set, capturing longer temporal extents, is required to deal with the wide range of sounds occurring in videos.

Another common method to represent audio signals is the Bag of Audio Words (BoAW) approach [63, 69, 5], which aggregates frame features such as MFCC into a histogram. BoAW discards the temporal order of the frame level features, thus suffering from considerable information loss.

1.5.5 Transfer Learning

Our works on video analysis is also related to transfer learaning. The success of deep learning is driven, in part, by large datasets such as ImageNet [84] or Sports1M [78]. These kinds of datasets are, however, only available in a limited set of research areas. Naturally, the question of whether CNN representations are transferable to other tasks arose [85, 86]. Indeed, as these works have shown, using CNN features trained on ImageNet provides performance improvements in a large array of tasks such as attribute detection or image and object instance retrieval, compared to traditional features [86]. Several follow-up works have analysed pooling mechanisms for improved domain transfer, *e.g.* [87, 88, 89]. Video CNN features have also been successfully transferred to other tasks [83, 90]. For this work, we have been inspired by these works and propose deep convolutional features trained on audio event recognition and that are transferable to video analysis. To the best of our knowledge, no other such features exist to date.

1.6 Structure of this Thesis

Reset of the thesis is organized according to the processing flow described in Figure1.1 and organized as follow: In chapter 2, we address problems in the audio source separation tasks. In chapter 3, ASR for under-resourced languages is discussed. Then we move on to the problem of AER. In chapter 5, we discuss how the AER model described in Chapter 4 is used for video analysis. Finally, we conclude the thesis in Chapter 6.

Chapter 2

Audio Source Separation

Audio Source Separation (SS) typically operates on magnitude spectra domain. We first investigate the DNN architectures that estimate the target magnitude spectrogram from the mixture spectrogram and propose a novel architectures devised for SS problem. Then, we investigate the phase reconstruction problem in Section 2.2. Finally, we address the problem of separating unknown number of speakers in Section 2.3.

2.1 DNN Architecture for Audio Source Separation

Early stage of deep learning based SS model used a standard feedforward fully connected network (FNN) to estimate source spectra [23]. A common way to exploit temporal contexts is to concatenate multiple frames as the input. However, the number of frames that can be used is limited in practice to avoid the explosion of the model size. In [24], long short-term memory (LSTM), was used to model longer contexts. However, the model size tends to become excessively large and the training becomes slow owing to the full connection between the layers and the gate mechanism in an LSTM cell. Recently, convolutional neural networks (CNNs) [72] have been successfully applied to audio modeling of spectrograms [91, 92], although CNNs were originally introduced to address the transition-invariant property of images. A CNN significantly reduces the number of parameters and improves generalization by sharing parameters to model local patterns in the input. However, a standard CNN requires considerable depth to cover long contexts, making training difficult. Another problem in applying a two dimensional convolution to a spectrogram is the biased distribution of the local structure in the spectrogram. Unlike an image, a spectrogram has different local structures depending on the frequency bands. Complete sharing of convolutional kernels over the entire frequency range may reduce modeling flexibility. To address these problems, we propose MMDenseNet architecture in the next section. Then, we further propose an improved architecture, MMDenseLSTM, which combines LSTM and DenseNet in multiple scales and multiple bands, improving separation quality while retaining a small model size.

2.1.1 DenseNet

In a standard feed forward network, the output of the lth layer is computed as $x_l = H_l(x_{l-1})$, where the network input is denoted as x_0 and $H_l(\cdot)$ is a non-linear transformation which can be a composite function of operations such as Batch Normalization (BN) [93], rectified linear units (ReLU) [94], pooling, or convolution. In order to mitigate difficulties of training very deep models, ResNet [95] employs a skip connection which adds an identity mapping of the input to the non-linear transformation:

$$x_l = H_l(x_{l-1}) + x_{l-1}. \tag{2.1}$$

The skip connection allows the network to propagate the gradient directly to the preceding layers, making the training of deep architectures easier. DenseNet [96] further improves the information

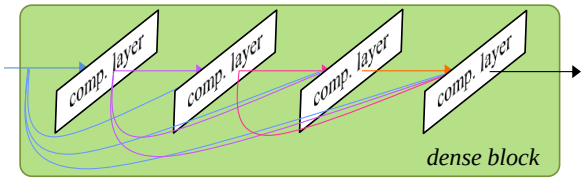

Figure 2.1: dense block architecture. The input of a composite layer is the concatenation of outputs of all preceding layers.

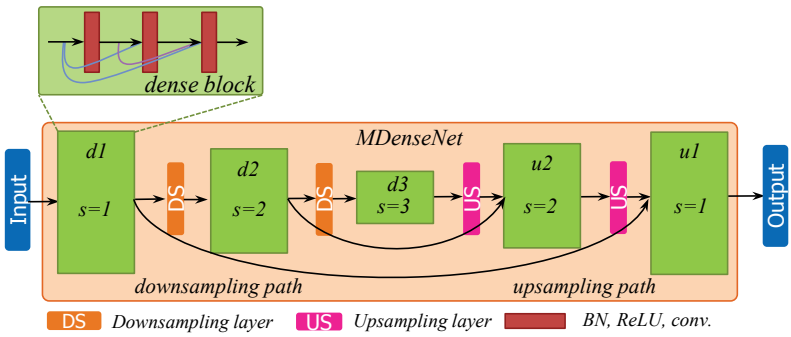

Figure 2.2: MDenseNet architecture. Multi-scale dense blocks are connected though down- or up-sampling layer or through block skip connections. The figure shows the case $s = 3$.

flow between layers by replacing the simple addition of the output of a single preceding layer with a concatenation of all preceding layers:

$$x_l = H_l([x_{l-1}, x_{l-2}, \ldots, x_0]), \tag{2.2}$$

where $[\ldots]$ denotes the concatenation operation. Such dense connectivity enables all layers not only to receive the gradient directly but also to reuse features computed in preceding layers. This avoids the re-calculation of similar features in different layers, making the network highly parameter efficient. Fig. 2.1 illustrate the dense block. In DenseNet, H_l comprises of BN, followed by ReLU and convolution with k feature maps. In the reminder of this paper, k is referred to as *growth rate* since the number of input feature maps grows linearly with depth in proportion to k (e.g. the input of lth layer have $l \times k$ feature maps).

For image recognition tasks, a pooling layer, which aggregates local activation and maps to the lower dimension, is essential to capture the global information efficiently. A *down-sampling layer* defined as a 1×1 convolution followed by a 2×2 average pooling layer is introduced to facilitate pooling. By alternately connecting dense blocks and down-sampling layers, the feature map dimension is successively reduced and finally fed to a softmax classification layer after global pooling layer. In the next section, We discuss how to apply these ideas to audio source separation.

2.1.2 Multi-scale DenseNet

Dense blocks and down-sampling layers comprise the down-sampling path of the proposed multi-scale DenseNet. Down-sampled feature maps enable the dense block network to model longer contexts and wider frequency range dependency while alleviating computational expense. In order to recover the original resolution from lower resolution feature maps, we introduce an up-sampling layer defined as a transposed convolution whose filter size is same as the pooling size. We again alternate up-sampling layers and dense blocks to successively recover the higher resolution feature maps. In order to allow forward and backward signal flow without passing though lower resolution blocks, we also introduce *inter-block skip connection* which directly connect two dense blocks of the same scale. With this connection, dense blocks in the down-sampling path are enabled to receive supervision and send the extracted features without compressing and decompressing them. The idea of the entire architecture is depicted in Fig. 2.2 in case that the number of different scales s

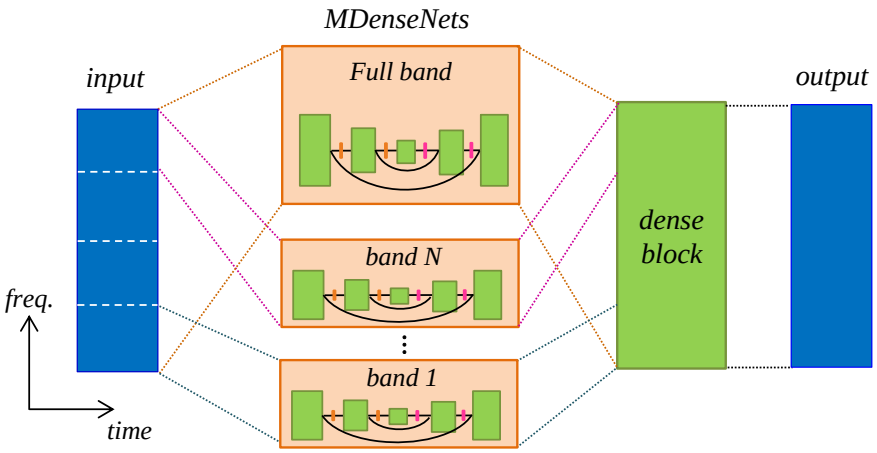

Figure 2.3: MMDenseNet architecture. Outputs of MDenseNets dedicated for each frequency band including full band are concatenated and the final dense block integrate features from these bands to create final output.

is 3 which can be tuned depends on a data complexity and resource availabilities. Hereafter, we refer to this architecture as *MDenseNet*. Note that the proposed architecture is fully convolutional and thus can be applied to arbitrary input length.

2.1.3 Multi-band Multi-scale DenseNet

In the architecture discussed in Sec. 2.1.2, the kernels of the convolution layer are shared across the entire input field. This is reasonable if the local input patterns appear in any position in the input, as is the case for objects in natural photos. In audio, however, different patterns occur in different frequency bands, though a certain amount of translation of patterns exists, depending on the relatively small pitch shift. Therefore, limiting the band that share the kernels is more suitable for efficiently capturing local patterns. Indeed, limited kernel sharing has been shown to be effective in speech recognition [97]. We split the input into multiple bands and apply multi-scale DenseNet to each band. However, simply splitting frequency band and modeling each band individually may hinder the ability to model the entire structure of spectrogram. Hence, we build in parallel an MDenseNet for the full band input and concatenate its output with outputs from multiple sub-band MDenseNets, as shown in Fig. 2.3. Note that in this architecture, since fine structure can be captured by band limited MDenseNets, the full band MDenseNet can focus on modeling rough global structure, thus simpler and less expensive model can be used. We refer to the architecture as MMDenseNet.

2.1.4 Combining LSTM with MMDenseNet

Blending two systems gives better performance even when one system consistently outperforms the other [24]. The improvement tends to be more significant when two very different architectures are blended such as a CNN and RNN, rather than the same architectures with different parameters. However, the blending of architectures increases the model size and computational cost additively, which is often undesirable when deploying the systems. Therefore, we propose combining the dense block and *LSTM block* in a unified architecture. The *LSTM block* consists of a 1×1 convolution that reduces the number of feature maps to 1, followed by a bi-directional LSTM layer, which treats the feature map as sequential data along the time axis, and finally a feedforward linear layer that transforms back the input frequency dimension f^s from the number of LSTM units m^s. We consider three configurations with different combinations of the dense and LSTM blocks as shown in Fig. 2.4. The *Sa* and *Sb* configurations place the LSTM block after and before the dense block, respectively, while the dense block and LSTM block are placed in parallel and concatenated in the

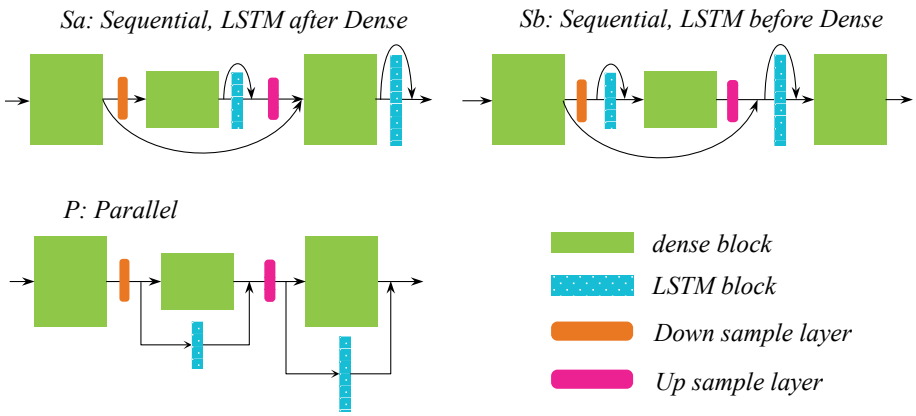

Figure 2.4: Configurations with different combinations of dense and LSTM blocks. LSTM blocks are inserted at some of the scales

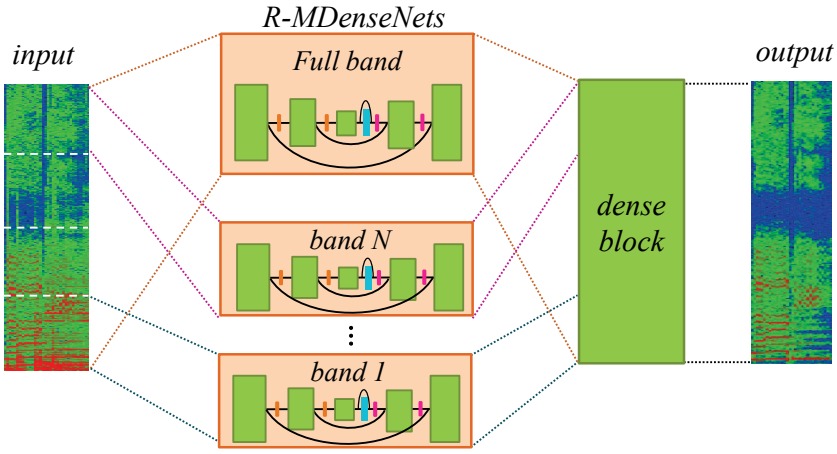

Figure 2.5: MMDenseLSTM architecture. Outputs of MDenseLSTM dedicated to different frequency band including the full band are concatenated and the final dense block integrates features from these bands to create the final output.

P configuration. We focus on the use of the Sa configuration since a CNN is effective at modeling the local structure and the LSTM block benefits from local pattern modeling as it covers the entire frequency at once. This claim will be empirically validated in Sec. 2.1.5.

Naively inserting LSTM blocks at every scale greatly increases the model size. This is mostly due to the full connection between the input and output units of the LSTM block in the scale $s = 1$. To address this problem, we propose the insertion of only a small number of LSTM blocks in the upsampling path for low scales ($s > 1$). This makes it easier for LSTM blocks to capture the global structure of the input with a much smaller number of parameters. On the other hand, a CNN is advantageous for modeling fine local structures; thus placing only dense block at $s = 1$ is suitable. The multi-band structure is also beneficial for LSTM blocks since the compression from the input frequency dimension f^s to m^s LSTM units is relaxed or it even allows the dimension ($f^s < m^s$) to be increased while using fewer LSTM units, increasing the modeling capabilities as discussed in [98]. The entire proposed architecture is illustrated in Fig. 2.5. To capture the pattern that spans the bands, MDenseLSTM for the full band is also built in parallel along with the band dedicated MDenseLSTM. The outputs of the MDenseLSTMs are concatenated and integrated by the final dense block, as MMDenseNet.

Table 2.1: The proposed architectures. All dense blocks are equipped with 3×3 kernels with L layers and k growth rate. The pooling size and transposed convolution kernel size are 2×2.

Layer	scale	MMDenseNet			MDenseNet
		low	high	full	
band split	1	first half	last half	-	-
conv (t×f,ch)		3×4, 32	3×3, 32	3×4, 32	3×4, 32
dense 1 (k,L)		14, 4	10, 3	6, 2	12, 4
down sample	$\frac{1}{2}$	pool	pool	pool	pool
dense 2 (k,L)		16, 4	10, 3	6, 2	12, 4
down sample	$\frac{1}{4}$	pool	pool	pool	pool
dense 3 (k,L)		16, 4	10, 3	6, 2	12, 4
down sample	$\frac{1}{8}$	pool	pool	pool	pool
dense 4 (k,L)		16, 4	10, 3	6, 4	12, 4
up sample	$\frac{1}{4}$	t.conv	t.conv	t.conv	t.conv
concat.		low dense 3	high dense 3	full dense 3	dense 3
dense 5 (k,L)		16, 4	10, 3	6, 2	12, 4
up sample	$\frac{1}{2}$	t.conv	t.conv	t.conv	t.conv
concat.		low dense 2	high dense 2	full dense 2	dense 2
dense 6 (k,L)		16, 4	10, 3	6, 2	12, 4
up sample	1	t.conv	t.conv	t.conv	t.conv
concat.		low dense 1	high dense 1	full dense 1	dense 1
dense 7 (k,L)		16, 4	10, 3	6, 2	12, 4
concat. (axis)	1	freq		-	-
concat. (axis)		channel			
dense 8 (k,L)		4, 2			4, 2
conv(t×f,ch)		1×2, 2			1×2, 2

2.1.5 Experiments

Setup

We evaluated the proposed method on the DSD100 and MUSDB18 datasets, prepared for SiSEC 2016 [99] and SiSEC 2018 [100], respectively. MUSDB18 has 100 and 50 songs while DSD100 has 50 songs each in the *Dev* and *Test* sets. In both datasets, a mixture and its four sources, *bass, drums, other* and *vocals*, recorded in stereo format at 44.1kHz, are available for each song. Short-time Fourier transform magnitude frames of the mixture, windowed at 2048 samples with 50% overlap for MDenseNet and MMDenseNet, and at 4096 samples with 75% overlap for MMDenseLSTM, with data augmentation [24] were used as inputs. The networks were trained to estimate the source spectrogram by minimizing the mean square error with the Adam optimizer. For the evaluation on MUSDB18, we used the *museval* package [100], while we used the BSSEval v3 toolbox [101] for the evaluation on DSD100 for a fair comparison with previously reported results. The SDR values are the median of the average SDR of each song.

Architectural details

Details of the proposed network architectures of MDenseNet, MMDenseNet and MMDenseLSTM are described in Table 2.1 and 2.2, respectively. The effective context window sizes of MDenseNet and MMDenseneNet architectures are 153 frames while that of MMDenseLSTM is 356 frames. One advantage of Multi-band architecture is that we can design suitable architectures for each band individually and assign computational resources according to the importance of each band which may differ depending on the target source or application. We design a relatively larger model for the lower frequency band. For MMDenseNet, we split the frequency into two bands in the middle while for MMDenseLSTM, we split the input into three bands at 4.1kHz and 11kHz. The LSTM blocks of MMDenseLSTM are only placed at bottleneck blocks and at some blocks at $s = 2$ in the upsampling path, which greatly reduces the model size. The final dense block has three layers with growth rate $k = 12$. The effective context size of the architecture is 356 frames. Note that MMDenseLSTM can be applied to an input of arbitrary length since it consists of convolution and LSTM layers.

Table 2.2: The proposed architecture. All dense blocks are equipped with 3×3 kernels with growth rate k. l and m^s denote the number of layer and LSTM units of LSTM block, respectively. ds denotes scale s in the downsampling path while us is that in the upsampling path.

band	k	scale	d1	d2	d3	d4	d5	u4	u3	u2	u1
1	14	l	5	5	5	5	-	-	5	5	5
		m^s	-	-	-	128	-	-	-	128	-
2	4	l	4	4	4	4	-	-	4	4	4
		m^s	-	-	-	32	-	-	-	-	-
3	2	l	1	1	-	-	-	-	-	1	1
		m^s	-	-	8	-	-	-	-	-	-
full	7	l	3	3	4	5	5	5	4	3	3
		m^s	-	-	-	128	-	-	-	128	-

Table 2.3: Comparison of SDR on DSD100 dataset.

Method	SDR in dB				
	Bass	Drums	Other	Vocals	Acco.
DeepNMF [16]	1.88	2.11	2.64	2.75	8.90
NUG [22]	2.72	3.89	3.18	4.55	10.29
FNN [23]	2.54	3.75	2.92	4.47	11.12
BLSTM [24]	2.89	4.00	3.24	4.86	11.26
BLEND [24]	2.98	4.13	3.52	5.23	11.70
MDenseNet	2.74	4.37	3.33	4.91	11.21
MMDenseNet [102]	**3.91**	5.37	3.81	6.00	12.10
MMDenseLSTM	3.73	**5.46**	**4.33**	**6.31**	**12.73**

Comparison with state-of-the-art methods

We compared our method with other state-of-the-art approaches on DSD100 dataset.

- DeepNMF [16]: Non-negative deep network architecture which results from unfolding NMF iterations and untying their parameters.
- NUG [22]: This approach estimates source spectra using DNN, and iteratively updates the spatial and spectral estimates using expectation-maximization. This approach was referred as NUG1 in [22].
- FNN [23]: The source spectra was estimated by feed forward fully connected DNN trained with an additional dataset (MedleyDB [103]). Final outputs were obtained by applying single-channel Wiener filter to each channel individually.
- BLSTM [24]: Three layer bidirectional long short time memory (BLSTM) was used to estimate source spectrogram. This system marked second best score in SiSEC 2016 competition [99] and can be considered as a good baseline since it also uses MWF, thus the performance difference between these system highlight the effect of our proposed network architectures.
- BLEND [24]: This approach linearly blend the estimates of FNN and BLSTM before applying MWF. The best score on SiSEC 2016 competition was obtained with this approach.

The task was to separate the four sources and accompaniment, which is the residual of the vocal extraction, from the mixture. Here, the multichannel Wiener filter was applied to FNN, BLSTM, BLEND, MDenseNet, MMDenseNet and MMDenseLSTM outputs as in [24, 102].

Table 2.3 shows the signal to distortion ratio (SDR) computed using the BSS Eval toolbox [101]. Among the state-of-the-art baselines, BLEND showed the best performance, which was

Table 2.4: Comparison of SDR on MUSDB18 dataset.

Method	#params [$\times 10^6$]	SDR in dB Bass	Drums	Other	Vocals	Acco.
IBM	-	5.30	6.87	6.42	7.50	10.83
BLSTM [24]	30.03	3.99	5.28	4.06	3.43	14.51
MMDenseNet [102]	0.33	**5.19**	6.27	4.64	3.87	15.41
BLEND2	30.36	4.72	6.25	4.75	4.33	16.04
MMDenseLSTM	1.22	**5.19**	**6.62**	**4.93**	**4.94**	**16.40**

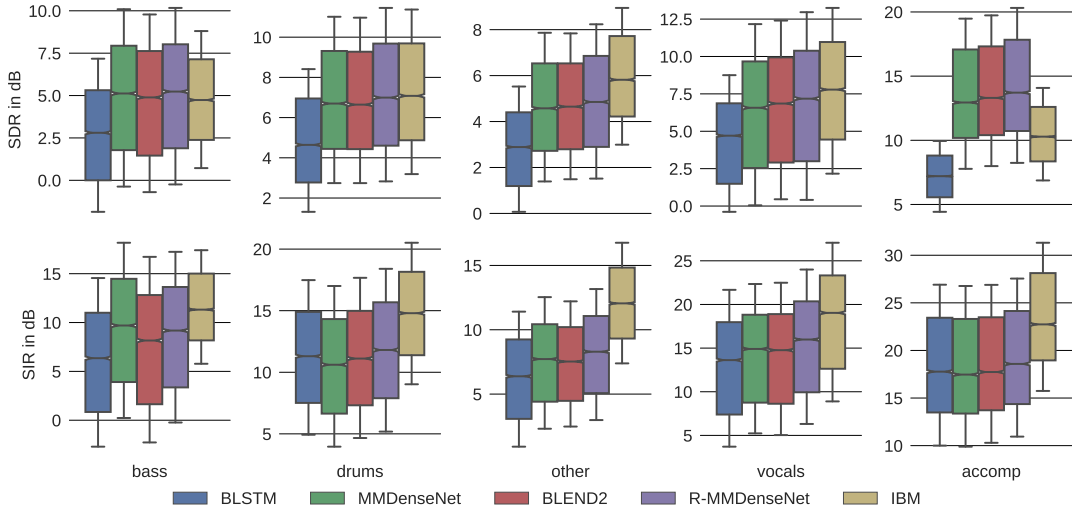

Figure 2.6: SDR comparison. Boxes indicate the 50% percentile and horizontal lines indicate the median.

a fusion of BLSTM and FNN. MDenseNet performed as good as BLSTM, which also utilized MWF. This suggests that the multi-scale architecture successfully learned to utilize long term contexts using the stack of convolution layers instead of the recurrent architecture. This claim will be further investigated in the next subsection. MMDenseNet significantly improved performance and largely outperformed all baselines, showing the effectiveness of the multi-band architecture. MMDenseLSTM architecture further improves the performance for most sources, improves SDRs by an average of 0.2dB compared with MMDenseNet.

To further improve the capability of music source separation and utilize the full modeling capability of MMDenseLSTM, we next trained models with the MUSDB *dev* set and an internal dataset comprising 800 songs resulting in a 14 times larger than the DSD100 *dev* set. The proposed MMDenseLSTM was compared with BLSTM [24], MMDenseNet [102] and a blend of these two systems (BLEND2) as in [24]. All baseline networks were trained with the same training set, namely 900 songs. For a fair comparison with MMDenseNet, we configured it with the same base architecture as in Table 2.1, with an extra layer in the *dense blocks*, corresponding to the *LSTM block* in our proposed method. We also included the IBM as an upper baseline since it uses oracle separation. Table 2.4 shows the result of this experiment. We obtained average improvements of 0.43dB over MMDenseNet and 0.41dB over BLEND2, achieving state-of-the-art results in SiSEC2018 [100]. The proposed method even outperformed the IBM for *accompaniment*. Table 2.4 also shows that MMDenseLSTM can efficiently utilize the sequence modeling capability of LSTMs in conjunction with MMDenseNet, having 24 times fewer parameters than the naive combination of BLSTM and MMDenseNet.

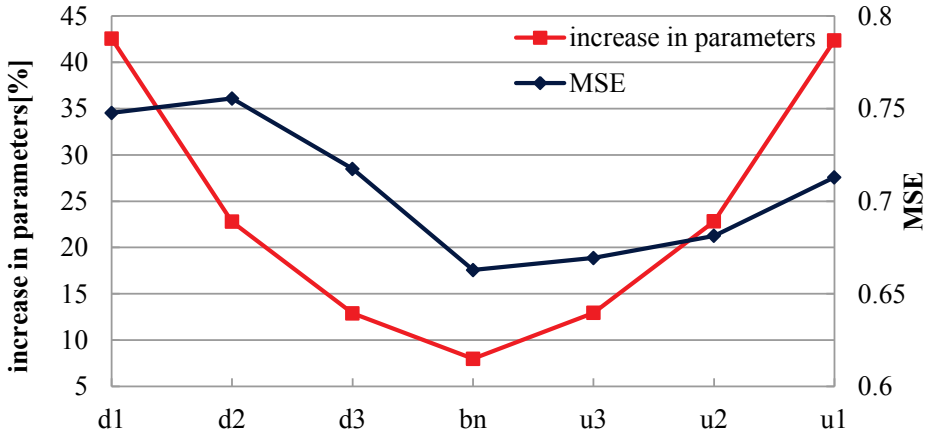

Figure 2.7: Effect of LSTM block at different scales.

Table 2.5: Comparison of MMDenseLSTM configurations.

type	Sa	Sb	P
SDR	**2.83**	2.31	2.47

Architecture validation

In this section we validate the proposed architecture for the singing voice separation task on MUSDB18.

Combination structure The SDR values obtained by the *Sa*-, *Sb*- and *P*- type MMDenseLSTMs are tabulated in Table 2.5. These results validate our claim (Sec. 2.1.3) that the *Sa* configuration performs the best because the LSTM layer can efficiently model the global modulations utilizing the local features extracted by the dense layers at this scale. Henceforth, all experiments use the *Sa* configuration.

LSTM insertion scale The efficiency of inserting the LSTM block at lower scales was validated by comparing seven MMDenseLSTMs with a single 64 unit LSTM layer inserted at different scales in band 1 (all other LSTM layers in Table 2.1 are omitted). Figure 2.7 shows the percentage increase in the number of parameters compared with that of the base architecture and the mean square error (MSE) values for the seven networks. It is evident that inserting LSTM layers at low scales in the up-scaling path gives the best performance.

Contribution of dense and LSTM layers We further compared the $l2$ norms of the feature maps (Fig.2.8) in the LSTM block $d4$ of band 1. It can be seen that the norm of the LSTM feature map is similar to the highest norm among the dense feature maps. Even though some dense feature maps have low norms, we confirmed that they tend to learn sparse local features.

2.2 Phase Reconstruction

Previous research on SS based on DNN mainly focuses on estimating the magnitude spectrum of target sources and typically, phase of the mixture signal is combined with the estimated magnitude spectra in an ad-hoc way. Although recovering target phase is assumed to be important for the improvement of separation quality, it can be difficult to handle the periodic nature of the phase with the regression approach. Unwrapping phase is one way to eliminate the phase discontinuity, however, it increases the range of value along with the times of unwrapping, making it difficult for DNNs to model. To overcome this difficulty, we propose to treat the phase estimation problem as a classification problem by discretizing phase values and assigning class indices to them. All the phase indices are equally treated in the discretized domain and the posterior probabilities for

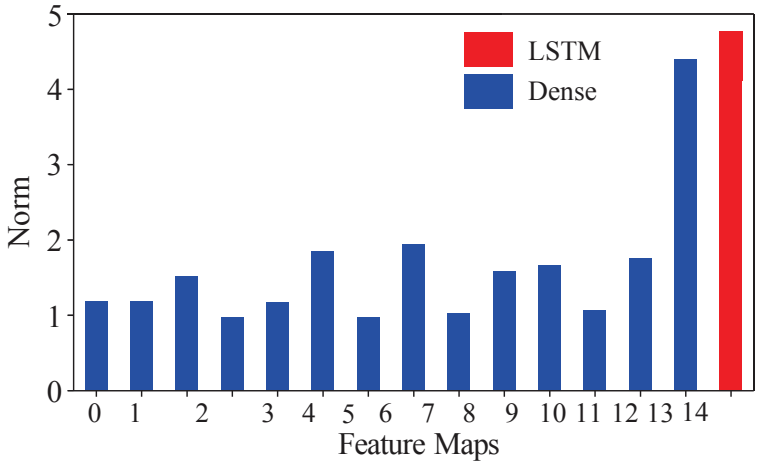

Figure 2.8: Average $l2$ norm of feature maps.

Table 2.6: Effect of Wiener Filtering (WF) on magnitude estimates of DNNs for MSS on DSD100 dataset. Values denote the Mean Squared Error (MSE) with respect to oracle magnitude.

Source	DNN estimate	WF estimate
Vocals	0.444	0.491

each class can be efficiently estimated by DNNs. The phase discretization or quantization has been intensively studied in speech/audio codings [104, 105]. However, to the best of our knowledge, this is the first attempt to apply source separation.

2.2.1 Phase Spectrogram in Source Separation

In audio source separation problems, the input signal is often transformed to the STFT domain to perform separation methods. The mask-based approaches estimate the target mask M and apply it to the input signal $X \in \mathbb{C}$. The target source estimate can be computed by $\hat{S} = M \odot X$, followed by inverse STFT (iSTFT) to obtain the time domain signal $\hat{s}$, where $\odot$ denotes element-wise product. The target mask M can be either estimated by DNNs directly or computed from the maginitude estimates $|\hat{S}|$ in the Wiener filtering way [24]. In the latter case, the mask is denoted as M^{WF} to distinguish it from the former case. In our preliminary experiment, we found that the magnitude of the target source estimated by DNN, $|\hat{S}|$, is more accurate than the magnitude of the filtered input $|M^{\mathrm{WF}}X|$. The Table 2.6 shows the mean square errors, which motivates us further to estimate the phase to improve the estimation of $\hat{s}$. Fig. 2.9 shows the magnitude and phase spectrogram of the clean source and the mixture where the effect of frame shift was corrected based on the sinusoidal model for the phase spectrogram. Unlike the magnitude spectrogram, the phase spectrogram does not show clear structure. This is partly due to the periodic nature of the phase. Even though the phase rotates smoothly around a complex plane, the phase value changes abruptly at the wrapping point (e.g, if the value range is $(-\pi, \pi]$, the wrapping point is π). One way to overcome the phase discontinuity is phase unwrapping. However, it increases the value range along with the times of unwrapping, where the value range at the later frame becomes larger than that at the earlier frame, making it difficult for DNNs to model.

2.2.2 Discrete Phase Modeling

We assume that the periodic nature of the phase is one of the reasons that makes it difficult to apply DNNs for phase estimation. Therefore, we address this problem by casting the phase regression problem to a classification problem. The Fig. 2.10 illustrates the signal flow of the proposed

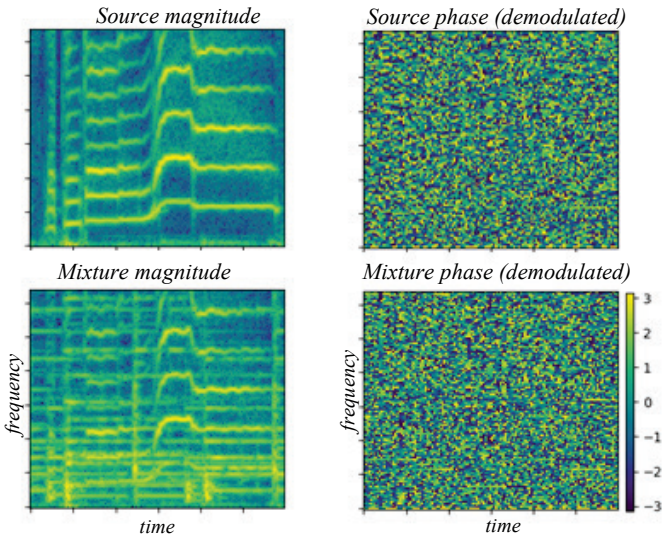

Figure 2.9: Magnitude and phase spectra of clean and mixed signal.

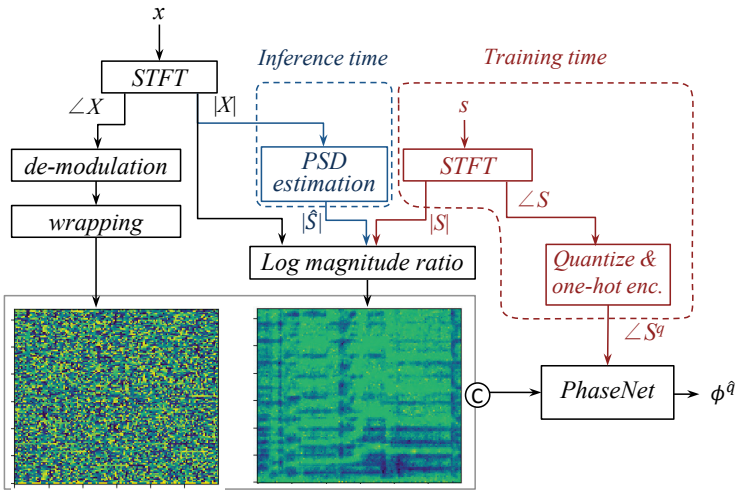

Figure 2.10: Signal flow of PhaseNet in training and inference time.

method. During the training time, the target phase values $\angle S$ are discretized (or quantized) and encoded to one-hot vectors $\angle S^q$, such as $(1, 0, \cdots 0)$ for index 0, so that DNNs can handle the problem as a classification problem. The DNNs are trained to predict the posterior probability of the quantized target phase indices given the mixture phase $\angle X$ through softmax distribution.

According to the sinusoidal model [106], the phase of slowly varying sinusoids can be written as:

$$\phi(f,t) = \phi(f,t-1) + 2\pi h\nu, \tag{2.3}$$

where $\phi(f,t)$, ν and h denote the phase at time frame t, the normalized frequency, and the hop size (in samples), respectively. Equation 2.3 suggests that the phase of the sinusoid varies depending on the TF bins and influenced by the frame shift of the STFT window. To mitigate this modulation effect for DNN phase estimation, we compensate the effect by subtracting $2\pi th\nu$ from each TF bin, which is denoted as de-modulation in Fig. 2.10, and wrap to $(-\pi, \pi]$.

The phase of mixture is dominantly affected by one of the sources if the magnitude of the source is dominant in a TF bin. If the magnitude of a target source is much higher than that of an interference, the mixture phase is most probably close to the target phase. On the other hand, if the target magnitude is at similar level or lower than the interference, the phase could be

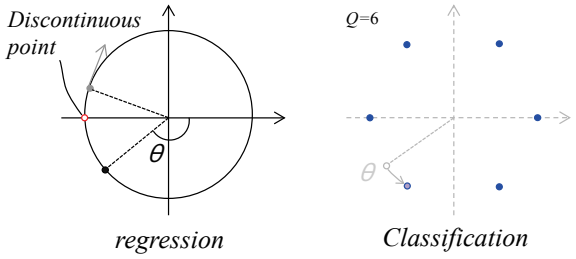

Figure 2.11: Phase representations in regression approach and classification approach. During the training of regression approach, phase value change along with the unit circle. On the other hand, each point is treated as equal in the classification approach.

tweaked by the interference and the phase of these TF bins need to be estimated. To incorporate this characteristic property, we also feed the log magnitude ratio:

$$R = \log\left(\frac{|S|}{|X|}\right) \tag{2.4}$$

to the network by concatenating it along channel dimension. The network is trained to minimize the cross entropy loss L:

$$L(\theta) = -\sum_{i} \angle S_i^q \log P(\phi^q | \angle X_i, R_i, \theta), \tag{2.5}$$

where $\angle S_i^q (q = 1, \cdots Q)$ denotes the index of one-hot encoded quantized phase, $P(\phi^q | \angle X_i, R_i, \theta)$ is the softmax output of DNN for quantized phase ϕ^q given ith sample. The quantization level and the network parameters are denoted as Q and θ respectively. During the inference time, when the magnitude of target source $|S|$ is not available, it is estimated by any method to provide the log magnitude ratio $\hat{R}$ as an input to DNN. We can also use the estimated source magnitude $|\hat{S}|$ for training or fine tuning to improve the phase estimation. The index that has the maximum probability, $\hat{q} = argmax_q P(\phi^q | \angle X_i, \hat{R}_i, \theta)$ is used to transform back to the quantized phase value $\phi^{\hat{q}}$. Hereafter, we call the DNN trained with this approach as PhaseNet. Recent works show that even when the data is implicitly continuous, the discrete softmax distribution works better [107, 108]. Moreover, the recent success of DNN based image classification methods suggest that converting continuous image to discrete class would not be a problem. In the discrete representation, every quantized point is treated equally and there is no explicit assumption on data, e.g., no periodic nature as Fig. 2.11 illustrates. However, the PhaseNet successfully learned a meaningful relationship among phase classes as discussed in Section 2.2.3.

2.2.3 Experiments

Quantization level

To assess the impact of the quantizing phase, we first conducted a subjective test. Speech signals from the Wall Street Journal (WSJ0) corpus were transformed into STFT, the phase was uniformly quantized by a different number of levels and was transformed back to the time domain signal. Ten audio engineers participated in the subjective test. Audio is presented with Sony's headphone 900ST. Six sentences from 3 male and 3 female speakers and 3 quantization levels (4, 8 and 12) per sentence were prepared for the test. The subjective test was conducted in a similar way to the double-blind triple-stimulus with hidden reference format (ITU-R BS.1116), where the reference was the original speech signal and one among A and B was same as the reference, the other being the quantized phase presented in a random order. The subjects were asked to identify which one was the same as the reference signal among A and B. This resulted in 60 evaluations for each quantization level. The Fig. 2.12 summarizes results. In the figure, blue bars indicate the accuracy of finding the reference signal from A and B at quantization levels 4, 8 and 12. The

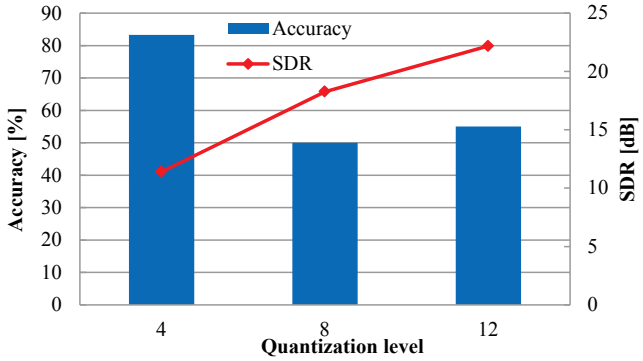

Figure 2.12: Effects of quantization level on perceptual quality and SDR. Blue bars indicate the accuracy of finding the reference signal. Red plot indicates the average SDR values.

Table 2.7: PhaseNet architecture based on MDenseNet.

scale	1	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{8}$	$\frac{1}{4}$	$\frac{1}{2}$	1
l	4	4	5	5	5	5	4	4	4
k	16	18	16	16	16	16	16	18	#Q×#ch

red plot indicates the average SDR values for each corresponding quantization level. As can be observed, the accuracy of finding the correct reference signal is closer to the chance rate (50%) for quantization levels 8 and 12. From this subjective test, we interpreted that at quantization level 8 and above, there is no noticeable difference from the reference signal perceptually.

Single channel speech enhancement (SCSE)

Next, we evaluated the proposed method on the single channel speech enhancement task. The dataset used for training was the speaker independent subset of the WSJ0 corpus. For noise source, the 3rd CHiME challenge (4 types noise) and AE Dataset [91, 92] (41 types noise) were used. AE Dataset was down sampled to 16kHz to match the sampling rate and the original train/test split was used. For the noise data from CHiME, we used session number 040 as a test set. The training data was prepared by randomly mixing sources of varying SNR from -7 to 6 dB. The STFT was performed with a frame size of 1024 samples with 75 % overlap. The PhaseNet architecture was adapted from the MDenseNet architecture. Table2.7 presents the details of the architecture, where l denotes the number of layers and k denotes growth factor of each dense block. The final layer of PhaseNet has $\#Q \times \#ch$ number of feature maps, where $\#Q$ is the number of quantization levels equal to 16 and $\#ch$ is the number of channels in the audio equal to 1. The PhaseNet was trained with Adam optimizer until the loss curve plateaued.

We consider three baselines for comparison, the lower baseline which uses mixture phase, the upper baseline which uses an oracle phase, and phase from a DNN trained with regression approach (DNN-R). The DNN architecture of DNN-R is identical to PhaseNet except the last layer where the softmax output for classification is replaced with a standard convolution output. The input of DNN-R is same as PhaseNet and it is trained to estimate the difference of the target phase and mixture phase ($\angle S - \angle X$) by minimizing the mean square error (MSE).

For reconstructing the time domain target signal, we considered two cases, namely oracle magnitude and noisy magnitude, since the magnitude of the target source is estimated by some method in inference time, and that estimate is usually not perfect. We simulated the noisy magnitude estimate by mixing the noise source in the input with -18 dB attenuation. Table 2.8 compares signal to distortion ratios (SDRs) of estimated target signal are compared with baselines in three SNR scenario, namely -6, -3, 0 dB. The results shows that the proposed method consistently outperform lower baselines and the regression approach. As the SNR becomes low, the input phase is more likely to be dominated by noise. Even in this case, PhaseNet improve the SDR more robustly. It

Table 2.8: Comparison of SDR with different SNRs for speech enhancement.

Magnitude	SNR [dB]	Baselines		Phase model	
		Lower	Upper	DNN-R	PhaseNet
Oracle	-6	7.90	-	8.73	**8.84**
	-3	9.60	-	10.59	**10.75**
	0	11.46	-	12.59	**12.74**
Noisy	-6	5.30	13.64	5.85	**6.16**
	-3	7.39	16.64	8.11	**8.45**
	0	9.54	19.64	10.39	**10.75**

Table 2.9: Comparison of SDRs with different phase on DSD100.

Magnitude	Phase	SDR
Oracle	Mixture	10.58
	CAW[30]	13.81
	DNN-R	12.09
	PhaseNet	**13.83**
Estimates	Oracle	7.04
	Mixture	4.95
	CAW[30]	5.02
	DNN-R	5.42
	PhaseNet	**6.49**

should be noted that even though the PhaseNet was trained only on the clean source magnitude, it significantly improved the performance even when the source magnitude was not perfect.

Music source separation (MSS)

In this section we describe the evaluation of the proposed method on the music source separation task. Specifically, focused on singing voice separation, where the vocals need to be extracted from a mixture of musical sources. For the evaluation we used the Demixing Secrets Database (DSD100), released as part of the SiSEC campaign [99], downsampled to a sampling rate of 22.05kHz. In DSD100, the mixture and its four sources - *bass, drums, vocals, and other*, are available. Thus, our task was to recover the phase of vocals $\angle S$ from the song x. For the MSS task, we used quantization level $\#Q$ as 20. The STFT was performed with frame size of 2048 samples with 75 % overlap. The PhaseNet architecture was the same as that used in the SCSE task up to the final layer, where it was changed based on the $\#Q$ and $\#ch$ values. The network was trained to estimate the quantized phase index $\angle S^q$ with the CE loss with Adam optimizer. The initial learning rate of 0.001, reduced to 0.0001 after training curve saturated. Similar to the SCSE task, to reconstruct the time domain signal, we considered two scenarios, oracle magnitude and estimated magnitude. For a realistic evaluation, we used a MMDenseNet to estimate the magnitude of the target source. In addition to the baselines mentioned in section 2.2.3, we compared PhaseNet with consistent anisotropic Wiener filtering (CAW), which showed superior performance to Wiener filtering, consistent Wiener filtering and anisotropic Wiener filtering [30].

The SDR values on Test set are compared in Table 2.9. From the results, it can be observed that the phase estimated by PhaseNet gives an absolute improvement of about 3.2dB SDR over lower baseline with oracle magnitude and 1.5dB SDR with estimated magnitude. Also worth noting is that PhaseNet performs as well as CAW with oracle magnitude, but more robustly improves performance in the realistic scenario of estimated magnitudes.

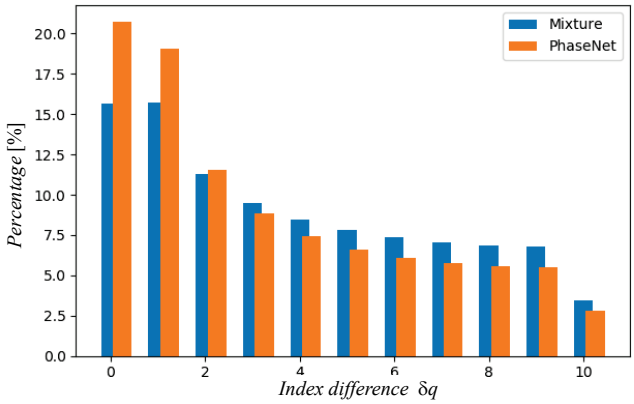

Figure 2.13: Histogram of 'index difference' between the quantized target indices and its estimates.

Estimated phase distribution

As described in Section 2.2.2, since PhaseNet is trained as a classification problem to predict quantized target phase indices, there is no assumption about the data such as periodicity and closeness of discretized points. Therefore, it is worth investigating how PhaseNet outputs are distributed. Fig. 2.13 shows a normalized histogram of the difference of indices δq between the target phase and the phase inferred by PhaseNet in Section 2.2.3. $\delta q = 0$ indicates that the phase is correctly recovered, $\delta q = 1$ indicates that the phase is wrongly estimated to a closest neighboring point, $\delta q = 2$ indicates the estimate is the second neighbor of the target, and so on ($\delta q = 10$ indicates the estimate is the opposite phase in case $\#Q = 20$). For comparison, the histogram of the mixture-source index difference was also presented. The histograms show that PhaseNet shifted the peak of the histogram to $\delta q = 0$ and more rapidly decayed toward the opposite phase, in comparison with the mixture-source index difference. It suggests that the PhaseNet learned a natural posterior distribution that has clear peak at target phase, and was aware of "neighboring points".

2.3 Recursive Speech Separation for Unknown Number of Speakers

In section 2.1 and 2.2, we have discussed separating audio sources depending on type of sound sources. In this section, we are discussing a separation of same type of sources, namely speech separation. Speech communication often occurs in a multi-talker environment. In such a scenario, speech separation is required to selectively process each speaker individually. For example, automatic speech recognition first requires the separation of individual speakers from overlapping speech to successfully transcribe the target speech. Compared with other source separation problems that aim to separate different types of sources such as instrument types in music, speech separation has been considered very challenging for decades since the statistics of sources are similar or the same in the case of speaker independent speech separation problem.

Most recently, a time domain method has surpassed the ideal frequency masks performance under a two-speaker condition[44]. However, most of these methods assume that the number of speakers is known in advance. For example, the deep clustering approach requires information of the number of speakers to cluster embeddings and obtain time-frequency (T-F) masks, although a unified model can be used for 2 and 3 speakers mixture[40]. In actual cases, however, the number of speakers is often unknown or varies, making it difficult to robustly estimate the number of speakers in a mixture. In [42, 46], this problem is partially solved by assuming the maximum number of speakers M in the mixture. The networks are trained to always output M channels regardless of the actual number of speakers N in the input, but when N is smaller than M, $M - N$ channels are enforced to output silent signals. At the test time, the number of speakers is determined by

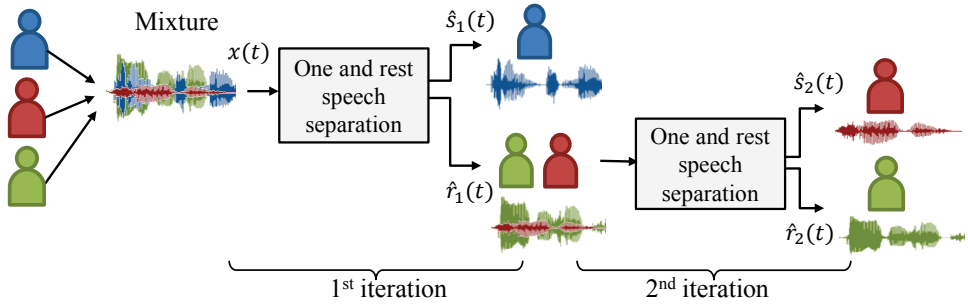

Figure 2.14: Illustration of recursive speech separation when $N = 3$. The speech separation model is trained to separate one speaker from remaining speakers with OR-PIT, and is recursively applied to the second output.

detecting the silent channels. Although the method is shown to work when $M = 3$ [42, 46], it fails when $M < N$.

One way to handle the separation of many speakers is to use visual information to leverage the correlation between speech and mouth movement. In [109], spatio-temporal representations of speakers' faces computed by a neural network trained on the lip reading task are concatenated with an audio signal and a separation network is trained to separate speech sources that correspond to visual information. It is shown to work up to five speakers in [109]. However, such visual information is often not available due to occlusion, frame out or lack of cameras. Thus, speech separation for an unknown number of speakers that operate with audio only is clearly required.

To address this problem, we propose to progressively separate speeches by applying a speech separation network recursively. Instead of separating all speakers in a mixture at once, the proposed model separates only one speaker from a mixture at a time and the residual signal is fed back to the separation model for the recursion to separate the next speaker, as shown in Fig. 2.14. To this end, we propose one-and-rest permutation invariant training (OR-PIT). The proposed method can handle different numbers of speakers using a single model by controlling the number of iterations. Moreover, the proposed method can separate mixture of multi-speakers whose number is larger than any of that seen during the training time. We further propose a method of robustly determining when to stop the iteration for an unknown number of speakers. With the proposed iteration termination criteria, we can more accurately identify the number of speakers than the number of speaker classifier that accept the mixture as the input, and separate speakers of unknown number.

Another advantage of the proposed method is that it tends to separate first a speaker that is easy to separate and sequentially tackle those that are harder to separate. Thus the first separation usually has the highest quality and the quality gradually decreases with increasing number of iterations. This is a preferable property since one can design a system that focuses on separating some of the clearest speakers, that is, a few speakers who are close to the microphone. Recently, similar recursive separation approaches are proposed [110, 111]. However, their works focus on the case where the number of speakers are same or less than training time, and evaluated only up to 2 speaker mixture. Moreover, [111] requires speaker ID during the training time. On the other hand, we show that our approach works even for 4 speaker mixture, which is greater than the number of speakers in the mixtures used for training.

Our contributions are fourfold:

1. We propose a recursive speech separation method for separating a mixture of different numbers of speakers with a single model, even for mixtures which have more number of speakers than the mixtures used for training. To train the recursive separation model, we propose OR-PIT.

2. We further propose a robust and efficient recursion stopping method that enables to operate the recursive speech separation model for an unknown number of speakers.

3. Experimental results showed that our proposed method achieves state-of-the-art results on WSJ0-2mix and WSJ0-3mix datasets using a single model. Moreover, the proposed model

can work surprisingly well for a four-speaker mixture, which was never encountered during the training.

4. We further showed that our proposed approach can more accurately detect the number of speakers in a mixture than the naive approach of directly classifying the number of speakers.

2.3.1 Recursive Speech Separation

Time domain single channel speech separation entails estimation of N speaker sources $s_1(t), s_2(t), ...s_N(t)$ from a mixture signal $x(t)$, where $x(t) = \sum_{i=1}^{N} s_i(t)$. In our work we consider the number of speakers N to be unknown. At the jth recursion step, the recursive speech separator separates one speaker source $\hat{s}^j(t)$ and the mixture of residual speaker sources $\hat{r}^j(t)$ from $\hat{r}^{j-1}(t)$ as

$$\hat{s}^j(t), \hat{r}^j(t) = F(\hat{r}^{j-1}(t)), \tag{2.6}$$

where $F()$ denotes the recursive speech separator modeled as a neural network. We define $\hat{r}^0(t) = x(t)$. The residual signal estimated at each step is input to F recursively to obtain subsequent speaker sources; thus, $\hat{r}^j(t)$ consists of $N - j$ speakers. The procedure is illustrated in Fig. 3.1. The criterion for deciding the number of recursion steps required to estimate all N speaker sources is described in 2.3.3.

2.3.2 One-and-Rest PIT

According to Eq. (2.6) the separation model F is to be trained to separate one speaker at a time and be recursively applicable. However, the choice of one speaker is ambiguous, e.g., there are N valid choices of one target speaker $s_i(t)$ and corresponding residual signal $r_i = \sum_{n \neq i} s_n(t)$. The training with a random or constant choice of the target speaker fails since we do not assume any prior on the order of sources, e.g., we do not assume s_1 to be female and s_2 to be male or so on, and the model becomes confused how to choose the speaker during the test time. To address this problem, we propose novel training method called OR-PIT. Inspired by uPIT [42], OR-PIT computes the error l between the network output and the target for N possible splits of one and rest assignment, $s_i(t), r_i(t)$. The assignment that yields the lowest loss is used for the training objective L to optimize the network,

$$L = \min_i l(\hat{s}(t), s_i(t)) + \frac{1}{N-1} l(\hat{r}(t), \sum_{n \neq i} s_n(t)). \tag{2.7}$$

We omit j for simplicity. The error of the residual signal output channel is divided by the number of speaker sources in the residual mixture to balance it with the error of the single-speaker output channel. For the error, in this work, we use the scale-invariant signal-to-noise ratio (SI-SNR)[1], which has successfully been used in speech separation in the literature [40, 43, 44]. SI-SNR is formulated as:

$$\begin{cases} s_{target} := \frac{\langle \hat{s}, s \rangle s}{\|s\|^2} \\ e_{noise} := \hat{s} - s_{target} \\ l_{\text{SI-SNR}}(\hat{s}, s) := 10 \log_{10} \frac{\|s_{target}\|^2}{\|e_{noise}\|^2} \end{cases} \tag{2.8}$$

where $\hat{s}$ and s are the mean normalized estimates and targets, respectively. The mean normalization of the sources ensures the scale invariance property of the loss function.

In the case when the input to the network is a two speaker mixture, OR-PIT is equivalent to the conventional uPIT [42]. However, when the input is a mixture of more than two speakers, the permutations are computed by taking combinations of one speaker source and the sum of other speaker sources. Thus, the number of permutations in our case is N rather than $N!$, which is the case in uPIT. Another key difference from uPIT is that the sum of rest speaker sources (residual, $\hat{r}_j(t)$) is always trained to be on the second output channel. The purpose of OR-PIT is to ensure

[1] Also denoted as SI-SDR in [41, 112].

Table 2.10: SI-SNR improvement (dB) for 2- and 3-speaker separations before and after fine tuning

Model	WSJ0-2mix	WSJ0-3mix
Before fine tune	15.0	12.2
After fine tune	14.8	12.6

that the best combination of one speaker source and residual speaker sources are separated during the training. This allows the model to be used recursively in the second output channel until the stopping criterion is met.

One notable advantage of the proposed method is that it is not required to predefine the maximum number of speakers and can be applied to an arbitrary number of sources even those never seen during the training. We verify this in Sec. 2.3.4.

2.3.3 Iteration Termination Criteria

As the proposed method recursively separates one speaker from a mixture at a time, we obtain J speaker sources by J recursion steps, where $J \leqq N$. If we wish all speaker sources to be separated, the number of iteration steps should be equal to the number of speakers, i.e., $J = N$. ($J - 1 = N$ is also possible since the residual signal at the $J-1$th recursion step contains a single speaker.) A naive approach is to estimate the number of speakers N using a neural network [113]. We argue that estimating the number of speakers directly from the mixture is relatively difficult and propose to leverage the recursive speech separation model. We propose a simple deep neural network based binary classifier that accepts the residual outputs $\hat{r}^j, (j \geqq 1)$ and predicts whether the signal is speech or not at each recursion step j. If $\hat{r}^j$ is predicted as speech, we proceed to the next recursion step. Otherwise, we stop the recursion and estimate N as j. Note that the energy based approach in [42, 46] cannot be applied to our approach since we use SI-SNR as a training objective and separating a single speaker input does not guarantee to produce a silent signal in one of the outputs, $\hat{s}^j(t), \hat{r}^j(t)$.

2.3.4 Experiments

Network Training

We trained our model for the speech separation task using the Wall Street Journal data set (WSJ0). The model was trained concurrently with 2-speaker and 3-speaker mixture inputs. Following [40], the input mixtures were generated by randomly selecting utterances of different speakers from WSJ0 and mixing them at random SNR between -2.5 dB and 2.5 dB. The mixture was resampled to 8 kHz to reduce computations.

As the network architecture, we adopted TASNet [44], which is a recently proposed time domain speech separation network and produced state-of-the-art results on WSJ-2mix and WSJ-3mix datasets. We used the best performing configuration described in [44]. We replaced the softmax non-linearity used to generate the masks with a ReLU non-linearity in our architecture as we found that it worked better in our recursive model. We chose a time domain approach as the phase reconstruction is shown to be important for source separation to improve the performance [41] and operating directly on time domain signals is possible. However, our proposed method is also applicable to the T-F domain approach.

While training, we forced the first network output channel to always have one speaker and the second channel to collect all the remaining speakers in the mixture input. The model was trained using the OR-PIT with the SI-SNR loss function explained in Sec. 2.3.2. The network was initially trained for 100 epochs with 2- and 3-speaker mixture inputs. The initial learning rate was set to $1e^{-3}$. The Adam optimizer was used with a weight decay of $1e^{-5}$. The input mixtures were 4 seconds long with 50% overlap between two successive frames. In the decoder, the overlapping segments were added together to generate the final reconstructions as in [44].

To further improve the performance of recursion, the model was fine tuned on a 2-speaker mixture obtained from a separation of the first iteration of 3-speaker mixture, instead of clean 2-

Table 2.11: Performance comparison of models trained on WSJ0 data sets. SiSNRi(dB), SDRi(dB) and PESQ of models on WSJ0-2mix, WSJ0-3mix and WSJ0-4mix are compared. ('N/A' - Not applicable, '-' - Data not available)

Method		WSJ0-2mix			WSJ0-3mix			WSJ0-4mix		
		SI-SNRi	SDRi	PESQ	SI-SNRi	SDRi	PESQ	SI-SNRi	SDRi	PESQ
2 speaker model	DPCL++[40]	10.8	-	-	N/A	N/A	N/A	N/A	N/A	N/A
	uPIT-BLSTM-ST[42]	-	10.0	-						
	DANet[45]	10.5	-	2.64						
	ADANet[46]	10.4	10.8	2.82						
	Conv-TasNet-gLN[44]	14.6	**15.0**	**3.25**						
3 speaker model	DPCL++[40]	N/A	N/A	N/A	7.1	-	-	N/A	N/A	N/A
	uPIT-BLSTM-ST[42]				-	7.7	-			
	DANet[45]				8.6	8.9	1.92			
	ADANet[46]				9.1	9.4	2.16			
	Conv-TasNet-gLN[44]				11.6	12.0	2.5			
2&3 speaker model	DPCL++[40]	10.5	-	-	7.1	-	-	N/A	N/A	N/A
	uPIT-BLSTM-ST[42]	-	10.1	-	-	7.8	-			
	ADANet[46]	10.4	-	-	8.5	-	-			
OR-PIT (Proposed)		**14.8**	**15.0**	3.12	**12.6**	**12.9**	**2.60**	**10.2**	**10.6**	**2.26**
Oracle mask (Ideal binary mask)		13.0	13.5	3.33	13.2	13.6	2.91	11.8	12.0	2.42

speaker mixture. The loss was accumulated on both the first iteration (clean 3-speaker separation) and the second iteration (residual 2-speaker separation), and back-propagated. Although the fine tuning slightly decreased the performance of 2-speaker separation, it more significantly improved the performance of the 3-speaker separation. The SI-SNR improvement of our model before and after fine tuning is shown in Table 2.10.

Comparison with other approaches

We compared the proposed method with other state-of-the-art methods [40, 45, 42, 46, 44] on WSJ0-2mix and WSJ0-3mix datasets[114]. The baseline methods are categorized into three groups, namely the *2-speaker model* which is trained for the 2-speaker separation task, the *3-speaker model* which is trained for 3-speaker separation task, and the *2&3-speaker model* which is trained so that it can be applied to both 2- and 3-speaker separation tasks with a unified model. Note that our model is not grouped as *2&3-speaker model* since it can be applied to arbitrary number of speakers even though it was trained on two and three speaker mixture. On the other hand, [45, 46], *2&3-speaker model* is designed to handle an unknown number of speakers which is smaller than the maximum number of speakers it is trained for. To confirm whether our proposed method can handle the number of speakers that was never seen during the training time, we also evaluated the proposed method on a newly created four-speaker mixture. The 4-speaker evaluation set (WSJ0-4mix) was created from the WSJ0-3mix by adding one more speaker source to each of the 3-speaker mixture and mixing them at random SNRs between -3 and 3dB. We also include the ideal binary mask as the baseline.

The SI-SNR improvement (SI-SNRi), signal-to-distortion ratio improvement (SDRi) [101] and perceptual evaluation of speech quality score (PESQ) [115] are shown in Table 2.11. In the cases where the number of speakers in the mixture is different from that of the model target, we marked them as Not Applicable (N/A). We assume an oracle iteration termination for OR-PIT. The termination method is discussed in Sec. 2.3.4. As shown in the table, the proposed method achieved the best results on SI-SNRi and SDRi on both WSJ0-2mix and 3mix datasets. Even when compared with the models specifically trained for 2- or 3-speaker separation, the proposed method outperforms most baselines with a single model. It is worth noting that our model uses the same network architecture as Conv-TasNet-gLN [44] except the nonlinearlity of the last layer. Comparison with these models suggests the effectiveness of the proposed recursive separation method and generalization capability to the number of input speakers. This effectiveness is further supported by the evaluation on four-speaker mixture. Even though the proposed method never encountered the four-speaker mixture during the training, it separated four speaker surprisingly well with three recursions.

Table 2.12: Test accuracy of speech or noise binary classifier and multi-class count speakers classifier

Model	Accuracy
Binary classifier	**95.7%**
Multi-class classifier	77.9 %

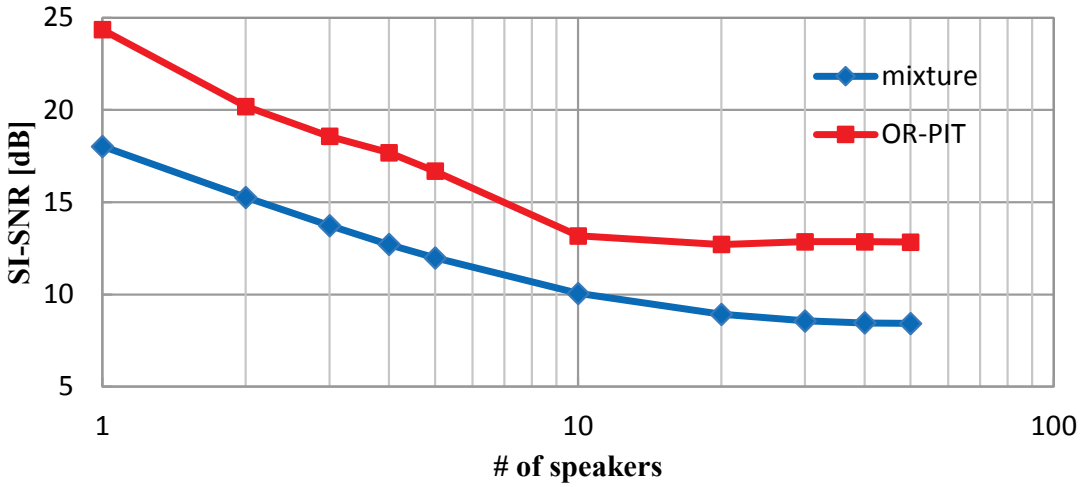

Figure 2.15: SI-SNR of dominant-speaker separation on various numbers of interference speakers.

Identification of number of speakers

Since our model can perform speech separation of an unknown number of speakers in input by recursion, it is very important to know when to terminate the recursion. We evaluated the proposed iteration termination criteria described in Sec.2.3.3. We trained the Alexnet model [116] for the task of binary classification of speech or noise on the residual inputs coming out from the second channel. As a baseline, we also trained the Alexnet model for a multiclass classification task of counting the number of speakers in the input mixture. The baseline model can be used with, for example, DPCL[40] to decide the number of clusters. Please note that the capability of the multiclass classifier is limited by the maximum number of speakers in a mixture input in the training set. On the other hand, the binary classifier is independent of the number of speakers in the mixture. For both models, input segments of 10 seconds long mel spectrogram with window length 1024, 50% overlap and downsized to 128 mel bands were used as input features. The test set consisted of 3000 samples each of clean 1, 2 and 3 speakers from the WSJ0 evaluation sets, WSJ0-2mix and WSJ0-3mix, respectively. For the binary classifier, when the network predicts all the iterations except the last as speech, it is considered as a successful classification and vice versa. As shown in Table 2.12, the binary classifier more accurately detects the number of speakers than the multiclass classifier. This clearly indicates the effectiveness of leveraging the recursive speech separation model for the detection of the number of speakers.

Dominant speech separation

A notable property of the proposed method is that it tends to separate the most dominant (easiest) speaker first and successively tackles the separation of less dominant (harder) speakers. This is useful when we consider extracting a few speakers close to the microphone in a crowd since we can often assume that the conversation takes place within a small area and one of speakers can hold or attach a microphone. As a special case, we consider extracting the most dominant speaker from a mixture of a large number of speakers. To simulate a speech in a crowd recorded by a microphone attached to the target speaker, we created an evaluation dataset by adding N interference speakers to the target speaker. We vary N from 1 to 50 and created an evaluation set of 500 samples for

each of the cases. The first interference speaker level is scaled to be 18dB less than target speaker and every Nth speaker was scaled to have 0.5 dB lower than the $N-1$th interference speaker. The same model in Sec.2.3.4 was used and the model was never trained or fine-tuned for this task. , i.e., the model never saw a mixture of more than three speakers during the training. Fig. 2.15 shows the SI-SNR of the proposed method and the mixture as a baseline. It is shown that the proposed method consistently improved SI-SNR and achieved high SI-SNR even for a mixture of more than 10 speakers. It indicates that the proposed method can be robustly applied for a dominant-speaker separation from a mixture of a large number of speakers.

2.4 Conclusion of this Chapter

We first investigated the deep neural network architecture that efficiently model the magnitude spectrogram. To this end, proposed MMDenseLSTM that model the spectrogram in multi-resolution and multi-band with combination of DenseNet and LSTM. The proposed architecture achieves state-of-the-art results on DSD100 and MUSDB18 datasets, showing the effectiveness of multi-band multi-scale structure with the dense connection and combination of convolutional and recurrent layers. To reconstruct the phase, we proposed to treat the phase estimation problem as a classification problem, and proposed PhaseNet that predict the quantized phase index of target phase. For speech separation, we proposed a novel recursive speech separation approach that deal with different numbers of speakers cases using a single model. Experimental results show that our proposed method achieves state-of-the-art results on two and three speaker mixture with the same model and even worked on a four-speaker mixture even though the model has never seen the four-speakers mixture during the training.

Chapter 3

Automatic Speech Recognition

The three principal resources typically required for developing a phoneme based automatic speech recognizer (ASR) are: transcribed acoustic data for acoustic model estimation, text data for language model estimation, and a pronunciation dictionary to map words to sequences of sub-word units. Manual preparation of such resources requires significant investment and expertise. Therefore, an automatic generation of pronunciation dictionary from the data is clearly required for many dialects and languages.

The main focus of this thesis is to design an ASR based on an automatically generated dictionary that outperforms commonly used phoneme based ASRs. While most of the solutions proposed to find a pronunciation based on multiple utterances of a word are n-best type heuristics [54, 117, 118], in this thesis, we employ an approximation of the K-dimensional Viterbi algorithm proposed in our previous works [119, 56]. This approach gives us the maximum-likelihood estimates of the pronunciations. These high-quality pronunciations are one of the key factors to outperform phoneme based ASRs. Moreover, to learn proper sub-word units, we combine the strength of Gaussian mixture models (GMM) and deep neural network (DNN) based acoustic modeling. We formulate this problem as an instance of a semi-supervised self-learning process. By taking advantage of the robustness of hidden Markov models (HMM) with GMM based observation probability distribution against labeling errors, we train the first set of sub-word units and output the first set of pronunciations. We then use this dictionary to re-label the data and employ the higher expressiveness of DNNs to improve the modeling of sub-word units and the dictionary in an iterative process. In each iteration round, a new dictionary is generated and by means of this new dictionary the data is re-labeled. This data is again used to train the DNN. As shown in the experiments, the proposed results achieves more than 10% absolute improvement over the phoneme based approach on TIMIT data in a continuous speech recognition task.

The reminder of this chapter is organized as follows. The proposed framework and its components for joint sub-word units and dictionary learning are introduced in Section 3.1. In Section 3.2 the experimental results are demonstrated and finally, conclusions are summarized in Section 3.3.

3.1 Semi-supervised joint Dictionary and Acoustic Model Learning

3.1.1 Framework

In the rest of this chapter, we refer to data-driven sub-word units as abstract acoustic elements (AAEs) in contrast to phones. Our goal is to jointly learn the pronunciation dictionary $d^* = \{\omega_1, \cdots, \omega_L\}$ of L pronunciations ω_i and N AAE models $\lambda^* = \{A_1, \cdots, A_N\}$ that maximize the joint likelihood:

$$\lambda^*, d^* = \arg\max_{\Lambda, D} P(\mathbf{X}|\mathbf{T}, \Lambda, D) \tag{3.1}$$

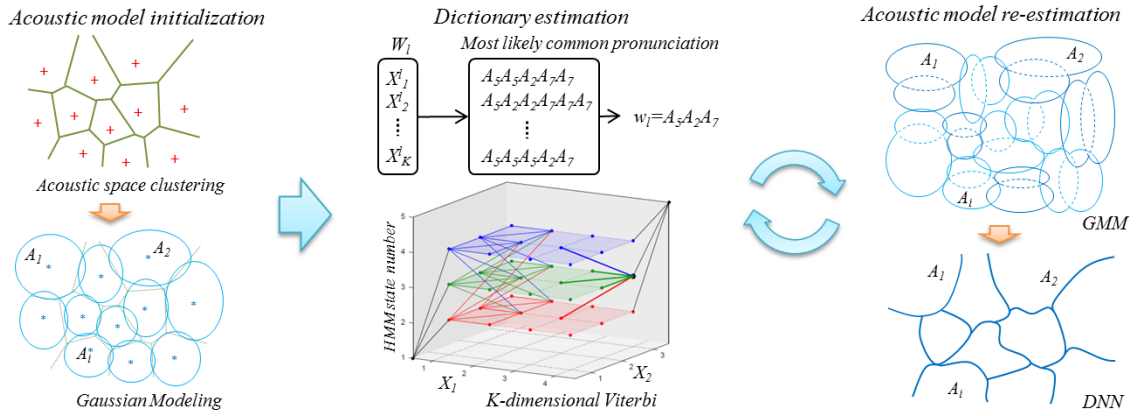

Figure 3.1: Framework of joint sub-word and dictionary learning. K-dimensional Viterbi illustrated in case of $K = 2$.

where $\mathbf{X} = (X_1, \cdots, X_M)$ is the set of training utterances, $\mathbf{T} = (T_1, \cdots, T_M)$ is the set of corresponding orthographic transcriptions, M is the number of utterances, Λ is the universe of all possible sets of N AAEs and D is the universe of all the dictionaries which map words to AAEs sequences. It is hard to find the optimal solution for the optimization problem in (3.1) due to its complex non-linear nature. It is thus decomposed into two simpler optimization problems which can be solved iteratively.

$$d^i = \arg\max_D P(X|T, \lambda^i, D) \tag{3.2}$$

$$\lambda^{i+1} = \arg\max_\Lambda P(X|T, \Lambda, d^i) \tag{3.3}$$

Since the pronunciation of each word can be estimated independently from other words, the dictionary estimation in (3.2) can be decomposed into L maximum likelihood estimations as follows:

$$\begin{aligned} \omega_l = \arg\max_\omega \prod_{j \in \Omega_l} \max_{\mathbf{S}_j} P(X_j, \mathbf{S}_j|\lambda) \\ \text{subject to: } \mathbf{S}_j \in \mathbb{S}_\omega \end{aligned} \tag{3.4}$$

where Ω_l is the set of indices of utterances of word W_l, $\mathbf{S}_j$ is a sequence of AAEs and $\mathbb{S}_\omega$ denotes a set of all possible AAE sequences of the pronunciation ω. For instance in $\mathbb{S}_\omega$, if the pronunciation is $\omega = A_1A_2A_3$, some samples in $\mathbb{S}_\omega$ may be $A_1A_1A_1A_2A_3$, $A_1A_2A_2A_3A_3$ and $A_1A_1A_2A_3A_3A_3$. The constraint in (3.4) implies that all AAE sequences should be samples of the same pronunciation. For the case where λ is modeled by a left-to-right HMM without skips, which is the most common topology in HMM based ASRs, a solution of (3.4) has been proposed in [119] (Details are in Section 3.1.3.). In (3.3), since the dictionary is fixed, the problem results in a common acoustic model estimation given the dictionary. However, the labels re-assigned by using the estimated dictionary are very noisy since the dictionary is automatically estimated from data without any expert supervision. Therefore, a robust model is required at early stage of the training iteration while a more expressive and powerful model such as a DNN [120, 121] can be used after the reliable dictionary is obtained.

The joint dictionary and AAE learning framework is illustrated in Figure 3.1 and summarized as follows:

3.1.2 Acoustic Model Initialization

Initial AAE models can simply be obtained by clustering the acoustic space. The acoustic space can be described by any feature as long as it is informative enough to discriminate between different

Algorithm 1 Semi-supervised joint AAEs and dictionary learning

$i = 0$
// Initialize AAE models λ^0 (Section 3.1.2)
Clustering the acoustic space.
Model each cluster by GMM and set as λ^0.
// Start joint AAEs and dictionary learning
while (Performance is improved) **do**
    Given AAE models λ^i, update dictionary d^i by maximizing joint likelihood multiple utterances (Section 3.1.3).
    Given dictionary d^i, double the number of mixtures and update AAE models λ^{i+1} (Section 3.1.4).
    $i \leftarrow i + 1$
end while
Replace GMM by DNN and train AAE model using labels obtained by HMM-GMM (Section 3.1.4).
while (Performance is improved) **do**
    Given AAE models λ^i, update dictionary d^i by maximizing joint likelihood multiple utterances.
    Given dictionary d^i, re-train DNN based AAE models λ^{i+1} (Section 3.1.4).
    $i \leftarrow i + 1$
end while

words. We employed the Linde-Buzo-Gray (LBG) algorithm [122] with a squared-error distortion measure to cluster the acoustic feature vectors. The LBG clustering algorithm tends to assign more codebook vectors to high-density areas which is a useful property in order to obtain discriminative initial AAEs. Each cluster is then modeled by a GMM with a single Gaussian component. These models are used as the initial models for AAEs.

3.1.3 Dictionary Generation

The solution of (3.4) proposed in [119] is an extension of the standard one-dimensional Viterbi algorithm to K dimensions. The K-dimensional Viterbi algorithm calculates the most probable HMM state sequence which is common to K given utterances. While this algorithm is rigorous, its complexity grows exponentially with the number of utterances, which consequently makes it infeasible to apply it to more than a few utterances. An efficient approximation of the K-dimensional Viterbi algorithm has been proposed in [56] where the problem to find the joint alignment and the optimal common sequence for K utterances is decomposed into $K-1$ applications of two-dimensional Viterbi algorithm. This approximation starts with finding the best alignment between two utterances. Then, while keeping the alignment between the already processed utterances fixed, the next utterance is aligned with this master utterance. The AAE sequence of the final master utterance is the approximation of the K-dimensional Viterbi pronunciation.

3.1.4 Acoustic Modeling

Once the dictionary is updated, all utterances are decoded based on the new pronunciation of the words in the dictionary and the AAEs are re-estimated according to the new labels. The AAEs can be modeled by commonly used models such as HMM/GMM or HMM/DNN. However, at the beginning of the training iteration, the model and dictionary are not accurate enough and more probable to get stuck in a bad local optimum if the model's degree of freedom is too high. In order to avoid this situation, we start the training with a simple model, namely one Gaussian component for each AAE with a diagonal covariance matrix. In each iteration, the dictionary gets more accurate. Thus, the number of mixture components are doubled in order to increase the modeling power. Once the performance is saturated the GMM is replaced with the DNN in order to utilize more expressive modeling capability. This process makes the semi-supervised DNN training feasible and

prevents it to get stuck in a bad local optimum. The HMM state-level transcription is obtained by force-aligned decoding with optimised HMM-GMM and dictionary. This transcription provides labels for DNN training. The DNN is trained to estimate HMM posterior states by minimizing the cross entropy loss L with l_1 regularization using back propagation:

$$\arg\min_{W} \sum_{i,j} L(\mathbf{x}_j^i, y_j^i, W) + \rho\|W\|_1 \tag{3.5}$$

where $\mathbf{x}_j^i \in X_i$ is the jth feature vector of the ith utterance, y_j^i is the corresponding label and W is the set of network parameters, respectively. ρ is a constant parameter which is set to 10^{-6} in this work.

3.2 Experiments

We conducted several sets of experiments on the TIMIT corpus [123]. The TIMIT corpus provides a manually prepared dictionary and phone-level transcriptions with 61 phones. As a baseline, 61 phone models were trained using the TIMIT dictionary and the provided transcriptions. We used 12 mel frequency cepstral coefficients (MFCCs) and energy with their deltas and delta-deltas as descriptors of the acoustic space. The speech data was analyzed using a 25 ms Hamming window with a 10 ms frame shift. We evaluated phone based DNN-HMM, GMM-HMM and AAE based GMM-HMM model as baselines. The DNN architecture was comprised of 7 hidden layers. The first hidden layer had 2048 nodes, next 5 layers had 1024 nodes and the number of nodes at the last layer was equal to the number of HMM states to be predicted. All hidden layers were equipped with the Rectified Linear Unit (ReLU) non-linearity [124]. The input to the network was 11 contiguous frames of MFCCs. The networks were trained using mini-batch gradient descent based on back propagation with momentum. We applied dropout [120] to all hidden layers with dropout probability 0.5. The batch size was set to 128. HMMs had left-to-right, no-skipping topology with three states for each phoneme as opposed to one state for each AAE. HMMs were trained using a modified version of HTK [125] and DNNs were implemented using Lasagne [126].

3.2.1 Isolated Word Recognition

The first set of experiments were on the isolated word recognition to test the performance of the proposed methods and investigate the effects of hyper parameters such as the number of mixture components and the number of AAEs. For joint pronunciation estimation and acoustic models training, we collected a pronunciation training set comprising of words with more than 10 utterances from the TIMIT training set. The total number of utterances in the pronunciation training set was 12800. After excluding words with less than 4 characters (e.g., a and the), 339 distinct words were collected from the TIMIT test set for the isolated word speech recognition task, resulting in 3900 utterances in total. The baseline GMM based phone models were trained with 32 mixture components. During the GMM based AAE model training the number of mixtures was doubled for each iteration until it reached 128 mixtures as described in Section 3.1.4.

Comparison with phonetic approach

The word error rates (WER) of each method are shown in Table 3.1. The results show that the proposed data-driven method clearly outperforms the baseline methods. The proposed AAE-DNN method achieved 10.3% and 2.4% improvement over GMM and DNN based phonetic acoustic models, respectively. This suggests that a more accurate dictionary and better acoustic models can be obtained directly from training data without any human expertise. Moreover, AAE-DNN method improves the performance by 3.2% over the AAE-GMM method. This indicates that the DNN was successfully trained in the semi-supervised manner and the final model could effectively use the its expressive modeling power.

Table 3.1: Comparison of word error rates of each method on 339 words isolated word recognition (%). Baseline phone models are trained by using the TIMIT dictionary.

Method	WER
Phone GMM	18.18
Phone DNN	10.31
AAE GMM	11.15
AAE DNN	**7.93**

Table 3.2: Word error rates in % of AAE based recognizers with different number of AAEs and GMM mixture. The best performance for each number of AAE is plotted in Figure 3.2.

# of AAE	# of mixture			
	16	32	64	128
64	19.48	18.33	17.52	**16.93**
128	14.33	13.87	**13.09**	13.70
192	13.39	13.31	**12.68**	13.98
256	11.97	**11.56**	12.65	14.33
320	11.46	**11.15**	11.69	14.10
384	11.63	**11.56**	12.14	13.75
448	**11.20**	11.33	12.45	-

Number of AAEs

Our second experiment focused on the effects of the number of AAEs, i.e. N. We trained the dictionary and AAE models with $N = 64, 128, 192, 256, 320, 384, 448$. The word error rates of DNN and GMM based AAE models are illustrated in Figure 3.2. The number of mixtures of the GMMs were determined experimentally as shown in Table 3.2. For DNN based AAE models, the best result are obtained with 384 AAEs in contrast to with 320 AAEs for the GMM based models. Interestingly, the optimal number of AAE states is far higher than the number of states of the phone models (61 phonemes $\times$ 3 states = 183 states). This is an indication that the proposed data-driven approach to jointly generate the sub-word units and dictionary models the acoustic space more precisely than the linguistically motivated phonetic units and the manually designed dictionary. It is also worthwhile to mention that the optimal number of DNN based AAE models was higher than that of GMM based models. This is perhaps due to the fact that the DNN was trained discriminatively, allowing to efficiently model the interaction between higher number of AAEs.

3.2.2 Continuous Speech Recognition

Unlike phoneme based ASRs, the proposed AAE based approach does not depend on linguistic knowledge. It is therefore interesting to compare these approaches on a real-world continues speech recognition (CSR) task. For this purpose, we used the SX records of the TIMIT corpus which contains 450 sentences spoken by 7 speakers, i.e. 3150 utterances in total. We prepared the test set by randomly selecting and putting aside one speaker for each sentence from the SX recordings and used the remaining samples as the training set (450 sentences $\times$ 6 speaker = 2700 utterances). We also included the SA and SI recordings of the TIMIT corpus in the training set. The number of AAEs was 384. The number of mixture components in the GMM based phone models was 64. The performance was evaluated in two scenarios: with and without language model. The language model employed in the baseline and the proposed methods is a simple bigram model.

Table 3.3 shows that the proposed AAE-DNN based approach significantly outperforms baseline methods in both scenarios. The performance improvements over the phone based HMM-DNN method in with and without the language model scenarios were 10.68% and 5.11%, respectively. The results suggest that the proposed data-driven dictionary and the AAE models are also useful

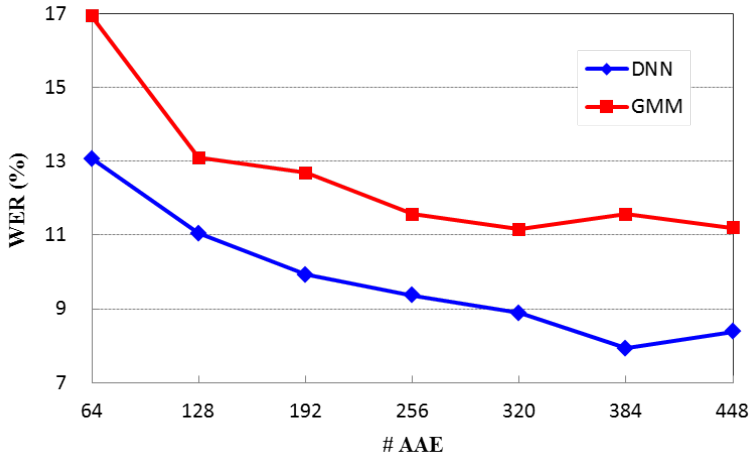

Figure 3.2: Performance of AAE based recognizers with different number of AAEs on test set with 339 words.

Table 3.3: Comparison of word error rate of each method on continuous speech recognition. In column "No LM", no language model was used.

Method	No LM	Bigram
Phone GMM	71.11	43.54
Phone DNN	50.18	20.89
AAE GMM	59.52	32.36
AAE DNN	**39.05**	**15.78**

for CSR and a more accurate representation of speech signals can be learned automatically. We observed that all 384 AAEs were actually used in the trained dictionary, and the dictionary tend to assign 39% more HMM states on average to each word as compare with the TIMIT phonetic dictionary. This means that in AAEs, the stay-in-state probability is smaller resulting in more frequent state transitions. This suggests that by using AAEs, the acoustic space was modeled at a higher resolution. This consequently increased the precision of the word pronunciations.

3.3 Conclusion of this Chapter

In this Chapter we proposed a novel joint dictionary and sub-word unit learning framework for ASRs. The proposed method does not require linguistic expertise, and can automatically create the set of sub-word units and the corresponding pronunciation dictionary. In our method, reliable pronunciations are estimated from multiple utterances by an efficient approximation of K-dimensional Viterbi algorithm which estimates the most probable HMM state sequence common to multiple utterances of a word. Experimental results show that the proposed method significantly outperforms the phone based methods which even get manually prepared dictionary and hand crafted transcriptions as inputs. We further investigated the effects of the number of data-driven sub-word units and showed that the optimal number of sub-word units is much higher than the total number of HMM states of the 61 phones. The future works will be directed towards applying the proposed method to speech recognition for under-resourced languages and large vocabulary continuous speech recognition tasks.

Chapter 4

Audio Event Recognition

Next, we discuss the recognition of general audio event. As we discussed in the introduction 1, we are aiming at creating a generic way to represent audio signals that can be used for solving various audio analysis tasks in a unitary way. Existing audio event recognition such as [127, 128] consist of complex events with multiple sound sources in a class (e.g. the "birthday party" class may contain sounds of voices, hand claps, music and crackers). Classifiers learnt from these datasets are task specific and not transferred well for other tasks such as generic video analysis since other classes such as "Wedding ceremony" also would contain the sounds of voices, hand claps or music. To this end, we introduce a novel dataset in which the audio event classes are more factorized and generic so that the hidden representation of DNN classifier learnt on the dataset can be useful for other tasks. We discuss the learnt feature transferability in Chapter 5.

In order to design the classifier, we introduce novel deep convolutional neural network (CNN) architectures with up to 9 layers and a large input field. The large input field allows the networks to directly model several seconds long audio events with time information and be trained end-to-end. The large input field, capturing audio features for video segments, is suitable for video analysis since this is typically conducted on segments several seconds long. Our feature descriptions keep information on the temporal order, something which is lost in most previous approaches [63, 4, 59]. In order to train our networks, we further propose a novel data augmentation method, which helps with generalization and boosts the performance significantly. The proposed network architectures show superior performance on AER over BoAW and conventional CNN architectures which typically have up to 3 layers.

In the following sections, we introduce a novel CNN based AER including a new dataset, novel network architectures and a data augmentation strategy. Then the proposed methods are validated in Section 4.2.

4.1 Architectural and Training Novelties

4.1.1 Large Input Field

In ASR, few-frame descriptors are typically concatenated and modeled by a GMM or DNN [120, 121]. This is reasonable since they aim to model sub-word units like phonemes which typically last less than a few hundred ms. The sequence of sub-word units is typically modeled by a HMM. Most works in AER follow similar strategies, where signals lasting from tens to hundreds of ms are modeled first. These small input field representations are then aggregated to model longer signals by HMM, GMM [129, 59, 71, 130, 131] or a combination of BoAW and SVM [67, 68, 69]. Yet, unlike speech signals, non-speech signals are much more diverse, even within a category, and it is doubtful whether a sub-word approach is suitable for AER. Hence, we decided to design a network architecture that directly models the entire audio event, with signals lasting multiple seconds handled as a single input. This also enables the networks to optimize its parameters end-to-end.

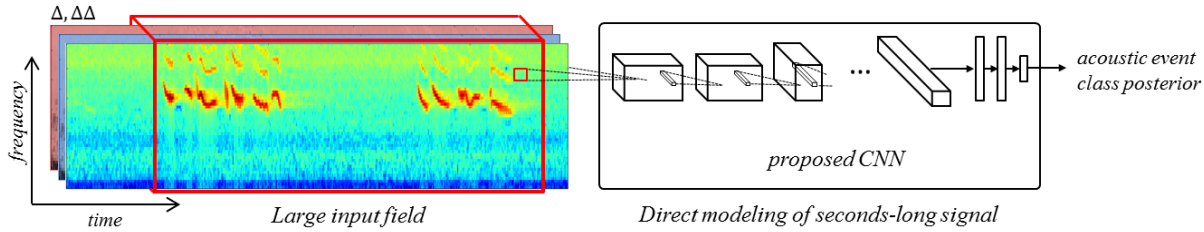

Figure 4.1: Our deeper CNN models several seconds of audio directly and outputs the posterior probability of classes.

4.1.2 Deep Convolutional Network Architecture

Since we use large inputs, the audio event can occur at any time and last only for a part, as depicted in Table 4.1. There the audio event occurs only at the beginning and the end of the input. Therefore, it is not a good idea to model the input with a fully connected DNN since this would induce a very large number of parameters that we could not learn properly. In order to model the large inputs efficiently, we used a CNN [72] to leverage its translation invariant nature, suitable to model such larger inputs. CNNs have been successfully applied to the audio domain, including AER [91, 71]. The convolution layer has kernels with a small receptive field which are shared across different positions in the input and extract local features. As stacking convolution layers, the receptive field of deeper layer covers larger area of input field. We also apply convolution to the frequency axis to deal with pitch shifts, which are shown to be effective for speech signals [97]. Our network architecture is inspired by "VGG Net" [74], which obtained the second place in the ImageNet 2014 competition and was successfully applied for ASR [75]. The main idea of VGG Net is to replace large (typically 9×9) convolutional kernels by a stack of 3×3 kernels without pooling between these layers. Advantages of this architecture are (1) additional non-linearities, hence more expressive power, and (2) a reduced number of parameters (i.e. one 9×9 convolution layer with C maps has $9^2C^2 = 81C^2$ weights while a three-layer 3×3 convolution stack has $3(3^2C^2) = 27C^2$ weights). We have investigated many types of architectures, including the number of layers, pooling sizes, and the number of units in fully connected layers, to adapt the VGG Net of the image domain to AER. As a result, we propose two architectures as outlined in *Table 4.1.* Architecture A has 4 convolutional and 3 fully connected layers, while Architecture B has 9 weight layers: 6 convolutional and 3 fully connected. In this table, the convolutional layers are described as conv(input feature maps, output feature maps). All convolutional layers have 3×3 kernels, thus henceforth kernel size is omitted. The convolution stride is fixed to 1. The max-pooling layers are indicated as $time \times frequency$ in Table 4.1. They have a stride equal to the pool size. Note that since the fully connected layers are placed on top of the convolutional and pooling layers, the input size to the fully connected layer is much smaller than that of the input to the CNN, hence it is much easier to train these fully connected layers. All hidden layers except the last fully-connected layer are equipped with the Rectified Linear Unit (ReLU) non-linearity. In contrast to [74], we do not apply zero padding before convolution, since the output size of the last pooling layer is still large enough in our case. The networks were trained by minimizing the cross entropy loss L with l_1 regularization using back-propagation:

$$\arg\min_{W} \sum_{i,j} L(\mathbf{x}_j^i, y_j^i, W) + \rho\|W\|_1 \tag{4.1}$$

where $\mathbf{x}_j$ is the jth input vector, y_j is the corresponding class label and W is the set of network parameters, respectively. ρ is a constant parameter which is set to 10^{-6} in this work.

4.1.3 Data Augmentation

Since the proposed CNN architectures have many hidden layers and a large input, the number of parameters is high, as shown in the last row of *Table 4.1.* A large number of training data is vital to train such networks. Jaitly *et al.* [132] showed that data augmentation based on Vocal Tract

Table 4.1: The architecture of our deeper CNNs. Unless mentioned explicitly convolution layers have 3×3 kernels. The input size of each model is discussed in Sec 4.2.3.

	Baseline		Proposed CNN	
#Fmap	DNN	Classic CNN	A	B
64		conv5×5 (3,64) pool 1×3 conv5×5(64,64)	conv(3,64) conv(64,64) pool 1×2	conv(3,64) conv(64,64) pool 1×2
128			conv(64,128) conv(128,128) pool 2×2	conv(64,128) conv(128,128) pool 2×2
256				conv(128,256) conv(128,256) pool 2×1
FC	FC4096 FC2048 FC2048 FC28	 FC1024 FC1024 FC28	 FC1024 FC1024 FC28	 FC2048 FC2048 FC28
	softmax			
#param	258×10^6	284×10^6	233×10^6	257×10^6

Length Perturbation (VTLP) is effective to improve ASR performance. VTLP attempts to alter the vocal tract length during the extraction of descriptors, such as a log filter bank, and perturbs the data in a certain non-linear way.

In order to introduce more data variation, we propose a different augmentation technique. For most sounds coming with an event, mixed sounds from the same class also belong to that class, except when the class is differentiated by the number of sound sources. For example, when mixing two different ocean surf sounds, or of breaking glass, or of birds tweeting, the result still belongs to the same class. Given this property we produce augmented sounds by randomly mixing two sounds of a class, with randomly selected timings. In addition to mixing sounds, we further perturb the sound by moderately modifying frequency characteristics of each source sound by boosting/attenuating a particular frequency band to introduce further varieties while keeping the sound recognizable. An augmented data sample s_{aug} is generated from source signals for the same class as the one both s_1 and s_2 belong to, as follows:

$$s_{aug} = \alpha\Phi(s_1(t), \psi_1) + (1-\alpha)\Phi(s_2(t-\beta T), \psi_2) \tag{4.2}$$

where $\alpha, \beta \in [0, 1)$ are uniformly distributed random values, T is the maximum delay and $\Phi(\cdot, \psi)$ is an equalizing function parametrized by ψ. In this work, we used a second order parametric equalizer parametrized by $\psi = (f_0, g, Q)$ where $f_0 \in [100, 6000]$ is the center frequency, $g \in [-8, 8]$ is a gain and $Q \in [1, 9]$ is a Q-factor which adjusts the bandwidth of a parametric equalizer. An arbitrary number of such synthetic samples can be obtained by randomly selecting the parameters α, β, ψ for each data augmentation. This data augmentation helps networks to be more robust and thus generalize to moderate fluctuation of the number of sources, loudness, and sound color. We refer to this approach as Equalized Mixture Data Augmentation (EMDA).

4.1.4 Dataset

In order to learn a discriminative and universal set of audio features, a dataset on which the feature extraction network is trained needs to be carefully designed. If the dataset contains only a small number of audio event classes, the learned features could not be discriminative. Another concern is that the learned features would be too task specific if the target classes are defined at too high a semantic level (e.g. Birthday Party or Repairing an Appliance), as such events would present the system with rather typical mixtures of very different sounds. Therefore, we design the target classes according to the following criteria: 1) The target classes cover as many audio events which may happen in consumer videos as possible, 2) The sound events should be atomic (no composed events) and non-overlapping. As a counterexample, "Birthday Party" may consist of Speech, Cracker Explosions and Applause, so it is not suitable. 3) Then again, the subdivision

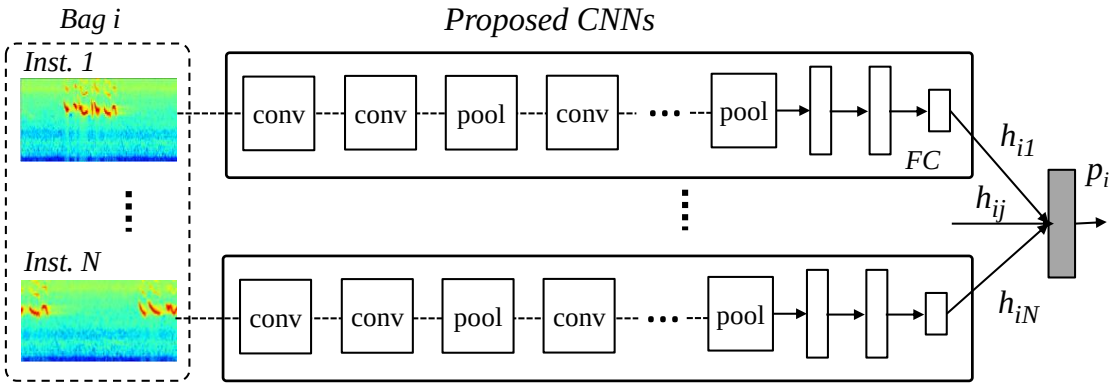

Figure 4.2: Architecture of our deeper CNN model adapted to MIL. The softmax layer is replaced with an aggregation layer.

of event classes should also not made too fine-grained. This will also higher the chance that a sufficiently large number of samples can be collected. For instance, "Church Bell" is better not subdivided further, e.g. in terms of its pitch.

In order to create such a novel audio event classification database, we harvested samples from Freesound [133]. This is a repository of audio samples uploaded by users. The database consists of 28 events as described in *Table 4.2*[1]. Note that since the sounds in the repository are tagged in free-form style and the words used vary a lot, the harvested sounds contain irrelevant sounds. For instance, a sound tagged 'cat' sometime does not contain a real cat meow, but instead a musical sound produced by a synthesizer. Furthermore sounds were recorded with various devices under various conditions (e.g. some sounds are very noisy and in others the audio event occurs during a short time interval between longer silences). This makes our database more challenging than previous datasets such as [134]. On the other hand, the realism of our selected sounds helps us to train our networks on sounds similar to those in actual consumer videos. With the above goals in mind we extended this initial freesound dataset, which we introduced in [91]), to 41 classes, including more diverse classes from the RWCP Sound Scene Database [134].

In order to reduce the noisiness of the data, we first normalized the harvested sounds and eliminated silent parts. If a sound was longer than 12 sec, we split the sound into pieces so that the split sounds lasted shorter than 12 sec. All audio samples were converted to 16 kHz sampling rate, 16 bits/sample, mono channel.

4.1.5 Multiple Instance Learning

Since we used web data to build our dataset (see *Sec. 4.1.4*), the training data is expected to be noisy and to contain outliers. In order to alleviate the negative effects of outliers, we also employed multiple instance learning (MIL) [135, 136]. In MIL, data is organized as bags $\{X_i\}$ and within each bag there are a number of instances $\{x_{ij}\}$. Labels $\{Y_i\}$ are provided only at the bag level, while labels of instances $\{y_{ij}\}$ are unknown. A positive bag means that at least one instance in the bag is positive, while a negative bag means that all instances in the bag are negative. We adapted our CNN architecture for MIL as shown in *Fig. 4.2*. N instances $\{x_1, \cdots, x_N\}$ in a bag are fed to a replicated CNN which shares its parameters. The last softmax layer is replaced with an aggregation layer where the outputs from each network $h = \{h_{ij}\} \in R^{M \times N}$ are aggregated. Here, M is the number of classes. The distribution of class of bag p_i is calculated as $p_i = f(h_{i1}, h_{i2}, \cdots, h_{iN})$ where $f()$ is an aggregation function. In this work, we investigate 2 aggregation functions: max

[1]The dataset is available at https://data.vision.ee.ethz.ch/cvl/ae_dataset

Table 4.2: The statistics of the dataset.

Class	Total minutes	# clip	Class	Total minutes	# clip
Acoustic guitar	23.4	190	Hammer	42.5	240
Airplane	37.9	198	Helicopter	22.1	111
Applause	41.6	278	Knock	10.4	108
Bird	46.3	265	Laughter	24.7	201
Car	38.5	231	Mouse click	14.6	96
Cat	21.3	164	Ocean surf	42	218
Child	19.5	115	Rustle	22.8	184
Church bell	11.8	71	Scream	5.3	59
Crowd	64.6	328	Speech	18.3	279
Dog barking	9.2	113	Squeak	19.8	173
Engine	47.8	263	Tone	14.1	155
Fireworks	43	271	Violin	16.1	162
Footstep	70.3	378	Water tap	30.2	208
Glass breaking	4.3	86	Whistle	6	78
			Total	768.4	5223

aggregation

$$p_i = \frac{exp(\hat{h_i})}{\sum_i exp(\hat{h_i})} \tag{4.3}$$

$$\hat{h_i} = \max_j(h_{ij}) \tag{4.4}$$

and Noisy OR aggregation [137],

$$p_i = 1 - \prod_j (1 - p_{ij}) \tag{4.5}$$

$$p_{ij} = \frac{exp(h_{ij})}{\sum_i exp(h_{ij})}. \tag{4.6}$$

Since it is unknown which sample is an outlier, we can not be sure that a bag has at least one positive instance. However, the probability that all instances in a bag are negative exponentially decreases with N, thus the assumption becomes very realistic.

4.2 Architecture Validation and Audio Event Recognition

We first evaluated our proposed deep CNN architectures and data augmentation method on the audio event recognition task. The aim here is to validate the proposed method and find an appropriate network architecture, since we can assume that a network that is more discriminative for the audio event recognition task gives us more discriminative AENet features for the other video analysis tasks.

4.2.1 Implementation Details

Through all experiments, 49 band log-filter banks, log-energy and their delta and delta-delta were used as a low-level descriptor, using 25 ms frames with 10 ms shift, except for the BoAW baseline described in *Sec. 4.2.2*. The input patch length was set to 400 frames (i.e. 4 sec). The effects of this length were further investigated in *Sec. 4.2.3*. During training, we randomly crop 4 sec for each sample. The networks were trained using mini-batch gradient descent based on back propagation with momentum. The learning rate was initially set to 0.001 and was reduced by a factor of 10 when the training error plateaued. The networks were trained for up to 5 $\times 10^4$ iterations. We applied dropout [138] to each fully-connected layer with as keeping probability 0.5. The batch size

Table 4.3: Accuracy of the deeper CNN and baseline methods, trained with and without data augmentation (%).

Method	Data augmentation	
	without	with
BoAW+SVM	74.7	79.6
BoAW+DNN	76.1	80.6
DNN+HMM	54.6	75.6
CNN+HMM	67.4	86.1
DNN+Large input	62.0	77.8
CNN+Large input	77.6	90.9
A	77.9	91.7
B	80.3	**92.8**

was set to 128, the momentum to 0.9. For data augmentation we used VTLP and the proposed EMDA. The number of augmented samples is balanced for each class. During testing, 4 sec patches with 50% shift were extracted and used as input to the Neural Networks. The class with the highest probability was considered the detected class. The models were implemented using the Lasagne library [126].

Similar to [131], the data was randomly split into training set (75%) and test set (25%). Only the test set was manually checked and irrelevant sounds not containing the target audio event were omitted.

4.2.2 State-of-the-art Comparison

In our first set of experiments we compared our proposed deeper CNN architectures to three different state-of-the-art baselines, namely, BoAW [63], HMM+DNN/CNN as in [139], and a classical DNN/CNN with large input field.

BoAW We used MFCC with delta and delta-delta as low-level descriptor. K-means clustering was applied to generate an audio word code book with 1000 centers. We evaluated both a SVM with a χ^2 kernel and a 4 layer DNN as classifiers. The layer sizes of the DNN classifier were (1024, 256, 128, 28).

DNN/CNN+HMM We evaluated the DNN-HMM system. The neural network architectures are described in the left 2 columns of Table 4.1. Both the DNN and CNN models are trained to estimate HMM state posteriors. The HMM topology consists of one state per audio event, and an ergodic architecture in which all states have equal transitions probabilities to all states, as in [71]. The input patch length for the CNN/DNN is 30 frames with 50% shift.

DNN/CNN+Large input field In order to evaluate the effect of using the proposed CNN architectures, we also evaluated the baseline DNN/CNN architectures with the same large input field, namely, 400 frame patches.

The classification accuracies of these systems – trained with and without data augmentation – are shown in Table 4.3. Even without data augmentation, the proposed CNN architectures outperform all previous methods. Furthermore, the performance is significantly improved by applying data augmentation, yielding a 12.5% improvement for the *B* architecture. The best result was obtained by the *B* architecture with data augmentation. It is important to note that the *B* architecture outperforms the classical DNN/CNN even though it has fewer parameters, as shown in *Table 4.1*. This result corroborates the efficiency of deeper CNNs with small kernels for modelling large input fields. This observation coincides with that made in earlier work in computer vision in [74].

4.2.3 Effectiveness of a Large Input Field

Our second set of experiments focuses on input field size. We tested our CNN with different patch size 50, 100, 200, 300, 400 frames (i.e. from 0.5 to 4 sec). The *B* architecture was used for this

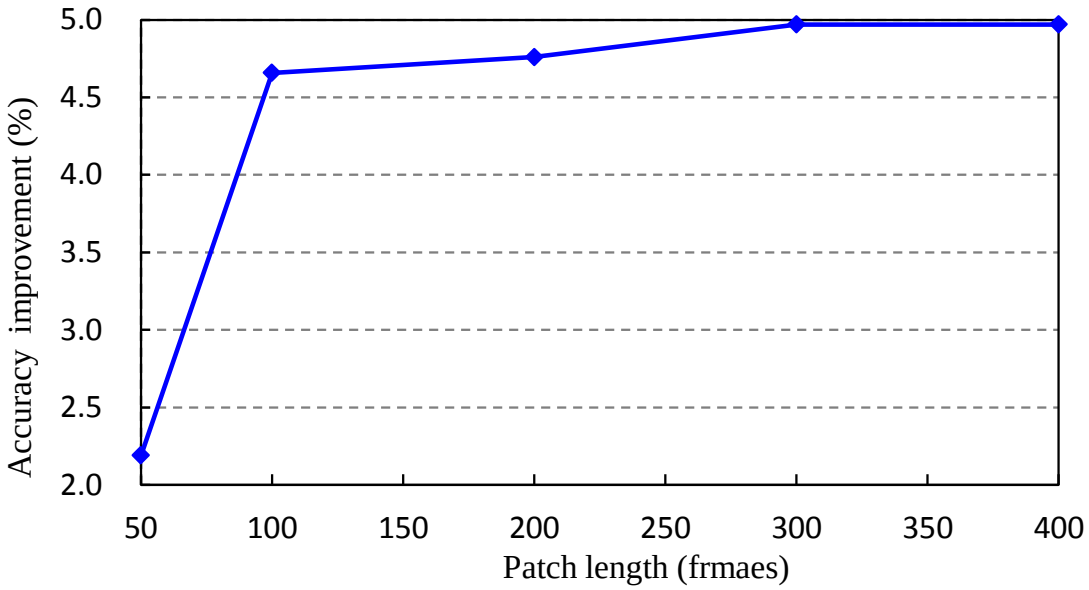

Figure 4.3: Performance of our network for different input patch lengths. The plot shows the increase over using a CNN+HMM with a small input field of 30 *frames.*

experiment. As a baseline we evaluated the CNN+HNN system described in *Sec. 4.2.2* but using our architecture B, rather than a classical CNN. The performance improvement over the baseline is shown in *Fig. 4.3*. The result shows that larger input fields improve the performance. Especially the performance with patch length less than 1 sec sharply drops. This proves that modeling long signals directly with a deeper CNN is superior to handling long sequences with HMMs.

4.2.4 Effectiveness of Data Augmentation

We verified the effectiveness of our EMDA data augmentation method in more detail. We evaluated 3 types of data augmentation: EMDA only, VTLP only, and a mixture of EMDA and VTLP (50%, 50%) with different numbers of augmented samples 10k, 20k, 30k, 40k. Fig. 4.4 shows that using both EDMA and VTLP always outperforms EDMA or VTLP only. This shows that EDMA and VTLP perturb the original data and thus create new samples in a different way (VTLP changes speed or pitch while EMDA changes number of sources, loudness, and sound color.). Applying both provides a more effective variation of data and helps to train the network to learn a more robust and general model from a limited amount of data.

4.2.5 Effects of Multiple Instance Learning

The A and B architectures with a large input field were adapted to MIL, to handle the noise in the database. The number of parameters were identical since both the max and Noisy OR aggregation methods are parameter-free. The number of instances in a bag was set to 2. We randomly picked 2 instances from the same class during each epoch of the training. Table 4.4 shows that MIL didn't improve performance in this case. However, MIL with a medium size input field (i.e. 2 sec) performs as good as or even slightly better than single instance learning with a large input field. This is perhaps due to the fact that the MIL took the same size input length (2 sec $\times$2 instances = 4 sec), while it had fewer parameter. Thus it managed to learn a more robust model. We also tried training the networks with 4 instances in a bag with 2 sec input field. However, we could not observe improved performance compared to using 2 instances.

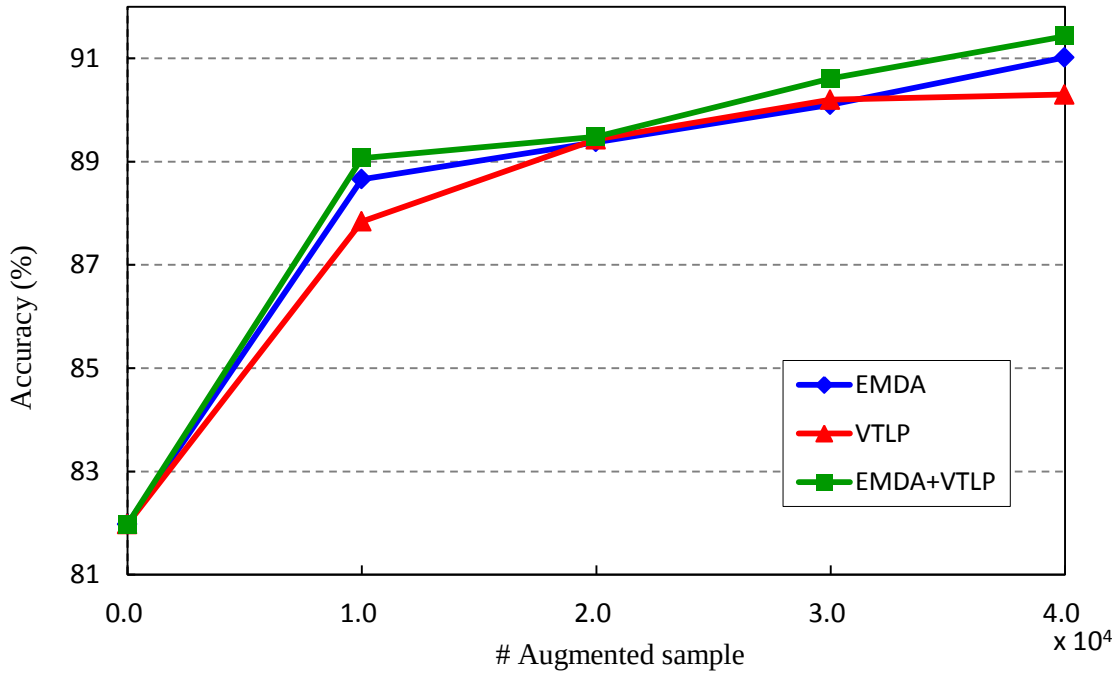

Figure 4.4: Effects of different data augmentation methods with varying amounts of augmented data.

Table 4.4: Accuracy of MIL and normal training (%).

Architecture	Single instance	MIL		
		Noisy OR	Max	Max (2sec)
A	91.7	90.4	92.6	**92.9**
B	92.8	91.3	92.4	92.8

4.3 Conclusions of this Chapter

We proposed a new, scalable deep CNN architecture to learn a model for entire audio events end-to-end, and outperforming the state-of-the-art on this task. Experimental results showed that deeper networks with smaller filters perform better than previously proposed CNNs and other baselines. We further proposed a data augmentation method that prevents over-fitting and leads to superior performance even when the training data is limited.

Chapter 5

Video Analysis using AENet Features

Finally, we are tacking video analysis by leveraging the audio event recognition (AER) networks discussed in the previous Chapter.

Our work is partially motivated by the success of deep features in vision, e.g. in image [140] and video [83] analysis. The features learnt in these networks (activations of the last few layers) have shown to perform well on transfer learning tasks [140]. Yet, a large and diverse dataset is required so that the learnt features become sufficiently generic and work in a wide range of scenarios. Unfortunately, most existing audio datasets are limited to a specific category, e.g. speech [123], music, environmental sounds in offices [141]). In Chapter 4, we designed an audio event dataset so that such more general deep audio features can be trained. In this Chapter, we investigate if learnt CNN deep feature is useful for video analysis tasks, namely action recgnition and video highlight detection.

5.1 Audio Event Net (AENet) Feature

Once the network was trained, it can be used as a feature extractor for audio and video analysis tasks. An audio stream is split into clips whose lengths are equal to the length of the network's input field. In our experiments, we took a 2 sec length (200 frame) patch since it did not degrade the performance considerably (Fig. 4.3) but gave us a reasonable time resolution with easily affordable computational complexity. Through our experiment we split audio streams with 50% overlap, although clips can be overlapped with arbitrary length depending on the desired temporal resolution. These clips are fed into the network architecture "A" and activations of the second last fully connected layer are extracted. The activations are then L2 normalized to form audio features. We call these features 'AENet features' from now on.

5.2 Action Recognition

We evaluated the AENet features on the USF101 dataset [142]. This dataset consists of 13,320 videos of 101 human action categories, such as Apply Eye Makeup, Blow Dry Hair and Table Tennis.

5.2.1 Baselines

The AENet features were compared with several baselines: visual only and with two commonly used audio features, namely MFCC and BoAW. Thirteen-dimensional MFCCs and its delta and delta delta were extracted, with 25 ms window with 10 ms shift and averaged for a clip. In order to form BoAW features, MFCC, delta and delta delta were clustered by K-means to obtain 1000 codebook elements. The audio features are then concatenated with the visual features.

Table 5.1: Accuracy of the deeper CNN and baseline methods, trained with and without data augmentation (%).

Method	accuracy
C3D	82.2
C3D+MFCC	82.5
C3D+BoAW	82.9
C3D+AENet	**85.3**

5.2.2 Setup

The AENet features were averaged within a clip. We did not fine-tune the network since our goal is to show the generality of the AENet features. For all experiments, we use a multi-class SVM classifier with a linear kernel for fair comparison. As visual features, we used C3D features [83] since it is a standard method for fast video analysis. More recent methods, such as [82] have higher performance, but have a high computational cost, as they rely on optical flow inputs. We observed that half of the videos in the dataset contain no audio. Thus, in order to focus on the effect of the audio features, we used only videos that do contain audio. This resulted in 6837 videos of 51 categories. We used the three split setting provided with this dataset and report the averaged performance.

5.2.3 Results

The action recognition accuracy of each feature set are presented in Table 5.1. The results show that the proposed AENet features significantly outperform all baselines. Using MFCC to encode audio on the other hand, does not lead to any considerable performance gain over visual features only. One difficulty of this dataset could be that the characteristic sounds for certain actions only occur very sparsely or that sounds are very similar, thus making it difficult to characterize sound tracks by averaging or taking the histogram of frame-based MFCC features. AENet features, on the other hand, perform well without fine-tuning. This suggests that AENet learned more discriminative and general audio representations.

In order to further investigate the effect of AENet features, we show the difference between the confusion matrices when using C3D vs C3D+AENet in Table 5. Positive diagonal values indicate an improvement in classification accuracy for the corresponding classes, whereas positive values on off-diagonal elements indicate increased mis-classification. The class indices were ordered according to descending accuracy gain. The Fig. 5.1 shows that the performance was improved or remains the same for most classes by using AENet features. The off-diagonal elements of the confusion matrix difference also show some interesting properties, e.g. the confusion of Playing Dhal (index 8) and Playing Cello (index 10) was descreased by adding the AENet features. This may be due to the clear difference of the cello and Dhal sounds while their visual appearance is sometimes similar: a person holding a brownish object in the middle and moving his hands arround the object. The confusion between Playing Cello and Playing Daf (index 2) , on the other hand, was slightly increased by using AENet features, since both are percussion instruments and the sound from these instruments may be reasonably similar.

5.3 Video Highlight Detection

We further investigate the effectiveness of AENet features for finding better highlights in videos. Thereby the goal is to find domain-specific highlight segments [8] in long consumer videos.

5.3.1 Dataset

The dataset consists of 6 domains, “skating”, “gymnastics”, “dog”, “parkour”, “surfing”, and “skiing”. Each domain has about 100 videos with various lengths, harvested from Youtube. The total

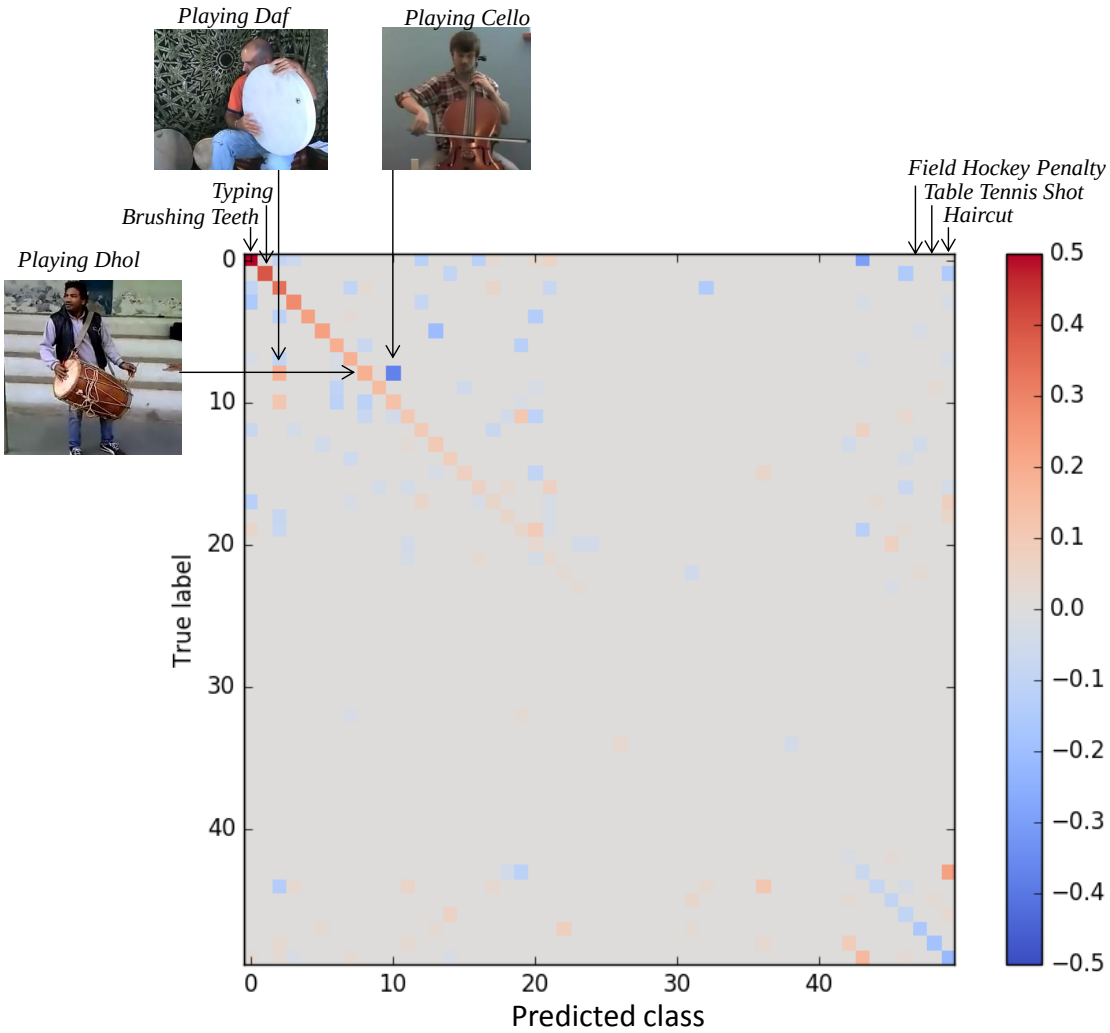

Figure 5.1: Difference of confusion matrices of C3D+AENet and C3D only. Positive diagonal values indicate a performance improvement for this class, while negative, off-diagonal values indicate that the mis-classification increased. The performance was improved or remains the same for most classes by using AENet features.

accumulated time is 1430 minutes. The dataset was split in half for training and testing. Highlights for the training set were automatically obtained by comparing raw and edited pairs of videos. The segments (moments) contained in the edited videos are labeled as highlights while moments only appearing in the raw videos are labeled as non-highlights. See [8] for more information.

5.3.2 Setup

If a moment contained multiple features, they were averaged within the moment. We used the C3D features for the visual appearance and concatenated then with AENet features. A H-factor y was estimated by neural networks which had two hidden layers and one output unit. A higher H-factor value indicates highlight moments, while a lower value indicates a non-highlight moment, as in [8]. Note that the classification model can not be applied since highlights are not comparable among videos: a highlight in one video may be boring compared to a non-highlight moment in other videos. A training objective which only depends on the relative 'highlightness' of moments

from a video is more suitable. Therefore, we followed [8] and used a ranking loss

$$L_{ranking} = \sum_i max(1 - y_i^{pos} + y_i^{neg}) \tag{5.1}$$

where $\{y_{pos}\}$ and $\{y_{neg}\}$ are the outputs of the networks for the highlight moments and non-highlight moments of a video. Eq. (5.1) required the network to score highlight moments higher than non-highlight moments within a video, but does not put constraints on the absolute values of the scores. Since all moments are labeled as a highlight if the moments are included in the edited video, the label tends to be redundant and noisy. To overcome this, we modified the ranking loss by applying the Huber loss [90]

$$L_{Huber} = \begin{cases} 1/2L_{ranking}^2, & \text{if } L_{ranking} < \delta \\ \delta(-L_{ranking} + 1/2\delta), & \text{otherwise} \end{cases} \tag{5.2}$$

and further by replacing the ranking loss by a multiple instance ranking loss

$$L_{miranking} = max(1 - max_i(y_i^{pos}) + y^{neg}). \tag{5.3}$$

The Huber loss has a smaller gradient for margin violations, as long as the positive example scores are higher than the negative, which alleviates the effect from ambiguous samples and leads to a more robust model. Eq. (5.3) takes I highlight moments $\{y_i^{pos} | i = 1, ..., I\}$ and requires only the highest scoring segment among them to rank higher than the negative. It is thus more robust to false positive samples, which exist in the training data, due to the way it was collected [8]. We used $I = 2$ in our experiment and the network was trained five times and the scores were averaged.

5.3.3 Baselines

As for action recognition, we consider three baselines, C3D features only, C3D with MFCC, and BoAW. MFCC features were averaged within a moment and the BoAW was calculated for each moment. A DNN based highlight detector was trained in the same manner as the ranking loss.

5.3.4 Results

The mean average precisions (mAP) of each domain, averaged over all videos on the test set, are presented in figure 5.2. For most of the domains, AENet features perform the best or are competitive with the best competing features. The overall performance of AENet features significantly outperforms the baselines, achieving 56.6% mAP which outperforms the current state-of-the-art of [8]. For "skating" and "surfing", all audio features help to improve performance, probably due to the fact that videos of these domains contain characteristic sounds at highlight moments: when a skater performs some stunt or a surfer starts to surf. For "skiing" and "parkour", AENet features improve performance while some other features do not. In the "parkour" domain, pulsive sounds such as foot steps which may occur when a player jumps, typically characterize the highlights. The MFCC features might have failed to capture the characteristics of the foot step sound because of the averaging of the features within the moment. BoAW features could keep the characteristics by taking a histogram, but AENet features are far better to characterize such pulsive sounds. For "dog" and "gymnastics", audio features do not improve performance or even slightly lower it. We observed that many videos in these domains do not contain any sounds which characterize the highlights, but contain constant noise or silence for the entire video. This may cause over-fitting to the training data. We further investigated the effects of loss functions. Table 5.2 shows the mAPs trained with the ranking loss in Eq. (5.1), Huber loss in Eq. (5.2) and the multiple instance ranking loss (MIRank) in Eq. (5.3). The Huber loss and MIRank both increase the performance by 1.2% and 2.4%, respectively. This shows that more robust loss functions help in this scenario, where the labels are affected by noise and contain false positives.

Qualitative evaluation: In figure 5.3, 5.4 and 5.5, we illustrate some typical examples of highlight detection results for the domains *parkour*, *skating* and *surfing*, i.e. the domains that were most improved by introducing AENet features. The last two rows give examples of highlight

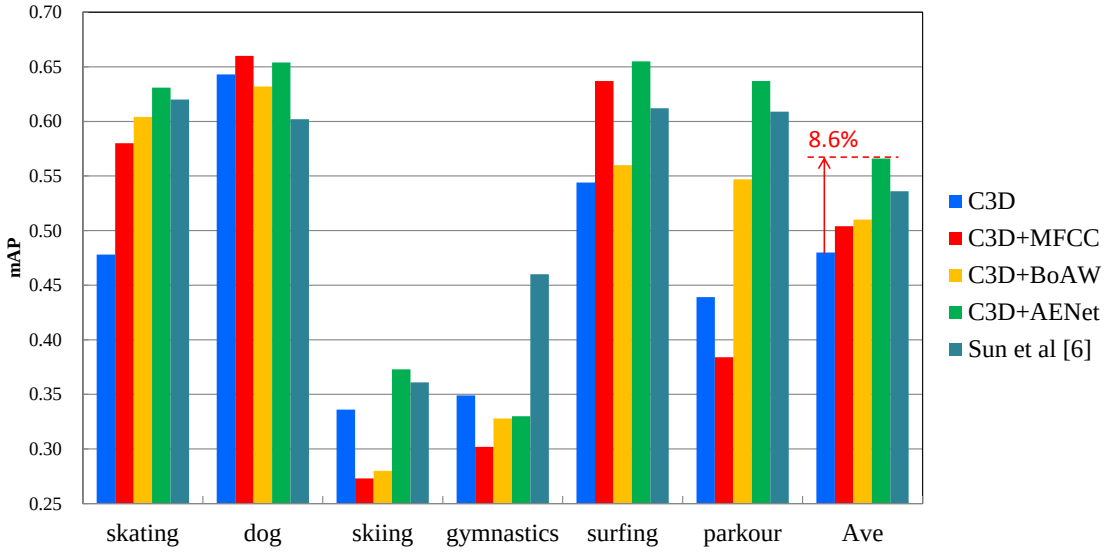

Figure 5.2: Domain specific highlight detection results. AENet features outperform other audio features with an average improvement of 8.6% over the C3D (visual) features.

Table 5.2: Effects of loss function. Mean average precision trained with different loss functions.

Method	mAP
Sum et al. [8]	53.6
C3D+AENet ranking loss	53.0
C3D+AENet Huber loss	54.2
C3D+AENet Huber loss MIRank	**56.6**

videos which were created by taking the moments with the highest H-factors so that the video length would about 20% of original video. In the video of parkour shown in figure 5.3, a higher H-factor was assigned around the moments in which a man was running, jumping and turning a somersault, when we used AENet features, as shown in the second row. On the other hand, the moments which clearly show the man in the video and have less camera motion tend to get a higher H-factor when only visual (C3D) features were used. The AENet features could characterize a footstep sound and therefore detected the highlights more reliably. Figure 5.4 illustrates a video with seven jumping scenes. With the AENet features, we can observe peaks in the H-factor at all jumps, since AENet features effectively capture the sound made by a skater jumping. Without audio information, the highlight detector failed to detect some jumping scenes and tended to pick moments with general motion including camera motion. For *surfing*, highlight videos created by using AENet features capture the whole sequence from standing on the board up to falling into the sea, while a highlight video created from visual features only sometimes misses some sequence of surfing and contains boring parts showing somebody just floating and waiting for the next wave. By including AENet features, the system really recognizes the difference in sound when somebody is surfing or not, and it detects highlights more reliably.

5.4 Conclusions of this Chapter

We proposed the AENet feature, activations of the networks trained on ASR task. We showed the generality of proposed AENet feature by showing superior performance on action recognition and video highlight detection, compared to commonly used audio features. We believe that our new audio features will also give similar improvements for other video analysis tasks, such as action

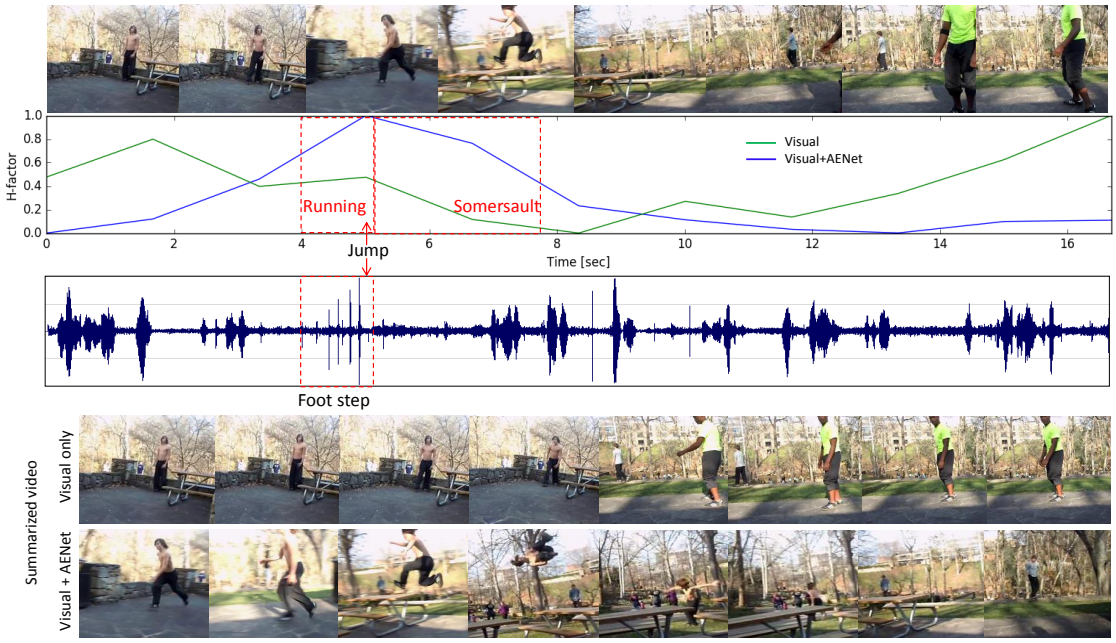

Figure 5.3: An example of an original video (top), h-factor (2nd row), wave form (3rd row), summarized video by using only visual features (4th row) and summarized video by using audio and visual features for parkour. AENet features capture the footstep sound and more reliably derive high h-factors around the moments when a person runs, jumps or performs some stunt such as a somersault.

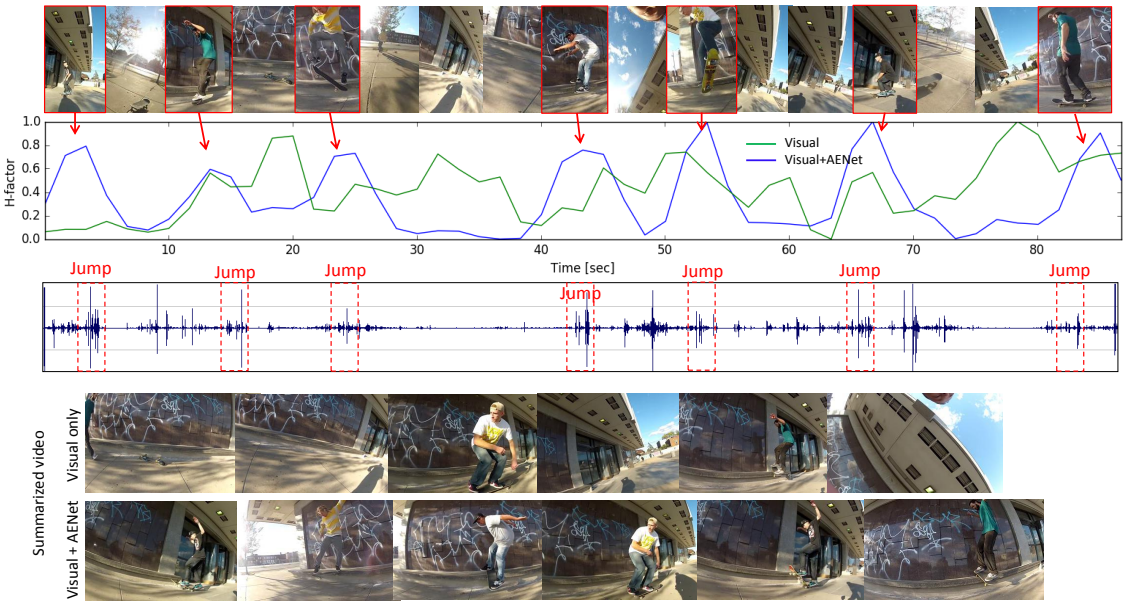

Figure 5.4: An example of an original video (top), h-factor (2nd row), wave form (3rd row), summarized video by using only visual features (4th row) and summarized video by using audio and visual features for skating. In the video, there are six scenes where skaters jump and perform a stunt. These moments are clearly indicated by the h-factor calculated from both the visual and AENet features. Sounds made by skaters jumping are characterized by the AENet well and help to reliably capture such moments.

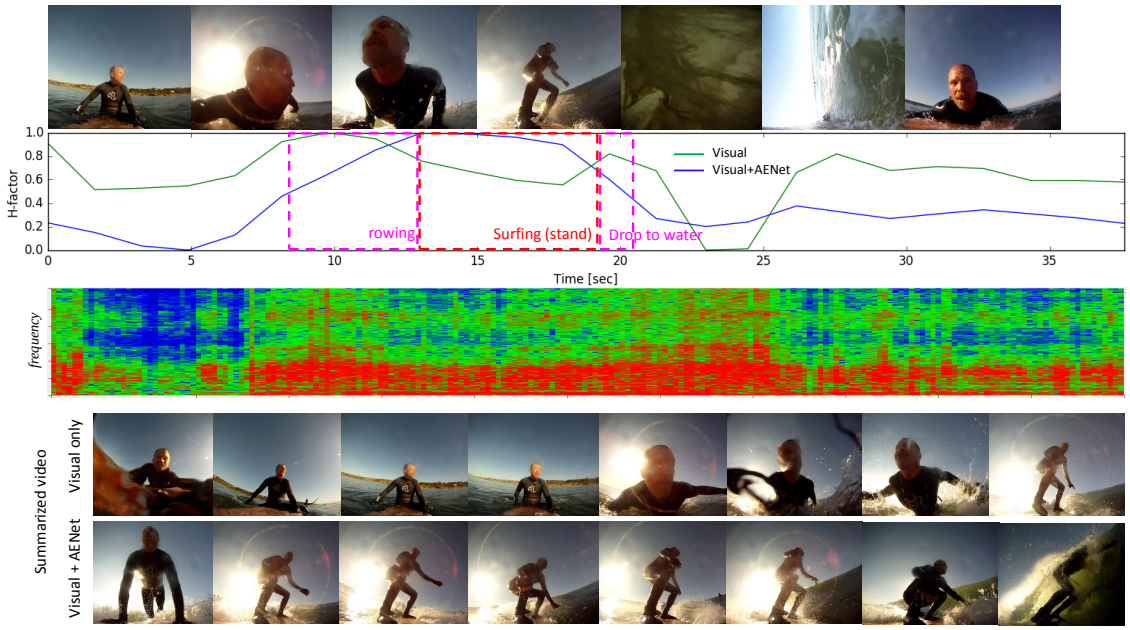

Figure 5.5: An example of an original video (top), h-factor (2nd row), spectrogram of audio (3rd row), summarized video by using only visual features (4th row) and summarized video by using audio and visual features for surfing. Surfing scenes contain louder and high-frequency sounds compared to scenes where a surfer is floating on the sea and waiting. This information helped to more reliably detect the surfing scene.

localization and temporal video segmentation.Investigating recently proposed architectures such as ResNet will be future work.

Chapter 6

Conclusion

We addressed different kinds of problems that we are facing to realize machine multimedia understanding in real world and proposed sets of deep learning based audio technologies to overcome them, namely audio source separation, joint acoustic model and pronunciation dictionary learning for low resource languages, audio event recognition for general audio analysis, and audio features for video analysis. Here, we summarize novelty, findings and utilities of our works and discuss future works.

6.1 Novelty and Findings

Most of works in this thesis were done in early stage of deep learning evolution. Not many works had been done in domains of audio source separation, low-resource speech recognition, audio event recognition, and audio visual video analysis. Our works showed how to apply deep learning approach in an efficient and effective ways and demonstrated superior performance over existing methods.

In audio source separation, we proposed a deep neural network architectures that model the spectrogram in multi-resolution and multi-band with combination of DenseNet and LSTM. Experimental results show the effectiveness and efficiency of multi-resolution and multi-band modeling with combination of DenseNet and LSTM, achieving state-of-the-art performance in DSD100 and MUSDB18 datasets. Moreover, we proposed a classification based phase recovery method phase to handle a periodic nature of phase. The proposed method outperforms a regression based phase recovery method as well as the state-of-the-art consistent anisotropic Wiener filtering. This indicates the effectiveness of the proposed method that discritize phase and estimate target phase by classification approach. Furthermore, we also address the problem of separating a mixture of unknown number of speakers by recursively separating a speaker one by one. For the first time, we show that a single model can separate speakers even when the number of speakers in the mixture is more than that seen during the model training.

To built automatic speech recognition for low resource languages, we proposed a novel joint dictionary and sub-word unit learning framework for ASRs. The proposed method does not require linguistic expertise, and can automatically create the set of sub-word units and the corresponding pronunciation dictionary. In our method, reliable pronunciations are estimated from multiple utterances by an efficient approximation of K-dimensional Viterbi algorithm which estimates the most probable HMM state sequence common to multiple utterances of a word. Experimental results show that the proposed method significantly outperforms the phone based methods which even get manually prepared dictionary and hand crafted transcriptions as inputs.

For audio event recognition, we proposed a new, scalable deep CNN architecture to learn a model for entire audio events end-to-end, and outperforming the state-of-the-art on this task. Experimental results showed that deeper networks with smaller filters perform better than previously proposed CNNs and other baselines. We further proposed a data augmentation method that prevents over-fitting and leads to superior performance even when the training data is limited.

Although more advanced network architectures are proposed and applied to AER after our work had been completed, basic concept on this work is widely used to date.

We used the learned network activations as audio features for video analysis and showed that they generalize well. Using the proposed features led to superior performance on action recognition and video highlight detection, compared to commonly used audio features. We believe that our new audio features will also give similar improvements for other video analysis tasks, such as action localization and temporal video segmentation.

6.2 Utilities

The audio source separation is applicable not only to machine multimedia understanding but also to multi-channel audio creation and Karaoke. At Sony, we have been applying audio source separation technology to our products, contents and services since recent years. For instance, we integrate speech-noise separation to a dog robot 'aibo' to robustly react owners voice in various conditions. We also start to using the source separation for movie and music creation. The proposed method has just started to be implemented to our society.

The automatic speech separation for low resource languages is useful event today. More than 1000 hours of speech corpus is available for major languages such as English, there are many local languages that have very limited amount of data. For instance, there are hundreds of local languages are spoken in India. Even when we consider major ones, about 20 languages are used in India and very few languages have ASR systems due to the limited corpus. Our proposed method has potential to address this problem since it does not require human experts to built pronunciation dictionaries for these languages.

AER and making it use for video analysis becomes more and more demanding. An audio tagging system for a recorded audio on mobile phone for an efficient search has been recently announced. AER for surveillance is also tested for a service launch. machine has just started to "listen" non-speech sounds in the real environment.

6.3 Future Works

In this thesis, we proposed sets of technologies for multimedia analysis in real world environment such as existence of overlapping, many different kinds of sounds and low resource availability. Combining and integrating individual methods to one system would be one possible direction of future works.

Application of individual methods can be further explored. For instance, we discussed two video analysis tasks to showcase the usefulness of the proposed audio feature. However, human cognition is much more diverse: human recognize much more things and higher level of concept. Increasing the number of concept or level of concept to be recognized is also important direction for future works.

Another direction could be make DNN models more computationally efficient so that they can operate on edge devices. Although network band width becomes wider and cloud computing becomes more popular, uploading data sometime has problem due to privacy, copy rights, or latency. In this case, making models low footprint becomes more important to save power consumption, avoid heat problem, or make product cost low.

Bibliography

[1] Y.-G. Jiang, G. Ye, S.-F. Chang, D. Ellis, and A. C. Loui, "Consumer video understanding: A benchmark database and an evaluation of human and machine performance," in *Proc. ICMR*, pp. 1–8, 2011.

[2] Y. Li and B. Mérialdo, "Video summarization based on balanced AV-MMR," in *Proc. International Conference on Multimedia Modeling*, pp. 1–6, 2012.

[3] T. Zhang and C.-C. J.Kuo, "Audio content analysis for online audiovisual data segmentation and classification," *IEEE Trans. on Speech and Audio Processing*, vol. 9, no. 4, pp. 441–457, 2001.

[4] K. Lee and D. P. W. Ellis, "Audio-based semantic concept classification for consumer video," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 18, no. 6, pp. 1406–1416, 2010.

[5] J. Liang, Q. Jin, X. He, Y. Gang, J. Xu, and X. Li, "Detecting semantic concepts in consumer videos using audio," in *ICASSP*, pp. 2279–2283, 2015.

[6] Q. Wu, Z. Wang, F. Deng, Z. Chi, and D. D. Feng, "Realistic human action recognition with multimodal feature selection and fusion," *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans*, vol. 43, no. 4, pp. 875–885, 2013.

[7] D. Oneata, J. Verbeek, and C. Schmid, "Action and event recognition with fisher vectors on a compact feature set," in *Proc. ICCV*, pp. 1817–1824, 2013.

[8] M. Sun, A. Farhadi, and S. Seitz, "Ranking domain-specific highlights by analyzing edited videos," in *ECCV*, pp. 787–802, 2014.

[9] M. R. Naphade and T. S. Huang, "Indexing , Filtering , and Retrieval," *IEEE Transactions on Multimedia*, vol. 3, no. 1, pp. 141–151, 2001.

[10] W. Hu, N. Xie, L. Li, X. Zeng, and S. Maybank, "A Survey on Visual Content-Based Video Indexing and Retrieval," *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 41, no. 6, pp. 797–819, 2011.

[11] Y. Wang, S. Rawat, and F. Metze, "Exploring audio semantic concepts for event-based video retrieval," in *Proc. ICASSP*, pp. 1360–1364, 2014.

[12] M. Gygli, H. Grabner, and L. Van Gool, "Video summarization by learning submodular mixtures of objectives," in *CVPR*, pp. 3090–3098, 2015.

[13] N. Q. K. Duong, E. Vincent, and R. Gribonval, "Under-determined reverberant audio source separation using a full-rank spatial covariance model," *IEEE Trans. Audio, Speech & Language Processing*, vol. 18, no. 7, pp. 1830–1840, 2010.

[14] D. Fitzgerald, A. Liutkus, and R. Badeau, "PROJET - spatial audio separation using projections," in *Proc. ICASSP*, pp. 36–40, 2016.

[15] A. Liutkus, D. Fitzgerald, and R. Badeau, "Cauchy nonnegative matrix factorization," in *Proc. WASPAA*, pp. 1–5, 2015.

[16] J. LeRoux, J. R. Hershey, and F. Weninger, "Deep NMF for speech separation," in *Proc. ICASSP*, p. 66–70, 2015.

[17] Y. Mitsufuji, S. Koyama, and H. Saruwatari, "Multichannel blind source separation based on non-negative tensor factorization in wavenumber domain," in *Proc. ICASSP*, pp. 56–60, 2016.

[18] A. Liutkus, D. Fitzgerald, Z. Rafii, B. Pardo, and L. Daudet, "Kernel additive models for source separation," *IEEE Trans. Signal Processing*, vol. 62, no. 16, pp. 4298–4310, 2014.

[19] A. Ozerov and C. Févotte, "Multichannel nonnegative matrix factorization in convolutive mixtures for audio source separation," *IEEE Trans. Audio, Speech & Language Processing*, vol. 18, no. 3, pp. 550–563, 2010.

[20] A. Liutkus, D. Fitzgerald, and Z. Rafii, "Scalable audio separation with light kernel additive modelling," in *Proc. ICASSP*, pp. 76–80, 2015.

[21] D. Fitzgerald, A. Liutkus, and R. Badeau, "Projection-based demixing of spatial audio," *IEEE/ACM Trans. Audio, Speech & Language Processing*, vol. 24, no. 9, pp. 1560–1572, 2016.

[22] A. A. Nugraha, A. Liutkus, and E. Vincent, "Multichannel music separation with deep neural networks," in *Proc. EUSIPCO*, pp. 1748–1752, 2015.

[23] S. Uhlich, F. Giron, and Y. Mitsufuji, "Deep neural network based instrument extraction from music," in *Proc. ICASSP*, pp. 2135–2139, 2015.

[24] S. Uhlich, M. Porcu, F. Giron, M. Enenkl, T. Kemp, N. Takahashi, and Y. Mitsufuji, "Improving music source separation based on deep networks through data augmentation and network blending," in *Proc. ICASSP*, pp. 261–265, 2017.

[25] D. Griffin and J. Lim, "Signal estimation from modified short-time fourier transform," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 32, no. 2, pp. 236–243, 1984.

[26] J. L. Roux, N. Ono, and S. Sagayama, "Explicit consistency constraints for stft spectrograms and their application to phase reconstruction," in *Proc. ISCA SAPA*, pp. 23–28, 2008.

[27] J. L. Roux and E. Vincent, "Consistent wiener filtering for audio source separation," *IEEE Signal Processing Letters*, vol. 20, pp. 217–220, March 2013.

[28] D. Gunawan and D. Sen, "Iterative phase estimation for the synthesis of separated sources from single-channel mixtures," *IEEE Signal Processing Letters*, vol. 17, pp. 421–424, May 2010.

[29] N. Sturmel and L. Daudet, "Informed source separation using iterative reconstruction," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 21, pp. 178–185, Jan 2013.

[30] P. Magron, J. L. Roux, and T. Virtanen, "Consistent anisotropic wiener filtering for audio source separation," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 269–273, Oct 2017.

[31] D. S. Williamson, Y. Wang, and D. Wang, "Complex ratio masking for monaural speech separation," *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, vol. 24, no. 3, pp. 483–492, 2016.

[32] L. Drude, B. Raj, and R. Haeb-Umbach, "On the appropriateness of complex-valued neural networks for speech enhancement," in *Interspeech 2016*, pp. 1745–1749, 2016.

[33] Y.-S. Lee, C.-Y. Wang, S.-F. Wang, J.-C. Wang, , and C.-H. Wu, "Fully complex deep neural network for phase-incorporating monaural source separation," in *Proc. ICASSP*, pp. 281–285, 2017.

[34] F. R. Bach and M. I. Jordan, "Learning Spectral Clustering, with Application to Speech Separation," *The Journal of Machine Learning Research*, vol. 7, pp. 1963–2001, Jan 2006.

[35] K. Hu and D. Wang, "An Unsupervised Approach to Cochannel Speech Separation," *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, vol. 21, pp. 122—131, Jan 2013.

[36] M. N. Schmidt and R. K. Olsson, "Single-channel speech separation using sparse non-negative matrix factorization," in *Proc. Interspeech*, 2006.

[37] T. Virtanen and A. T. Cemgil, "Mixtures of gamma priors for non-negative matrix factorization based speech separation," in *Proc. Independent Component Analysis and Signal Separation (ICA)*, pp. 646–653, 2009.

[38] G. J. Mysore and P. Smaragdis, "A non-negative approach to language informed speech separation," in *Proc. Latent Variable Analysis and Signal Separation (LVA/ICA)*, pp. 356–363, 2012.

[39] Z. Wang and F. Sha, "Discriminative non-negative matrix factorization for single-channel speech separation," in *Proc. ICASSP*, pp. 3749–3753, 2014.

[40] Y. Isik, J. L. Roux, Z. Chen, S. Watanabe, and J. R. Hershey, "Single-channel multi-speaker separation using deep clustering," in *Proc. Interspeech*, pp. 545–549, 2016.

[41] Z.-Q. Wang, J. L. Roux, D. Wang, and J. R. Hershey, "End-to-end speech separation with unfolded iterative phase reconstruction," in *Proc. Interspeech*, pp. 2708–2712, 2018.

[42] M. Kolbæk, D. Yu, Z.-H. Tan, and J. Jensen, "Multitalker speech separation with utterance-level permutation invariant training of deep recurrent neural networks," *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, vol. 25, p. 1901–1913, 2017.

[43] Y. Luo and N. Mesgarani, "Tasnet: Time-domain audio separation network for real-time, single-channel speech separation," in *Proc. ICASSP*, 2018.

[44] Y. Luo and N. Mesgarani, "Tasnet: Surpassing ideal time-frequency masking for speech separation," *CoRR*, 2018.

[45] Z. Chen, Y. Luo, and N. Mesgarani, "Deep attractor network for single-microphone speaker separation," in *Proc. ICASSP*, 2017.

[46] Y. Luo, Z. Chen, and N. Mesgarani, "Speaker-independent speech separation with deep attractor network," *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, vol. 26, pp. 787–796, April 2018.

[47] A. Das and M. Hasegawa-Johnson, "Cross-lingual transfer learning during supervised training in low resource scenarios," in *Proc. Interspeech*, pp. 1–5, 2015.

[48] Y. Qian, D. Povey, and J. Liu, "State-level data borrowing for low-resource speech recognition based on subspace GMMs," in *Proc. Interspeech*, pp. 553–556, 2011.

[49] L. Besacier, E. Barnard, A. Karpov, and T. Schultz, "Automatic speech recognition for under-resourced languages : A survey," *Speech Communication*, vol. 56, pp. 85–100, 2014.

[50] M. Saraçlar, H. Nock, and S. Khudanpur, "Pronunciation modeling by sharing Gaussian densities across phonetic models," *Computer Speech & Language*, vol. 14, no. 2, pp. 137–160, 2000.

[51] M. Wester, "Pronunciation modeling for ASR - Knowledge-based and data-derived methods," *Computer Speech and Language*, vol. 17, no. 1, pp. 69–85, 2003.

[52] T. Hain, "Implicit modelling of pronunciation variation in automatic speech recognition," *Speech Communication*, vol. 46, no. 2, pp. 171–188, 2005.

[53] I. Mcgraw, I. Badr, and J. R. Glass, "Learning lexicons from speech using a pronunciation mixture model," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 21, no. 2, pp. 357–366, 2013.

[54] R. Singh, B. Raj, and R. M. Stern, "Automatic generation of subword units for speech recognition systems," *IEEE Trans. on Speech and Audio Processing*, vol. 10, no. 2, pp. 89–99, 2002.

[55] A. Ghoshal, D. Povey, M. Agarwal, P. Akyazi, N. Goel, M. Karafi, A. Rastrow, R. C. Rose, P. Schwarz, S. Thomas, and I. Allahabad, "Approaches to automatic lexicon learning with limited trainging examples," in *Proc. ICASSP*, pp. 5094–5097, 2010.

[56] T. Naghibi, S. Hoffmann, and B. Pfister, "An efficient method to estimate pronunciation from multiple utterances," in *Proc.Interspeech*, pp. 1951–1955, 2013.

[57] T. Holter and T. Svendsen, "Combined optimisation of baseforms and model parameters in speech recognition based on acoustic subword units," in *IEEE Workshop Automatic Speech Recognition*, pp. 199–206, 1997.

[58] M. Bacchiani and M. Ostendorf, "Joint lexicon, acoustic unit inventory and model design," *Speech Communication*, vol. 29, no. 2, pp. 99–114, 1999.

[59] A. J. Eronen, V. T. Peltonen, J. T. Tuomi, A. P. Klapuri, S. Fagerlund, T. Sorsa, G. Lorho, and J. Huopaniemi, "Audio-based context recognition," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 14, no. 1, pp. 321–329, 2006.

[60] Z. Huang, Y. C. Cheng, K. Li, V. Hautamäki, and C. H. Lee, "A blind segmentation approach to acoustic event detection based on I-vector," in *Proc. INTERSPEECH*, pp. 2282–2286, 2013.

[61] A. Temko, E. Monte, and C. Nadeu, "Comparison of sequence discriminant support vector machines for acoustic event classification," in *Proc. ICASSP*, vol. 5, pp. 721–724, 2006.

[62] H. Phan, L. Hertel, M. Maass, R. Mazur, and A. Mertins, "Representing nonspeech audio signals through speech classification models," in *Proc. INTERSPEECH*, pp. 3441–3445, 2015.

[63] S. Pancoast and M. Akbacak, "Bag-of-Audio-Words Approach for Multimedia Event Classification," in *Proc. Interspeech*, (Portland, OR, USA), pp. 2105–2108, 2012.

[64] W. Choi, S. Park, D. K. Han, and H. Ko, "Acoustic event recognition using dominant spectral basis vectors," in *Proc. INTERSPEECH*, pp. 2002–2006, 2015.

[65] J. Beltrán, E. Chávez, and J. Favela, "Scalable identification of mixed environmental sounds, recorded from heterogeneous sources," *Pattern Recognition Letters*, vol. 68, pp. 153–160, 2015.

[66] S. Chu, S. Narayanan, and C.-C. J.Kuo, "Environmental sound recognition with time frequency audio features," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 17, no. 6, pp. 1142–1158, 2009.

[67] X. Lu, P. Shen, Y. Tsao, C. Hori, and H. Kawai, "Sparse representation with temporal max-smoothing for acoustic event detection," in *Proc. INTERSPEECH*, pp. 1176–1180, 2015.

[68] H. Lim, M. J. Kim, and H. Kim, "Robust sound event classification using LBP-HOG based bag-of-audio-words feature representation," in *Proc. INTERSPEECH*, pp. 3325–3329, 2015.

[69] F. Metze, S. Rawat, and Y. Wang, "Improved audio features for large-scale multimedia event detection," in *Proc. ICME*, pp. 1–6, 2014.

[70] K. Ashraf, B. Elizalde, F. Iandola, M. Moskewicz, J. Bernd, G. Friedland, and K. Keutzer, "Audio-based multimedia event detection with DNNs and Sparse Sampling Categories and Subject Descriptors," in *Proc. ICMR*, pp. 611–614, 2015.

[71] M. Espi, M. Fujimoto, K. Kinoshita, and T. Nakatani, "Exploiting spectro-temporal locality in deep learning based acoustic event detection," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2015, no. 26, pp. 1–12, 2015.

[72] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient based learning applied to document recognition," in *Proc. of the IEEE*, vol. 86, pp. 2278–2324, 1998.

[73] F. M. Yun Wang, Leonardo Neves, "Audio-based multimedia event detection using deep recurent neural nework," in *Proc. ICASSP*, pp. 2742–2746, 2016.

[74] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. ICLR*, pp. 1–14, 2015.

[75] T. Sercu, C. Puhrsch, B. Kingsbury, and Y. LeCun, "Very deep multilingual convolutional neural networks for LVCSR," in *Proc. ICASSP*, pp. 4955–4959, 2016.

[76] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Action recognition by dense trajectories," in *Computer Vision and Pattern Recognition (CVPR)*, pp. 3169–3176, IEEE, 2011.

[77] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *Proc. ICCV*, pp. 3551–3558, 2013.

[78] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proc. CVPR*, pp. 1725–1732, 2014.

[79] L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," in *Proc. CVPR*, pp. 4305–4314, 2015.

[80] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in Neural Information Processing Systems*, pp. 568–576, 2014.

[81] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," *arXiv preprint arXiv:1604.06573*, 2016.

[82] C. Feichtenhofer, A. Pinz, and R. P. Wildes, "Spatiotemporal residual networks for video action recognition," *arXiv preprint arXiv:1611.02155*, 2016.

[83] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning Spatiotemporal Features with 3D Convolutional Networks," in *Proc. ICCV*, pp. 4489–4497, 2014.

[84] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, pp. 248–255, 2009.

[85] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition.," in *ICML*, pp. 647–655, 2014.

[86] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-the-shelf: an astounding baseline for recognition," in *Proc. CVPR Workshops*, pp. 806–813, 2014.

[87] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *European Conference on Computer Vision (ECCV)*, pp. 346–361, Springer, 2014.

[88] Y. Gong, L. Wang, R. Guo, and S. Lazebnik, "Multi-scale orderless pooling of deep convolutional activation features," in *European Conference on Computer Vision (ECCV)*, pp. 392–407, Springer, 2014.

[89] M. Cimpoi, S. Maji, and A. Vedaldi, "Deep filter banks for texture recognition and segmentation," in *Proc. CVPR*, pp. 3828–3836, 2015.

[90] M. Gygli, Y. Song, and L. Cao, "Video2gif: Automatic generation of animated gifs from video," in *CVPR*, pp. 1001–1009, 2016.

[91] N. Takahashi, M. Gygli, B. Pfister, and L. Van Gool, "Deep Convolutional Neural Networks and Data Augmentation for Acoustic Event Detection," in *Proc. Interspeech*, pp. 2982–2986, 2016.

[92] N. Takahashi, M. Gygli, and L. Van Gool, "Aenet: Learning deep audio features for video analysis," *IEEE Trans. on Multimedia*, vol. 20, pp. 513–524, 2017.

[93] S. Ioffe and C. Szegedy, " Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, pp. 448–456, 2015.

[94] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. AISTATS*, pp. 315–323, 2011.

[95] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, pp. 770–778, 2016.

[96] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely connected convolutional networks," *arXiv preprint arXiv:1608.06993*, 2016.

[97] O. Abdel-hamid, L. Deng, and D. Yu, "Exploring Convolutional Neural Network Structures and Optimization Techniques for Speech Recognition," in *Interspeech*, pp. 3366–3370, 2013.

[98] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *Proc. BMVC*, pp. 87.1–87.12, 2016.

[99] A. Liutkus, F. Stöter, Z. Rafii, D. Kitamura, B. Rivet, N. Ito, N. Ono, and J. Fontecave, "The 2016 Signal separation evaluation campaign," in *Proc. LVA/ICA*, pp. 66–70, 2017.

[100] A. Liutkus, F.-R. Stöter, and N. Ito, "The 2018 signal separation evaluation campaign," in *Proc LVA/ICA*, pp. 293–305, 2018.

[101] E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 14, pp. 1462–1469, 2006.

[102] N. Takahashi and Y. Mitsufuji, "Multi-scale multi-band DenseNets for audio source separation," in *Proc. WASPAA*, pp. 261–265, 2017.

[103] R. M. Bittner, J. Salamon, M. Tierney, C. C. M. Mauch, , and J. P. Bello, ""MedleyDB: A multitrack dataset for annotation-intensive MIR research," in *Proc.ISMIR*, pp. 66–70, 2014.

[104] E. W. M. and C.-F. Chan, "Phase modeling and quantization for low-rate harmonic+noise coding," in *Proc. European Signal Processing Conference*, 2002.

[105] D.-S. Kim, "Perceptual phase quantization of speech," *IEEE Trans. on Speech and Audio Processing*, vol. 11, no. 4, pp. 355–364, 2003.

[106] R. J. McAuley and T. F. Quatieri, "Speech analysis/synthesis based on a sinusoidal representation," *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. 34, no. 4, pp. 744–754, 1986.

[107] A. V. Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," in *Proc. the 33rd International Conference on Machine Learning (ICML)* (M. F. Balcan and K. Q. Weinberger, eds.), vol. 48 of *Proc. Machine Learning Research*, (New York, New York, USA), pp. 1747–1756, PMLR, 20-22 Jun 2016.

[108] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[109] T. Afouras, J. S. Chung, and A. Zisserman, "The conversation: Deep audio-visual speech enhancement," in *Proc. Interspeech*, pp. 3244–3248, 2018.

[110] K. Kinoshita, L. Drude, M. Delcroix, and T. Nakatani, "Listening to each speaker one by one with recurrent selective hearing networks," in *Proc. ICASSP*, pp. 5064–5068, 2018.

[111] J. Shi, J. Xu, G. Liu, and B. Xu, "Listen, think and listen again: Capturing top-down auditory attention for speaker-independent speech separation," in *Proc. IJCAI*, pp. 4353–4360, 2018.

[112] J. L. Roux, S. Wisdom, H. Erdogan, and J. R. Hershey, "SDR - half-baked or well done?," in *Proc. ICASSP*, pp. 3244–3248, 2019.

[113] F.-R. Stoeter, S. Chakrabarty, B. Edler, and E. A. P. Habets, "Classification vs. regression in supervised learning for single channel speaker count estimation," in *Proc. ICASSP*, 2018.

[114] J. R. Hershey, Z. Chen, J. LeRoux, and S. Watanabe, "Deep clustering: Discriminative embeddings for segmentation and separation," in *Proc. ICASSP*, pp. 31–35, 2016.

[115] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, "Perceptual evaluation of speech quality (pesq)-a new method for speech quality assessment of telephone networks and codecs," in *Proc. ICASSP*, p. 749–752, 2001.

[116] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25* (F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds.), pp. 1097–1105, Curran Associates, Inc., 2012.

[117] T. Svendsen, "Pronunciation modeling for speech technology," in *International Conference on Signal Processing and Communica- tions (SPCOM)*, pp. 11–16, 2004.

[118] H. Mokbel and D. Jouvet, "Derivation of the optimal set of phonetic transcriptions for a word from its acoustic realizations," in *Speech Communication*, vol. 29, pp. 49–64, 1999.

[119] M. Gerber, T. Kaufmann, and B. Pfister, "Extended Viterbi algorithm for optimized word HMMs," in *ICASSP*, pp. 4932–4935, 2011.

[120] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition," *Signal Processing Magazine*, pp. 82–97, 2012.

[121] O. Abdel-Hamid, A. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid NN-HMM model for Speech Recognition," in *ICASSP*, pp. 4277–4280, 2012.

[122] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. on Communications*, vol. 28, no. 1, pp. 84–95, 1980.

[123] W. Fisher, G. Doddington, and K. Goudie-Marshall, "The DARPA speech recognition research database: Specifications and status," in *Proc. DARPA Workshop on Speech Recognition*, pp. 93–99, 1986.

[124] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in *Proc. ICASSP*, pp. 8609–8613, 2013.

[125] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, "The HTK Book (for HTK Version 3.4.1)." `http://htk.eng.cam.ac.uk`, 2009. University of Cambridge, UK.

[126] E. Battenberg, S. Dieleman, D. Nouri, E. Olson, C. Raffel, J. Schlüter, S. K. Sønderby, D. Maturana, M. Thoma, and et al., "Lasagne: First release.." `http://dx.doi.org/10.5281/zenodo.27878`, Aug. 2015.

[127] NIST, "2011 trecvid multimedia event detection evalua- tion plan." http://www.nist.gov/itl/iad/mig/upload/MED11-EvalPlan-V03-20110801a.pdf.

[128] P. Over, J. Fiscus, and G. Sanders, "Trecvid 2013 – an introduction to the goals, tasks, data, eval- uation mechanisms, and metrics." http://www-nlpir.nist.gov/projects/tv2013/.

[129] X. Zhuang, X. Zhou, M. a. Hasegawa-Johnson, and T. S. Huang, "Real-world acoustic event detection," *Pattern Recognition Letters*, vol. 31, no. 12, pp. 1543–1551, 2010.

[130] M. Xu, C. Xu, L. Duan, J. S. Jin, and S. Luo, "Audio keywords generation for sports video analysis," *ACM Trans. on Multimedia Computing, Communications, and Applications*, vol. 4, no. 2, pp. 1–23, 2008.

[131] S. Deng, J. Han, C. Zhang, T. Zheng, and G. Zheng, "Robust minimum statistics project coefficients feature for acoustic environment recognition," in *Proc. ICASSP*, pp. 8232–8236, 2014.

[132] N. Jaitly and G. E. Hinton, "Vocal tract length perturbation (VTLP) improves speech recognition," *ICML*, vol. 28, 2013.

[133] "freesound." http://www.freesound.org/.

[134] S. Nakamura, K. Hiyane, and F. Asano, "Acoustical sound database in real environments for sound scene understanding and hands-free speech recognition," in *International Conference on Language Resources & Evaluation*, pp. 2–5, 2000.

[135] J. Wu, Yinan Yu, Chang Huang, and Kai Yu, "Deep multiple instance learning for image classification and auto-annotation," in *Proc. CVPR*, pp. 3460–3469, 2015.

[136] P. Viola, J. C. Platt, and C. Zhang, "Multiple instance boosting for object detection," in *Proc. NIPS*, vol. 74, pp. 1769–1775, 2005.

[137] H. David, "A tractable inference algorithm for diagnosing multiple diseases.," in *Proc. UAI*, pp. 163–171, 1989.

[138] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv: 1207.0580*, 2012.

[139] O. Gencoglu, T. Virtanen, and H. Huttunen, "Recognition of acoustic events using deep neural networks," *Proc. EUSIPCO*, no. 1-5 Sept. 2014, pp. 506–510, 2014.

[140] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition," in *Proc. ICML*, pp. 647–655, 2014.

[141] D. Mostefa, N. Moreau, K. Choukri, G. Potamianos, S. M. Chu, A. Tyagi, J. R. Casas, J. Turmo, L. Cristoforetti, F. Tobia, A. Pnevmatikakis, V. Mylonakis, F. Talantzis, S. Burger, R. Stiefelhagen, K. Bernardin, and C. Rochet, "The CHIL audiovisual corpus for lecture and meeting analysis inside smart rooms," *Language Resources and Evaluation*, vol. 41, no. 3-4, p. 389–407, 2007.

[142] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A Dataset of 101 Human Action Classes From Videos in The Wild," in *CRCV-TR-12-01*, 2012.

Bibliography (written by Naoya Takahashi)

Journal

(1) N. Takahashi, M. Gygli, L. V. Gool, "AENet: Learning Deep Audio Features for Video Analysis", IEEE Transaction Multimedia, Vol.20, Issue 3, March 2018, pp. 513 - 524

International conference papers

(2) N. Takahashi, P. Sudarsanam, N.Goswami, Y. Mitsufuji, "Recursive speech separation for unknown number of speakers", Interspeech2019

(3) N. Takahashi, P.Agrawal, N.Goswami, Y. Mitsufuji, "PhaseNet: Discretized Phase Modeling with Deep Neural Networks for Audio Source Separation", Interspeech 2018, pp.2713-2717

(4) N. Takahashi, N.Goswami, Y. Mitsufuji, "MMDenseLSTM: an Efficient Combination of Convolutional and Recurrent Neural Networks for Audio Source Separation", IWAENC 2018, pp.106-110

(5) N. Takahashi, Y. Mitsufuji, "Multi-scale Multi-band DenseNets for Audio Source Separation", WASPAA 2017, pp21-25

(6) N. Takahashi, M. Gygli, B. Pfister, L. Van Gool, Deep Convolutional Neural Networks and Data Augmentation for Acoustic Event Recognition, Interspeech2016, pp.2982-2986

(7) N. Takahashi, T. Naghibi, B. Pfister, Automatic Pronunciation Generation by Utilizing a Semi-supervised Deep Neural Networks, Interspeech 2016, pp1141-1145