SINGLE CHANNEL AUDIO SEPARATION USING DEEP NEURAL NETWORKS AND MATRIX FACTORIZATIONS

Di Wu

BEng

A thesis submitted to the Newcastle University for the degree of Doctor of Philosophy

School of Electrical and Electronic Engineering

Faculty of Science, Agriculture and Engineering

November 2017

ABSTRACT

Source Separation has become a significant research topic in the signal processing community and the machine learning area. Due to numerous applications, such as automatic speech recognition and speech communication, separation of target speech from the mixed signal is of great importance. In many practical applications, speech separation from a single recorder is most desirable from an application standpoint. In this thesis, two novel approaches have been proposed to address this single channel audio separation problem. This thesis first reviews traditional approaches for single channel source separation, and later elicits a generic approach, which is more capable of feature learning, *i.e.* deep graphical models.

In the first part of this thesis, a novel approach based on matrix factorization and hierarchical model has been proposed. In this work, an artificial stereo mixture is formulated to provide extra information. In addition, a hybrid framework that combines the generalized Expectation-Maximization algorithm with a multiplicative update rule is proposed to optimize the parameters of a matrix factorization based approach to approximatively separate the mixture. Furthermore, a hierarchical model based on an extreme learning machine is developed to check the validity of the approximately separated sources followed by an energy minimization method to further improve the quality of the separated sources by generating a time-frequency mask. Various experiments have been conducted and the obtained results have shown that the proposed approach outperforms conventional approaches not only in reduction of computational complexity, but also the separation performance.

In the second part, a deep neural network based ensemble system is proposed. In this work, the complementary property of different features are fully explored by ‘wide’ and ‘forward’ ensemble system. In addition, instead of using the features learned from the output layer, the features learned from the penultimate layer are investigated. The final embedded features are classified with an extreme learning machine to generate a binary mask to separate a mixed signal. The experiment focuses on speech in the presence of music and the obtained results demonstrated that the proposed ensemble system has the ability to explore the complementary property of various features thoroughly under various conditions with promising separation performance.

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my supervisors Dr Wai Lok Woo and Professor Satnam Dlay for their dedications towards work, professionalism, and allowing me to freely pursue and explore research topics that I have interests in. I am very appreciative of my supervisors who give me countless support, guidance, encouragement, and trust over the years. Without them, this dissertation would not have been possible.

I would like to thank my thesis examination committee members for their time, patience, criticism and feedback.

I owe my heartfelt appreciation to my father, Guihai Wu and my mother Lanqin Feng for their unconditional love and persistent support over the years. I have been truly blessed to have them as my parents. I would like to specially thank my wife, Ying Pi, who shares my frustration, excitement, sadness and happiness along the journey. It's her who supports me all the time and makes me a better person.

LIST OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS/ACRONYMS

AMS	Amplitude Modulation Spectrum
AR	Auto-Regressive
CASA	Computational Auditory Scene Analysis
DBN	Deep Belief Network
DEE	DNN Ensemble Embedding
DES	DNN Ensemble Stacking
DNMF	Discriminative Nonnegative Matrix Factorization
DNN	Deep Neural Network
DSELM	Deep Sparse Extreme Learning Machine
EEG	Electroencephalography
ELM	Extreme Learning Machine
EM	Expectation-Maximization
FFT	Fast Fourier Transform
GBRBM	Gaussian-Bernoulli Restricted Boltzmann Machine
GEM	Generalized Expectation-Maximization
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
IBM	Ideal Binary Mask
IRM	Ideal Ratio Mask
IS	Itakura-Saito
IS-NMF	Itakura-Saito Nonnegative Matrix Factorization
KL	Kullback-Leibler
LC	Local Criterion
LS	Least Square
MAP	Maximum a Posterior
MFCC	Mel-Frequency Cepstral Coefficients

MGD	Multiplicative Gradient Decent
ML	Machine Learning
MLP	Multilayer Perceptron
MSE	Multi-view Spectral Embedding
MU	Multiplicative Update
MRF	Markov Random Field
NMF	Nonnegative Matrix Factorization
NMF-2D	Nonnegative Two-Dimensional Matrix Factorization
PESQ	Perceptual Evaluation of Speech Quality
RASTA-PLP	Relative Spectral Transform and Perceptual Linear Prediction
RBM	Restricted Boltzmann Machine
RL	Reinforcement Learning
SAR	Signal-to-Artifacts Ratio
SCSS	Single Channel Source Separation
SDR	Signal-to-Distortion Ratio
SLFNs	Single Layer Feedforward Neural Networks
SELM	Sparse Extreme Learning Machine
SIR	Signal-to-Interference Ratio
SNR	Signal-to-Noise Ratio
SS	Source Separation
STFT	Short Time Fourier Transform
STOI	Short-Time Objective Intelligibility
SVM	Support Vector Machine
TF	Time-Frequency
WDO	W-Disjoint Orthogonality

LIST OF PUBLICATIONS

- D. Wu, W. L. Woo, S. S. Dlay, "Single Channel Audio Separation using Deep Sparse Extreme Learning Machine with EM-based Nonnegative Matrix Factorization", *Revision, IEEE Transactions on Audio, Speech and Language Processing*.

- D. Wu, W. L. Woo, S. S. Dlay, "Deep Neural Networks Ensemble System for Single-Channel Audio Separation", *Revision*, *IEEE Transactions on Audio, Speech and Language Processing*.

- D. Wu, W. L. Woo, S. S. Dlay, "NMF-2D based Source Separation using Extreme Learning Machine", *2nd IET International conference on Intelligent Signal Processing 2015 (ISP)*, Dec. 2015

- D. Wu, W. L. Woo, S. S. Dlay, "Single Channel Audio Separation using Matrix Factorization with assistance of Deep Neural Networks", *Accepted for publication in 17th IEEE International Conference of Communication Technology*

- D. Wu, W. L. Woo, S. S. Dlay, "Ensemble Deep Neural Networks with Transfer Learning for Single Channel Source Separation", *Accepted for publication in 2nd International Conference on Signal and Image Processing (CCISP)*

Chapter 1. Introduction of thesis

1.1 The background of source separation

Source separation which aims at separating different sources from a mixed signal, has been one of the most challenging problems and has received considerable attention from both the signal processing community and the machine learning area in recent years. Many approaches have been proposed to address the separation problem during the past several decades. These approaches can be classified as Multi Channel Source Separation (MCSS) if multiple instances or recordings of the mixture signal are given or Single Channel Source Separation (SCSS) if only a single recording is available. SCSS is perhaps the most challenging problem as the underlying problem is severely under-determined. Hence, the prior knowledge of sources may be required to solve this problem. Depending on the requirement of training information, the approaches to solve the separation problem can be classified as supervised or unsupervised source separation. When the training information of the sources is provided, the separation is classified as supervised source separation. Otherwise, the separation is classified as unsupervised source separation.

During the last decade, tremendous developments have been achieved in the area of SCSS. The techniques used in SCSS have been successfully applied in various fields such as hearing aids, automatic speech recognition, advanced statistical modeling, data mining, pattern recognition, communication system, intelligence system, geophysics, econometrics and neural network.

1.2 SCSS problem formulation

Generally, the SCSS problem can be mathematically expressed by the following equation

$$x_{mix}(t) = \sum_{j=1}^{J} x_j(t) \tag{1.1}$$

where $x_{mix}(t)$ denotes the mixed signal consisting J sources (e.g. music, speech) and $x_j(t)$ represents the j^{th} source in the time domain, t is the time index. The goal of SCSS is to recover the original sources from the mixture. It is worth mentioning that $Eq.(1.1)$ denotes a simplified model, in which some factors such as non-linear distortion, propagation delay are not taken into account. However, the above model is sufficient to represent most realistic applications and cases.

1.3 Category of Source Separation

According to the review of current literature, the source separation problem is sorted into three categories: Linear and Nonlinear; Instantaneous and Convolutive; Overcomplete and Underdetermined. The majority of researchers are working in the field of linear source separation due to its simplicity of analysis. The linear source separation is represented by a linear combination of each source, as defined in $Eq.(1.1)$. For the nonlinear source separation, the nonlinear distorted signals are taken into consideration. Compared with existing linear algorithms, the nonlinear algorithms is able to offer a more accurate representation of a realistic environment, but more adaption, computation and processing are required. In the second category, the source separation is consider a convolutive system if the observed signals consist of combinations of multiple time-delayed versions of original sources. Otherwise, the source separation is defined as instantaneous system. In the last category, the source separation is classified into the overcomplete system where the number of source signals is smaller than the number of observed signals and underdetermined system where the number of source signals is greater than the number of observed signals.

1.4 Applications of Source Separation

Source separation has received much attention in both the industry area as well as the academic field because of its substantial importance and various applications. During the last decade, thousands of applications have been developed, particularly in communication system, medical signal processing, automatic speech recognition, and hearing aid design [1-7]. Speech is crucial for human communication. However, just like the "cocktail party" problem, where a number of people are talking simultaneously in a noisy room, speech is often corrupted by various acoustic interferences, such as multiple speech, music instrument and singing, causing issues for both human and machine listeners [8]. The goal of "cocktail party" is to recover the original component signals from the mixture in a practical sense.

In the communication system, the antenna array in combination with source separation techniques can be used to separate multiple overlapping tag signals [9]. In the field of medical signal processing, the neurophysiological information such as electroencephalography (EEG) signal is required to be separated from the mixture of artefacts and noise for better analysis. The source separation method is useful to track and isolate the target signal, leading to a more accurate analysis [10]. Source separation has also been applied in the field of finance area. The factors that affect the financial stance of a currency or stock market can be isolated and analyzed to predict the future trends [11]. Source separation has been applied in the chemometrics to

determine the spectra in an unresolved mixture [12]. The automatic speech processing system, such as Apple Siri and Amazon Echo, have produced usable performance over the past few years. The Bose noise cancellation earphones are able to isolate the sound from the environment noise.

Seen from the industrial point of view, source separation is one of the most important cases, especially only one recording is available. Lots of industrial practical cases where source separation are needed, for example

1. The source separation technique is able to isolate the target speech while answering an important conversation in a crowded area.
2. The performance of Automatic Speech Recognition relies on the quality of the speech. When dealing with speech in the presence of acoustic interferences, the source separation techniques are strongly needed as the recognition performance could be improved dramatically if the speech can be separated from the mixture.
3. In speech transcription, it is helpful if the desired speech source can be isolated from the mixture.
4. Acoustic interferences create troubles for people with hearing loss. With source separation technique, the target speech can be isolated, enhanced and played for the listeners.

Since only one mixed signal is observed, the solution to the problem is very limited. To obtain good separation performance, generally extra information such as prior information of the desired source is required to help distinguish sources from the mixed signal.

The source separation problem often solved through two domains: the Time Domain (TD) and the Time Frequency domain (TF). In general, the time domain mixture is represented as $Eq.(1.1)$. Although very fast computations, due to the lack of sparseness and the large quantities of the data, very limited approaches have been developed in time domain.

The corresponding TF domain expression of $Eq.(1.1)$ is presented as

$$X_{mix}(t,f) = \sum_{j=1}^{J} X_j(t,f) \tag{1.2}$$

where $X_{mix}(t,f)$, $X_j(t,f)$ denote the TF components of mixture and j-th source, respectively. The time slots are given by $t = 1, \dots, T$ while frequencies are given by $f = 1, \dots, F$. T and F denote the total time slots and frequency in the TF domain, respectively. The most commonly used TF representation is obtained by applying Short-Time Fourier Transform (STFT). This

representation is regarded as well balanced between sparse representation and computation complexity of the sources [13].

1.5 Contribution

This thesis contributes two novel approaches, namely multi-layered Extreme Learning Machine (ELM) with Expectation-Maximization (EM) based Non-Negative Matrix (NMF) factorization; and Deep Neural Networks (DNN) ensemble system, for the single-channel source separation problem.

The contribution of this thesis are listed as follows

1. A unified perspective of the widely used existing SCSS methods, especially ML approaches is introduced. The theoretical aspects of SCSS are presented to provide sufficient background knowledge relevant to the thesis.
2. A novel approach that combines an artificial stereo mixture with generalized EM based NMF followed by multi-layered ELM refining is developed.
3. A novel approach that based on an ensemble learning concept using DNN is proposed to generate more discriminative and robust representations that can be used to estimate TF mask to separate mixed signal.
4. The separation performance of the proposed approaches are evaluated and compared with the existing state-of-the-art SCSS approaches.

1.6 Thesis Outline

This thesis focuses on the supervised source separation. Two novel approaches are proposed. The theoretical part of each approach is described in details. All the experiments are conducted on Matlab. The experimental results are evaluated and compared with other existing approaches. The comparison results demonstrate that the proposed approaches achieved better performance.

The thesis outline is as follows

In Chapter 2, a comprehensive review of recent SCSS method is introduced. This chapter starts of introducing the model based method, including Hidden Markov Model (HMM) and NMF. Computational Auditory Scene Analysis (CASA) and supervised separation approaches including DNN and ELM are sequentially presented.

In Chapter 3, the neural networks for deep architectures are presented. The main mathematical concepts of Energy-Based Model (EBM) and the building block of DNN named Restricted Boltzmann Machine (RBM) are described in details.

In Chapter 4, a novel approach that combines NMF and multi-layered ELM is introduced. Furthermore, a hybrid framework that combine the generalized EM algorithm with the multiplicative update rule is proposed to optimize the parameters of the NMF. In addition, artificial stereo channel termed as pseudo stereo mixture is adapted to increase the dimensionality of the mixing channel matrix to reduce the ambiguity between estimating the mixing coefficients and the source signals. Furthermore, a joint energy minimization method based on the trained multi-layered ELM is developed to improve the quality of the coarsely separated signals previously obtained from the NMF.

In Chapter 5, a novel approach based on the ensemble learning concept using DNNs is proposed to estimate an Ideal Binary Mask (IBM) to separate the mixed signal. In this approach, the complementary property of the raw acoustic features are fully explored by the ensemble learning. Furthermore, instead of using the representations learned from the output layer of each component, the representations learned from the penultimate layers are investigated. The learned representations are fused together to form a new feature vector which will be classified using the ELM classifier to generate an IBM to separate the mixed signal.

The conclusion to the thesis is presented in Chapter 6, along with the future research and the latest approaches in speech separation.

Chapter 2. Literature review of Single Channel Source Separation

In this chapter, the review of the existing Single Channel Source Separation (SCSS) approaches is described in details. We first review the traditional separation approaches including the model based source separation and the Computational Auditory Scene Analysis (CASA) followed by reviewing the supervised source separation.

2.1 Model based source separation

Model based source separation builds Machine Learning (ML) models for speech and interference. The models are trained as a prior knowledge by using some or the entire information of the mixed signal. Different mixture representations can be formed by combining different patterns learnt from different sources via training.

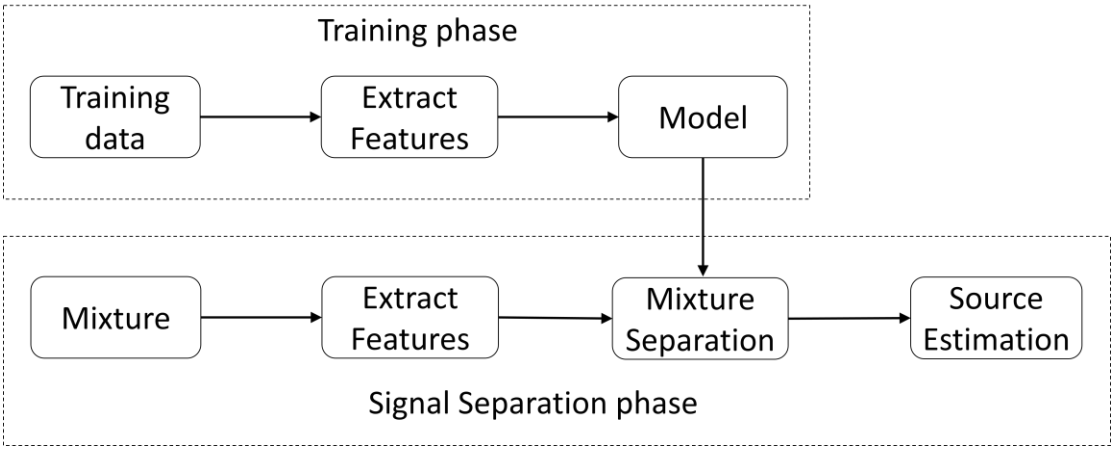


Fig. 2.1 A general framework for Model-based SCSS

Fig. 2.1 illustrates a general framework of the model based source separation. In the training phase, the source models are either constructed directly from the knowledge of the sources, or by learning from the training data. (e. g. Gaussian Mixture Model (GMM)). In the separation phase, the mixed signal is transformed into an appropriate representation domain followed by performing the separation. The models and the data are combined to yield estimates of the sources, either directly or through a signal reconstruction step.

There are two widely used model based approaches named GMM and Hidden Markov Model (HMM). The work in [14] investigated the phone-level dynamics using HMMs to impose temporal constraints on speech signals for separation. A factorial HMM adopted in [15] was used to model a speaker, and then the estimated sources were used to generate a binary mask to separate the mixed signal. A feature modeling approach combining GMM and factorial HMM was presented in [16]. HMM-GMM user-generated exemplar source to solve the single

channel speech separation was proposed in [17]. The details of GMM and HMM are described in Section 2.1.1 and 2.1.2.

2.1.1 *Gaussian Mixture Model*

A GMM is commonly used as a parametric model where the parameters are estimated from training data using the iterative Expectation-Maximization (EM) algorithm or the Maximum a Posteriori (MAP) estimation [18]. The GMM based approaches offer the advantages of being sufficiently general and applicable to a wide variety of audio signals. GMM based method have been successfully used to separate speech within a single channel [19] and some particular musical instruments [20]. For SCSS, GMM is trained on a training set that contains samples which should have characteristics similar to those of the sources to be separated (*i.e.* speech, music, drum, etc.). The trained GMM can be used to represent each source in the mixed signal by a set of characteristic spectral patterns.

Technically, a Gaussian mixture model can be expressed as

$$p(\boldsymbol{x}|\lambda) = \sum_{i=1}^{I} \boldsymbol{w}_i \, g(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i^2) \tag{2.1}$$

where $\boldsymbol{x} \in \Re^D$ is a D-dimensional continuous data vector (*i.e.* features), $\lambda = \{\boldsymbol{w}_i, \ \boldsymbol{\mu}_i, \ \boldsymbol{\Sigma}_i^2\}, i = 1, \dots, I$ denotes the parameters of model, $\boldsymbol{w}_i$ denotes the mixture weights, $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i^2$ denotes the mean vector, and its covariance matrix, respectively. The term $g(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i^2)$ is the Gaussian density function, which can be expressed as

$$g(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i^2) = \frac{1}{(2\pi)^{D/2}\left|\boldsymbol{\Sigma}_i^2\right|^{1/2}} exp\left\{-\frac{1}{2\boldsymbol{\Sigma}_i^2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^T(\boldsymbol{x}-\boldsymbol{\mu}_i)\right\} \tag{2.2}$$

The matrix weights satisfy the constraint that $\sum_{i=1}^{I} \boldsymbol{w}_i = 1$.

Given the training data, the goal of GMM is to estimate the parameter λ that best matches the distribution of the training feature vector. The most popular method to update the model parameters is the maximum likelihood algorithm

$$p(\boldsymbol{X}|\lambda) = \prod_{k=1}^{K} p(\boldsymbol{x}_k|\lambda) \tag{2.3}$$

where $\boldsymbol{X} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_K\}$ is a sequence of K training vectors. The parameters are updated by using the iterative EM algorithm. Begin with the initial model λ, a new model $\bar{\lambda}$ which satisfy the condition $p(\boldsymbol{x}|\bar{\lambda}) \geq p(\boldsymbol{x}|\lambda)$ is estimated. Then the new model is regarded as the initial model for the next iteration. The procedure is repeated until a convergence threshold is achieved. On each EM iteration, to guarantee a monotonic increases in the model's likelihood value, the following re-estimation formulas are utilized

$$\bar{\boldsymbol{w}}_i = \frac{1}{K}\sum_{k=1}^{K} P(i|\boldsymbol{x}_k, \lambda) \tag{2.4}$$

$$\bar{\boldsymbol{\mu}}_i = \frac{\sum_{k=1}^{K} P(i|\boldsymbol{x}_k, \lambda)\boldsymbol{x}_k}{\sum_{k=1}^{K} P(i|\boldsymbol{x}_k, \lambda)} \tag{2.5}$$

$$\bar{\boldsymbol{\Sigma}}_i^{\,2} = \frac{\sum_{k=1}^{K} P(i|\boldsymbol{x}_k, \lambda)\boldsymbol{x}_k^2}{\sum_{k=1}^{K} P(i|\boldsymbol{x}_k, \lambda)} - \bar{\boldsymbol{\mu}}_i^{\,2} \tag{2.6}$$

The posterior probability for component i is given by

$$P(i|\boldsymbol{x}_k, \lambda) = \frac{\boldsymbol{w}_i g(\boldsymbol{x}_k|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i^2)}{\sum_{j=1}^{M} \boldsymbol{w}_k g(\boldsymbol{x}_k|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k^2)} \tag{2.7}$$

Although the GMM based approach has the ability to separate the mixed signal successfully, this approach suffers from two big issues [21]. One issue is the number of Gaussians in the observation density. It grows exponentially with the increment of the number of sources, which often leads to an intractable problem. The other issue is the computational complexity. For instance, in [15], to describe a speaker's speech, GMMs with 8000 states were required. This is because the model attempts to capture every possible instance of the signal.

2.1.2 *Hidden Markov Model*

HMM is a tool for representing probability distributions over sequences of observations [22]. It can be considered as a generalization of a mixture model where the hidden variables are related through Markov model rather than independent of each other. HMM has been successfully used to tackle the single channel source separation problem. In [19], Roweis discussed the use of a factorial HMM with a GMM observation model. In their approach, a trained HMM/GMM model is utilized to separate the mixed sources. Promising results have been achieved on a single channel speech separation problem using an extended HMM/GMM approach in a log power spectral representation [23]. In this approach, the element-wise observation model is discussed and an approximation based on Laplace's method is adopted. HMM with GMM emissions is proposed in [24] for modelling the log power spectrum of each source. In addition, eigenvoice adaptation is included across all states to account for any mismatch between signal level in the training and testing data.

Define a state alphabet set $S = \{s_1, s_2, \dots, s_N\}$, an observation alphabet set $V = \{v_1, v_2, \dots, v_M\}$, a fixed state sequence $Q = \{q_1, q_2, \dots, q_T\}$ and a sequence of corresponding observations $O = \{o_1, \dots, o_T\}$. HMM parameters are initialized as $\lambda = (A, B, \pi)$, where $A = [a_{ij}], a_{ij} = p(q_t = s_j|q_{t-1} = s_i)$ is a transition array, storing the probability of state j following state i. $B = [b_i(k)], b_i(k) = p(x_t = v_k|q_t = s_i)$ is the observation array, storing the probability of

observation k being produced from the state j, independent of t. $\pi = [\pi_i], \pi_i = p(q_1 = s_i)$ is the initial probability array.

The objective of HMM is to calculate the probability of the observation sequence given a model

$$p(O|\lambda) = \sum_{Q} p(O|Q,\lambda)p(Q|\lambda) \tag{2.8}$$

The probability of the observations O for a specific state sequence Q is given as

$$p(O|Q,\lambda) = \prod_{t=1}^{T} p(o_t|q_t,\lambda) = b_{q1}(o_1) \times b_{q2}(o_2) \cdots b_{qT}(o_T) \tag{2.9}$$

and the probability of the state sequence is given as

$$p(Q|\lambda) = \pi_{q1} a_{q1q2} a_{q2q3} \cdots a_{qT-1qT} \tag{2.10}$$

Define a forward probability variable

$$\begin{aligned} \alpha_t(j) &= p\left(o_1 o_2 \cdots o_t, q_t = s_j \middle| \lambda\right) \\ &= \left[\sum\nolimits_{i=1}^{N} \alpha_{t-1}(j) a_{ij}\right] b_j(o_t) \end{aligned} \tag{2.11}$$

where $1 \le t \le T-1$ and $1 \le j \le N$. Therefore, $p(O|\lambda)$ can be solved by calculating forward part of forward-backward algorithm. The backward probability variable are defined as follows

$$\begin{aligned} \beta_t(i) &= p(o_{t+1} o_{t+2} \cdots o_T, q_t = s_i|\lambda) \\ &= \sum_{j=1}^{N} a_{ij} b_j(o_{t+1})\, \beta_{t+1}(j) \end{aligned} \tag{2.12}$$

where $t = T-1, T-2, \ldots, 1$ and $1 \le i \le N$. In addition, the posterior probability of HMM components can be expressed as

$$\gamma_t(i) = p(q_t = s_i|O,\lambda) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} \tag{2.13}$$

where $\gamma_t(i)$ is the probability of being in state s_i at time t, given by O and λ.

The model parameters $\lambda = (A, B, \pi)$ can be re-estimated by maximizing the probability of the observation sequence by the forward-backward algorithm. The goal of re-estimate HMM parameter λ is to calculate the optimal variances of each source. The new set of the model parameters are updated by a number of iterations until convergence.

For SCSS, HMM assumes that signals have Gaussian centered priors $S_i(t,f) \sim \mathcal{N}\left(0, diag\left((\sigma_{k_i}^2(f))\right)\right)$, with a diagonal covariance matrix $\sum_i = diag(\sigma_i^2(f))$, where $S_i(t,f)$ denotes the TF representation of i-th source and $\sigma_i^2(f)$ denotes the covariance matrix. In the training phase, HMM is used to model the structure of sources through the covariance matrix which is estimated using the GMM and is used to compute the initial observation probability of the HMM. In the separation phase, the mixture $S(t,f) = S_1(t,f) + S_2(t,f)$ is

centered as Gaussian distribution $S(t,f) \sim \mathcal{N}\left(0, diag\left(\sigma_{k_1}^2(f) + \sigma_{k_2}^2(f)\right)\right)$ with the covariance matrix $\sigma_{k_1}^2(f) + \sigma_{k_2}^2(f)$. The posterior probability of the mixture $\gamma_{k_1,k_2}(t) = p\big(q_1 = s_k, q_2 = s_k | S(t_1,f), \dots, S(t_N,f)\big)$is calculated using forward and backward algorithm as $Eq.(2.11)$ and $Eq.(2.12)$. At last, Wiener filters are established from the posterior probability and the covariance matrix as follows to estimate the source signals

$$\hat{S}_1(f,t) = \left[\sum_{k_1=1}^{N}\sum_{k_2=1}^{N} \frac{\sigma_{k_1}^2(f)}{\sigma_{k_1}^2(f) + \sigma_{k_2}^2(f)} \gamma_{k_1,k_2}(t)\right] S(t,f)$$
$$\hat{S}_2(f,t) = \left[\sum_{k_1=1}^{N}\sum_{k_2=1}^{N} \frac{\sigma_{k_2}^2(f)}{\sigma_{k_1}^2(f) + \sigma_{k_2}^2(f)} \gamma_{k_1,k_2}(t)\right] S(t,f) \quad (2.14)$$

The training procedure of source separation by generating Wiener filter using GMM and HMM is illustrated as in Fig. 2.2.

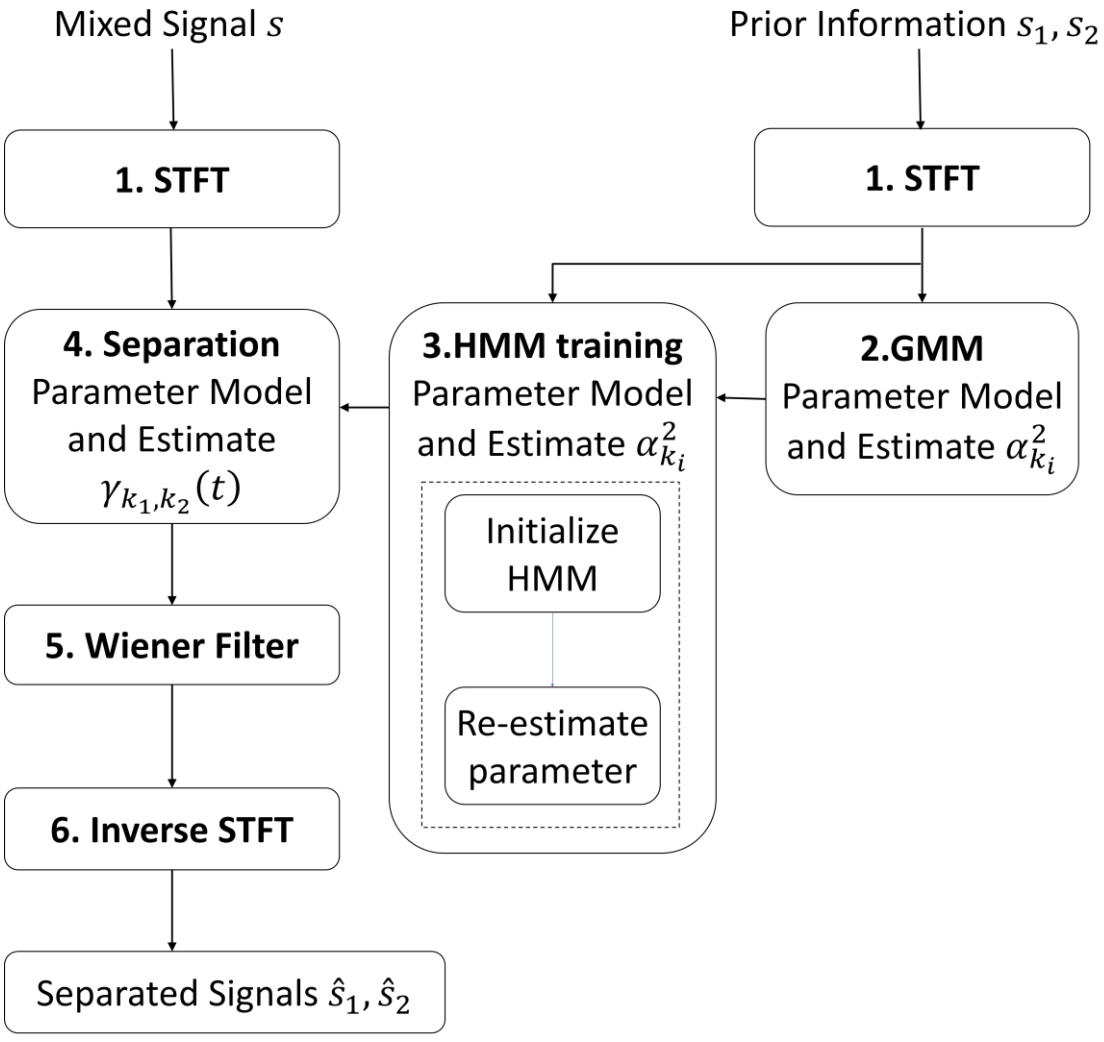


Fig. 2.2 Procedure of source separation by Wiener filter using GMM and HMM

Good results can be obtained using this method if a priori information is provided sufficiently in the training phase. However, to accurately model complex sources such as speech using HMM, a very large number of states may be needed. This will make HMM based method consume long time.

2.1.3 Matrix Factorization based models

Another approach within model based SCSS is matrix factorization model. One of the most popular techniques is Non-negative Matrix Factorization (NMF) in which a mixture **Y** is modeled as weighted sums of a non-negative basis matrix **D** and an encoding matrix **H** with non-negative elements [25-31]:

$$\mathbf{Y} \approx \mathbf{DH} \quad s.t. \mathbf{D}, \mathbf{H} \geq \mathbf{0} \tag{2.15}$$

where $\mathbf{Y} \in \mathbb{R}_{+}^{F \times T_s}$ is the TF representation of mixture $y(t)$, $\mathbf{D} \in \mathbb{R}_{+}^{F \times I}$ and $\mathbf{H} \in \mathbb{R}_{+}^{I \times T_s}$. In the expression, $\mathbf{D}, \mathbf{H} \geq \mathbf{0}$ means that all elements of **D** and **H** are non-negative and $\mathbb{R}_{+} = [0, \infty)$ denotes the non-negative real number. To reduce the data matrix **D** to its integral component such as **D** only containing spectral basis vectors and **H** only containing the temporal basis vectors, I is chosen to be $I < T_s$.

NMF has a multitude of applications in audio processing, including feature extraction, sound classification, and source separation. In [32], NMF is used to train a set of basis vectors for individual sources and to decompose the magnitude spectra of a mixture into a linear combination of the basis and the encoding matrix to generate a mask to separate a target source from the mixed signal. Many different generalizations and extensions to NMF have been proposed. An effective and discriminative approach for training NMF is proposed in [33-36]. The proposed Discriminative NMF (DNMF) is able to optimize all basis vectors jointly to reconstruct both clean signals and mixed signals. Source-filter model based NMF is another type of model based SCSS which is proposed in [37-39] for analyzing polyphonic audio signals. This approach model the spectral basis of a polyphonic signal as source-filter representation where the filter is characterized by a FIR filter.

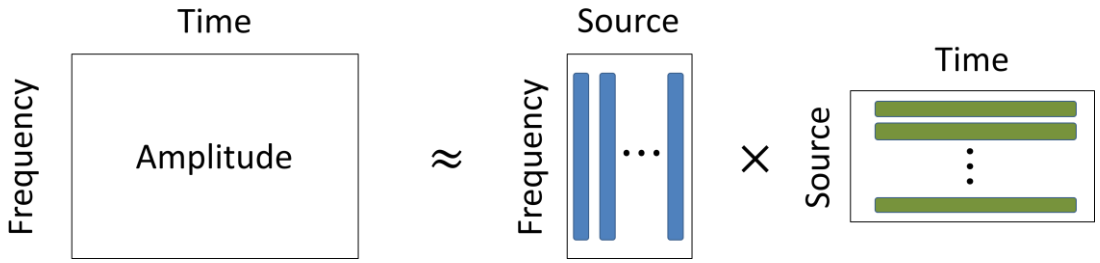


Fig. 2.3 Illustration of NMF decomposition of audio

For source separation, audio signals are computed by NMF in a TF representation as illustrated in Fig. 2.3.

Audio is transferred from the time domain to the TF domain using STFT. The magnitude TF representation is adopted to ensure the matrix is non-negative. The decomposition finds a set of time-varying sources with constant spectrum.

A wide range of cost functions have been proposed for NMF. There are two most commonly used cost functions expressed as follows, one is Least Square (LS) and the other is generalized Kullback-Leibler (KL) divergence

$$C_{LS}^{NMF} = \frac{1}{2}\sum_{f,t_s}\left(\mathbf{Y}_{f,t_s} - \mathbf{Z}_{f,t_s}\right)^2 \tag{2.16}$$

$$C_{KL}^{NMF} = \sum_{f,t_s}\left(\mathbf{Y}_{f,t_s}\log\frac{\mathbf{Y}_{f,t_s}}{\mathbf{Z}_{f,t_s}} - \mathbf{Y}_{f,t_s} + \mathbf{Z}_{f,t_s}\right) \tag{2.17}$$

where $\mathbf{Z}_{f,t_s} = \mathbf{DH}$. In above, C_{LS}^{NMF} corresponds to computing the maximum likelihood estimation of **D** and **H** under the assumption that the residual is additive independent and identically distributed (i.i.d.) Gaussian distributed. C_{KL}^{NMF} measures the relative entropy between the data and the approximate factorization. Lee and Seung [40] proposed an algorithm that minimize the chosen cost function by initializing the entries of **D** and **H** with random positive values, and then update those iteratively using multiplicative rules [41].

The update rule for the LS distance

$$\mathbf{D} \leftarrow \mathbf{D}\cdot\frac{\mathbf{YH^T}}{\mathbf{DHH^T}} \qquad \mathbf{H} \leftarrow \mathbf{H}\cdot\frac{\mathbf{D^TY}}{\mathbf{D^TDH}} \tag{2.18}$$

The update rule for the KL divergence:

$$\mathbf{D} \leftarrow \mathbf{D}\cdot\frac{\left(\frac{\mathbf{Y}}{\mathbf{DH}}\right)\mathbf{H^T}}{\mathbf{1H^T}} \qquad \mathbf{H} \leftarrow \mathbf{H}\cdot\frac{\left(\frac{\mathbf{Y}}{\mathbf{DH}}\right)\mathbf{D^T}}{\mathbf{1D^T}} \tag{2.19}$$

where '$\cdot$' denotes the element-wise multiplication, '$\mathbf{1}$' denotes an all-one matrix with dimension $F \times T_s$, and '$\frac{\boldsymbol{A}}{\boldsymbol{B}}$' denotes the element-wise division.

The NMF technique with a concatenated basis matrix is based on an assumption that the subspace of the separate sources span should be orthogonal to each other. However, the source subspaces often overlap, which makes the target source separation likely to fail.

The recently developed Nonnegative Matrix Factorization Two-Dimensional Deconvolution (NMF-2D) extends the NMF to a two-dimensional convolution of **D** and **H** *i.e.* $\mathbf{Y} \approx \mathbf{Z} = \sum_{\tau,\phi} \overset{\downarrow\phi}{\mathbf{D}^{\tau}}\overset{\rightarrow\tau}{\mathbf{H}^{\phi}}$ where the vertical arrow in $\downarrow\phi$ denotes downward shift of all elements in the matrix $\mathbf{D}^{\tau}$ by ϕ rows, and the horizontal arrow $\rightarrow\tau$ means right shift of all elements in the matrix $\mathbf{H}^{\phi}$ by τ columns [42, 43]. This can be interpreted as follow

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}, \quad \overset{\downarrow 2}{\mathbf{A}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 2 & 3 \end{bmatrix}, \quad \overset{\rightarrow 1}{\mathbf{A}} = \begin{bmatrix} 0 & 1 & 2 \\ 0 & 4 & 5 \\ 0 & 7 & 8 \end{bmatrix}.$$

The factorization decomposes the information matrix into two-dimensional convolution of factor matrices represent the spectral dictionary and temporal code as illustrated in Fig. 2.4

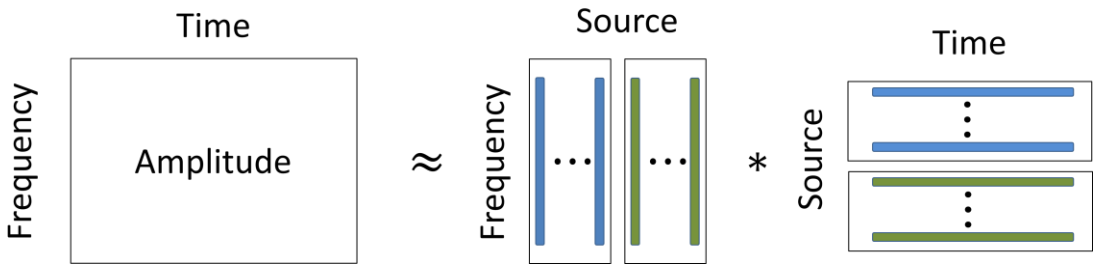


Fig. 2.4 Illustration of NMF-2D decomposition of audio

The advantage of NMF-2D is that each source can be represented by a single TF profile convolved in both time and frequency by a time-pitch weight matrix.

Similar to NMF, the two widely used cost functions are given as follows:

$$C_{LS}^{NMF2D} = \frac{1}{2}\sum_{f,t_s}\left(\mathbf{Y}_{f,t_s} - \mathbf{Z}_{f,t_s}\right)^2 \tag{2.20}$$

$$C_{KL}^{NMF2D} = \frac{1}{2}\sum_{f,t_s}\mathbf{Y}_{f,t_s}\, log\, \frac{\mathbf{Y}_{f,t_s}}{\mathbf{Z}_{f,t_s}} - \mathbf{Y}_{f,t_s} + \mathbf{Z}_{f,t_s} \tag{2.21}$$

for $\forall f \in F, \forall t_s \in T_s$. The goal is to minimize the chosen cost function by randomly initializing **D** and **H** with positive values, and then using the multiplicative rules to iteratively update the parameters until the algorithm is converges. Too see this, let's express each element in $\mathbf{Z}_{f,t_s}$ as

$$\mathbf{Z}_{f,t_s} = \sum_{\tau}\sum_{\phi}\sum_{k}\mathbf{D}_{f-\phi,k}^{\tau}\,\mathbf{H}_{k,t_s-\tau}^{\phi} \tag{2.22}$$

The derivative of a given element $\mathbf{Z}_{f,t_s}$ with respect to a given element $\mathbf{D}_{f',k'}^{\tau'}$ and $\mathbf{H}_{k',t_s'}^{\phi'}$ are given as

$$\begin{aligned} \frac{\partial \mathbf{Z}_{f,t_s}}{\partial \mathbf{D}_{f',k'}^{\tau'}} &= \frac{\partial \sum_{\tau,\phi,k}\mathbf{D}_{f-\phi,k}^{\tau}\,\mathbf{H}_{k,t_s-\tau}^{\phi}}{\partial \mathbf{D}_{f',k'}^{\tau'}} = \mathbf{H}_{k',t_s-\tau'}^{f-f'} \\ \frac{\partial \mathbf{Z}_{f,t_s}}{\partial \mathbf{H}_{k',t_s'}^{\phi'}} &= \frac{\partial \sum_{\tau,\phi,k}\mathbf{D}_{f-\phi,k}^{\tau}\,\mathbf{H}_{k,t_s-\tau}^{\phi}}{\partial \mathbf{H}_{k',t_s'}^{\phi'}} = \mathbf{D}_{f-\phi',k'}^{t_s-t_s'} \end{aligned} \tag{2.23}$$

For the LS cost function, minimizing the squared error corresponds to maximizing the likelihood of a Gaussian noise model. Differentiating C_{LS}^{NMF2D} with respect to $\mathbf{D}_{f',k'}^{\tau'}$ gives:

$$\frac{\partial C_{LS}^{NMF2D}}{\partial \mathbf{D}_{f',k'}^{\tau'}} = \frac{\partial}{\partial \mathbf{D}_{f',k'}^{\tau'}} \frac{1}{2} \sum_{f,t_s} \left(\mathbf{Y}_{f,t_s} - \mathbf{Z}_{f,t_s}\right)^2 = -\sum_{f,t_s} \left(\mathbf{Y}_{f,t_s} - \mathbf{Z}_{f,t_s}\right) \mathbf{H}_{k',t_s-\tau'}^{f-f'}$$

$$= -\sum_{\phi,t_s} \left(\mathbf{Y}_{f'+\phi,t_s} - \mathbf{Z}_{f'+\phi,t_s}\right) \mathbf{H}_{k',t_s-\tau'}^{\phi}, f = f' + \phi \tag{2.24}$$

Similarly for a given element $\mathbf{H}_{k',t_s'}^{\phi'}$

$$\frac{\partial C_{LS}^{NMF2D}}{\partial \mathbf{H}_{k',t_s'}^{\phi'}} = -\sum_{\tau,f} \left(\mathbf{Y}_{f,t_s'+\tau} - \mathbf{Z}_{f,t_s'+\tau}\right) \mathbf{D}_{f-\phi',k'}^{\tau}, t_s = t_s' + \tau \tag{2.25}$$

The recursive update steps for the gradient descent of a given element $\mathbf{D}_{f',k'}^{\tau'}$ are given as

$$\mathbf{D}_{f',k'}^{\tau'} \leftarrow \mathbf{D}_{f',k'}^{\tau'} - \epsilon \frac{\partial C_{LS}^{NMF2D}}{\partial \mathbf{D}_{f',k'}^{\tau'}} \tag{2.26}$$

The step size ϵ can be chosen as follows to cancel the first term of $Eq.(2.26)$ [40]

$$\epsilon = \frac{\mathbf{D}_{f',k'}^{\tau'}}{\sum_{\phi,t_s} \mathbf{Z}_{f'+\phi,t_s} \mathbf{H}_{k',t_s-\tau'}^{\phi}} \tag{2.27}$$

Therefore, the simple multiplicative rule can be obtained as

$$\mathbf{D}_{f',k'}^{\tau'} \leftarrow \mathbf{D}_{f',k'}^{\tau'} \frac{\sum_{\phi,t_s} \mathbf{Y}_{f'+\phi,t_s} \mathbf{H}_{k',t_s-\tau'}^{\phi}}{\sum_{\phi,t_s} \mathbf{Z}_{f'+\phi,t_s} \mathbf{H}_{k',t_s-\tau'}^{\phi}} \tag{2.28}$$

A similar step size can be found for a given element $\mathbf{H}_{k',t_s'}^{\phi'}$.

$$\mathbf{H}_{k',t_s'}^{\phi'} \leftarrow \mathbf{H}_{k',t_s'}^{\phi'} \frac{\sum_{\tau,f} \mathbf{D}_{f-\phi',k'}^{\tau} \mathbf{Y}_{f,t_s'+\tau}}{\sum_{\tau,f} \mathbf{D}_{f-\phi',k'}^{\tau} \mathbf{Z}_{f,t_s'+\tau}} \tag{2.29}$$

Consequently, the update function for LS cost function can be written in matrix notation as follows

$$\mathbf{D}_{LS}^{\tau} \leftarrow \mathbf{D}^{\tau} \cdot \frac{\sum_{\phi} \overset{\uparrow\phi}{\mathbf{Y}} \overset{\rightarrow\tau}{\mathbf{H}^{\phi}}^{\mathrm{T}}}{\sum_{\phi} \overset{\uparrow\phi}{\mathbf{Z}} \overset{\rightarrow\tau}{\mathbf{H}^{\phi}}^{\mathrm{T}}} \qquad \mathbf{H}_{LS}^{\phi} \leftarrow \mathbf{H}^{\phi} \cdot \frac{\sum_{\tau} \overset{\downarrow\phi}{\mathbf{D}^{\tau}}^{\mathrm{T}} \overset{\leftarrow\tau}{\mathbf{Y}}}{\sum_{\tau} \overset{\downarrow\phi}{\mathbf{D}^{\tau}}^{\mathrm{T}} \overset{\leftarrow\tau}{\mathbf{Z}}} \tag{2.30}$$

For the KL divergence cost function, the update function can be obtained as

$$\mathbf{D}_{KLd}^{\tau} \leftarrow \mathbf{D}^{\tau} \cdot \frac{\sum_{\phi} \overset{\uparrow\phi}{\left(\frac{\mathbf{Y}}{\mathbf{Z}}\right)} \overset{\rightarrow\tau}{\mathbf{H}^{\phi}}^{\mathrm{T}}}{\sum_{\phi} \mathbf{1} \cdot \overset{\rightarrow\tau}{\mathbf{H}^{\phi}}^{\mathrm{T}}} \qquad \mathbf{H}_{KLd}^{\phi} \leftarrow \mathbf{H}^{\phi} \cdot \frac{\sum_{\tau} \overset{\downarrow\phi}{\mathbf{D}^{\tau}}^{\mathrm{T}} \overset{\leftarrow\tau}{\left(\frac{\mathbf{Y}}{\mathbf{Z}}\right)}}{\sum_{\tau} \overset{\downarrow\phi}{\mathbf{D}^{\tau}}^{\mathrm{T}} \cdot \mathbf{1}} \tag{2.31}$$

Many subsequent works enhance the NMF-2D by incorporating more constraints and regulations [44], such as sparsity constraints [25] and alternative cost functions. For instance, a Sparse NMF-2D, which uses a double convolution to model both the spreading of spectral basis and the variation of temporal structure is developed in [45]. The experiments have been

illustrated that Sparse NMF-2D have achieved good results in separating single channel mixed signals [25]. Compared with the standard NMF, the capability of NMF-2D is dramatically increased due to the decrement of the number of components that needed to model each source. However, the obtained results still not fully achieve the required performance. This is because SCSS is a highly underdetermined problem where only a single recording is available to estimate more than one source signals. Hence, given only the mixed signal, potentially innumerable number of solutions exists. Therefore, extra information may be needed to solve this problem.

2.2 CASA based source separation

CASA aims to mimic the human auditory system to build a machine hearing system based on the Auditory Scene Analysis (ASA) principles. ASA attempts to explain the remarkable capability of the human auditory system that has the ability to segregate an acoustic signal into perceptual streams that correspond to different sources [46]. This approach has become a significant research topic in the speech and signal processing community and the machine learning area. The general procedure of CASA is presented in Fig. 2.5.

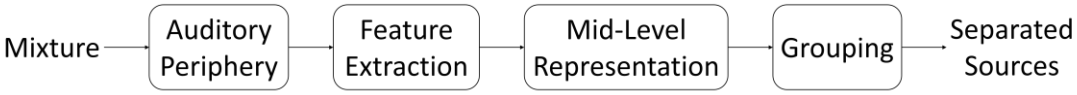


Fig. 2.5 General procedure of CASA

The first stage of CASA is to create a TF representation of the input audio mixture. A gammatone filterbank derived from psychophysical observations of the auditory periphery is a typical model of cochlear filtering. Secondly, the features such as pitch, onset, offset and amplitude modulation are extracted. In the next stage, the mid-level representations such as voiced and unvoiced representation are formed based on the extracted features. Next, each TF unit is classified into the corresponding group to construct a mask, which can be binary or real-valued. The original signals can be estimated by masking the TF plane of the mixture.

The primary computational goal of CASA is to estimate an ideal TF mask for speech separation [47]. The motivation is mainly because the masking phenomenon in hearing: a weak sound tends to be masked by a stronger sound in a critical band, which leads to the weak sound inaudible [48]. Each element of TF mask is a multiplicative weight applying to the corresponding TF unit of the signal. Typically, TF mask is defined based on the local signal-to-noise ratio (SNR) of the mixture. A low weight means high attenuation of the unit while a high weight denotes low attenuation. There are two important TF masks, one is the ideal ratio mask (IRM) and the other is the ideal Binary mask (IBM). The main difference between these

two masks is the weight. In the case of the former one, the weights are continues and derived as the speech energy divided by the energy of the mixed signal. For the IBM, by comparing the energy of the speech and the energy of interfering, the weights of the IBM takes 0 or 1. The weight of TF unit is set to 1 if the speech energy is greater than the interfering energy by a threshold; otherwise the TF unit is set to 0. Although the simplicity of IBM, it has achieved promising achievement in the speech intelligibility [49-51].

2.3 Supervised source separation

By considering the computational goal of CASA, a new separation paradigm, namely supervised source separation, has grown out. Taking a pre-mixed signal as input, supervised source separation aims to generate a TF mask to separate each source in the mixture. Different from the traditional separation approach, the supervised separation becomes a data-driven problem. Many approaches have been proposed to tackle this problem. Multi-layer neural network is one of the most popular instances to address this problem. Multi-layer neural network is a feed-forward neural network, which maps the input data onto the appropriate output. However, the training of the deep architecture is more difficult than that of the shallow architecture [52, 53]. Many experimental results suggest that the training of multi-layer neural network (starting from random initialization) gets stuck in local minima, and that as the architecture gets deeper, it becomes more difficult to obtain good generalization [53]. However, G. E. Hinton and R. Salakhutdinov [54, 55] has discovered that much better results can be acquired when the deep architecture is trained layer-by-layer with an unsupervised learning algorithm. This training approach is known as Deep Learning (DL) or Deep Neural Network (DNN). One of the most widely used unsupervised pre-learning models is Restricted Boltzmann Machine (RBM). The details of RBM and DNN are presented in Chapter 3. DNN can be used in many areas such as pattern recognition, speech recognition, financial data analysis and so on [56, 57]. In this thesis, we focus on the signal processing, especially the source separation.

The first DNN based supervised source separation is proposed by Wang and Wang [58]. In [58], DNN is utilized to extract more separable and discriminative features from the raw acoustic features. The target speech is recovered by applying the IBM which is estimated by classifying the extracted features into the target domain and the interfering domain. Their proposed system is trained and tested on a variety of acoustic conditions and the results outperform the traditional speech enhancement algorithms. In [59], DNN is employed to capture the nonlinear features. Instead of generating IBM, a smoothed IRM is estimated in the Mel frequency domain using the DNN and a set of TF unit level features [60]. In [61], a regression based speech enhancement framework using DNN is presented. DNN is used to estimate the nonlinear mapping from the

observed noisy speech to the desired clean signals. In [62], the DNN-based speech enhancement system is improved by introducing several techniques, including global variance equalization. The dropout and noise-aware training strategies are also developed to further improve the generalization capability of the DNN, especially for the unseen noise conditions.

A joint optimization of DNN and Recurrent Neural Network with an extra masking layer is proposed in [63] to address the two-talker separation problem. Furthermore, the discriminative training criterion is explored to further enhance the separation performance. In [64], Huang *et al*. explore deep RNN with different temporal connections for singing-voice separation from monaural recordings in a supervised setting. Their model achieves significant improvement compared to the previous methods. Tu *et al* [65] use DNN to generate the speech features of both the target speaker and the interferer for the two-talker separation problem. The nonlinear relationship between the speech features of the mixed signals is estimated directly using the DNN.

Supervised source separation is not limited to SCSS only. Nugraha *et al* [66] propose to use DNN to address the multichannel audio source separation problem. The framework is applied to model the source spectra, which is then combined with the classical multichannel Gaussian model to explore the spatial information. Jiang *et al*. [67] employ DNN to address binaural reverberant audio separation problem. The binaural features combined with the gammatone frequency cepstral coefficients are treated as the main auditory features to train the DNN to predict an IBM. Their proposed approach shows well generalization ability to the unseen spatial configurations and the reverberant conditions.

Neural network can also be used to classify the TF unit of the mixed signals to estimate a mask to separate the mixtures. In [68], the mixed signal is separated by a TF mask, which is estimated by a neural network classifier trained with a novel approach named Extreme Learning Machine (ELM) [69]. ELM is a simple and efficient learning algorithm of Single Layer Feedforward Neural Networks (SLFNs) [70, 71]. Unlike the other traditional learning algorithms such as back-propagation based multi-layer neural network, or support vector machine (SVM), the parameters of the hidden layers of ELM are initialized randomly and need not to be fine-tuned. Theoretically, Huang *et al* [72-74] have proved that the universal approximation capability can be maintained by randomly initializing the hidden neurons of SLFNs and calculating the output weights by regularized least square [75]. Due to its unique characteristics such as extremely fast training, good generalization, and universal approximation and classification capability, ELM has obtained promising results in various applications, such as face classification and gesture recognition. The deep architecture of multi-layer ELM is proposed by Kasun *et al* [71]. In their paper, an ELM autoencoder with ℓ_2 penalty is developed to construct the deep

architecture. In [76], a deep ELM is proposed by combining with multi-layer ELM and ELM with kernel. Their model applied to tackle the EEG classification problem and the experimental results demonstrate that deep ELM has the advantage of the least training time and good efficiency.

2.4 Summary

In this chapter, various approaches for SCSS have been reviewed. The approaches can be classified as Model-based SCSS, CASA and supervised source separation. The Model-based approach delivers relatively good separation performance, however, the training process requires various criterion for producing a good model. Therefore, this causes high computational complexity. The NMF-2D is a ground breaking for musical mixtures, but the obtained results still not fully achieve the required performance. To tackle the SCSS problem, extra information may be needed. Furthermore, Model based approach such as NMF-2D is not complex enough to describe audio signal in details. Therefore, more powerful machine learning approaches such as ELM or DNN are needed. Supervised source separation, especially DNN based separation, has achieved promising results across a range of tasks, such as pattern recognition and nature language processing. DNN is a suitable candidate for supervised source separation duo to its abilities like excellent performance and good scalability. A key element to supervised separation is generalization, however, current proposed DNN approaches may not fulfil this element as the extracted features of single DNN may not discriminative and robustness enough. In this thesis, two novel SCSS approaches have been developed to tackle the problems mentioned above. The design of each approach is described in the Chapter 4 and Chapter 5.

Chapter 3. Neural Networks for Deep Architecture

3.1 Multi-Layer Neural Network

In this chapter, the neural networks for deep architecture are presented in details. A typical diagram of the multi-layer neural network is illustrated in Fig. 3.1.

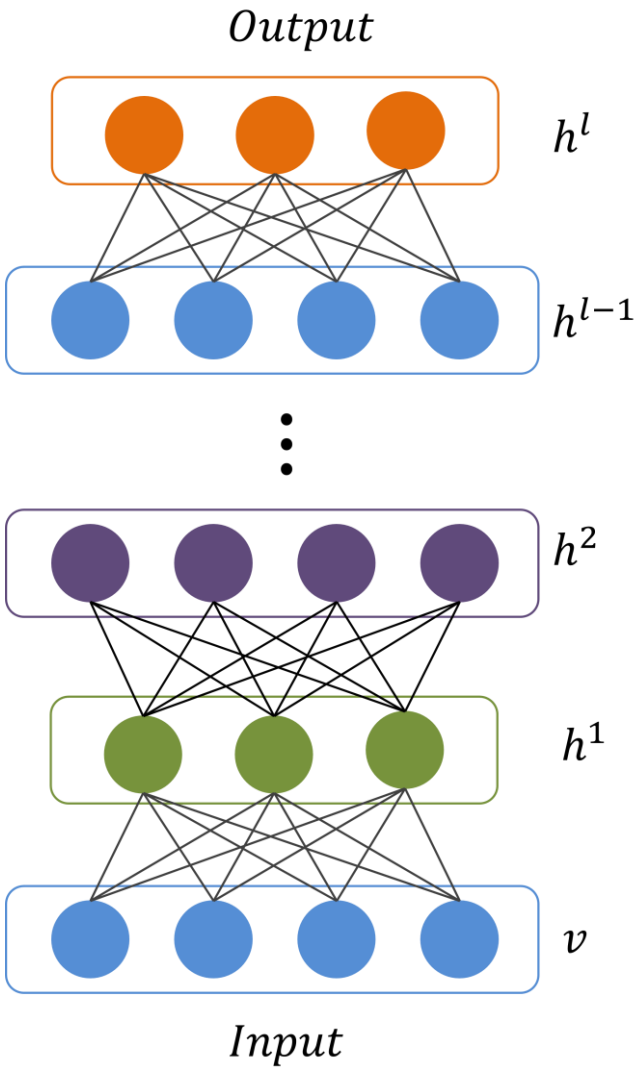


Fig. 3.1 Diagram of the multi-layer neural network. Typically used in supervised learning to make a prediction or classification through a series of layers. Deterministic transformations are computed in a feedforward way from the input v, through the hidden layers, to the network output h^l, which gets compared with a label y to obtain the loss $Cost(h^L, y)$ to be minimized.

Starting with the input vector $v = h^0$, layer l computes an output vector h^l using the output h^{l-1} of previous layer

$$h^l = g(W^l h^{l-1} + b^l) \tag{3.1}$$

where W^l denotes the matrix of weights between layer $l-1$ and layer l. b^l denotes the bias of l^{th} layer. $g(\cdot)$ is an activation function. The commonly used activation functions are sigmoid function and hyperbolic tangent function:

$$g(z) = \frac{1}{1+e^{-z}} \qquad g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{3.2}$$

The top layer h^l is used to make a prediction. Typically, the output layer is combined with a label y into a loss function $Cost(h^l, y)$. The activation function of output layer may different from the one used in other layers, such as a softmax function

$$h_i^l = \frac{e^{W_i^l h^{l-1} + b_i^l}}{\sum_j e^{W_j^l h^{l-1} + b_j^l}} \tag{3.3}$$

where W_i^l is the i-th row of W^l, h_i^l is the output of i^{th} unit and $\sum_i h_i^l = 1$. The softmax output h_i^l can be used as an estimator of $p(Y = i|x)$, with the interpretation that Y is the class associated with input x.

Compared with shallow architectures, multi-layer neural network has ability to extract features more precisely by a multi-layer feature representation with higher layers represent more abstract features than those from lower ones. However, the traditional training procedure of multi-layer neural network is ineffective.

Traditional training procedure of multi-layer neural network is randomly initialize all the parameters and optimize the network with back-propagation algorithm, given enough labeled data. As mentioned before, this training procedure is more difficult than that of training shallow architectures such as neural network with 1 or 2 hidden layers due to the local minima and the poor generalization over the depth of network [52, 53, 77, 78]. Recently experiments have found that pre-training each layer with an unsupervised learning algorithm is able to improve the results dramatically [54]. This training approach named as greedy layer-wise unsupervised learning algorithm is first introduced for Deep Belief Networks (DBN), which is a generative model formed by stacking a number of RBMs. The training strategy for a DBN can be described as follows [78]. Firstly, the input layer and the first hidden layer is treated as a RBM model and trained using unsupervised learning algorithm, giving rise to an initial set of parameters for the first layer of the deep architecture. Then the activation probabilities of its hidden units is treated as the input data for the next hidden layer above. Next the first hidden layer and the second hidden layer is treated as a RBM model and similarly unsupervised learning algorithm is utilized to train this RBM model. The activation probabilities of the second RBM are then used as input data for the third-layer RBM. This layer-by-layer fashion can be repeated as many times as desired. As illustrated in Fig. 3.2, input layer v and first hidden layer h^1 can be formed

as a RBM model. This RBM will be trained iteratively using unsupervised learning algorithm until some criterion are satisfied. Then the learned activation probabilities h^1 of this RBM are treated as the 'data' for training the next RBM above. Hinton et al. has justified that the stacking procedure improves the variational lower bound on the likelihood of the training data under the composite model, which means the greedy layer-wise pre-training achieves approximate maximum likelihood learning [54, 55, 79, 80].

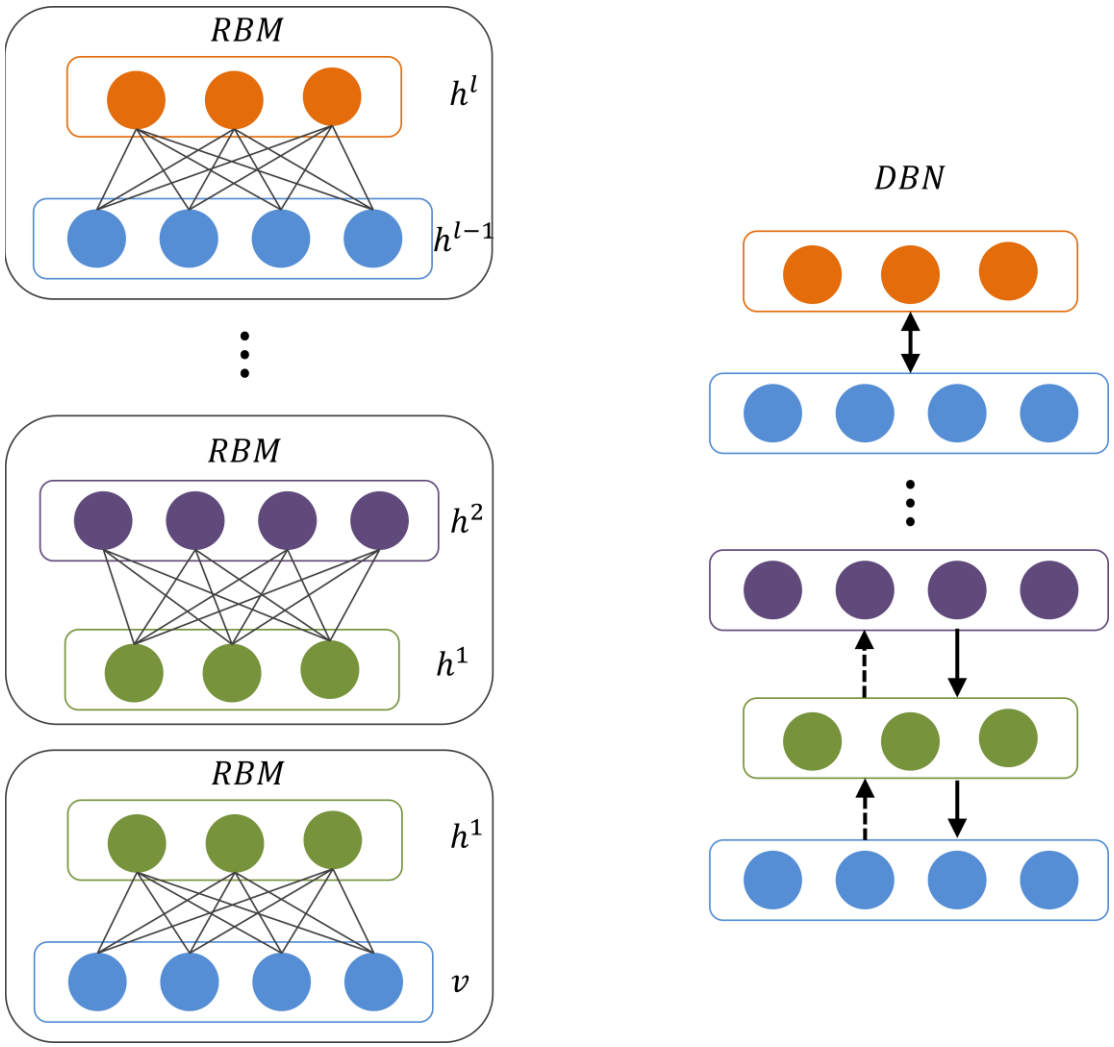


Fig. 3.2 Diagram of DBN constructed by stacking RBM

The parameters of the whole network can be obtained by using layer-wise pre-training method. However, the learned parameters of the deep network is not optimal. Therefore, fine-tuning is necessity to optimize the parameters to improve the performance of the network. When applied to a classification task, a layer of variables that represent the desired labels is added. Then the back-propagation algorithm is applied to refine the network parameters in the same way as for the standard feed-forward neural networks [54]. Compared to the traditional multi-layer neural

networks, pre-training based on the use of RBM to construct DBN has achieved many promising results on a wide variety of tasks [52, 55, 78-80].

Consequently, in DNN scheme each layer is trained in a greedy layer-wise learning algorithm first followed by a back-propagation fine-tuning stage to optimize the parameters of the whole network. In the next subsection, the widely used DNN is presented. Because DNN is constructed by stacking RBMs, which is an *energy-based model* (EBM), therefore the main mathematical concepts of EBM and RBM is described in details.

3.2 Energy-Based model

EBM associate a scalar energy to each configuration of the variables of interest. The task of learning from the EBM can be explained as modifying the energy function so that its shape has desirable properties. For example, desirable configurations that have low energy.

Consider a physical system with many degrees of freedom. Let p_i denotes the probability of occurrence of a state i with the following properties:

$$p_i \geq 0 \quad for\ all\ i \tag{3.4}$$

And

$$\sum_i p_i = 1 \tag{3.5}$$

Let E_i denotes the energy of the system when it is in the state i. A fundamental result from statistical mechanics tells us that when the system is in thermal equilibrium with its surrounding environment, state i occurs with a probability defined by

$$p_i = \frac{1}{Z} e^{\left(-\frac{E_i}{k_B T}\right)} \tag{3.6}$$

where T is the absolute temperature in kelvins, k_B is Boltzmann's constant, and Z is a constant that is independent of all states.

$Eq.(3.5)$ defines the condition for the normalization of probabilities. Imposing this condition on $Eq.(3.6)$, we get

$$\sum_i p_i = \sum_i \frac{1}{Z} e^{\left(-\frac{E_i}{k_B T}\right)} = 1 \tag{3.7}$$

$$Z = \sum_i e^{\left(-\frac{E_i}{k_B T}\right)} \tag{3.8}$$

The normalizing quantity Z is called the *partition function*. The probability distribution of $Eq.(3.6)$ is called the *canonical distribution*, or *Gibbs distribution*.

The following two points are noteworthy from the Gibbs distribution [81].

i. States of low energy have a higher probability of occurrence than states of high energy.

ii. As the temperature T is reduced, the probability is concentrated on a smaller subset of low-energy states.

Set the constant k_B and T equal to unity and redefine the probability p_i and partition function Z as follows:

$$p_i = \frac{1}{Z} e^{(-E_i)} \tag{3.9}$$

$$Z = \sum_i e^{(-E_i)} \tag{3.10}$$

3.2.1 EBM with Hidden Variables

In many cases of interest, some component variables v_i cannot be observed simultaneously, or some non-observed variables are introduced to increase the expressive power of the model. Consider an observed part (denoted v) and a hidden part (denoted h), a *joint probability* can be expressed as

$$p(v,h) = \frac{1}{Z} e^{(-E(v,h))} \tag{3.11}$$

$$Z = \sum_{v,h} e^{(-E(v,h))} \tag{3.12}$$

Because only v is observed, the marginal

$$p(v) = \sum_h p(v,h) = \frac{1}{Z} \sum_h e^{(-E(v,h))} \tag{3.13}$$

In such cases, to map $Eq.(3.10)$ to the one similar to $Eq.(3.9)$, a *free energy* is introduced

$$F(v) = -log \sum_h e^{(-E(v,h))} \tag{3.14}$$

Therefore, the marginal function can be rewritten as

$$p(v) = \frac{1}{Z} e^{(-F(v))} \tag{3.15}$$

$$Z = \sum_v e^{(-F(v))} \tag{3.16}$$

From $Eq.(3.14)$ and $Eq.(3.15)$, it can be concluded that the free energy is a marginalization of energies in the log-domain.

$$logp(v) = -F(v) - logZ \tag{3.17}$$

The log-likelihood gradient has a particularly interesting form. Let us introduce θ to present parameters of the model. The negative log-likelihood of the data can be obtained

$$-\frac{\partial logp(v)}{\partial \theta} = \frac{\partial F(v)}{\partial \theta} - \sum_{\tilde{v}} \mathrm{p}(\tilde{v}) \frac{\partial F(\tilde{v})}{\partial \theta} \tag{3.18}$$

The two terms in the right hand side of the above equation are denoted as positive phase and negative phase. The positive phase raise the probability of the observed data by lowering the free energy while the negative phase decreases the probability of samples generated by the model [82]. However, the gradient of the second term is intractable because $\mathbb{E}_p\left[\frac{\partial F(v)}{\partial \theta}\right]$ is the expectation of all possible configurations of the observable v times their corresponding probability distribution p. Therefore, an approximation approach is required.

3.3 Restricted Boltzmann Machine

Boltzmann Machine (BM) is a special structure of Markov Random Field (MRF).

An RBM is a MRF associated with a bipartite undirected graph as shown in Fig. 3.3. It contains of m visible units $V = (v_1, \dots, v_m)$ representing the observable data and n hidden units $H = (h_1, \dots, h_n)$ to capture the features between the observed variables.

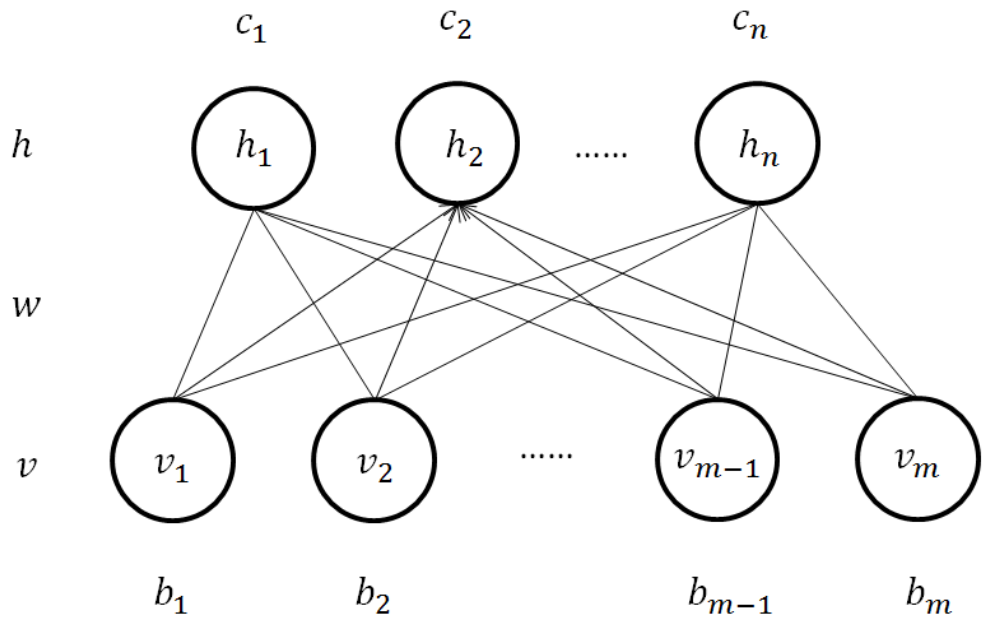


Fig. 3.3 Architecture of RBM

In binary RBM, the random variable (V, H) takes the value $v \in \{0,1\}$ and $h \in \{0,1\}$. The energy function of an RBM is defined as follows:

$$E(v,h) = -\sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij} h_i v_j - \sum_{j=1}^{m} b_j v_j - \sum_{i=1}^{n} c_i h_i \tag{3.19}$$

For all $i \in \{1,2,\dots,n\}$ and $j \in \{1,2,\dots,m\}$, w_{ij} is the weight associated with visible unit v_j and hidden unit h_i, b_j and c_i are bias terms associated with the j^{th} visible and the i^{th} hidden variable, respectively. The aim of training RBM is to adjust the parameters $\theta = \{w, b, c\}$ such that the probability distribution the model represents fits the training data as well as possible.

The graph of an RBM has connections only between the layer of hidden and the layer of visible variables, but not between two variables of the same layer. In terms of probability, this means the hidden variables are independent given the state of the visible variables and vice versa. Therefore, the conditional distributions $p(h|v)$ and $p(v|h)$ can be factorized nicely

$$p(h|v) = \prod_{i=1}^{n} p(h_i|v) \tag{3.20}$$

$$p(v|h) = \prod_{j=1}^{m} p(v_j|h) \tag{3.21}$$

As mentioned above, an RBM is a MRF with hidden variables. The Gibbs distribution of a MRF describes the joint probability distribution of (V, H) as follows:

$$p(v, h) = \frac{1}{Z} e^{(-E(v,h))} \tag{3.22}$$

where $Z = \sum_{v,h} e^{(-E(v,h))}$.

The probability that the network assigns to a visible vector is given by summing over all possible hidden vector:

$$p(v) = \sum_{h} p(v, h) = \frac{1}{Z} \sum_{h} e^{(-E(v,h))} \tag{3.23}$$

RBM is also a stochastic neural network. A standard way of estimating the parameters of a statistical model is the maximum-likelihood estimation. Applied to RBM, this corresponds to finding the RBM parameters θ that maximize the likelihood of training data.

Maximizing the likelihood is the same as maximizing the log-likelihood. For model of the form $Eq.(3.23)$ with parameters θ, the log-likelihood given a single training example v is

$$\begin{aligned} log\mathcal{L}(\theta|v) = logp(v|\theta) &= log \frac{1}{Z} \sum_{h} e^{(-E(v,h))} \\ &= log \sum_{h} e^{(-E(v,h))} - log \sum_{v,h} e^{(-E(v,h))} \end{aligned} \tag{3.24}$$

and for the gradient we get

$$\frac{\partial log\mathcal{L}(\theta|v)}{\partial\theta} = \frac{\partial}{\partial\theta}\left(log\sum_{h} e^{(-E(v,h))}\right) - \frac{\partial}{\partial\theta}\left(log\sum_{v,h} e^{(-E(v,h))}\right)$$

$$= \frac{1}{\sum_h e^{(-E(v,h))}}\sum_{h} e^{(-E(v,h))}\left(-\frac{\partial E(v,h)}{\partial\theta}\right)$$

$$-\frac{1}{\sum_{v,h} e^{(-E(v,h))}}\sum_{v,h} e^{(-E(v,h))}\left(-\frac{\partial E(v,h)}{\partial\theta}\right)$$

$$= -\sum_{h}\frac{e^{(-E(v,h))}}{\sum_h e^{(-E(v,h))}}\frac{\partial E(v,h)}{\partial\theta} + \sum_{v,h}\frac{e^{(-E(v,h))}}{\sum_{v,h} e^{(-E(v,h))}}\left(\frac{\partial E(v,h)}{\partial\theta}\right)$$

$$= -\sum_{h} p(h|v)\left(\frac{\partial E(v,h)}{\partial\theta}\right)$$

$$+\sum_{v,h} p(v,h)\left(\frac{\partial E(v,h)}{\partial\theta}\right) \tag{3.25}$$

In the last step the conditional probability is used

$$p(h|v) = \frac{p(v,h)}{p(v)} = \frac{\frac{1}{Z}e^{(-E(v,h))}}{\frac{1}{Z}\sum_h e^{(-E(v,h))}} = \frac{e^{(-E(v,h))}}{\sum_h e^{(-E(v,h))}} \tag{3.26}$$

Note that the last expression of $Eq.(3.25)$ is the difference between two expectations: the expected values of the energy function under the conditional distribution of the hidden variables given the training example and under the model distribution. Directly calculating this, which run over all values of the respective variables, leads to a computational complexity which is in general exponential in the number of variables of the MRF.

The first term of $Eq.(3.25)$ (*i.e.* the expectation of the energy gradient under the conditional distribution of the hidden variables given a training sample v) can be computed efficiently. The second term in $Eq.(3.25)$ (*i.e.* the expectation of the energy gradient under the model distribution) can also be written as $\sum_v p(v)\sum_h p(h|v)\frac{\partial E(v,h)}{\partial\theta}$, however, the computation remains intractable for regular sized RBM because its complexity is still exponential in the size of the smallest layer.

According to the energy function (3.19), three parameters $\{w, b, c\}$ need to be computed. The derivative of the log-likelihood of a single training pattern v with respect to the weight w_{ij} becomes

$$\frac{\partial log\mathcal{L}(\theta|v)}{\partial w_{ij}} = -\sum_h p(h|v)\left(\frac{\partial E(v,h)}{\partial w_{ij}}\right) + \sum_{v,h} p(v,h)\left(\frac{\partial E(v,h)}{\partial w_{ij}}\right)$$

$$= \sum_h p(h|v)h_i v_j - \sum_v p(v)\sum_h p(h|v)\, h_i v_j$$

$$= p(h_i = 1|v)v_j - \sum_v p(v)\, p(h_i = 1|v)v_j \tag{3.27}$$

where

$$\frac{\partial E(v,h)}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}}\left(-\sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij}h_i v_j - \sum_{j=1}^{m} b_j v_j - \sum_{i=1}^{n} c_i h_i\right) = -h_i v_j \tag{3.28}$$

Because $h_i \in \{0,1\}$,

$$\sum_h p(h|v)h_i = p(h_i = 1|v) \tag{3.29}$$

Analogously to $Eq.(3.27)$, the derivatives with respect to the bias parameters b_j of the j^{th} visible variable and c_i of the i^{th} hidden variable can be obtained as

$$\frac{\partial log\mathcal{L}(\theta|v)}{\partial b_j} = -\sum_h p(h|v)\left(\frac{\partial E(v,h)}{\partial b_j}\right) + \sum_{v,h} p(v,h)\left(\frac{\partial E(v,h)}{\partial b_j}\right)$$

$$= \sum_h p(h|v)v_j - \sum_v p(v)\sum_h p(h|v)\, v_j$$

$$= v_j - \sum_v p(v)\, v_j \tag{3.30}$$

$$\frac{\partial log\mathcal{L}(\theta|v)}{\partial c_i} = -\sum_h p(h|v)\left(\frac{\partial E(v,h)}{\partial c_i}\right) + \sum_{v,h} p(v,h)\left(\frac{\partial E(v,h)}{\partial c_i}\right)$$

$$= \sum_h p(h|v)h_i - \sum_v p(v)\sum_h p(h|v)\, h_i$$

$$= p(h_i = 1|v) - \sum_v p(v)\, p(h_i = 1|v) \tag{3.31}$$

The conditional probability of a single variable being one can be interpreted as the firing rate of a neuron with the sigmoid activation function $sigm(x) = \frac{1}{1+e^{-x}}$

$$p(h_i = 1|v) = sigm\left(\sum_{j=1}^{m} w_{ij}v_j + c_i\right) \tag{3.32}$$

$$p(v_j = 1|h) = sigm\left(\sum_{i=1}^{n} w_{ij}h_i + b_j\right) \tag{3.33}$$

To see this, let v_{-k} denote the state of all visible units except the k-th one and let us define

$$\alpha_k(h) = -\sum_{i=1}^{n} w_{ik} h_i - b_k \tag{3.34}$$

$$\beta(v_{-k}, h) = -\sum_{i=1}^{n} \sum_{j=1, j\neq k}^{m} w_{ij} h_i v_j - \sum_{j=1, j\neq k}^{m} b_j v_j - \sum_{i=1}^{n} c_i h_i \tag{3.35}$$

Therefore $E(v,h) = \beta(v_{-k}, h) + v_k \alpha_k(h)$, where $v_k \alpha_k(h)$ collects all terms involving v_k and we can write

$$\begin{aligned} p(v_k = 1|h) = p(v_k = 1|v_{-k}, h) &= \frac{p(v_k = 1, v_{-k}, h)}{p(v_{-l}, h)} = \frac{\mathrm{e}^{-E(v_k=1, v_{-k}, h)}}{\mathrm{e}^{-E(v_k=1, v_{-k}, h)} + \mathrm{e}^{-E(v_k=0, v_{-k}, h)}} \\ &= \frac{\mathrm{e}^{-\beta(v_{-k}, h) - 1\cdot\alpha_k(h)}}{\mathrm{e}^{-\beta(v_{-k}, h) - 1\cdot\alpha_k(h)} + \mathrm{e}^{-\beta(v_{-k}, h) - 0\cdot\alpha_k(h)}} = \frac{\mathrm{e}^{-\beta(v_{-k}, h)} \cdot e^{-\alpha_k(h)}}{\mathrm{e}^{-\beta(v_{-k}, h)} \cdot \mathrm{e}^{-\alpha_k(h)} + \mathrm{e}^{-\beta(v_{-k}, h)}} \\ &= \frac{\mathrm{e}^{-\beta(v_{-k}, h)} \cdot e^{-\alpha_k(h)}}{\mathrm{e}^{-\beta(v_{-k}, h)} \cdot \mathrm{e}^{-\alpha_k(h)+1}} = \frac{e^{-\alpha_k(h)}}{\mathrm{e}^{-\alpha_k(h)+1}} = \frac{1}{1 + e^{\alpha_k(h)}} = sigm\big(-\alpha_k(h)\big) \\ &= sigm\left(\sum_{i=1}^{n} w_{ik} h_i + b_k\right) \end{aligned} \tag{3.36}$$

Since h and v play a symmetric role in the energy function, a similar derivation of $p(h_i = 1|v)$ could be computed and sampled efficiently.

According to $Eq.(3.32)$, the first term of the log-likelihood gradient can be computed. However, when calculating the second term of the log-likelihood gradient, it will suffer exponential complexity of summing over all values of the visible variables. To avoid this problem, one can approximate this expectation by sampling from the model distribution. These samples can be obtained by Gibbs sampling. But Gibbs sampling requires running the Markov chain "long enough" to ensure the convergence to stationary. Since the computational costs are still too large to yield an efficient learning algorithm, another approach named Contrastive Divergence (CD) learning was proposed [79]. The k-step CD can be described as follows: The Gibbs chain is initialized with a training example $v^{(0)}$ of the training set and yields the sample $v^{(k)}$ after k step. Each step t consists of sampling $h^{(t)}$ from $p(h|v^{(t)})$ and subsequently sampling $v^{(t+1)}$ from $p(v|h^{(t)})$. The gradient $Eq.(3.25)$ with regard to θ of the log-likelihood for one training sample $v^{(0)}$ is then approximated by

$$CD_k\big(\theta, v^{(0)}\big) = -\sum_h p\big(h\big|v^{(0)}\big) \frac{\partial E\big(v^{(0)}, h\big)}{\partial \theta} + \sum_h p\big(h\big|v^{(k)}\big) \frac{\partial E\big(v^{(k)}, h\big)}{\partial \theta} \tag{3.37}$$

The derivatives in the direction of each single parameter are obtained by estimating the expectations over $p(v)$

$$CD_1(\mathrm{w}_{ij}, v^{(0)}) = -\sum_h p(h|v^{(0)}) \frac{\partial E(v^{(0)}, h)}{\partial \theta} + \sum_h p(h|v^{(1)}) \frac{\partial E(v^{(1)}, h)}{\partial \theta}$$

$$= p(h_i = 1|v^{(0)}) v_j^{(0)} - p(h_i = 1|v^{(1)}) v_j^{(1)} \tag{3.38}$$

$$CD_1(\mathrm{b}_j, v^{(0)}) = -\sum_h p(h|v^{(0)}) \frac{\partial E(v^{(0)}, h)}{\partial \theta} + \sum_h p(h|v^{(1)}) \frac{\partial E(v^{(1)}, h)}{\partial \theta}$$

$$= v_j^{(0)} - v_j^{(1)} \tag{3.39}$$

$$CD_1(\mathrm{c}_i, v^{(0)}) = -\sum_h p(h|v^{(0)}) \frac{\partial E(v^{(0)}, h)}{\partial \theta} + \sum_h p(h|v^{(1)}) \frac{\partial E(v^{(1)}, h)}{\partial \theta}$$

$$= p(h_i = 1|v^{(0)}) - p(h_i = 1|v^{(1)}) \tag{3.40}$$

A pseudo-code is shown in Table 3.1.

Table 3.1 1-step Contrastive Divergence

Input: RBM $(v_1, \dots, v_m, h_1, \dots, h_n)$, training samples x.

Output: gradient approximation $\Delta w_{ij}, \Delta b_j \; and \; \Delta c_i \; for \; i = 1, \dots, n; j = 1, \dots, m$

1) Initialize $\Delta w_{ij} = \Delta b_j = \Delta c_i = 0 \; for \; i = 1, \dots, n; j = 1, \dots, m$
2) For all the samples x, do
3) $v^{(0)} \leftarrow x$
4) For $t = 1, \dots k$(training period), do
5) For $i = 1, \dots, n$ do

$$p\left(h_i^{(0)} = 1 \middle| v^{(0)}\right) = sigm\left(\sum_{j=1}^{m} w_{ij} v_j^{(0)} + c_i\right)$$

Sample $h_i^{(0)} \in \{0,1\}$ from $p\left(h_i^{(0)} \middle| v^{(0)}\right)$

6) For $j = 1, \dots, m$ do

$$p\left(v_j^{(1)} = 1 \middle| h^{(0)}\right) = sigm\left(\sum_{i=1}^{n} w_{ij} h_i + b_j\right)$$

Sample $v_j^{(1)} \in \{0,1\}$ from $p\left(v_j^{(1)} \middle| h^{(0)}\right)$

7) For $i = 1, \dots, n$ do

$$p\left(h_i^{(1)} = 1 \middle| v^{(1)}\right) = sigm\left(\sum_{j=1}^{m} w_{ij} v_j^{(1)} + c_i\right)$$

8) For $i = 1, \dots, n; j = 1, \dots, m$ do

$$\Delta w_{ij} \leftarrow \Delta w_{ij} + \left(p\left(h_i^{(0)} = 1 \middle| v^{(0)}\right) v_j^{(0)} - p\left(h_i^{(1)} = 1 \middle| v^{(1)}\right) v_j^{(1)}\right)$$

9) For $j = 1, \dots, m$ do

$$\Delta b_j \leftarrow \Delta b_j + (v_j^{(0)} - v_j^{(1)})$$

10) For $i = 1, \dots, n$ do

$$\Delta c_i \leftarrow \Delta c_i + \left(p\left(h_i^{(0)} = 1 \middle| v^{(0)}\right) - p\left(h_i^{(1)} = 1 \middle| v^{(1)}\right)\right)$$

If the input data are continuous rather than binomial data, the hidden units of the RBM remain binary, but the visible units are replaced by linear units with Gaussian noise. This RBM termed as Gaussian Bernoulli RBM (GBRBM), which the energy function can be expressed as:

$$E(v, h) = -\sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij} h_i \frac{v_j}{\sigma_j} - \sum_{j=1}^{m} \frac{(v_j - b_j)^2}{2\sigma_j^2} - \sum_{i=1}^{n} c_i h_i \tag{3.41}$$

If the noise has zero mean and unit variance, then the update rule for the hidden units remains the same but the update rule for the visible units becomes as follows:

$$p(h_i = 1|v) = sigm\left(\sum_{j=1}^{m} w_{ij} v_j + c_i\right) \tag{3.42}$$

$$p\left(v_j \middle| h\right) = \mathcal{N}\left(v_j | \mu_j, \sigma_j^2\right) \tag{3.43}$$

where $\mu_j = \sum_{i=1}^{n} w_{ij} h_i + b_j$ and $\mathcal{N}$ is the normal distribution.

The RBM discussed above is an unsupervised model, which characterizes the input data distribution using hidden variables and there is no label information provided. By stacking a number of RBMs learned layer-by-layer gives rise to DBN as illustrated on the right side of Fig. 3.2. However, the learned parameters of the deep network is not optimal. Therefore, fine-tuning is necessity. Supervised fine-tuning can be performed by adding a final layer of variables that represent the desired outputs. Given labelled data, the parameters of the whole network can be bettered further to model the input data by supervised fine-tuning.

3.4 Deep Belief Network

DBN is a powerful multilayer generative model, which can be viewed as a composition of RBMs [54, 55, 83-90]. DBN learns to extract a deep hierarchical representation of the training data. The joint distribution between observed vector v and the hidden layer h^k is expressed as follows

$$p(v, h^1, \dots, h^l) = \left(\prod_{k=0}^{l-2} p(h^k | h^{k+1})\right) p(h^{l-1}, h^l) \tag{3.44}$$

where $v = h^0$, $p(h^{k-1}|h^k)$ is a conditional distribution for the visible units conditioned on the hidden units of the RBM at level k, and $p(h^{l-1}, h^l)$ is the visible-hidden joint distribution in the top-level RBM.

As for source separation, the problem often solved through TF domain. Due to the input data is continuous, therefore, GBRBM is utilized to train the input layer and first hidden layer. There are two categories while using DNN to solve the separation problem. In the first category, DNN is trained as a regression model to recover the target source directly. In this category, DNN is utilized as a regression model to predict the log-power spectral features of the target source given the input log-power spectral features of the mixed signals. The training of DNN consists of two stages: unsupervised pre-training with RBM and supervised fine-tuning with back-propagation. While in the fine-tuning stage, the objective is to minimize a certain cost function between the predicted output and the provided clean features of the target source.

In the second category, DNN is trained to estimate a TF mask by classifying each TF unit of the mixed signal to its corresponding domain. In this category, DNN is utilized as a discriminative model to estimate a TF mask of the target source. The unsupervised pre-training procedure is the same as in the first category but the supervised fine-tuning is different. Instead of providing clean acoustic features of the target source, the labels of the sources are given. The objective is to calculate the probability of a given features that belongs to the target source of the interfering source.

3.5 Summary

This section focuses on the RBM and DBN. The EBM is presented in details followed by RBM. The use of the CD update rule for training RBM is provided in details. The GBRBM for modelling real valued data is also introduced.

Chapter 4. Pseudo channel GEM-MU based NMF-2D with Deep Sparse Extreme Learning Machine

In this chapter, a novel framework called Deep Sparse Extreme Learning Machine (DSELM) for Single Channel Source Separation (SCSS) with assistance of channel diversity realized by a pair of artificial stereo signals is proposed. The proposed approach employs the Nonnegative Matrix Factorization Two-Dimensional Deconvolution (NMF-2D) to perform a coarse separation followed by a more refined estimation using the developed DSELM.

The NMF-2D is optimized using a hybrid framework that combines the Generalized Expectation-Maximization (EM) with the Multiplicative Update rule (GEM-MU) algorithm. Furthermore, an artificial stereo channel termed as the pseudo stereo mixture has been adapted to alleviate the problem for only one observation with several unknown variables. Given a single observation, the model has ability to generate pseudo stereo mixture that has an artificial resemblance of the stereo signal. This model takes advantage of the relationship between the readily available mixture and the pseudo-stereo mixture to estimate the signature parameter of the original signals.

The DSELM extracts the features of the separated sources layer-by-layer and the extracted information is calculated to check for their validity during the fine-tuning stage. In addition, a joint energy minimization method based on the trained DSELM is proposed to improve the quality of the coarsely separated signals previously obtained from the NMF-2D.

This chapter is organized as follows: Section 4.1 introduces the pseudo stereo model and ELM. In Section 4.2, the GEM-MU algorithm is proposed to optimize the parameters of the NMF-2D. The DSELM and the joint energy minimization method are developed in section 4.3. Experimental results and series of comparisons with other approaches are presented in section 4.4. Section 4.5 concludes the chapter.

4.1 Background

4.1.1 *Pseudo stereo mixture*

Pseudo stereo mixture generates a pseudo mixture that is used along with the original mixture. The pseudo stereo mixture creates a pair of artificial mixed signals to increase the dimensionality of the mixing matrix, renders full-rank condition and effectively reduces the ambiguity between estimating the mixing coefficients and the source signals. For simplicity, we assume that there are two sources mixed together. Consider a single channel mixture $\tilde{x}_1(t)$ in time domain

$$\tilde{x}_1(t) = u\tilde{s}_1(t) + v\tilde{s}_2(t) + \tilde{b}(t) \tag{4.1}$$

where $\tilde{s}_j(t), j = 1,2.$ denotes the j^{th} source (assumed to normalize to unit variance), $\tilde{b}(t)$ is some additive noise, for $t = 1, \dots, T$, and u and v are the mixing gains. To generate a virtual mixture, we assume that $\tilde{s}_j(t)$ is a quasi-stationary autoregressive (AR) where the AR parameters are stationary within a frame but can change from frame to frame. Thus, the sources can be modeled as [91, 92]

$$\tilde{s}_1(t) = -\sum_{\tau=1}^{\mathcal{D}_{\tilde{s}_1}} c_{\tilde{s}_1}(\tau; t)\tilde{s}_1(t-\tau) + \tilde{z}_{\tilde{s}_1}(t) \tag{4.2}$$

$$\tilde{s}_2(t) = -\sum_{\tau=1}^{\mathcal{D}_{\tilde{s}_2}} c_{\tilde{s}_2}(\tau; t)\tilde{s}_2(t-\tau) + \tilde{z}_{\tilde{s}_2}(t) \tag{4.3}$$

where $c_{\tilde{s}_j}(\tau; t)$ denotes the τ^{th} order AR coefficient of the signal at time t, $\mathcal{D}_{\tilde{s}_j}$ denotes the maximum AR order, and $\tilde{z}_{\tilde{s}_j}(t)$ is a residue factor that is an independent identically distributed (i.i.d.) random signal with zero mean and variance $\sigma^2_{\tilde{z}_j}$. By weighting and time-shifting the observed signal $\tilde{x}_1(t)$, a virtual mixture can be formulated

$$\tilde{x}_2(t) = \frac{\tilde{x}_1(t) + \gamma\tilde{x}_1(t-\delta)}{1+|\gamma|} \tag{4.4}$$

where $\gamma \in \mathcal{R}$ is the weight parameter and δ is the time-delay between $\tilde{x}_2(t)$ and $\tilde{x}_1(t)$. The mixture in $Eq.(4.1)$ and $Eq.(4.4)$ are termed as 'pseudo stereo'. This is because the mixture has an artificial resemblance of a stereo signal except only one location is given. This will result in the same time-delay but different attenuation of the source signals. $Eq.(4.4)$ can be rewritten as

$$\begin{aligned}
\tilde{x}_2(t) &= \frac{\tilde{x}_1(t) + \gamma\tilde{x}_1(t-\delta)}{1+|\gamma|} \\
&= \frac{u\tilde{s}_1(t) + v\tilde{s}_2(t) + \tilde{b}(t)}{1+|\gamma|} + \frac{\gamma\left[u\tilde{s}_1(t-\delta) + v\tilde{s}_2(t-\delta) + \tilde{b}(t-\delta)\right]}{1+|\gamma|} \\
&= \frac{u\left(-c_{\tilde{s}_1}(\delta) + \gamma\right)}{1+|\gamma|}\tilde{s}_1(t-\delta) + \frac{v\left(-c_{\tilde{s}_2}(\delta) + \gamma\right)}{1+|\gamma|}\tilde{s}_2(t-\delta) \\
&\quad + \frac{u\left(\tilde{z}_{\tilde{s}_1}(t) - \sum_{\substack{\tau=1 \\ \tau\neq\delta}}^{\mathcal{D}_{\tilde{s}_1}} c_{\tilde{s}_1}(\tau)\tilde{s}_1(t-\tau)\right)}{1+|\gamma|} + \frac{v\left(\tilde{z}_{\tilde{s}_2}(t) - \sum_{\substack{\tau=1 \\ \tau\neq\delta}}^{\mathcal{D}_{\tilde{s}_2}} c_{\tilde{s}_2}(\tau)\tilde{s}_2(t-\tau)\right)}{1+|\gamma|} \\
&\quad + \frac{\tilde{b}(t) + \gamma\tilde{b}(t-\delta)}{1+|\gamma|}
\end{aligned} \tag{4.5}$$

The mixing coefficient is define as follows

$$a_1(\delta) = \frac{u\left(-c_{\tilde{s}_1}(\delta) + \gamma\right)}{1+|\gamma|}, \qquad a_2(\delta) = \frac{v\left(-c_{\tilde{s}_2}(\delta) + \gamma\right)}{1+|\gamma|} \tag{4.6}$$

The residue of the source signals is represented by

$$\tilde{r}_1(t) = \frac{u\left(\tilde{z}_{\tilde{s}_1}(t) - \sum_{\substack{\tau=1 \\ \tau\neq\delta}}^{\mathcal{D}_{\tilde{s}_1}} c_{\tilde{s}_1}(\tau)\tilde{s}_1(t-\tau)\right)}{1+|\gamma|}$$

$$\tilde{r}_2(t) = \frac{v\left(\tilde{z}_{\tilde{s}_2}(t) - \sum_{\substack{\tau=1 \\ \tau\neq\delta}}^{\mathcal{D}_{\tilde{s}_2}} c_{\tilde{s}_2}(\tau)\tilde{s}_2(t-\tau)\right)}{1+|\gamma|} \tag{4.7}$$

$\tilde{v}(t)$ denotes the noise obtained by weighting and time-shifting of the additive noise $\tilde{b}(t)$ plus the residues

$$\tilde{v}(t) = \tilde{r}_1(t) + \tilde{r}_2(t) + \frac{\tilde{b}(t) + \gamma\tilde{b}(t-\delta)}{1+|\gamma|} \tag{4.8}$$

Using $a_j, \tilde{r}_j(t)$ and $\tilde{v}(t)$, the overall noisy mixing model can be formulated as

$$\begin{aligned} \tilde{x}_1(t) &= u\tilde{s}_1(t) + v\tilde{s}_2(t) + \tilde{b}(t) \\ \tilde{x}_2(t) &= a_1(\delta)\tilde{s}_1(t-\delta) + a_2(\delta)\tilde{s}_2(t-\delta) + \tilde{v}(t) \end{aligned} \tag{4.9}$$

4.1.2 *Extreme Learning Machine*

Extreme Learning Machine (ELM) is a learning algorithm that proposed to train a single layer feed forward neural network [69, 75, 93, 94]. Unlike the traditional neural networks, ELM is able to maintain the universal approximation capability without updating the randomly generated hidden neurons [69, 93].

The ELM model with *L* hidden nodes can be expressed as

$$f_L(x) = \sum_{i=1}^{L} h_i(\boldsymbol{a}_i \cdot \boldsymbol{x} + b_i) \cdot \beta_i \tag{4.10}$$

where $h_i(\cdot)$ denotes the i^{th} hidden unit activation function, $\boldsymbol{a}_i \in \mathcal{R}^d$ is the input weight vector connecting the input layer to the i^{th} hidden unit, $b_i \in \mathcal{R}$ denotes the bias of i^{th} hidden unit, and $\beta_i \in \mathcal{R}$ is the output weight. Given *N* training samples $\{(\boldsymbol{x}_i, \boldsymbol{t}_i) | \boldsymbol{x}_i \in \mathcal{R}^{d_N}, \boldsymbol{t}_i \in \mathcal{R}^{d_M}\}$, the ELM can resolve the following learning problem

$$\boldsymbol{H\beta} = \boldsymbol{T} \tag{4.11}$$

Where $\boldsymbol{H}$ is the hidden layer output matrix

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{h}(\boldsymbol{x}_1) \\ \vdots \\ \boldsymbol{h}(\boldsymbol{x}_N) \end{bmatrix} = \begin{bmatrix} \boldsymbol{h_1}(\boldsymbol{x}_1) & \cdots & \boldsymbol{h_L}(\boldsymbol{x}_1) \\ \vdots & \ddots & \vdots \\ \boldsymbol{h_1}(\boldsymbol{x}_N) & \cdots & \boldsymbol{h_L}(\boldsymbol{x}_N) \end{bmatrix} \tag{4.12}$$

and $\boldsymbol{T}$ is the target label

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{t}_1^{\mathrm{T}} \\ \vdots \\ \boldsymbol{t}_N^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} t_{11} & \cdots & t_{1m} \\ \vdots & \ddots & \vdots \\ t_{N1} & \cdots & t_{NM} \end{bmatrix} \tag{4.13}$$

With the target of each training set, the weights between the hidden layer and the output layer can be calculated as follows

$$\boldsymbol{\beta} = \boldsymbol{H}^{\dagger}\boldsymbol{T} \tag{4.14}$$

where $\boldsymbol{H}^{\dagger}$ is the Moore-Penrose generalized inverse of matrix $\boldsymbol{H}$ [95].

4.2 Proposed method

The proposed approach consists of four stages as depicted in Fig. 4.1. Stage I involves the generation of pseudo-stereo mixtures. Stage II is the unsupervised separation where the mixture will be coarsely separated by the GEM-MU based NMF-2D algorithm. Stage III is the DSELM learning which consists of unsupervised training and supervised learning. Finally, in Stage IV, the trained DSELM will be used to calculate the weightage contributions of the coarsely separated sources followed by an energy minimization method to further improve the separation performance.

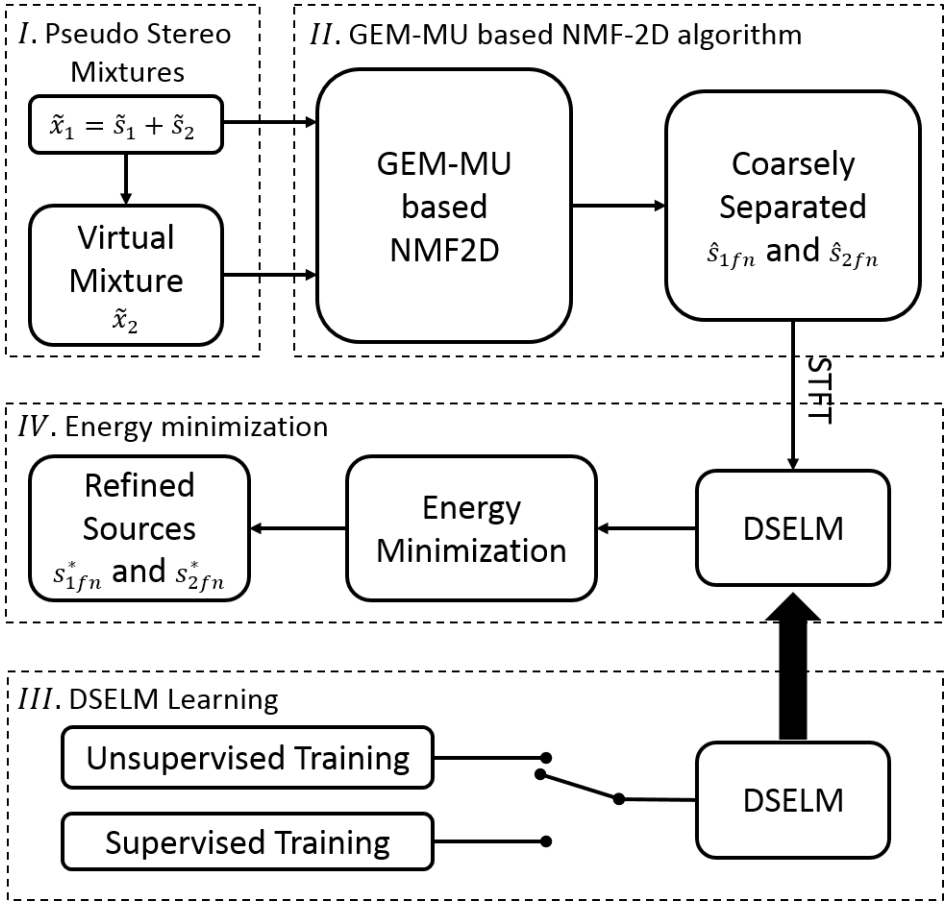


Fig. 4.1 Proposed approach

4.2.1 *GEM-MU algorithm*

In this section, GEM-MU algorithm is developed to optimize the parameters of the NMF-2D.

As mentioned before, the pseudo mixture model is expressed as $Eq.$ (4.9), the TF representation can be obtained by using the Short-Time Fourier Transform (STFT)

$$\begin{aligned} x_{1,f,n} &= us_{1,f,n} + vs_{2,f,n} + b_{f,n} \\ x_{2,f,n} &= a_1(\delta)e^{-i\omega\delta}s_{1,f,n-\delta} + a_2(\delta)e^{-i\omega\delta}s_{2,f,n-\delta} + v_{f,n} \end{aligned} \tag{4.15}$$

where $x_{j,f,n}$, $s_{j,f,n}$, $b_{f,n}$ and $v_{f,n}$ are the STFT of $\tilde{x}_j(t)$, $\tilde{s}_j(t)$, $\tilde{b}(t)$ and $\tilde{v}(t)$, respectively. $n = 1, \dots, N$, $f = 1, \dots, F$ are the time frame index and frequency bin index, respectively. By invoking stationarity of the source signals *i.e.*, $STFT[\tilde{s}_1(t-\delta)] = e^{-i\omega\delta}s_{1,f,n-\delta} \approx e^{-i\omega\delta}s_{1,f,n}$, $Eq.$ (4.16) can be expressed as

$$\boldsymbol{x}_{f,n} \cong \boldsymbol{A}_f \boldsymbol{s}_{f,n} + \boldsymbol{b}_f \tag{4.16}$$

Where $\boldsymbol{x}_{f,n} = \begin{bmatrix} x_{1,f,n} \\ x_{2,f,n} \end{bmatrix} \in \mathbb{C}^{2\times 1}$, $\boldsymbol{A}_f = \begin{bmatrix} u & v \\ a_{1,f}(\delta) & a_{2,f}(\delta) \end{bmatrix} \in \mathbb{C}^{2\times 2}$, $a_{j,f}(\delta) = a_j(\delta)e^{-i\omega\delta}$, $\boldsymbol{s}_{f,n} = \begin{bmatrix} s_{1,f,n} \\ s_{2,f,n} \end{bmatrix} \in \mathbb{C}^{2\times 1}$, and $\boldsymbol{b}_f = \begin{bmatrix} b_{f,n} \\ v_{f,n} \end{bmatrix} \in \mathbb{C}^{2\times 1}$. By comparing with the single channel mixture in $Eq.$ (4.1), the virtual mixture $\tilde{x}_2(t)$ contains the temporal feature of the source signals *i.e.* $\{a_{j,f}(\delta)\}$ which augments the dimensionality of the mixing matrix and eventually increases its matrix rank. Thus, the pseudo stereo mixture is proposed as a way to ameliorate the ambiguities associated with the underdetermined system.

The NMF-2D has the ability to specify the temporal and the spectral changes of the signal through its convolutive parameters (τ and ϕ), and the number of frequency basis (K). Each source in TF domain can be expressed by K complex-valued latent components, *i.e.* $s_{j,f,n} = \sum_{k=1}^{K_{s_j}} c_{k,f,n}^{s_j}$, for $k = 1, \dots K$, where $c_{k,f,n}^{s_j}$ can be expressed as

$$\begin{aligned} c_{k,f,n}^{s_j} &\sim \mathcal{N}_c\left(0, \sigma_{k,f,n}^{s_j{}^2}\right) \\ &= \mathcal{N}_c\left(0, \sum_{\tau=0}^{\tau_{max}} \sum_{\phi=0}^{\phi_{max}} w_{k,f-\phi,\tau}^{s_j} h_{k,\phi,n-\tau}^{s_j}\right) \end{aligned} \tag{4.17}$$

where $\mathcal{N}_c(\eta, \Sigma)$ is proper complex Gaussian distribution [96, 97], $w_{k,f,\tau}^{s_j}$ represents the spectral basis of the j^{th} source and $h_{k,\phi,n}^{s_j}$ represents the temporal code for each spectral basis element of the j^{th} source.

Assume the noise $b_{f,n}$ and $v_{f,n}$ are stationary and spatially uncorrelated, *i.e.* $b_{f,n} \sim \mathcal{N}_c\left(0, (\sigma_{b,f}^2)^2\right)$ and $\boldsymbol{\Sigma}_{b,f} = diag\left(\left[\sigma_{b,f}^2\right]_{b,f}\right)$. Let $\boldsymbol{X} = \{\boldsymbol{x}_{f,n}\}_{f,n}$ and $\boldsymbol{C} = \left\{c_{k,f,n}^{s_j}\right\}_{k,f,n}$ be the observations and latent variables, and $\boldsymbol{\theta} = \{\boldsymbol{A}, \boldsymbol{W}, \boldsymbol{H}, \boldsymbol{\Lambda}, \boldsymbol{\Sigma}_{\boldsymbol{b},f}\}$ as the parameters of the model where $\boldsymbol{A} = \{\boldsymbol{A}_f\}_f, \boldsymbol{W} = \{\boldsymbol{W}^{s_j}\}, \boldsymbol{H} = \{\boldsymbol{H}^{s_j}\}, \boldsymbol{\Lambda} = \{\boldsymbol{\Lambda}^{s_j}\}, \boldsymbol{\Sigma}_{\boldsymbol{b},f} = \{\boldsymbol{\Sigma}_{b,f}, \boldsymbol{\Sigma}_{v,f}\}$ with $\boldsymbol{W}^{s_j} =$

$\left\{w_{k,f,\tau}^{s_j}\right\}_{k,f,\tau}$, $\boldsymbol{H}^{s_j} = \left\{h_{k,\phi,n}^{s_j}\right\}_{k,\phi,n}$, $\boldsymbol{\Lambda}^{s_j} = \left\{\lambda_{k,\phi,n}^{s_j}\right\}_{k,\phi,n}$. It is worth pointing out that the tensor $\boldsymbol{\Lambda}$ contains the sparsity terms for $\boldsymbol{H}$ as each individual element in $\boldsymbol{H}$ is constrained to an exponential distribution with independent decay parameter $\lambda_{k,\phi,n}^{s_j}$, namely $p(\boldsymbol{H}^{s_j}|\boldsymbol{\Lambda}^{s_j}) = \prod_{k,\phi,n} p\left(h_{k,\phi,n}^{s_j}\middle|\lambda_{k,\phi,n}^{s_j}\right) = \prod_{k,\phi,n} \lambda_{k,\phi,n}^{s_j} \exp\left(-\lambda_{k,\phi,n}^{s_j} h_{k,\phi,n}^{s_j}\right)$. The model parameters and latent variables can be estimated via the Maximum a Posteriori (MAP) probability [96]

$$\widehat{\boldsymbol{\theta}}_{MAP} = \arg\max_{\theta} \log p(\boldsymbol{\theta}|\boldsymbol{X}) \tag{4.18}$$

where

$$\log p(\boldsymbol{\theta}|\boldsymbol{X}) \geq \int \mathcal{J}(\boldsymbol{C}) \log\left[\frac{p(\boldsymbol{C},\boldsymbol{\theta}|\boldsymbol{X})}{\mathcal{J}(\boldsymbol{C})}\right] d\boldsymbol{C} \tag{4.19}$$

the posterior probability is given by

$$p(\boldsymbol{C},\boldsymbol{\theta}|\boldsymbol{X}) = \frac{p(\boldsymbol{X},\boldsymbol{C}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\boldsymbol{X})} \propto p(\boldsymbol{X}|\boldsymbol{C},\boldsymbol{\theta})p(\boldsymbol{C}|\boldsymbol{\theta})P(\boldsymbol{\theta}) \tag{4.20}$$

Next step is to estimate the parameters using the GEM-MU algorithm [96, 98].

In the E-step, the source power spectrogram posterior estimate ($\hat{p}_{j,f,n}$), the mixing gain, and the noise covariance are estimated.

The complete data log-likelihood is given by

$$\begin{aligned}
-\log p(\boldsymbol{C},\boldsymbol{\theta}|\boldsymbol{X}) &= -\log p(\boldsymbol{X}|\boldsymbol{C},\boldsymbol{\theta}) - \log p(\boldsymbol{C}|\boldsymbol{\theta}) - \log p(\boldsymbol{\theta}) \\
&= \sum_{f,n}\Bigg[\log\left|\boldsymbol{\Sigma}_{\boldsymbol{b},f}\right| + \sum_{k=1}^{K_{s_1}} \log\left(\sum_{\tau=0}^{\tau_{max}}\sum_{\phi=0}^{\phi_{max}} w_{k,f-\phi,\tau}^{s_1} h_{k,\phi,n-\tau}^{s_1}\right) \\
&+ \sum_{k=1}^{K_{s_1}} \frac{\left|c_{k,f,n}^{s_1}\right|^2}{\sum_{\tau=0}^{\tau_{max}}\sum_{\phi=0}^{\phi_{max}} w_{k,f-\phi,\tau}^{s_1} h_{k,\phi,n-\tau}^{s_1}} + \sum_{k=1}^{K_{s_2}} \log\left(\sum_{\tau=0}^{\tau_{max}}\sum_{\phi=0}^{\phi_{max}} w_{k,f-\phi,\tau}^{s_2} h_{k,\phi,n-\tau}^{s_2}\right) \\
&+ \sum_{k=1}^{K_{s_2}} \frac{\left|c_{k,f,n}^{s_2}\right|^2}{\sum_{\tau=0}^{\tau_{max}}\sum_{\phi=0}^{\phi_{max}} w_{k,f-\phi,\tau}^{s_2} h_{k,\phi,n-\tau}^{s_2}}\Bigg] \\
&+ N\sum_{f} tr\left[\boldsymbol{\Sigma}_{\boldsymbol{b},f}^{-1}\boldsymbol{R}_{\boldsymbol{xx},f} - \boldsymbol{\Sigma}_{\boldsymbol{b},f}^{-1}\boldsymbol{A}_f\boldsymbol{R}_{\boldsymbol{xs},f}^{H} - \boldsymbol{\Sigma}_{\boldsymbol{b},f}^{-1}\boldsymbol{R}_{\boldsymbol{xs},f}\boldsymbol{A}_f^{H} + \boldsymbol{\Sigma}_{\boldsymbol{b},f}^{-1}\boldsymbol{A}_f\boldsymbol{R}_{\boldsymbol{ss},f}\boldsymbol{A}_f^{H}\right] \\
&- \log p(\boldsymbol{A}_f) - \log p(\boldsymbol{\Sigma}_{\boldsymbol{b},f}) - \log p(\boldsymbol{W}) - \log p(\boldsymbol{H}|\boldsymbol{\Lambda})
\end{aligned} \tag{4.21}$$

Where the superscript H is the Hermitian transpose. $\boldsymbol{R}_{\boldsymbol{xx},f} = \frac{1}{N}\sum_n \boldsymbol{x}_{f,n}\boldsymbol{x}_{f,n}^H$, $\boldsymbol{R}_{\boldsymbol{ss},f} = \frac{1}{N}\sum_n \boldsymbol{s}_{f,n}\boldsymbol{s}_{f,n}^H$, $\boldsymbol{R}_{\boldsymbol{xs},f} = \frac{1}{N}\sum_n \boldsymbol{x}_{f,n}\boldsymbol{s}_{f,n}^H$ and $\left|c_{k,f,n}^{s_j}\right|^2 = \left[\hat{\boldsymbol{c}}_{fn}\hat{\boldsymbol{c}}_{fn}^H + \widehat{\boldsymbol{\Sigma}}_{\boldsymbol{c},fn}\right]_{k,k}^{s_j}$.The source power spectrogram posterior can be estimated as

$$\hat{p}_{j,f,n} = \hat{R}_{\boldsymbol{ss},f,n}(j,j) \tag{4.22}$$

Where

$$\hat{R}_{\boldsymbol{ss},f,n} = \hat{\boldsymbol{s}}_{f,n}\hat{\boldsymbol{s}}_{f,\boldsymbol{n}}^{H} + \hat{\sum}_{s,f,n} \tag{4.23}$$

$$\hat{\boldsymbol{s}}_{f,n} = \boldsymbol{\Sigma}_{\boldsymbol{s},f,n}\boldsymbol{A}_f^H\boldsymbol{\Sigma}_{\boldsymbol{x},f,n}^{-1}\boldsymbol{x}_{f,n} \tag{4.24}$$

$$\hat{\boldsymbol{c}}_{f,n} = \boldsymbol{\Sigma}_{\boldsymbol{c},f,n}\left[\boldsymbol{A}_f \otimes \mathbf{1}_K\right]^H\boldsymbol{\Sigma}_{\boldsymbol{x},f,n}^{-1}\boldsymbol{x}_{f,n} \tag{4.25}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{x},f,n} = \boldsymbol{A}_f\boldsymbol{\Sigma}_{\boldsymbol{s},f,n}\boldsymbol{A}_f^H + \boldsymbol{\Sigma}_{\boldsymbol{b},f} \tag{4.26}$$

$$\hat{\boldsymbol{\Sigma}}_{\boldsymbol{s},f,n} = \left(\boldsymbol{I} - \boldsymbol{\Sigma}_{\boldsymbol{s},f,n}\boldsymbol{A}_f^H\boldsymbol{\Sigma}_{\boldsymbol{x},f,n}^{-1}\boldsymbol{A}_f\right)\boldsymbol{\Sigma}_{\boldsymbol{s},f,n} \tag{4.27}$$

$$\hat{\boldsymbol{\Sigma}}_{\boldsymbol{c},f,n} = \left(\boldsymbol{I} - \boldsymbol{\Sigma}_{\boldsymbol{c},f,n}\left[\boldsymbol{A}_f \otimes \mathbf{1}_K\right]^H\boldsymbol{\Sigma}_{\boldsymbol{x},f,n}^{-1}\left[\boldsymbol{A}_f \otimes \mathbf{1}_K\right]\right)\boldsymbol{\Sigma}_{\boldsymbol{c},f,n} \tag{4.28}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{s},f,n} = \begin{bmatrix} \sum_{k=1}^{K_{s_1}}\sum_{\tau=0}^{\tau_{max}}\sum_{\phi=0}^{\phi_{max}} w_{k,f-\phi,\tau}^{s_1} h_{k,\phi,n-\tau}^{s_1} & 0 \\ 0 & \sum_{k=1}^{K_{s_2}}\sum_{\tau=0}^{\tau_{max}}\sum_{\phi=0}^{\phi_{max}} w_{k,f-\phi,\tau}^{s_2} h_{k,\phi,n-\tau}^{s_2} \end{bmatrix} \tag{4.29}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{c},fn} = \begin{bmatrix} \boldsymbol{\Sigma}_{\boldsymbol{c}^{s_1},f,n} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{\boldsymbol{c}^{s_2},f,n} \end{bmatrix} \tag{4.30}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{c}^{s_j},f,n} = diag\left(\left[\sum_{\tau=0}^{\tau_{max}}\sum_{\phi=0}^{\phi_{max}} w_{k,f-\phi,\tau}^{s_j} h_{k,\phi,n-\tau}^{s_j}\right]_{k,k}\right) \tag{4.31}$$

In above, '$\otimes$' is the Kronecker product and $\mathbf{1}_K$ is a row vector with K unit element where K is the number of complex-valued latent components. Detailed derivations of $Eq.(4.23)-Eq.(4.31)$ follow immediately from the linear Gaussian process model [98, 99].

In the M-step, the parameters $\boldsymbol{W}$ and $\boldsymbol{H}$ are estimated by using the MU algorithm. By setting

$$\frac{\partial}{\partial \boldsymbol{A}_f}\langle \log p(\boldsymbol{X}|\boldsymbol{C},\boldsymbol{\theta}) + \log p\left(\boldsymbol{A}_f\right)\rangle_{P\left(\boldsymbol{C}|\boldsymbol{X},\boldsymbol{\theta}'\right)} = 0 \tag{4.32}$$

$$\frac{\partial}{\partial \boldsymbol{\Sigma}_{\boldsymbol{b},f}^{-1}}\langle \log p(\boldsymbol{X}|\boldsymbol{C},\boldsymbol{\theta}) + \log p(\boldsymbol{\Sigma}_{\boldsymbol{b},f})\rangle_{P\left(\boldsymbol{C}|\boldsymbol{X},\boldsymbol{\theta}'\right)} = 0 \tag{4.33}$$

where $p\left(\boldsymbol{A}_f\right)$ and $p(\boldsymbol{\Sigma}_{\boldsymbol{b},f})$ assume a uniform distribution. $\boldsymbol{A}_f$ and $\boldsymbol{\Sigma}_{\boldsymbol{b},f}$ can be obtained as

$$\boldsymbol{A}_f = \hat{\boldsymbol{R}}_{\boldsymbol{xs},f}\hat{\boldsymbol{R}}_{\boldsymbol{ss},f}^{-1} \tag{4.34}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{b},f} = diag\left(\boldsymbol{R}_{\boldsymbol{xx},f} - \boldsymbol{A}_f\hat{\boldsymbol{R}}_{\boldsymbol{xs},f}^H - \hat{\boldsymbol{R}}_{\boldsymbol{xs},f}\boldsymbol{A}_f^H + \boldsymbol{A}_f\hat{\boldsymbol{R}}_{\boldsymbol{ss},f}\boldsymbol{A}_f^H\right) \tag{4.35}$$

where $\hat{\boldsymbol{R}}_{\boldsymbol{xs},f} = \frac{1}{N}\sum_n \boldsymbol{x}_{f,n}\hat{\boldsymbol{s}}_{f,n}^H$ and $\hat{\boldsymbol{R}}_{\boldsymbol{ss},f} = \frac{1}{N}\sum_n \hat{R}_{\boldsymbol{ss},f,n}$.

The second term in the right hand side of $Eq.(4.20)$ can be expressed using the Itakura-Saito divergence. The third term involves the parametrization of $\{\boldsymbol{W},\boldsymbol{H},\boldsymbol{\Lambda}\}$. As in [45], the prior over $\{\boldsymbol{W}^{s_j}\}$ can be assumed flat such that each spectral component is factor-wise normalized to unit length *i.e.* $p(\boldsymbol{W}^{s_j}) = \prod_k \delta\left(\left\|\boldsymbol{W}_k^{s_j}\right\|_2 - 1\right)$ where $\left\|\boldsymbol{W}_k^{s_j}\right\|_2 = \sqrt{\sum_{f,\tau}\left(w_{k,f,\tau}^{s_j}\right)^2}$. Thus, the

conditional expectation of the negative logarithm of the second and third terms of $Eq.(4.20)$ can be express as

$$-\langle \log p(\boldsymbol{C}|\boldsymbol{W},\boldsymbol{H}) + \log p(\boldsymbol{W}) + \log p(\boldsymbol{H}|\boldsymbol{\Lambda})\rangle_{P(\boldsymbol{C}|\boldsymbol{X},\boldsymbol{\theta}')}$$
$$= \sum_{j,f,n} D_{IS}\left(\hat{p}_{j,f,n}\middle|\sum_{k,\tau,\phi} w_{k,f-\phi,\tau}^{s_j} h_{k,\phi,n-\tau}^{s_j}\right) - \sum_{j,k}\log\left(\delta\left(\left\|\boldsymbol{W}_k^{s_j}\right\|_2 - 1\right)\right)$$
$$+ \sum_{j,k,n,\phi}\left(\lambda_{k,\phi,n}^{s_j} h_{k,\phi,n}^{s_j} - \log\lambda_{k,\phi,n}^{s_j}\right) \tag{4.36}$$

where $\hat{p}_{j,f,n}$ is the j^{th} source power spectrogram estimated from $Eq.(4.22)$. The terms $\{\boldsymbol{W}^{s_j}\}$ and $\{\boldsymbol{H}^{s_j}\}$ can be estimated directly from the estimates of source one and source two obtained from the E-step. By letting $\vartheta_{f,n}^{s_j} = \sum_{k,\tau,\phi} w_{k,f-\phi,\tau}^{s_j} h_{k,\phi,n-\tau}^{s_j}$, $Eq.(4.36)$ reduces up to the constant terms to

$$\mathcal{J} \overset{c}{=} \sum_{f,n}\left(\hat{p}_{1,f,n}\,\vartheta_{f,n}^{{s_1}^{-1}} - \log\,\vartheta_{f,n}^{{s_1}^{-1}}\right) + \sum_{k,n,\phi}\lambda_{k,\phi,n}^{s_1} h_{k,\phi,n}^{s_1} - \sum_{k,n,\phi}\log\lambda_{k,\phi,n}^{s_1}$$
$$+ \sum_{f,n}\left(\hat{p}_{2,f,n}\,\vartheta_{f,n}^{{s_2}^{-1}} - \log\,\vartheta_{f,n}^{{s_2}^{-1}}\right) + \sum_{k,n,\phi}\lambda_{k,\phi,n}^{s_2} h_{k,\phi,n}^{s_2} - \sum_{k,n,\phi}\log\lambda_{k,\phi,n}^{s_2} \tag{4.37}$$

The MU approach

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}\cdot\frac{[\nabla\mathcal{J}]_-}{[\nabla\mathcal{J}]_+} \tag{4.38}$$

where $\nabla\mathcal{J} = [\nabla\mathcal{J}]_+ - [\nabla\mathcal{J}]_-$. This leads to

$$w_{k',f',\tau'}^{s_1} \leftarrow w_{k',f',\tau'}^{s_1}\left(\frac{\sum_{\phi,n}\hat{p}_{1,f'+\phi,n}\vartheta_{f'+\phi,n}^{{s_1}^{-2}} h_{k',\phi,n-\tau'}^{s_1}}{\sum_{\phi,n}\vartheta_{f'+\phi,n}^{{s_1}^{-1}} h_{k',\phi,n-\tau'}^{s_1}}\right) \tag{4.39}$$

$$h_{k',\phi',n'}^{s_1} \leftarrow h_{k',\phi',n'}^{s_1}\left(\frac{\sum_{f,\tau}\hat{p}_{1,f,n'+\tau}\vartheta_{f,n'+\tau}^{{s_1}^{-2}} w_{k',f-\phi',\tau}^{s_1}}{\sum_{f,\tau}\vartheta_{f,n'+\tau}^{{s_1}^{-1}} w_{k',f-\phi',\tau}^{s_1} + \lambda_{k',\phi',n'}^{s_1}}\right) \tag{4.40}$$

$$w_{k',f',\tau'}^{s_2} \leftarrow w_{k',f',\tau'}^{s_2}\left(\frac{\sum_{\phi,n}\hat{p}_{2,f'+\phi,n}\vartheta_{f'+\phi,n}^{{s_2}^{-2}} h_{k',\phi,n-\tau'}^{s_2}}{\sum_{\phi,n}\vartheta_{f'+\phi,n}^{{s_2}^{-1}} h_{k',\phi,n-\tau'}^{s_2}}\right) \tag{4.41}$$

$$h_{k',n',\phi'}^{s_2} \leftarrow h_{k',n',\phi'}^{s_2}\left(\frac{\sum_{f,\tau}\hat{p}_{2,f,n'+\tau}\vartheta_{f,n'+\tau}^{{s_2}^{-2}} w_{k',f-\phi',\tau}^{s_2}}{\sum_{f,\tau}\vartheta_{f,n'+\tau}^{{s_2}^{-1}} w_{k',f-\phi',\tau}^{s_2} + \lambda_{k',\phi',n'}^{s_2}}\right) \tag{4.42}$$

For the sparsity term, the update is obtained by solving $\frac{\partial}{\partial\lambda_{k',\phi',n'}^{s_i}}\langle \log p(\boldsymbol{C},\boldsymbol{\theta}|\boldsymbol{X})\rangle_{P(\boldsymbol{C}|\boldsymbol{X},\boldsymbol{\theta}')} = 0$ which leads to

$$\lambda_{k',n',\phi'}^{s_i} = \frac{1}{h_{k',\phi',n'}^{s_i}} \tag{4.43}$$

Once all the parameters are obtained, $\hat{\boldsymbol{s}}_{f,n}$ can be estimated by using the Wiener filtering $\left(\boldsymbol{\Sigma}_{\boldsymbol{s},f,n}\boldsymbol{A}_f^H\boldsymbol{\Sigma}_{\boldsymbol{x},f,n}^{-1}\right)$. The magnitude spectrogram of the separated sources will be further refined using the DSELM followed by an energy minimization method in the next stage.

4.2.2 *Deep Sparse Extreme Learning Machine*

In this section, the DSELM is developed to extract the acoustic features of the coarsely separated sources. The procedure of the deep architecture consists of two phases: 1) unsupervised feature mapping and 2) supervised feature learning. In the following, the training procedure of Sparse Extreme Learning Machine (SELM) autoencoder is described in details as it represents the basic building block of the DSELM [100].

Given a set of M training data $(\boldsymbol{X},\boldsymbol{Y}) = \{\boldsymbol{x}_i,\boldsymbol{y}_i\}_{i=1}^{M}$, where $\boldsymbol{x}_i = [x_{i1}, x_{i2}, \dots, x_{iu}] \in \mathcal{R}^{d_u}$ and $\boldsymbol{y}_i = [y_{i1}, y_{i2}, \dots, y_{iv}] \in \mathcal{R}^{d_v}$ are the training data and the corresponding target, respectively. The term x_{iu} is the magnitude spectrogram of the signal and y_{iv} is the target class. The

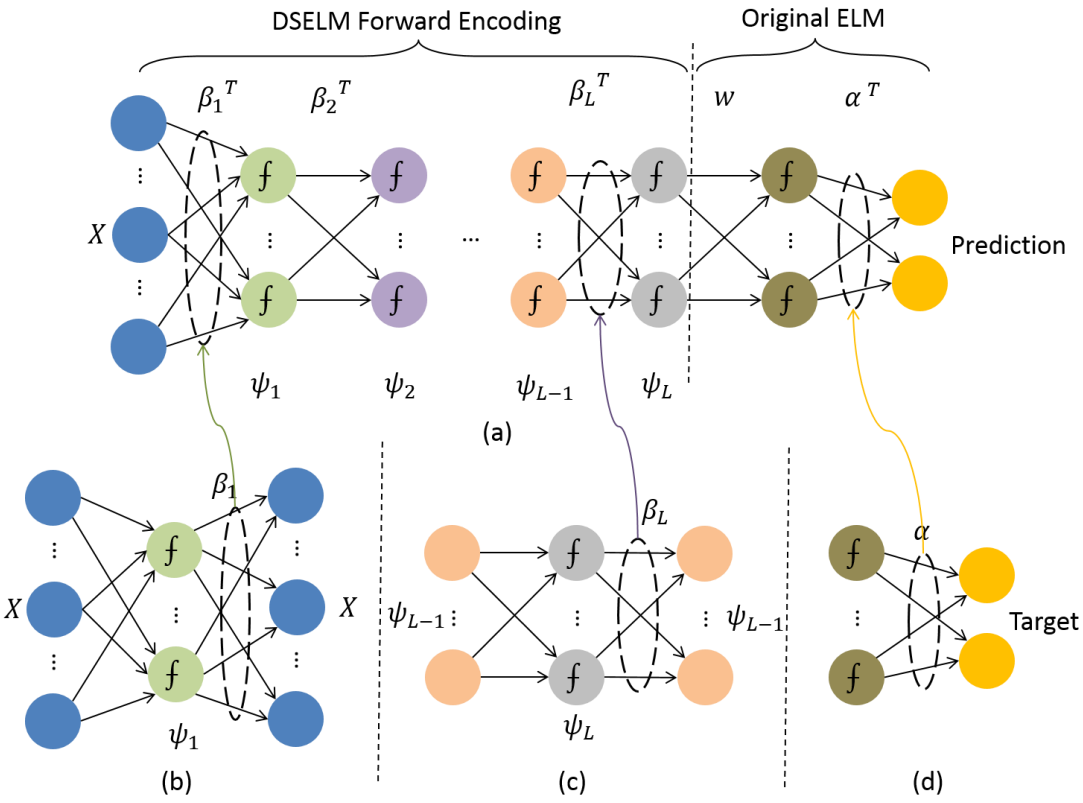


Fig. 4.2 Diagram of DSELM (a) Layer wise training of DSELM, which is consists of two phases: Forward learning followed by the original ELM classification. (b) Implementation of first hidden layer ELM based sparse auto-encoder. (c) Training procedure of L^{th} hidden layer ELM based sparse auto-encoder. (d) Analytically calculate the output weights of original ELM with labelled target using randomly initialized parameters.

parameters u and v denote the dimension of input and target, respectively. The output $f(\cdot)$ of SELM with L hidden nodes fully connect the input data to the outputs is represented by

$$f(\boldsymbol{x}_i) = \sum_{l=1}^{L} h_l(\boldsymbol{w}_l^T \boldsymbol{x}_i) \cdot \boldsymbol{\beta}_l, \qquad i = 1,2, \dots M \tag{4.44}$$

where $h(\cdot)$ is the activation function which we used is the sigmoid function

$$h_l(\boldsymbol{w}_l^T \boldsymbol{x}_i) = \frac{1}{1 + e^{-\boldsymbol{w}_l^T \boldsymbol{x}_i}} \tag{4.45}$$

$\boldsymbol{w}_l \in \mathcal{R}^{d_u}$ is the randomly generated parameters connecting input layer and the l^{th} hidden node, $\boldsymbol{\beta}_l \in \mathcal{R}^{d_v}$ is the output weight vector connecting the l^{th} hidden node and the output layer. The M equations in $Eq.(4.44)$ can be written compactly as

$$\boldsymbol{F}(\boldsymbol{\mathcal{P}}) = \boldsymbol{H}\boldsymbol{\beta} \tag{4.46}$$

where $\boldsymbol{\beta}$ is the output weight matrix, $\boldsymbol{H}$ is the $M \times L$ hidden feature mapping matrix with respect to input $\boldsymbol{X}$.

A SELM learns the parameters in two stages: 1) random feature mapping and 2) parameter solving. In the first stage, the input data is projected into a feature space with the randomly initialized parameters using the activation function $h(\cdot)$. It has been proven that any continuous function can be approximated with randomly initialized parameters [69]. Therefore, the only parameter that needs to be determined is the output weight $\boldsymbol{\beta}$. In the second stage, instead of solving the cost function with norm-2 penalty, the output weight $\boldsymbol{\beta}$ is optimized by solving the cost function with ℓ_1 penalty:

$$\underset{\boldsymbol{\beta} \in \mathcal{R}^{L \times d_v}}{\operatorname{argmin}} \qquad \|\boldsymbol{H}\boldsymbol{\beta} - \boldsymbol{F}\|^2 + \varepsilon \|\boldsymbol{\beta}\|_{\ell_1} \tag{4.47}$$

where ε is the hyperparameter, $\|\boldsymbol{\beta}\|_{\ell_1}$ is the ℓ_1 penalty, which is used to obtain compact and sparse hidden information [101, 102]. SELM autoencoder is designed to encode output to approximate the original input by minimizing the reconstruction errors.

For clear representation, $Eq.(4.47)$ can be rewritten as

$$argmin\{\varepsilon \|\boldsymbol{\beta}\|_{\ell_1} + g(\boldsymbol{\beta})\} \tag{4.48}$$

where $g(\boldsymbol{\beta}) = \|\boldsymbol{H}\boldsymbol{\beta} - \boldsymbol{F}\|^2$. To address the minimization problem, a gradient projection algorithm, which generates a sequence $\{\boldsymbol{\beta}_i\}$ is adapted

$$\boldsymbol{\beta}_0 \in \boldsymbol{\mathcal{R}}^n, \qquad \boldsymbol{\beta}_i = \boldsymbol{\beta}_{i-1} - \kappa_i \nabla g(\boldsymbol{\beta}_{i-1}) \tag{4.49}$$

where $\kappa_i > 0$ is a suitable stepsize. $Eq.(4.49)$ can be viewed as a proximal regularization of the linearized function $g(\cdot)$ at $\boldsymbol{\beta}_{i-1}$. Therefore, the non-smooth ℓ_1 regularized problem can be written as follows

$$\boldsymbol{\beta}_i = \underset{\boldsymbol{\beta}}{argmin}\left\{g(\boldsymbol{\beta}_{i-1}) + \langle \boldsymbol{\beta} - \boldsymbol{\beta}_{i-1}, \nabla g(\boldsymbol{\beta}_{i-1})\rangle + \frac{1}{2\kappa_i}\|\boldsymbol{\beta} - \boldsymbol{\beta}_{i-1}\|^2 + \varepsilon\|\boldsymbol{\beta}\|_{\ell_1}\right\}$$

$$= \underset{\boldsymbol{\beta}}{argmin}\left\{\left[g(\boldsymbol{\beta}_{i-1}) - \left(\kappa_i \nabla g(\boldsymbol{\beta}_{i-1})\right)^2\right] + \frac{1}{2\kappa_i}\|\boldsymbol{\beta} - (\boldsymbol{\beta}_{i-1} - \kappa_i \nabla g(\boldsymbol{\beta}_{i-1})\|^2 + \varepsilon\|\boldsymbol{\beta}\|_{\ell_1}\right\} \quad (4.50)$$

The constant terms can be ignored, thus $Eq.(4.50)$ can be rewritten as

$$\boldsymbol{\beta}_i = \underset{\boldsymbol{\beta}}{argmin}\left\{\frac{1}{2\kappa_i}\|\boldsymbol{\beta} - (\boldsymbol{\beta}_{i-1} - \kappa_i \nabla g(\boldsymbol{\beta}_{i-1})\|^2 + \varepsilon\|\boldsymbol{\beta}\|_{\ell_1}\right\} \quad (4.51)$$

Let $\nabla g = 2\boldsymbol{H}^T(\boldsymbol{H}\boldsymbol{\beta} - \boldsymbol{F})$ denotes the gradient of $g(\cdot)$ and $\boldsymbol{\mathcal{L}} \coloneqq \boldsymbol{\mathcal{L}}(g) = 2(\boldsymbol{F}^T\boldsymbol{F})$ denotes the Lipschitz constant of $\nabla g(\cdot)$. Define an operator $\boldsymbol{\varkappa}: \mathcal{R}^n \longrightarrow \mathcal{R}^n$, $\kappa = 1/\boldsymbol{\mathcal{L}}(g)$. The computation of output weight $\boldsymbol{\beta}$ can be represented as follows:

$$\boldsymbol{\beta}_i = \boldsymbol{\varkappa}(\boldsymbol{\varrho}_i) \quad (4.52)$$

where $\boldsymbol{\varrho}_i$ is a new point which is a specific linear combination of the previous two points $\{\boldsymbol{\beta}_{i-1}, \boldsymbol{\beta}_{i-2}\}$. The implementation details are as follows:

(a) Calculate the Lipschitz constant $\boldsymbol{\mathcal{L}}$ of the gradient of smooth convex function $\boldsymbol{\nabla g}(\cdot)$

(b) Take the initial value $\boldsymbol{\varrho_1 = \beta_0 \in \mathcal{R}^n, \kappa_1 = 1}$

(c) For $\boldsymbol{i \geqslant 1}$, compute

$$\boldsymbol{\beta}_i = \boldsymbol{\varkappa}(\boldsymbol{\varrho}_i) = \underset{\boldsymbol{\beta}}{argmin}\left\{\frac{1}{2\kappa_i}\left\|\boldsymbol{\beta} - \left(\boldsymbol{\varrho}_i - \kappa_i \nabla g(\boldsymbol{\varrho}_i)\right)\right\|^2 + \varepsilon\|\boldsymbol{\beta}\|_{\ell_1}\right\} \quad (4.53)$$

$$\kappa_{i+1} = \left(1 + \sqrt{1 + 4\kappa_i^2}\right)/2 \quad (4.54)$$

$$\boldsymbol{\varrho}_{i+1} = \boldsymbol{\beta}_i + \left(\frac{\kappa_i - 1}{\kappa_{i-1}}\right)(\boldsymbol{\beta}_i - \boldsymbol{\beta}_{i-1}) \quad (4.55)$$

The output weight $\boldsymbol{\beta}$ can be obtained by computing iterative steps. As a building block, the trained SELM can be used to construct DSELM, which is able to extract higher-level feature representations of input data by layer-wise comparison and the learned information will be classified to its corresponding domain by stacking an original ELM classifier at the top.

The coarsely separated sources obtained from first stage are updated by using a joint energy minimization method based on the trained DSELM to improve the separation performance in the second stage.

4.2.3 *Energy Minimization Method*

The goal of this stage is to estimate a TF mask, which is obtained by minimizing the joint energy function. In this stage, the DSELM is assumed to be fully trained and is act as prior knowledge

for refining the separated sources. For simplicity, we assume the noise is negligible. Thus (4.1) can be expressed as $\tilde{x}(t) = u\tilde{s}_1(t) + v\tilde{s}_2(t)$. Speech and music signals are approximately Window Disjoint Orthogonal (WDO) in the magnitude spectrogram domain *i.e.* $s_{1,f,n}s_{2,f,n} \approx 0$. Hence, to refine the estimation of the sources, we consider the magnitude spectrogram. The magnitude spectrogram of mixture given the initial estimate of the sources $\dot{s}_{j,f,n}$ from the NMF-2D can be expressed as $\bar{x}_{f,n} = \bar{u}\bar{s}_{1,f,n} + \bar{v}\bar{s}_{2,f,n}$ where $\bar{x}_{f,n} = |x_{f,n}|$, $\bar{s}_{j,f,n} = |\dot{s}_{j,f,n}|$, and $\bar{u}$ and $\bar{v}$ are the magnitude of the mixing gains, respectively. The separation problem can now be formulated as finding the unknown parameters $\rho = \{\bar{s}_{1,f,n}, \bar{s}_{2,f,n}, \bar{u}, \bar{v}\}$ by minimizing the following joint energy function

$$\{\check{s}_{1,f,n}, \check{s}_{2,f,n}, \check{u}, \check{v}\} = \underset{\bar{s}_{1,f,n}, \bar{s}_{2,f,n}, \bar{u}, \bar{v}}{argmin}\ \Omega(\rho, \bar{x}_{f,n}) \tag{4.56}$$

$$\Omega(\rho, \bar{x}_{f,n}) = \sum_j \Omega_j(\bar{s}_{j,f,n}) + \varsigma \mathfrak{e}(\rho, \bar{x}_{f,n}) + \xi \sum_j \varphi(\bar{s}_{j,f,n}) \tag{4.57}$$

where ς and ξ are the regularization parameters which are chosen experimentally. The term $\{\check{s}_{1,f,n}, \check{s}_{2,f,n}, \check{u}, \check{v}\}$ represents the optimum solution that minimizes the energy function. $\Omega_j(\cdot)$ is the least square cost function defined as

$$\Omega_1(\bar{s}_{1,f,n}) = \left(1 - \Gamma_1(\bar{s}_{1,f,n})\right)^2 + (\Gamma_2(\bar{s}_{1,f,n}))^2 \tag{4.58}$$

$$\Omega_2(\bar{s}_{2,f,n}) = (\Gamma_1(\bar{s}_{2,f,n}))^2 + (1 - \Gamma_2(\bar{s}_{2,f,n}))^2 \tag{4.59}$$

where $\Gamma_\iota(\cdot), \iota = 1,2$ is the ι^{th}output of the DSELM which indicates the proportional to the probabilities of each source respectively for a given frame of magnitude spectrum. Therefore, if the input data is from source one only, the expectation of $\Gamma_1(\bar{s}_{1,f,n}) = 1$ and $\Gamma_2(\bar{s}_{1,f,n}) = 0$, thus, the expectation of $\Omega_1(\bar{s}_{1,f,n})$ is equal to 0. The second term of $Eq.(4.57)$ is an error function, which denotes the differences between the coarsely separated sources and the mixed spectrum:

$$\mathfrak{e}(\rho, \bar{x}_{f,n}) = \frac{1}{2}\left\|\bar{u}\bar{s}_{1,f,n} + \bar{v}\bar{s}_{1,f,n} - \bar{x}_{f,n}\right\|_2^2 \tag{4.60}$$

Finally, an energy function $\varphi(\bar{s}_{j,f,n}) = \left(\min(\bar{s}_{j,f,n}, 0)\right)^2$ is selected in this work to ensure the nonnegativity of the reconstructed spectrum.

To minimize the joint energy function, we need to calculate the derivative with respect to the input data. The derivative of a sigmoid activation function $h(\cdot)$ is given as $dh(\cdot) = h(\cdot) \circ \left(1 - h(\cdot)\right)$, where $'\circ'$ is element-wise multiplication. Define the derivative of the input vector $\bar{\boldsymbol{s}}_{f,n}$ with respect to Γ_i as $\frac{\partial \Gamma_i}{\partial \bar{\boldsymbol{s}}_{f,n}} = \boldsymbol{\hbar}_{1,i}$, then

$$\boldsymbol{\hbar}_{\hbar,i} = \boldsymbol{\beta}_{\hbar} \circ \left(\boldsymbol{\hbar}_{\hbar+1,i} \circ \boldsymbol{H}_{\hbar} \circ (1 - \boldsymbol{H}_{\hbar})\right) \tag{4.61}$$

where $\boldsymbol{\beta}_{\hbar}$is the weight of $\hbar^{th}$ hidden layer and $\boldsymbol{H}_{\hbar}$ is the output value of $\hbar^{th}$ hidden layer for $\hbar = 1, \dots, \mathcal{K}$.

The mixing gains $\bar{u}$ and $\bar{v}$ are initialized by ℓ_2-norm of the source $\bar{s}_{1,f,n}$ and the source $\bar{s}_{2,f,n}$ divided by the ℓ_2 -norm of the mixed signal $\bar{x}_{f,n}$, respectively. The parameters $\{\check{s}_{1,f,n}, \check{s}_{2,f,n}, \check{u}, \check{v}\}$ can be obtained and utilized as a mask to regenerate the spectrogram of each source spectra

$$s^*_{1,f,n} = \frac{\left(\check{u}\check{s}_{1,f,n}\right)^2}{\left(\check{u}\check{s}_{1,f,n}\right)^2 + \left(\check{v}\check{s}_{2,f,n}\right)^2} \circ x_{f,n} \tag{4.62}$$

$$s^*_{2,f,n} = \frac{\left(\check{v}\check{s}_{2,f,n}\right)^2}{\left(\check{u}\check{s}_{1,f,n}\right)^2 + \left(\check{v}\check{s}_{2,f,n}\right)^2} \circ x_{f,n} \tag{4.63}$$

The term $s^*_{j,f,n}$ represents the j^{th} fine-tuned estimated source spectrogram using the energy minimization method. Due to the linearity of the STFT, the inverse-STFT can be used to transform it to the time domain [103].

Table 4.1 Proposed approach

1. Initialize $W^{s_j}_{k,f,\tau}$ and $H^{s_j}_{k,\phi,n}$, $j = 1,2$
2. Generate the pseudo mixture $\tilde{x}_2(t)$ as in $Eq.(4.5)$.
3. Apply the STFT on the mixture signal.
4. E-step: Compute $\hat{p}_{j,f,n}$ and $\hat{s}_{f,n}$ using $Eq.(4.22)$ and $Eq.(4.24)$.
5. M-step: Compute $\boldsymbol{A}_f$, $\Sigma_{\boldsymbol{b},f}$, $w^{s_j}_{k,f,\tau}$, $h^{s_j}_{k,\phi,n}$and $\lambda^{s_j}_{k,\phi,n}$, using $Eq.(4.23) - Eq.(4.31)$.
6. Normalize $w^{x}_{k,f,\tau} = w^{x}_{k,f,\tau} / \sqrt{\sum_{f,k,\tau}\left(w^{x}_{k,f,\tau}\right)^2}$
7. Repeat E-step and M-step, and the normalization until convergence is achieved
8. Optimize the parameters of DSELM
9. Compute the proportional to the probabilities of the coarsely separated sources for a given frame of magnitude spectrum with trained deep structure
10. Calculate $\check{s}_1, \check{s}_2, \check{u}, \check{v}$ by minimizing cost function $Eq.(4.56)$.
11. Generate masks as $Eq.(4.62), Eq.(4.63)$ to recover sources
12. Perform inverse STFT with dual synthetic window to estimate $\tilde{s}_1(t)$, and $\tilde{s}_2(t)$.

4.3 Results and Discussions

The performance of the proposed separation system is evaluated with the Itakura-Saito NMF (IS-NMF) [104] combined with clustering algorithm, the IS-NMF-2D algorithm [42] and the DNN with proposed GEM-MU based NMF-2D algorithm. IS-NMF has been previously shown to correctly capture the semantics of audio and is better suitable to the representation than the standard NMF [104]. The recently proposed IS-NMF-2D provides promising separation result for music mixture and it is deemed as a competitive approach to solve the separation problem [42]. We also compare our proposed approach with DNN to demonstrate the separation performance especially in term of computational efficiency. DNN with pre-training initialization is an artificial neural network model for representing the structure details and maintaining the key abstract information that can be used for classification and regression [58, 61-63, 65, 66, 84].

4.3.1 *Experimental data and evaluation criteria*

Experiments are conducted on the 'CHiME' database [105], which consists of 34 speakers speaking 500 utterances each. For training, 400 utterances of a speaker are selected to form the training data while 20 utterances different from the training data are randomly selected to form the mixture with guitar music selected from RWC database [106]. The separation performance is evaluated in terms of Signal-to-Distortion Ratio (SDR), which is a measure of the quality of a desired signal from a communications device.

4.3.2 *General system design*

1) Data pre-processing

For simplicity, we assume that the observed signal is the summation of two source signals with fixed mixing gains over time. It is necessary to explicitly normalize the source to unit variance. The training and testing data is the magnitude spectrogram of the clean sound. Both the training sources and mixtures are sampled at 16 kHz and transformed to TF domain using a Hamming window with 512 points length with 50% overlap and 256-point Fast Fourier Transform (FFT) to form the spectra.

2) Parameter initialization

In the coarse separation step, several parameters need to be determined in advance. The parameter time-delay δ must satisfy

$$|\omega_{max}\delta_{max}| < \pi \tag{4.64}$$

where $\omega_{max} = 2\pi f_{max}/f_s$, δ_{max} is the maximum time delay, f_{max} is the maximum frequency present in the sources and f_s is the sampling frequency. As long as the delay parameter is less than δ_{max}, this will avoid the phase ambiguity. In our cases, the signals are acquired using sampling frequency $f_s = 16kHz$. Some signals such as singing voice are characterized by high frequency band about $4kHz$. Thus the choice for δ is limited to either 1 or 2. The weight delay parameter γ can be determined during the experiments. Fig. 4.3 shows the SDR results with different γ. The highest SDR can be obtained between 0.1 and 0.2. Thus, we set the value of γ to 0.15.

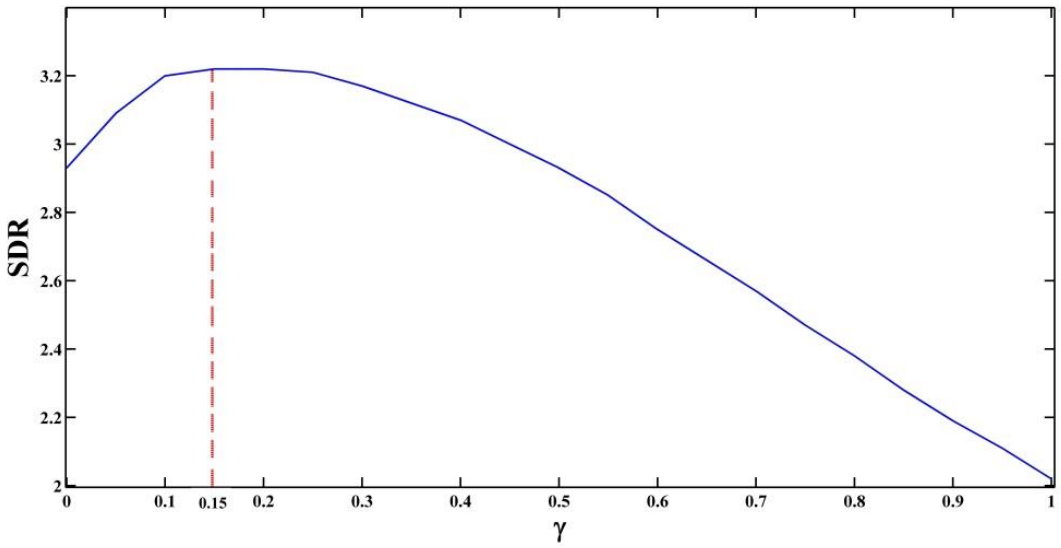


Fig. 4.3 SDR with respect to γ

The convolutive parameters (τ and ϕ) and the required number of frequency basis K will be optimized from the NMF-2D. For all kinds of mixtures, the parameters are selected within the range of $\tau = \{0, \dots, 10\}, \phi = \{0, \dots, 10\}$ and $K = \{0, \dots, 10\}$. For different τ, ϕ, and K, we estimated the SDR between the source signal and its approximation in order to evaluate the factorization performance. The obtained result shows that robust and reliable SDR performance can be achieved by setting the convolutive parameters $\tau = 9$, $\phi = 1$, and frequency basis $K = 2$.

3) Fine-tuning the coarsely separated sources

The signal in the TF domain obtained from the GEM-MU based NMF-2D is fed into the trained DSELM to calculate the proportion followed by the energy minimization method.

To determine the hyperparameter ε of the regularization term of $Eq.(4.47)$, we have tested all the experiments with haperparameter ranges from 0 to 1 with incremental step of 0.01. The obtained results in terms of the SDR show that the optimum range of hyperparameter is [0.14, 0.23] *i.e.* 0.185 ± 0.045. Thus the haperparameter has been set to 0.185. In addition, the

hyperparameters of the regularization term in the energy function are fixed at $\varsigma = 0.3$ and $\xi = 0.5$ which are obtained from experiments.

4.3.3 *DSELM spectral model*

1) Architecture:

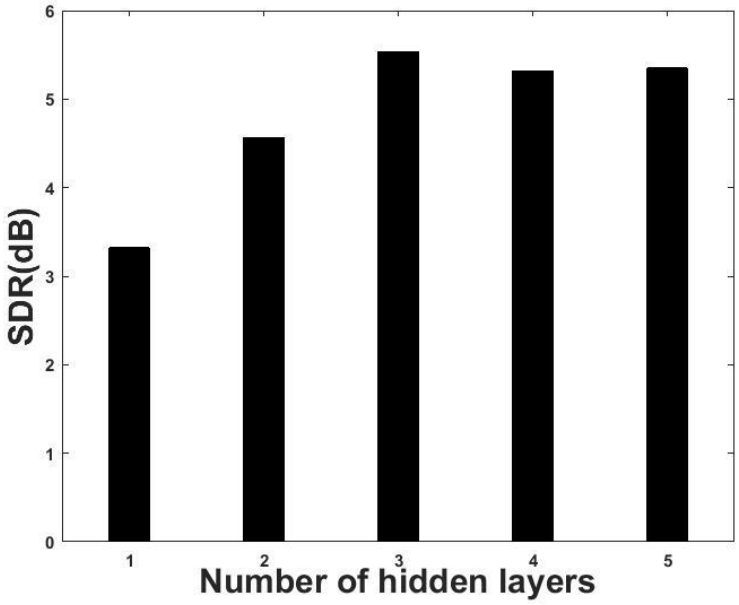


Fig. 4.4 SDR with respect to number of hidden layers

The architecture of the DSELM is identical to the Multi-Layer Perceptron (MLP). The number of hidden layers and the number of hidden nodes in each layer may vary. We have compared the performance in terms of number of hidden layers and the result is shown in Fig.4.4. In the experiments, we use the same training data to train DSELM with different number of hidden layers and the same separated sources obtained from the proposed GEM-MU based NMF-2D algorithm for testing. One hidden layer framework denotes the original ELM. From Fig. 4.4, we can see that adding a second hidden layer significantly improve the result over using a single hidden layer framework. The improvement is more significant with one more hidden layer added. It is interesting to note that the performance tends to degrade with four and five hidden layers. This may be due to the ceiling effects that more hidden layers are difficult to further improve the performance. Based on the experiments conducted, by considering the performance and computational complexity, a three-hidden layer DSELM is selected as small number of network parameters facilities fast training with reasonable good performance. The architecture of DSELM used in all experiments is 257-300-150-200-2, which indicates that the number of hidden node is 300, 150 and 200, respectively and the output node is 2.

2) *Training phase*

The DSELM is trained with layer-wise pre-training method where the hidden layers are added incrementally.

In the beginning, the magnitude spectrum is transferred into a space with randomly initialized input parameters. Unsupervised feature learning is performed to calculate the output parameters analytically. Once the output parameters are obtained, they will be used to initialize the weights between the input layer and the first hidden layer of DSELM and will be kept and need not to be fine-tuned. The output of the first hidden layer is then calculated with the fixed parameters and substituted by a new hidden and output layer to form a new one-hidden-layer network which will be trained using the same method. Once all the parameters of the deep structure are initialized, an original ELM will be trained and stacked at the top of the network to form the whole system. At last, the derivative of the output with respect to the input data is calculated to minimize the energy function to improve the coarsely separated sources obtained from the GEM-MU based NMF-2D algorithm.

4.3.4 *Performance measure*

In this section, the computing effectiveness and efficiency of the proposed approach is compared with a MLP trained with back-propagation algorithm and a DNN with Restricted Boltzmann Machine (RBM) pre-training. MLP is selected as a baseline of the deep architecture as higher-order correlations between input can be extracted using the hierarchical structure. However, MLP is easily stuck at local minima. Compared with MLP, DNN has achieved promising improvement [107]. However, DNN is augmented with high computational complexity and time consumption. The alternative is to use the aforementioned DSELM. We select 400 utterances from male and female each and guitar and bass music to form the training data to train the deep frameworks while 50 utterances different from the training data are selected to test the train models [106]. While training MLP and DNN, the input data is normalized to zero mean and unit variance. For MLP, we use 50 epochs for the back-propagation fine-tuning. For DNN, we use 50 epochs for pre-training and 50 epochs for the whole network fine-tuning. The learning rate that we used is 0.001 for the first Gaussian-Bernoulli RBM and 0.01 for the above Bernoulli-Bernoulli RBM.

The result presented in Table 4.2 illustrates the comparison of the DSELM with MLP and DNN based on the training time and classification accuracy. It should be mentioned that with the same training data, the execution time of DSELM is obviously much faster than that of MLP and DNN. This is mainly attributed to the training simplicity of DSELM without progressive fine-tuning. This is contrasted with MLP and DNN as MLP needs to be trained with back-

propagation algorithm iteratively and DNN needs to be fine-tuned multiple times before the network can be readily used. Referring to Table II, it is generally noted that DSELM not only outperforms MLP and DNN in training time, but also the classification accuracy. For all types of mixtures, the MLP and DNN delivers average accuracy of 93.57% ± 0.4% and 97.02% ± 0.2% while the DSELM with an average accuracy of 98.78% ± 0.2%.

Table 4.2 Comparison in terms of training time and classification accuracy

	Method	Training Time (s)	Classification Accuracy (%)
Guitar and Male	MLP	3335	93.8±0.4
	DNN	5667	97.4±0.2
	DSELM	**8.84**	**98.7±0.3**
Guitar and Female	MLP	3146	94.1±0.5
	DNN	5326	97.2±0.2
	DSELM	**8.39**	**99.2±0.2**
Bass and Male	MLP	3261	92.5±0.4
	DNN	5438	96.4±0.3
	DSELM	**8.21**	**98.3±0.3**
Bass and Female	MLP	3094	93.9±0.4
	DNN	5296	97.1±0.2
	DSELM	**8.36**	**98.9±0.2**

4.3.5 *Speech separation performance*

Apart from computational complexity, we also consider SDR to measure the separation performance. To evaluate the proposed approach, IS-NMF combined with clustering algorithm [108], IS-NMF-2D [42] and DNN are selected to compare with. To create the training set, we randomly choose 200 female utterances and 200 male utterances from the 'CHiME' database [105]. To create the test set, 25 utterances of a female speaker and 25 utterances of a male speaker are chosen. The selected utterances are mixed with guitar and bass music at 0 dB, respectively. For IS-NMF combined with clustering algorithm approach, the mixed signal is

factorized into $И = 2, 4, \ldots, 10$ components followed by a grouping method which is used to cluster the $И$ components to each source [108]. The best value of SDR of each case of the $И$ different configurations is retained for comparison. For IS-NMF-2D [42], the spectral and temporal features of the mixed signal are factorized in nonuniform TF domain produced by the gammatone filterbank. The obtained features are used to generate a binary mask to separate the mixed signal. With respect to the DNN, a 3 hidden-layer network is selected and trained using RBM pre-training and back-propagation algorithm. For fair comparison, the same separated sources obtained from pseudo stereo mixture GEM-MU based NMF-2D algorithm are utilized as the input data to the trained DNN followed by the energy minimization method.

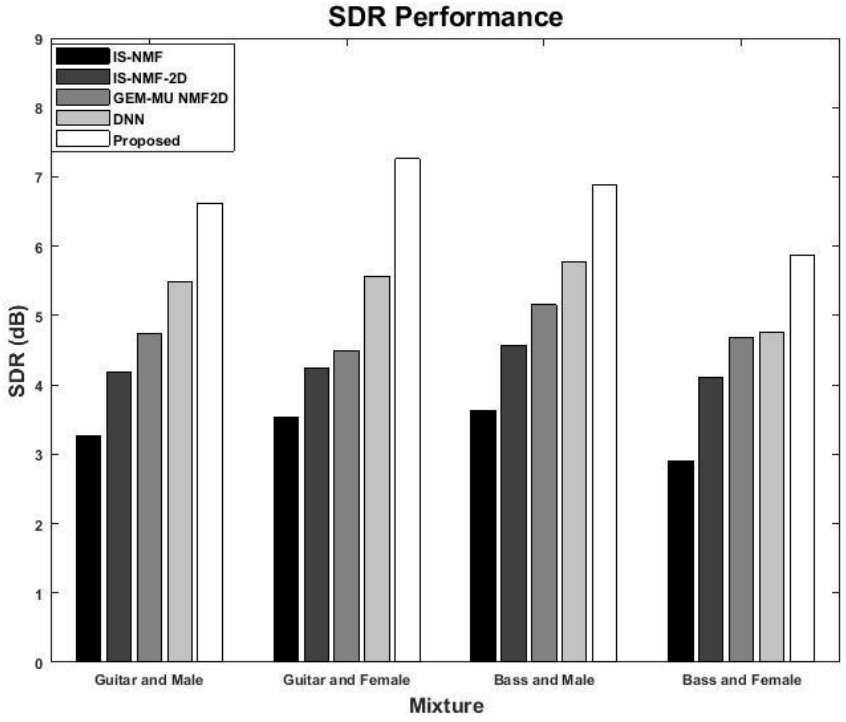


Fig. 4.5 SDR performance

Fig. 4.5 shows the comparison result of the proposed method with the IS-NMF combined with clustering algorithm, IS-NMF-2D algorithm, GEM-MU based NMF-2D and DNN. Judging from the SDR for all type of mixture, the IS-NMF algorithm gives an average SDR of 3.28 dB, the IS-NMF-2D algorithm 4.23 dB, the GEM-MU based NMF-2D algorithm 4.88 dB, and the DNN 5.59 dB. On the other hand, the average SDR of the proposed approach is 6.86 dB. The proposed method clearly performs better than other approaches. The separation performance of guitar and female speech mixture is the highest among all mixtures, which is 7.69 dB. Compared with pseudo-stereo mixture GEM-MU based NMF-2D algorithm, the result of the proposed approach has improved by 3.03 dB. The reason is likely because the contribution of the music source of each frame is alleviated during the fine-tuning stage. As the mixed signal

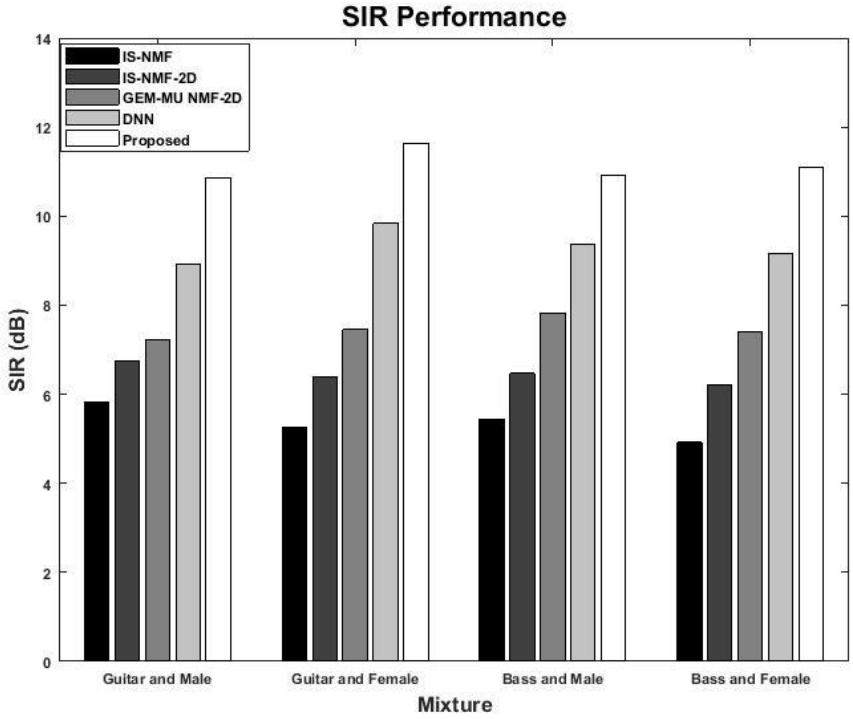


Fig. 4.6 SIR performance

cannot be separated thoroughly using pseudo-stereo mixture GEM-MU based NMF-2D algorithm, thus the separated sources still contain both speech and music. To refine the coarsely separated speech, we need to calculate the proportion of each source, which can be performed by DSELM. The output of DSELM measures the proportion of the source that contained in the input frame. Thus, the coarsely separated speech can be updated every iterations to reduce the music contribution by using the energy minimization function.

In order to evaluate the efficiency of the energy minimization method, we use the SIR as a measure of interefernce rejection. The SIR performance is shown in Fig. 4.6. It can be seen that our proposed pseudo stereo mixture GEM-MU based NMF-2D algorithm (mixture separation stage) gives releatively better performance compared with IS-NMF and IS-NMF-2D for all mixtures. This is because with pseudo stereo mixture, it becomes possible to uniquely estimate the mixing gains and therefore improves the estimation of the sources. It is also noted that the performance is improved significantly during the fine-tuning stage, especially with proposed DSELM. The reason is mainly because the remaining music portion of the coarsely separated speech signal is decreased by using the deep structure.

To further analyze the performance, we have plotted the mixture, original sources, the separated sources from pseudo stereo mixture GEM-MU based NMF-2D algorithm and the final estimated sources from the proposed approach. The results are shown in Fig. 4.7. Fig. 4.7(a)

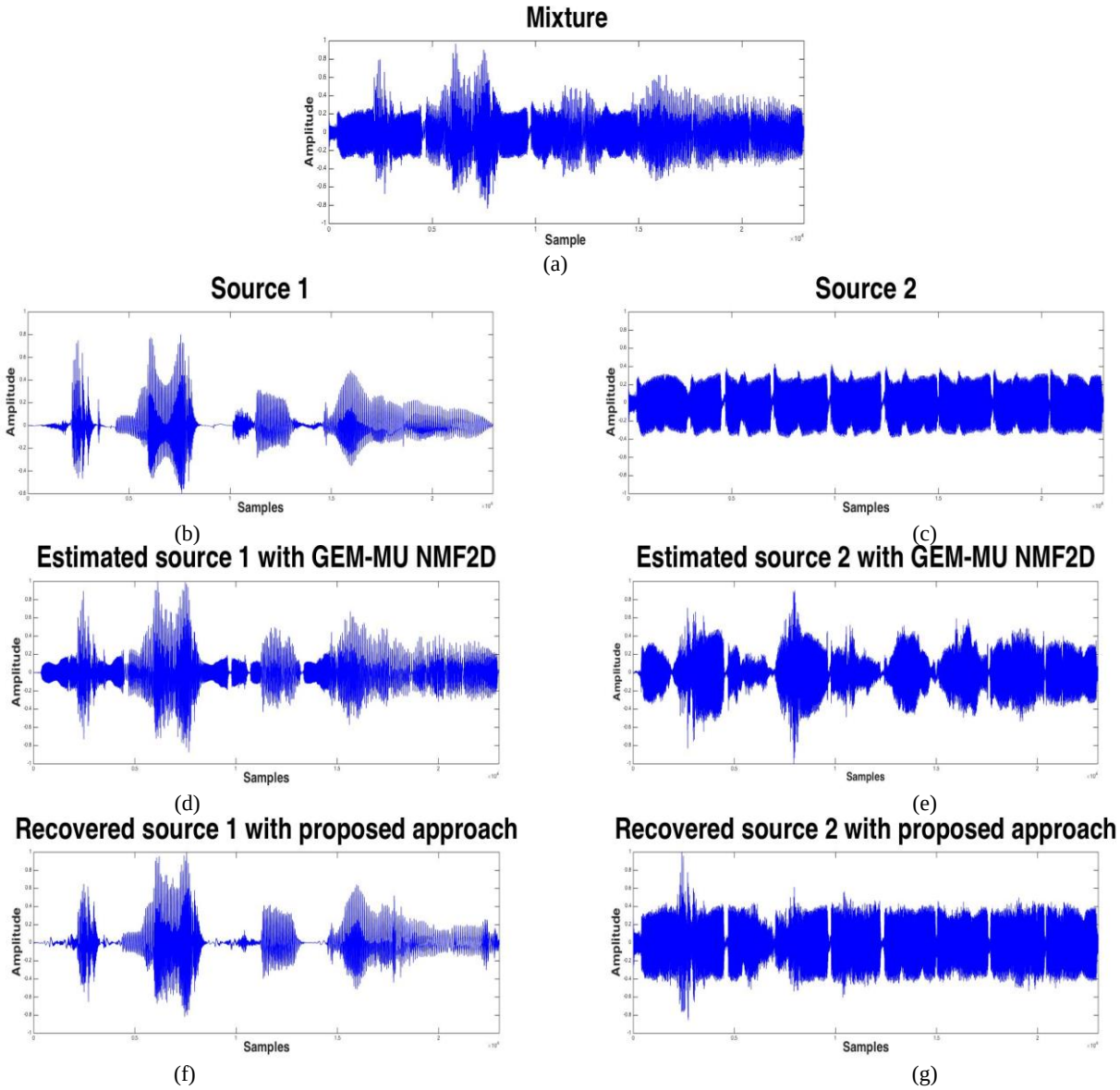
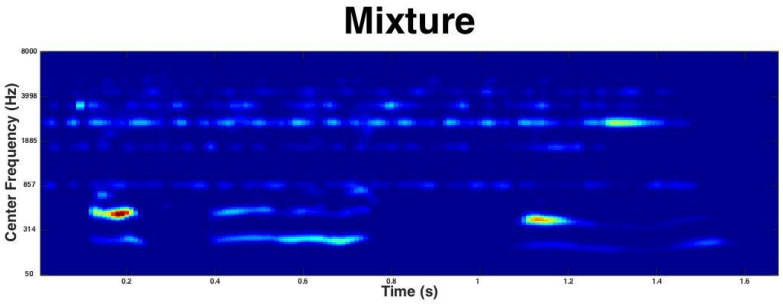


Fig. 4.7 Time domain separation results. (a) Mixture of guitar and female utterance. (b) Female utterance (c) Music. (d) Female utterance initialized with GEM-MU algorithm (e) Guitar music initialized with GEM-MU algorithm. (f) Improved separation result of female utterance using DSELM with Energy minimization function. (g) Improved separation result of music using DSELM with energy minimization function.

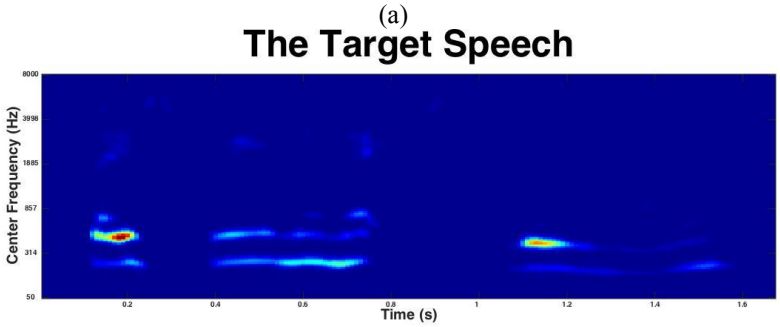
denotes the mixture of guitar music and female utterance. Fig. 4.7(b) and (c) are the original speech and music, respectively which we want to separate from the mixture. Fig. 4.7(d) and (e) are the coarsely separated speech and music using the proposed pseudo-stereo mixture GEM-MU based NMF-2D algorithm. Fig. 4.7(f) and (g) are the final estimated speech and music using DSELM with energy minimization method. Compare (d) and (f) in Fig. 4.7, it is clearly shown that the proposed approach exhibit good estimation of the speech. The music contribution is alleviated by using the DSELM with energy minimization. The coarsely separated speech and music result in SDR of 4.49 dB and 5.41dB while the final estimated sources give the SDR of 7.26 dB and 6.85 dB, respectively.

4.3.6 *Effects of pseudo-stereo mixture GEM-MU based NMF-2D algorithm*

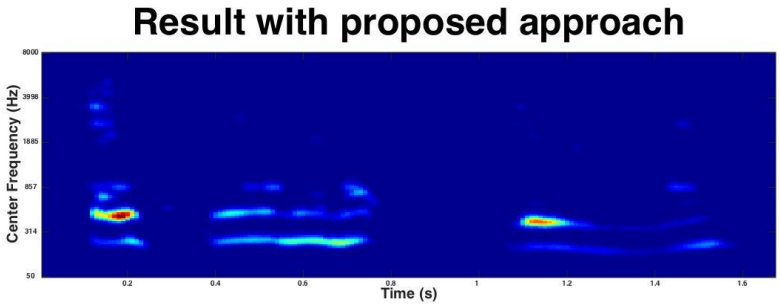
The proposed approach contains two main stages *i.e.* the pseudo stereo mixture with GEM-MU based NMF-2D coarse separation, and the DSELM with energy minimization refining. As



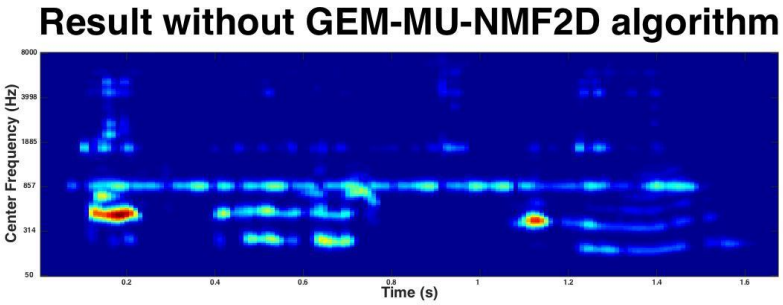
(a)



(b)



(c)



(d)

Fig. 4.8 Effects of pseudo stereo mixture GEM-MU based NMF-2D algorithm. . (a) Mixture of female and music. (b) Female speech. (c) Separation results of the proposed method. (d) Separation result in the absence of the pseudo-stereo mixture GEM-MU based NMF-2D.

the second stage is dependent on the accuracy of the coarsely separated sources, a question arises as to how important the first stage is. In this section, the approach with and without the part of the pseudo-stereo mixture GEM-MU based NMF-2D separation is evaluated. A female utterance is selected to generate a mixture with a piece of guitar music. For comparison purpose, the original mixture and its pseudo stereo mixture will be fed directly into the proposed system. The results are shown in Fig 4.8. In sharp contrast with Fig. 4.8(b), it is clearly seen that without the pseudo stereo mixture GEM-MU based NMF-2D separation (depicted in Fig. 4.8(d)), the separation result reveals that the output still contains mixture from both sources. This is mainly attributed to the features of each signal that are extracted from the mixed signal which have not been properly separated and therefore render the fine-tuning stage ineffective. On the other hand, Fig. 4.8(c) shows the separation result when the pseudo stereo mixture GEM-MU based NMF-2D algorithm is included in the system. It is noted that most of the music contribution has been removed while the speech still remains. This is likely because the coarse separation stage generates relatively good input data for DSELM, which enables more accurate decision to alleviate the proportion of interference. The proposed approach with the pseudo stereo mixture GEM-MU based NMF-2D has led to good level of separation performance with SDR of 5.94dB for the speech. On the other hand, without resourcing to using the pseudo-stereo mixture GEM-MU based NMF-2D, the SDR only reaches to 1.86dB.

4.3.7 *Different SNR conditions*

In this section, experiments are conducted to evaluate the effectiveness of proposed approach under different SNR conditions. To generate the mixtures under different SNR, we randomly select 25 utterances from male and female speakers. The selected utterances are mixed with guitar and bass music separately at SNR ranging from -6dB to 6dB with an increment of 3dB. In comparison with the proposed approach, we have included the pseudo stereo with GEM-MU based NMF-2D and DNN based approach. The separation results are shown in Fig. 4.9. In general, as the SNR increases the SDR also increases.

It is notable that our proposed approach outperforms the pseudo stereo mixture with GEM-MU based NMF-2D algorithm and the DNN approach for all input in different SNR conditions range from -6dB to 6dB. The average improvement compared with pseudo stereo mixture GEM-MU based NMF-2D is closed to 20%. This is due to the reduction of remaining music contribution in the coarsely separated speech source. In addition, the distortion part of speech is recovered by using the mask, which is obtained from the energy minimization method. The DNN based approach also performs with relative acceptable results compared with the DSELM. However,

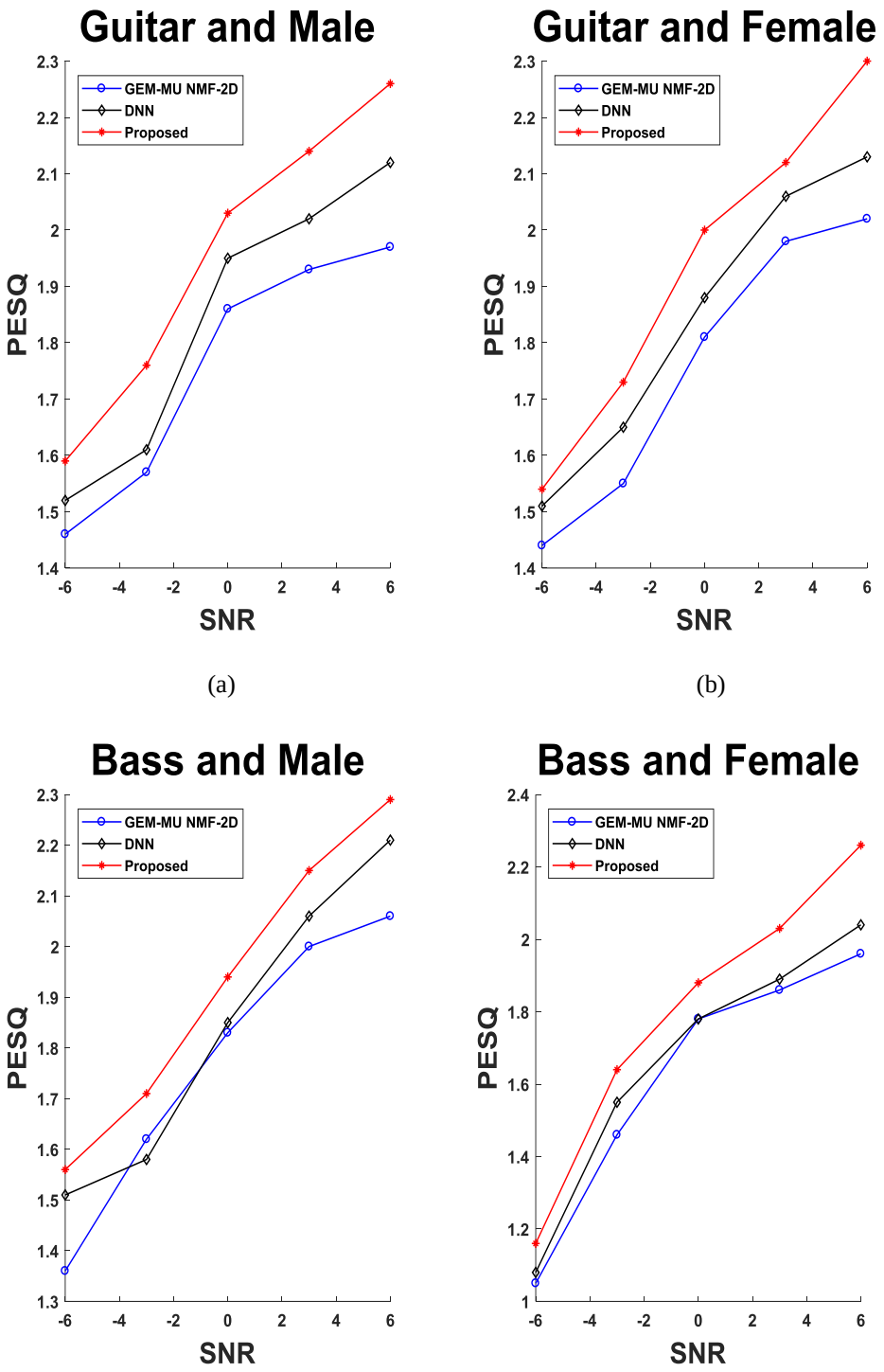


Fig. 4.9 PESQ score of speech under different SNR of mixtures

it is interesting to note that the improvement of DNN is less than that of DSELM, which shows that the DSELM is more efficient.

4.4 Conclusions

The impetus behind this research is that although matrix factorization based approaches have acquired considerable success in tackling single channel audio separation problem, its

performance level can be improved. In the proposed approach, a pseudo-stereo mixture GEM-MU based NMF-2D algorithm has been proposed. The artificial stereo signals provide extra information in increasing the identifiability of the mixing gain and reducing the ambiguity of estimating the sources. In addition, a deep sparse extreme learning machine with energy minimization function has been developed to further improve the separation performance by fine-tuning the estimation of the sources. Experiments have been conducted to evaluate our proposed approach. Obtained results have shown that the proposed approach offers considerably better separation performance compared with conventional methods.

Chapter 5. Deep Neural Networks ensemble system for Single Channel Audio Separation

From a classification viewpoint, a binary mask can be estimated by utilizing machine learning classifiers. This formulation leads to the so called classification based source separation. Within the framework of the computational auditory scene analysis (CASA), classification based speech separation, especially Ideal Binary Mask (IBM) has the optimum performance in Time-Frequency (TF) units. In this Chapter, a novel approach is proposed to estimate the IBM through binary classification. For classification, both features and classifiers are important. In this chapter, we study the performance of various acoustic features in terms of IBM estimation. In other words, the feature generalization issue. To explore a complementarity and discriminative feature set, a 'deep' and 'wide' learning system is proposed. The 'deep' and 'wide' system first extracts abstract representations from raw acoustic features and then the features are embedded and classified to its corresponding domain to estimate a TF binary mask to separate the pre-mixed signals. For the classifier, our previous work demonstrated that Extreme Learning Machine (ELM) is a good choice due to the high classification performance [68].

This chapter is organized as follows: Section 5.1 introduces the background of IBM and classification-based source separation. Section 5.2 provides overview of the proposed system and feature extraction. The detailed proposed approach is described in Section 5.3. Experimental results and comparison with other methods are presented in section 5.4. Section 5.5 concludes the chapter.

5.1 Background

As mentioned before, the IBM has been suggested as a primary goal within the framework of the CASA. The IBM is a TF mask that can be estimated from a mixed signal. Specifically, for each TF representation of a mixture, the corresponding mask element in IBM is set to 1 (target domain) if the signal-to-noise ratio (SNR) is greater than a local SNR criterion. Otherwise, the mask element is set to 0 (interference domain).

A series of studies have shown that the speech intelligibility can be improved by using a well-estimated IBM. In the speech recognition and separation community, many acoustic features have been explored, such as Mel-Frequency Cepstral Coefficients (MFCC) which are the coefficients of the Mel-cepstrum which is the cepstrum computed on the Mel-bands (scaled to human ear) instead of the Fourier spectrum [109]. However, simply concatenating acoustic features as the input to the classifier may lead to poor classification, especially when the feature

distribution of the test set is different from those in the training set. Therefore, a learning system that is capable of extracting robust and discriminative features from raw acoustic features is needed. Motivated by the success of Deep Neural Network (DNN), Wang *et al.* [58] first introduced DNN to perform binary classification for speech separation and the separation performance significantly outperforms earlier separation methods. However, their method did not address the redundancy problem, which may affect the efficiency and separation performance. Typically, a learning machine uses the concatenation of different features instead of an individual feature as its input to estimate the IBM. Different combinations of features may provide complementary information that can further improve the performance. Most researches are focused on single DNN to perform the feature learning. However, different initializations of DNN can give rise to different decision features, even if all other parameters are kept constants [110]. Therefore, the estimated IBM based on different initializations may vary and separation performance may be affected dramatically. Furthermore, to obtain relatively good performance, DNN needs to be trained with sufficient labeled data. However, if the labeled training instances are few, *i.e.* fine-tuning information is scarce, DNN can suffer from over-fitting.

To address the above problems, a deep and ensemble of DNN audio separation system is proposed. The ensemble system is a methodology that combines diverse models to obtain better performance. The key element to make ensemble system succeed is the diversity that make up the ensemble. DNN is a good choice to achieve diversity when trained by using different weight initializations, number of layers and hidden neurons, and cost functions. The ensemble system is constructed in two steps: training a number of individual DNN and combining the component output. As for training a DNN, a layer-wise pre-training method is employed. DNN can be viewed as hierarchical feature detector that capable of capturing higher-order correlations between input data. To combine the components output, the most prevailing approach is to average the output of each individual component of the system. However, simply averaging the output of each individual component may not achieve the system's optimum performance as different DNN have different contributions to the final features [111]. Simply averaging ignores the diversity of each DNN and thus it cannot efficiently explore the complementary property. To overcome the drawbacks, and to efficiently learn the complementary property of the input data, a Multi-view Spectral Embedding (MSE) is utilized. The MSE method learns a low-dimensional representation and sufficiently smooth embedding over all DNNs simultaneously. Furthermore, instead of using the output layer as the learned representation, we treat the penultimate layer as the learned intermediate features.

5.2 System overview and feature extraction

5.2.1 *System overview*

Fig. 5.1 shows the overall proposed system, which consists of three phases: 1) DNN training, 2) MSE and 3) ELM classification. For DNN training, the raw acoustic features of the source signals are extracted and utilized to train each individual DNN in each frequency channel. Then the MSE is utilized to fuse the learned features of the penultimate layer to a complementary feature vector. The obtained feature vector is then fed into the second module to extract more robust and discriminative feature followed by the ELM classifier to classify each TF unit to the speech domain or non-speech domain. For testing, each TF unit of the mixed signal is classified with the optimized ensemble system to generate IBM. The estimated time domain sources are resynthesized using the approach in [112] by weighting the mixture cochleagram by the mask and correcting phase shifts introduced during the gammatone filtering.

The architecture of the proposed framework is described as follows: The mixed signal with 16kHz sampling frequency is fed into a 64-channel gammatone filterbank [113], with center frequencies equally spaced from 50Hz to 8000Hz on the equivalent rectangular bandwidth rate scale. The output of each filter channel is divided into time frames with 50% overlap between consecutive frames. The TF units of all the filter outputs are then constructed to form the

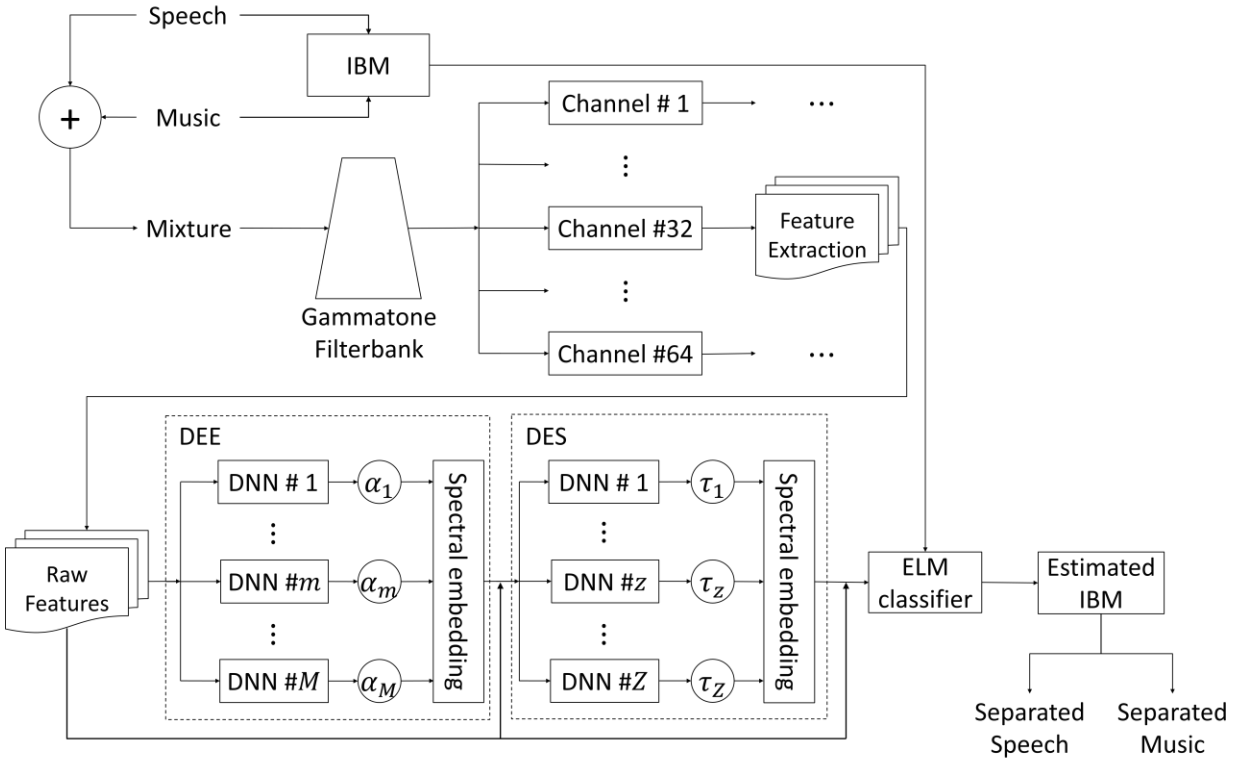


Fig. 5.1 DNN ensemble system including DNN Ensemble Embedding (DEE) and DNN Ensemble Stacking (DES)

cocholeagram [42]. The objective is to estimate IBM by classifying each TF unit to its corresponding domain. However, the spectral properties of source signals in different channels can be very different. Therefore, for each channel, we train a subband classifier to make the decision. We choose ELM classifier due to its high classification performance and low computational complexity [75, 114-116]. To perform classification, various features are extracted for each TF unit. The feature set consists of 15-Dimensional Amplitude Modulation Spectrogram (AMS), 31-Dimensional MFCC and 13-Dimensional Relative Spectral Transform and Perceptual Linear Prediction (RASTA-PLP). The PLP is a way of warping spectra to minimize the differences between speakers while preserving the important speech information. A special band-pass filter called RASTA was added to each frequency sub-band in traditional PLP algorithm in order to smooth out short-term noise variations and to remove any constant offset in the speech channel. The extracted features are concatenated to form a feature vector. Instead of feeding the feature vector into the classifier directly, we propose to pool several DNNs and create an ensemble system to learn more robust and discriminative representations. Furthermore, the penultimate layer of each individual DNN is embedded to explore the complementarity of the learned representations to further enhance the robustness of classification and hence improve the separation performance.

To obtain the features of each TF unit, we apply the conventional frame-level acoustic feature extraction method for the output of each gammatone filter channel and the concatenated feature vectors are taken as the raw acoustic feature set, which will be fed into the DNN ensemble system.

5.2.2 *Feature extraction*

In the following, the features used in our experiments are described. We use the RASTAMAT toolbox [117]for extracting MFCC, RASTA-PLP features.

- *Amplitude modulation spectrogram.* To produce AMS features, we compute the envelope of the mixture signal by full-wave rectification and decimated by a factor of 4. The decimated envelope is subsequently segmented into overlapping segments followed by Hanning windowing and zero-padding for a 256-point fast Fourier transform (FFT). The obtained magnitudes of FFT are multiplied by 15 triangular-shaped windows spaced uniformly across the 15.6-400 Hz to generate the 15-Dimensional AMS [118].
- *Relative spectral transform and perceptual linear prediction*. To generate RASTA-PLP features, the spectral amplitude is transformed through a compressing static nonlinear transformation. The time trajectory of each transformed spectral component is filtered

and expanded again followed by conventional PLP analysis [119, 120].

- *Mel-frequency cepstral coefficient*. MFCC is acquired by applying a short-time Fourier transform with Hamming window and then warped to the mel scale followed by a log operation and discrete cosine transform.

Delta features of RASTA-PLP are found to be helpful in speech separation [120]. So the original features are concatenated with the first order delta features and second order delta features of RASTA-PLP to form a combined feature vector for feature learning and classification. The final 85-Dimensional raw acoustic features set consists of 15-Dimensional AMS, 13-Dimensional RASTAPLP, 13-Dimensional RASTAPLPΔ, 13-Dimensional RASTA-PLPΔΔ, and 31-Dimensional MFCC. Δ and ΔΔ denote first order delta and second order delta.

5.3 The proposed DNN ensemble system

Given a mixed signal, we extract the acoustic features of each TF unit in cochleagram denoted as $\{x_n\}_{n=1}^{N}$, where N is the number of frames.

5.3.1 *DNN Ensemble Embedding*

Suppose that the DNN Ensemble Embedding (DEE) contains M DNNs ($M > 1$).

The m-th DNN learns a mapping function that can be formulated as

$$\mathcal{F}_m = f_m\left(w_{mA}g_{m(A-1)}\left(\dots w_{ma}g_{m(a-1)}\left(\dots w_{m2}g_{m1}\left(w_{m1}\{x_n\}_{n=1}^{N}\right)\right)\right)\right) \tag{5.1}$$

where $a = 1, \dots, A$ denotes the number of hidden layers, w_{ma} is the weight connecting the a^{th} hidden layer and the layer above, $g_{ma}(\cdot)$ denotes the activation function of the a^{th} hidden layer, $f_m(\cdot)$ denotes the output activation function. Note that the weight parameter $W = \{w_m\}_{m=1}^{M}$ of each DNN in the same module is different.

As mentioned in Chapter 2, the traditional training procedure for DNN is ineffective due to the poor generalization and local optima. To overcome these issues, the network is pre-trained using the RBM in a greedy layer-wise fashion as illustrated in Chapter 3 table 3-I. The extracted raw acoustic features are used as training data to train the first RBM, then the hidden activations are treated as the new 'data' for the second RBM, and so on. For training the first RBM, due to the input data is continuous, Gaussian-Bernoulli RBM (GBRBM), whose energy function is defined as follows, is utilized

$$E_{GBRBM}(v,h) = \sum_{\varphi \in vis} \frac{\left(v_\varphi - b_\varphi\right)^2}{2\sigma_\varphi^2} - \sum_{\upsilon \in hid} c_\upsilon h_\upsilon - \sum_{\varphi,\upsilon} w_{\varphi\upsilon} h_\upsilon \frac{v_\varphi}{\sigma_\varphi} \tag{5.2}$$

where v_φ and h_υ are the φ^{th} and υ^{th} unit of visible layer and hidden layer, respectively, b_φ denotes the bias of φ^{th} visible unit and c_υ means the bias of υ^{th} hidden unit, $w_{\varphi\upsilon}$ is the weight

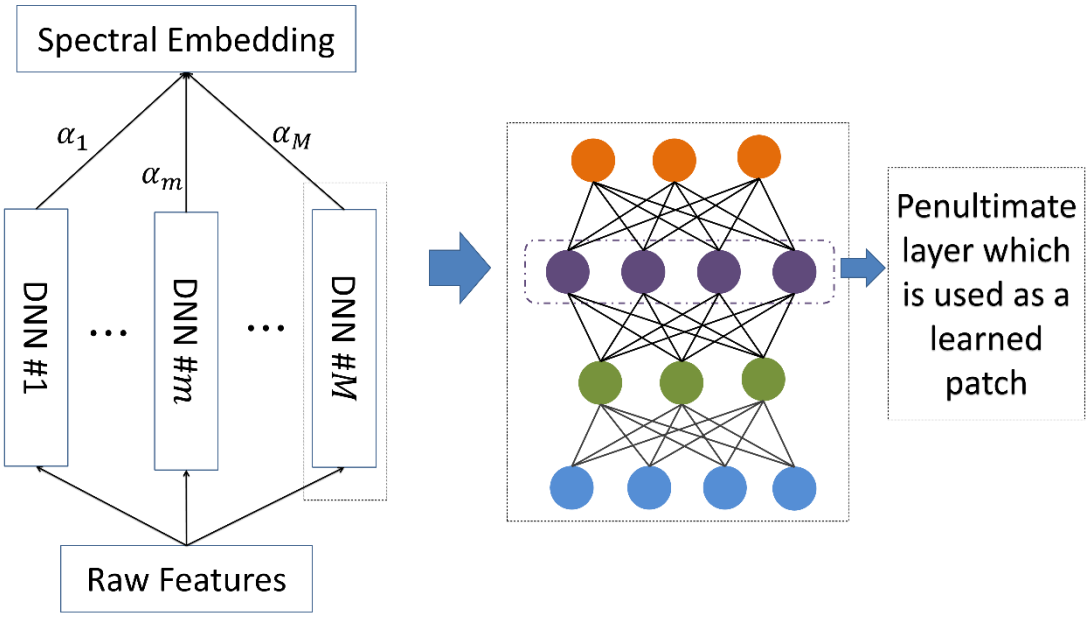


Fig. 5.2 Penultimate layer of DNN

between the φ^{th} visible unit and v^{th} hidden unit. Bernoulli-Bernoulli RMB are used for all remaining layers:

$$E_{RBM}(v,h) = \sum_{\varphi \in vis} b_\varphi v_\varphi - \sum_{v \in hid} c_v h_v - \sum_{\varphi,v} w_{\varphi v} h_v v_\varphi \tag{5.3}$$

A DBN can be formed by stacking RBMs. This way of constructing a network has empirically been found to aid the subsequent backpropagation fine-tuning and it is often critical for training a deep network having many hidden layers. To improve the performance, the whole network is then fine-tuned using the back-propagation algorithm. Instead of outputting the last layer activation, the penultimate layer activation expressed as P_m are treated as the learned intermediate representation after the network is sufficiently fine-tuned.

5.3.2 *Multi-view Spectral Embedding*

The learned intermediate representations of M DNNs $P = \{P_m \in \Re^{d_m \times n}\}_{m=1}^{M}$ are fed into a multispectral graph Laplacian to explore the complementary property [111, 121]. Different representations have different strengths that may tend to result in separation system make different errors [122]. MSE is a way to exploit complementary representations and to take the advantage of the strengths of particular representations as illustrated in Fig. 5.3. Given the m^{th} learned representation $P_m = [p_{m1}, p_{m2}, \dots, p_{mn}] \in \Re^{d_m \times n}$, consider an arbitrary point p_{mj} and its k related ones in the same features set (*e.g.*, nearest neighbors) $p_{mj1}, p_{mj2}, \dots, p_{mjk}$; the patch of p_{mj} is defined as $P_{mj} = [p_{mj}, p_{mj1}, p_{mj2}, \dots, p_{mjk}] \in \Re^{d_m \times (k+1)}$. For P_{mj}, there is a part mapping $\mathcal{H}_{mj}$: $P_{mj} \to R_{mj}$, where $R_{mj} = [r_{mj}, r_{mj1}, r_{mj2}, \dots, r_{mjk}] \in \Re^{v \times (k+1)}$, where v

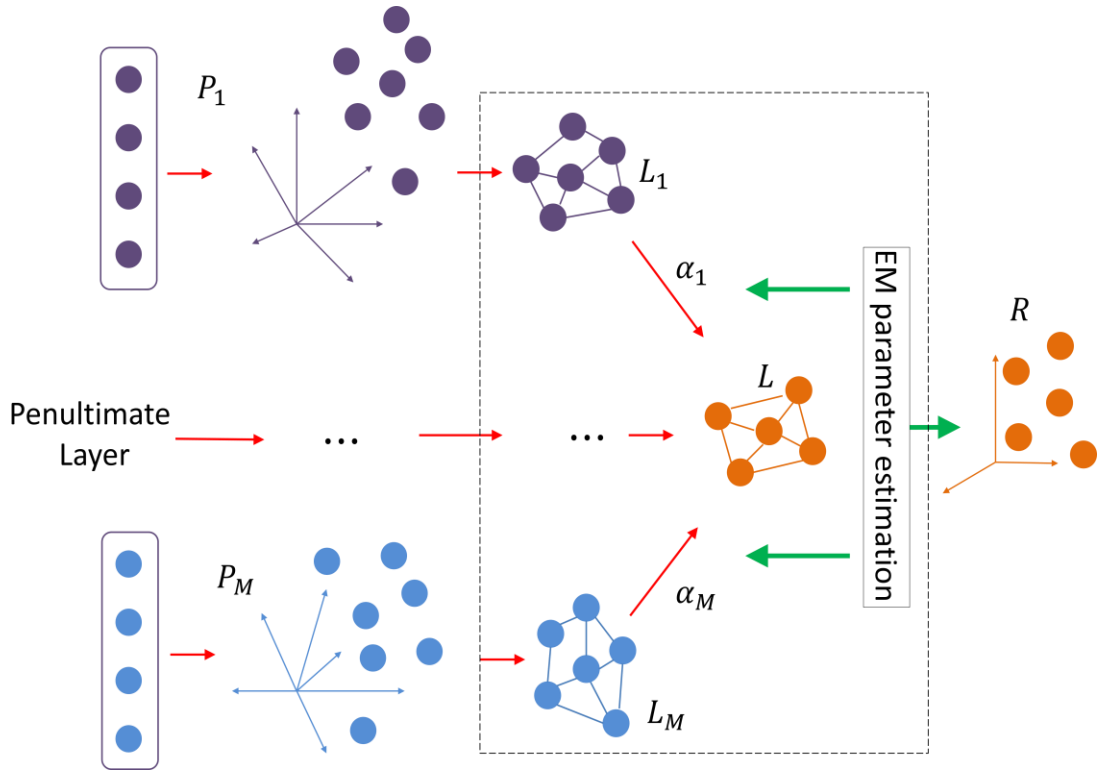


Fig. 5.3 Working principle of Multi-view Spectral Embedding

is a predefined number denotes the dimension of the aimed embedding. To preserve the locality in the projected low dimensional space, the part optimization for the j^{th} patch on the m^{th} feature set is

$$\underset{R_{mj}}{\operatorname{argmin}} \sum_{i=1}^{k} \left\| r_{mj} - r_{mji} \right\|^2 \left(\mu_{mj}\right)_i \tag{5.4}$$

where μ_{mj} is a k -dimensional column vector weighted by $\left(\mu_{mj}\right)_i = exp\left(-\left\|p_{mj} - p_{mji}\right\|^2/\gamma\right)$, γ controls the width of the neighborhoods. Therefore, the part optimization can be reformulated to

$$\begin{aligned} &\underset{R_{mj}}{\operatorname{argmin}}\, tr\left(\begin{bmatrix} \left(r_{mj} - r_{mj1}\right)^T \\ \dots \\ \left(r_{mj} - r_{mjk}\right)^T \end{bmatrix} \times \left[r_{mj} - r_{mj1}, \dots, r_{mj} - r_{mjk}\right] diag\left(\mu_{mj}\right)\right) \\ &= \underset{R_{mj}}{\operatorname{argmin}}\, tr\left(R_{mj} L_{mj} \left(R_{mj}\right)^T\right) \end{aligned} \tag{5.5}$$

where $tr(\cdot)$ is the trace operator, $L_{mj} = \begin{bmatrix} \sum_{i=1}^{k} \left(\mu_{mj}\right)_i & -\left(\mu_{mj}\right)^T \\ -\mu_{mj} & diag\left(\mu_{mj}\right) \end{bmatrix} \in \Re^{(k+1)\times(k+1)}$ encodes the objective function for the j^{th} patch on the m^{th} learned representation. By preserving the intrinsic structure of the j^{th} patch on the m^{th} learned representation, a sufficiently smooth low dimensional embedding R_{mj} can be found.

With different mapping parameters, the DNN ensemble system extracts various features that may have different contribution to the final low dimensional embedding. To explore the complementary property of different extracted features, a set of nonnegative weights $\alpha = [\alpha_1, \dots, \alpha_m]$ are imposed on part optimizations of different DNN independently. The larger α_m is, the more important role the P_{mj} plays in learning to obtain the low dimensional embedding R_{mj}. By summing over all m^{th} learned representation, the part optimization for the j^{th} patch is expressed as

$$\underset{R_j=\{R_{mj}\}_{m=1}^{M},\alpha}{\operatorname{argmin}} \sum_{m=1}^{M} \alpha_m tr\left(R_{mj} L_{mj} \left(R_{mj}\right)^T\right) \tag{5.6}$$

For each patch P_{mj}, there exist a low-dimensional embedding R_{mj}. All R_{mj} can be unified together as a whole one by assuming that the coordinate for $R_{mj} = [r_{mj}, r_{mj1}, r_{mj2}, \dots, r_{mjk}]$ is selected from the global coordinate $R = [r_1, r_2, r_3, \dots, r_n]$, *i.e.*, $R_{mj} = RV_{mj}$, where $V_{mj} \in \Re^{n\times(k+1)}$ is the selection matrix to encode the spatial relationship of samples in a patch in the original high-dimensional space. Therefore, (5.3.4) can be rewritten as

$$\underset{R_j,\alpha}{\operatorname{argmin}} \sum_{m=1}^{M} \alpha_m tr\left(RV_{mj} L_{mj} \left(V_{mj}\right)^T (R)^T\right) \tag{5.7}$$

By summing over all part optimizations, the global coordinate alignment is given by

$$\begin{aligned} &\underset{R,\alpha}{\operatorname{argmin}} \sum_{m=1}^{M} \alpha_m^{\varepsilon} tr(RL_m R^T) \\ &s.t. RR^T = I, \sum_{m=1}^{M} \alpha_m^{\varepsilon} = 1, \qquad \alpha_m \geq 0 \end{aligned} \tag{5.8}$$

where $L_m \in \Re^{n\times n}$ is the alignment matrix for the m^{th} learned representations, and it is defined as $L_m = \sum_{j=1}^{N} V_{mj} L_{mj} \left(V_{mj}\right)^T$. The constraint $RR^T = I$ is to uniquely determine R. Exponent ε is the coefficient for controlling the interdependency between different views and should satisfy $\varepsilon \geq 1$. By performing a normalization on L_m, we obtain a normalized graph Laplacian L_{sys} which is symmetric and positive semidefinite defined as

$$L_{sys} = {D_m}^{-\frac{1}{2}} L_m {D_m}^{-\frac{1}{2}} = I - {D_m}^{-\frac{1}{2}} Q_m {D_m}^{-\frac{1}{2}} \tag{5.9}$$

where Q_m is a degree matrix defined as the diagonal matrix with the degrees $d_i = \sum_{j=1}^{n} \mu_{mj}$ on the diagonal and unnormalized graph Laplacian matrix, which is defined as $L_m = D_m - Q_m$. $Eq.(5.8)$ is a nonlinearly constrained nonconvex optimization problem and the optimal solution can be obtained by using Expectation Maximization (EM) iterative algorithm [123]. The optimization iteratively updates R and α in an alternating fashion.

M-step: Fix R to update α.

By introducing a Lagrange multiplier λ and taking the constraint $\sum_{m=1}^{M} \alpha_m^{\varepsilon} = 1$ into consideration, the Lagrange function can be expressed as

$$L(\alpha, \lambda) = \sum_{m=1}^{M} \alpha_m^{\varepsilon} tr\left(RL_{sys}R^T\right) - \lambda\left(\sum_{m=1}^{M} \alpha_m - 1\right) \tag{5.10}$$

The solution for α_m can be given by

$$\alpha_m = \frac{\left(1/tr\left(RL_{sys}R^T\right)\right)^{1/(\varepsilon-1)}}{\sum_{m=1}^{M}\left(1/tr\left(RL_{sys}R^T\right)\right)^{1/(\varepsilon-1)}} \tag{5.11}$$

When R is fixed, $Eq.(5.11)$ gives the global optimal α.

E-step: Fix α to update R.

The optimization problem in $Eq.(5.8)$ is equivalent to

$$\min_{R}(RLR^T) \quad s.t. RR^T = I \tag{5.12}$$

where $L = \sum_{m=1}^{M} \alpha_m^{\varepsilon} L_{sys}$. Based on Ky-Fan theorem, $Eq.(5.12)$ has a global optimal solution when α is fixed [124]. The optimal R is given as the eigenvectors associated with the smallest d eigenvalues of the matrix L. Once the embedded feature R is obtained, it will be concatenated with the raw acoustic features to form a new feature vector since the raw acoustic features are able to provide global information that may helpful to estimate mask.

5.3.3 *DNN Ensemble Stacking*

To generate higher order and more robust and discriminative features, a second DNN ensemble termed as DNN Ensemble Stacking (DES) is stacked onto the DEE. The DEE is treated as a lower module while the DES is treated as a higher one. The embedded features extracted from the DEE are concatenated with the raw features as its input to the higher module. This allows higher order and more robust and discriminative features to be extracted. DES is a masking-based module in which DNN is trained with pre-training followed by supervised fine-tuning. In this stage, DES involves training $Z > 1$ DNNs, denoted as ϕ_Z. The learning procedure of the z^{th} DNN can be expressed as

$$\phi_z = f_z\left(w_{zB}g_{z(B-1)}\left(\dots w_{zb}g_{z(b-1)}\left(\dots w_{z2}g_{z1}(w_{z1}\sigma)\right)\right)\right) \tag{5.13}$$

where σ denotes the concatenation of embedded features and raw acoustic features from lower module and w denotes the model parameters. In DES, each individual DNN learns a masking function. Common activation functions in output layer include linear function, sigmoid function and softmax function. As the training target is the IBM whose value is either 0 or 1, we choose the sigmoid function for output layer. The combination feature set is used to train the first

GBRBM, whose hidden activations are then treated as a new training data for the second RBM. The pre-trained GBEBM, RBMs and sigmoid layer are combined together and fine-tuned with labeled data to obtain the internal discriminative representations. After the network is sufficiently fine-tuned, the outputs of the penultimate layer of the DES are fed into a multispectral graph Laplacian to explore the complementary property. The embedded features concatenated with raw features will be classified using the ELM classifier to estimate an IBM. The estimated time domain sources are resynthesized by weighting the mixture cochleagram by the mask.

5.4 Single Channel Audio Separation

In this section, the proposed separation system is evaluated on recorded audio signals. The speech data that we used is from the 'CHiME' database [105], which consists of 34 speakers speaking 500 utterances each. For training data generation, 10 utterances are randomly selected and mixed with the music [106] at 0 dB. The test set is created by mixing 25 utterances different from the training data with the music at 0 dB. The raw acoustic features of each channel are extracted and normalized to zero mean and unit variance before feeding into the system [117].

5.4.1 *Experiment Set-up*

For each DNN in the system, GBRBM is trained as the first layer between the visible layer and the first hidden layer while above layers are constructed using RBM pre-training. We use 50 epochs of gradient descent for pre-training and 50 epochs for the whole network fine-tuning. The learning rate of GBRBM is set to 0.001 and learning rate of above RBM is set to 0.01. The momentum of the first 5 epochs is set to 0.5, and the momentum of other epochs is set to 0.9. Considering the performance and computational complexity, relatively small DNN with two hidden layers are adopted. The small number of tunable network parameters facilitates fast and scalable training with reasonably good performance. For the MSE, the size of the nearest neighbors is set to be 10. The dimension of embedded features is set to 50. Note that the embedded features are always combined with raw acoustic features for training the ELM classifier.

The proposed system is compared with other machine learning methods including ELM-based, Support Vector Machine (SVM)-based, DNN-based and DNN-ELM-based. For ELM-based and SVM-based methods, the raw acoustic features are utilized to train the ELM and SVM. The DNN-based method is trained by mini-batch gradient descent with 50 epochs for RBM pre-training and with 50 epochs for the network fine-tuning. For DNN-ELM based methods, the output of the last hidden layer of the DNN is utilized to train an ELM. All four methods train a

classifier for each channel. Furthermore, we have selected Itakura-Saito NMF (IS-NMF) [104] and NMF-2D [42] algorithm as other comparison methods. IS-NMF has been previously shown to correctly capture the semantics of audio and is better suitable to the representation than the standard NMF [104]. The recently proposed NMF-2D [42] provides promising separation result for music mixture and it is deemed as a competitive approach to solve separation problems.

5.4.2 *Optimizing number of DNNs*

To determine the number of DNNs in each module, we compared the separation performance in terms of number of DNNs. We first set 1 DNN in DEE and DES (denoted as 1DEE-1DES), respectively and evaluate the separation performance. Then we set 2 DNNs in DES and 1 DNN in DEE (2DES-1DEE) and evaluate the performance. The experiments will be continued until all the settings (5DEE-5DES) are evaluated. The separation results are shown in Fig. 5.4. In all the experiments, we use the same training data. The evaluation metric that we used is Short-Time Objective Intelligibility (STOI) [125], which evaluates the objective speech intelligibility of time-domain signals. It has been shown empirically that the STOI scores are well correlated with speech intelligibility. The higher the STOI value is, the better the predicted intelligibility is.

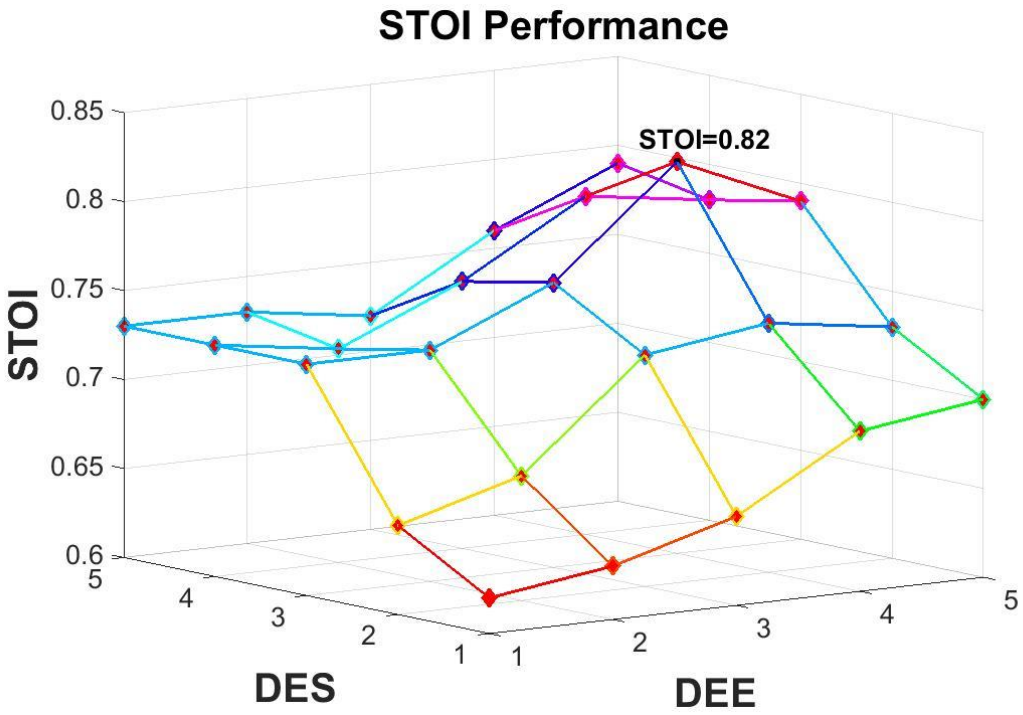


Fig. 5.4 STOI performance

From Fig. 5.4, it is observed that adding a second DNN in the DEE and the DES improves the separation performance over using a single DNN of each module. By adding one DNN to each module, the performance improves significantly compared with only one DNN in each module.

Not only that, it is observed that the improvement is more significant with subsequent addition of DNNs. This is especially more accentuated with additional DNNs in the DES module. The highest achievable STOI is 0.82. However, the separation performance improvement becomes less significant with more DNNs onwards. This may be due to additional DNNs being unable to extract extra discriminative features that are able to improve the separation performance.

To further investigate the effectiveness of number of DNNs in the learning system, we used Perceptual Evaluation of Speech Quality (PESQ) and Signal-to-Distortion Ratio (SDR) as other criteria to evaluate the proposed learning system. PESQ is an objective method to test the speech quality. The objectivity is based on the comparison to the traditional Mean Opinion Score method in which a group of listeners are used to rate the voice quality to a value ranging from 1(bad) to 5 (excellent) [126]. A higher score means a better speech quality.

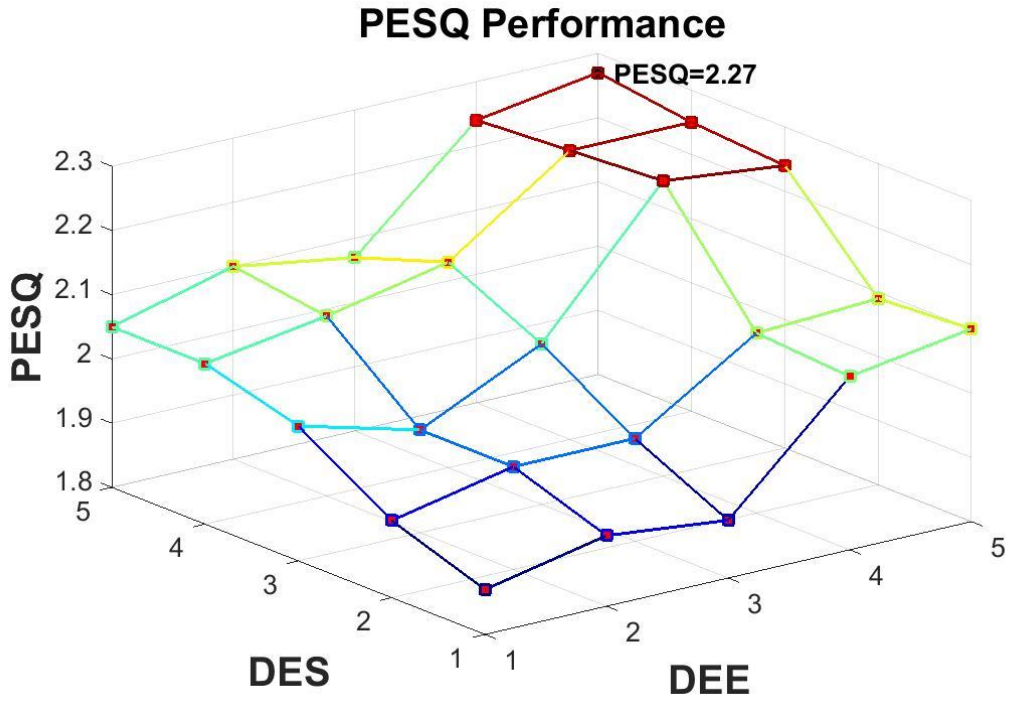


Fig. 5.5 PESQ performance

The results have shown in Fig. 5.5 and Fig. 5.6. From Fig. 5.5, we can observe that the separation performance with the setting of 4DEE-3DES improve significantly compared with 4DEE-2DES and 3DEE-3DES. Although the highest PESQ is obtain with 5DEE-5DES, the improvement is less significant compared with 4DEE-3DES. Fig. 5.6 has demonstrated that the separation performance of 4DEE-3DES gives result of 11.82 dB, which is considerably better than that of using single DNN in each module of the network. To conclude, the separation performance has been improved with each increment in the number of DNN in each module, however, the improvement is less significant after 4 DNNs in each module. Considering the computational complexity, using 4 DNNs in DEE and 3 DNNs in DES is a good choice.

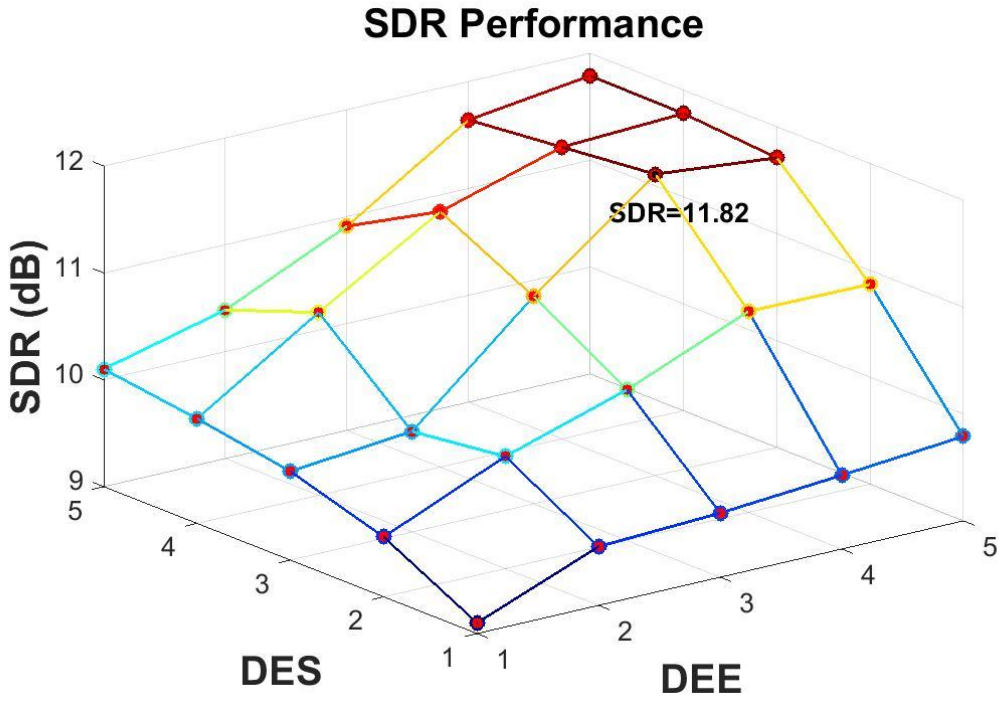


Fig. 5.6 SDR performance

5.4.3 *Speech separation performance*

In order to demonstrate the effectiveness of our proposed approach, we have compared the separation performance with the selected methods for different mixtures. To generate the training set, we randomly choose 10 utterances from a male and a female each. The selected utterances are mixed with guitar and bass music at 0 dB SNR. To test our system, we create a testing data set consists 30 utterances different from the training data mixed with guitar and bass music at 0 dB SNR. For pre-processing, the 85-Dimensional feature set of training and testing data of each TF unit is extracted.

In this experiment, the separation performance is evaluated in terms of SDR. For comparison, IS-NMF, NMF-2D, ELM-based, DNN-based and IBM method are selected. The IS-NMF is used in conjunction with a clustering algorithm where the mixed signal is factorized into $И = 2, 4, \ldots, 10$ components followed by a grouping method which is used to cluster the $И$ components to each source. The best value of result of each case of the $И$ different configurations has been retained for comparison. For the NMF-2D, the spectral and temporal features of the mixed signal are factorized in nonuniform TF domain produced by the gammatone filterbank. The obtained features are used to generate the binary mask to separate the mixed signal. For the IBM method, the mask is generated directly from the speech and music.

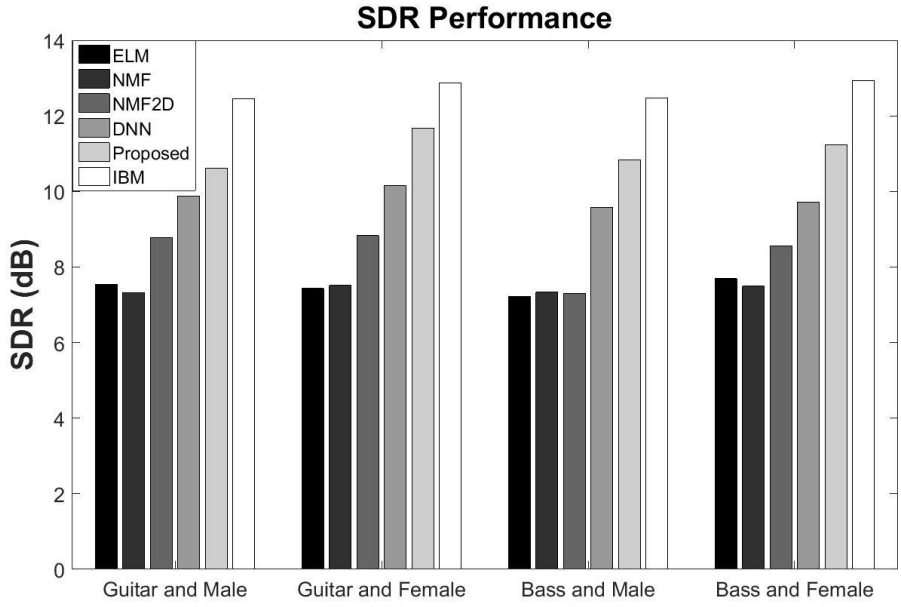


Fig. 5.7 SDR performance for different mixture

Referring to Fig. 5.7, it is noted that the SDR performance varies significantly depending on approaches used for separation. Judging from the SDR for all type of mixture, the ELM-based method gives an average SDR of 7.47 dB; NMF algorithm gives an average SDR of 7.24 dB; NMF-2D algorithm gives an average SDR of 8.37 dB; DNN-based method delivers an average SDR of 9.83 dB; our proposed method with an average SDR of 11.09 dB and, IBM gives an average SDR of 12.66 dB. It should be mentioned that the IBM delivers the highest score for the reason that it is generated from the target speech and music directly. It should be noticed that the obtained results of DNN-based method and our proposed system significantly outperform ELM-based method. This is attributed to the classified features learned by deep architecture which is more discriminative than the shallow network. It is also noted that both DNN and proposed system exhibit relatively high SDR performance. In addition, the performance of proposed approach is always better than that of DNN. This confirms our analysis that proposed system is able to extract more complementary features than single DNN. It also proved that the higher layers of deep architecture represent more abstract and discriminative features than those from lower ones.

To further analyze the separation performance of the proposed approach, we have conducted an experiment with a mixture of a female mixed with guitar music at 0 dB. The original utterance, music, mixture and the separation result are shown in Fig. 5.8. The SDR performance for speech is 11.69 dB while the SDR for music is 9.16 dB.

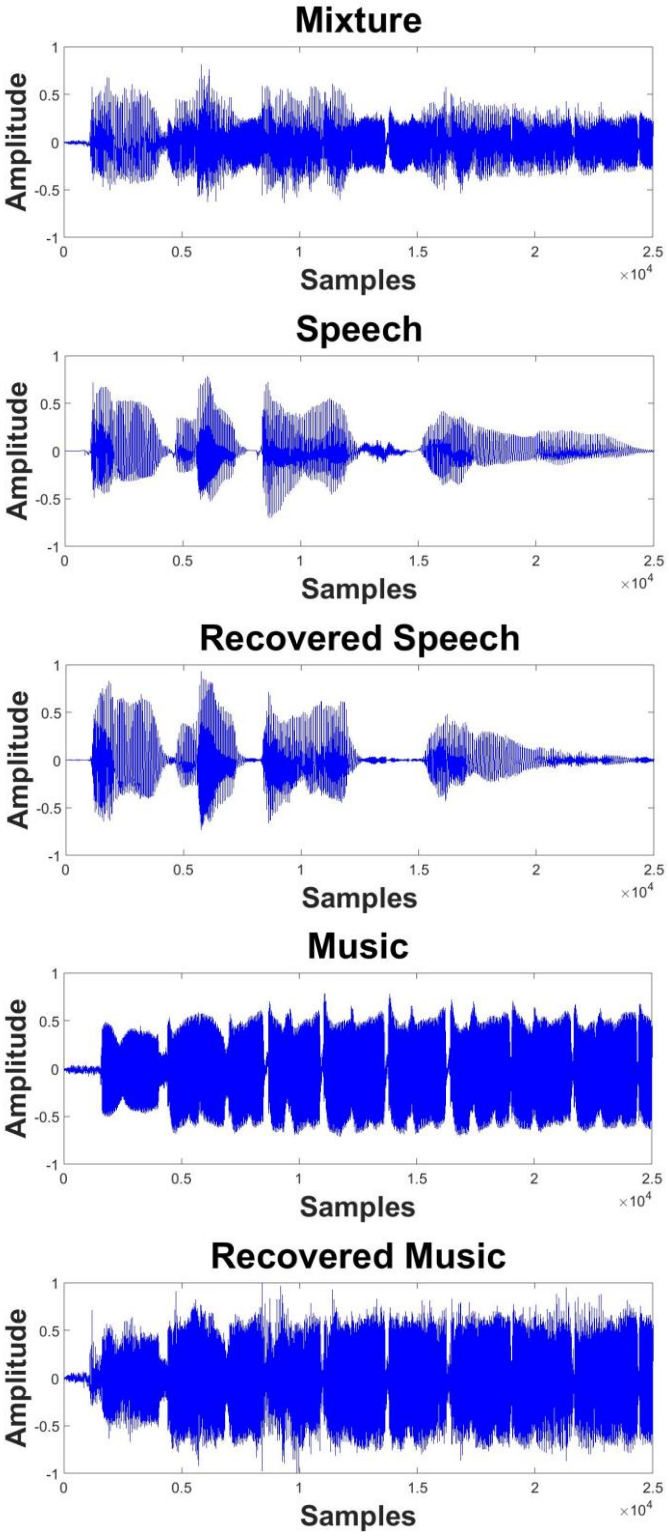


Fig. 5.8 Time domain separation results. (a) Mixture of guitar and female utterance. (b) Speech (c) Recovered speech (d) Music (e) Recovered music.

5.4.4 *Generalization under different SNR*

In this section, experiments are conducted to evaluate the effectiveness of the proposed approach under different SNR conditions. For SNR generalization, the training set contains mixtures at a single input SNR and the system will be tested on mixtures at different SNRs. 10

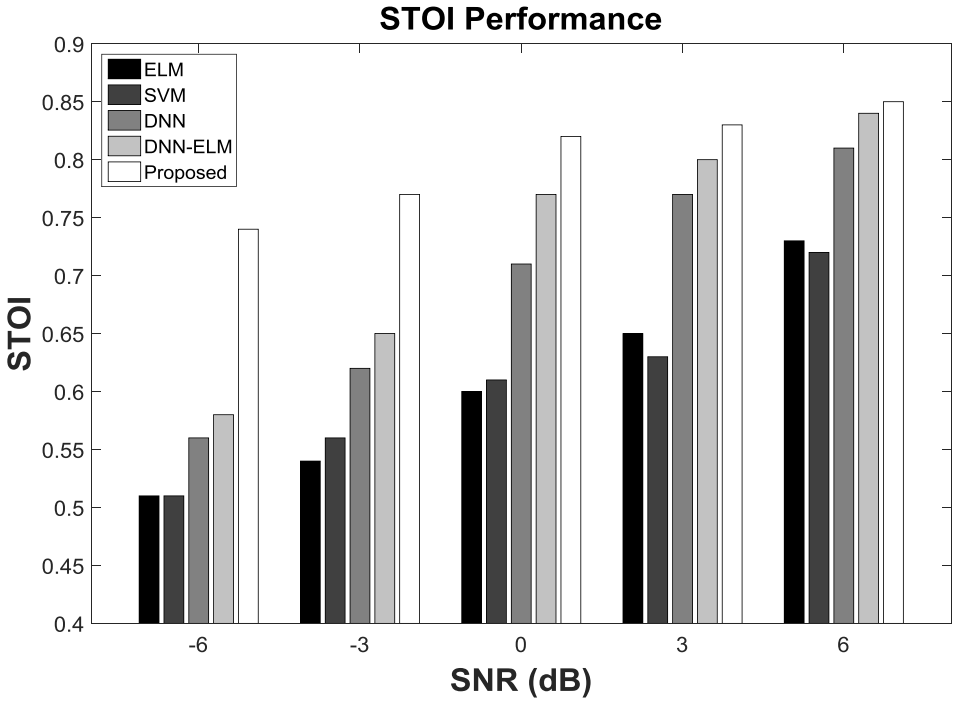


Fig. 5.9 STOI under different SNR

utterances of a speaker are selected and mixed with the music at 0 dB SNR while 20 utterances are selected and mixed with the music at SNR ranging from -6 dB to 6 dB with an increment of 3 dB to generate the test data.

For comparison, ELM-based, SVM-based, DNN-based and DNN-ELM-based methods are selected. Fig. 5.9 gives the STOI comparison of different separation methods. Several observations can be made. First of all, deep architectures (DNN, DNN-ELM and the proposed method) significantly outperform the shallow architectures (ELM, SVM) across all different input SNRs. For example, the proposed method leads to an average STOI improvement close to 24% compared with ELM. Especially at -6 SNR, our proposed method leads to 29% improvement. This is attributed to the deep architecture being able to extract features by a multilayer distributed feature representation with higher layers represent more abstract and discriminative features. Therefore, the IBM generated from deep architectures are more accurate than that of the shallow architectures. Secondly, the DNN-ELM gives better SNR results than that of DNN. This is due to the assistance of the ELM classifier. Although the DNN outputs have already formed an estimated IBM, the ELM is able to further generate the features extracted from the DNN. Thirdly, the proposed method delivers the best STOI result among the deep architectures. It should also be noted that the separation performance is not affected dramatically over the SNR. Compared with other methods, the proposed method has shown increased robustness as the STOI index changes only minutely. This is because the DNN ensemble with MSE is able to extract more complementary and robust features.

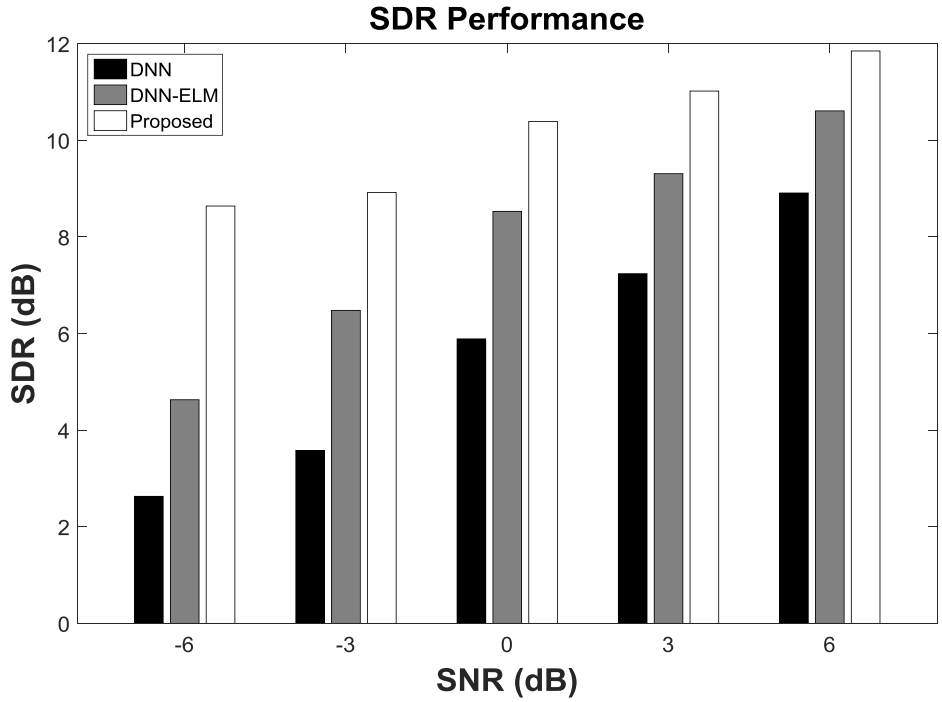


Fig. 5.10 SDR performance under different SNR

To further analyze the effectiveness of the proposed method, we have plotted the SDR performance. For comparison, we select deep architectures including DNN and DNN-ELM to learn and classify the input signals. The comparison results are shown in Fig. 5.10. It is clearly shown that our proposed method outperforms DNN and DNN-ELM across all different input SNRs. This has proved the point that the proposed method is able to extract more discriminative features than that of single DNN.

5.4.5 *Generalization to different input music*

To demonstrate the generalization ability of our proposed system, we have conducted the experiments with regard to the target dependent. That is to say, the interfering music in the test set is different from those in the training set but the testing speech is from the same speaker. The training set contains signals mixed with a piece of music at 0dB and the system is tested on mixtures of speech and unseen music. To create the training set, we randomly choose 10 male and female utterances each from the 'CHiME' dataset and mixed with guitar music at 0 dB SNR to train the proposed system. The feature set comprises 85-Dimensional raw acoustic features in total. To test our system, 30 male and female utterances different from the training data are selected and mixed with bass and piano music at 0 dB. For pre-processing, the 85-Dimensional feature set of testing data of each TF unit is extracted and normalized to zero mean and unit covariance.

For comparison, the ELM-based, DNN-based and IBM methods are selected. The comparison result is shown in Fig. 5.11. Firstly, it is observed that although the proposed method is trained

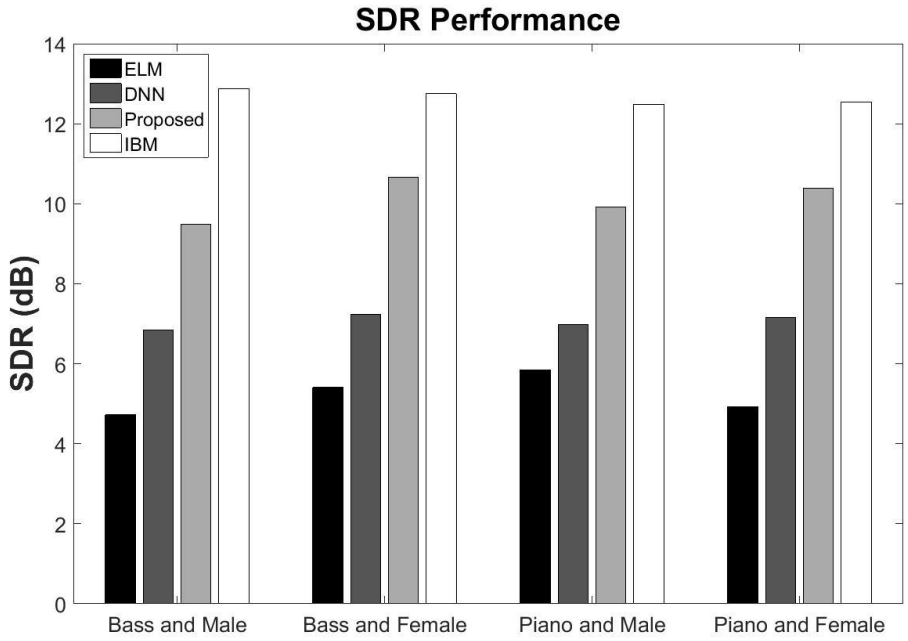


Fig. 5.11 SDR with unmatched music

with the selected music, its generalization to other music mixtures has rendered good performance as demonstrated by the result of Fig. 5.11. The SDR performance of bass and

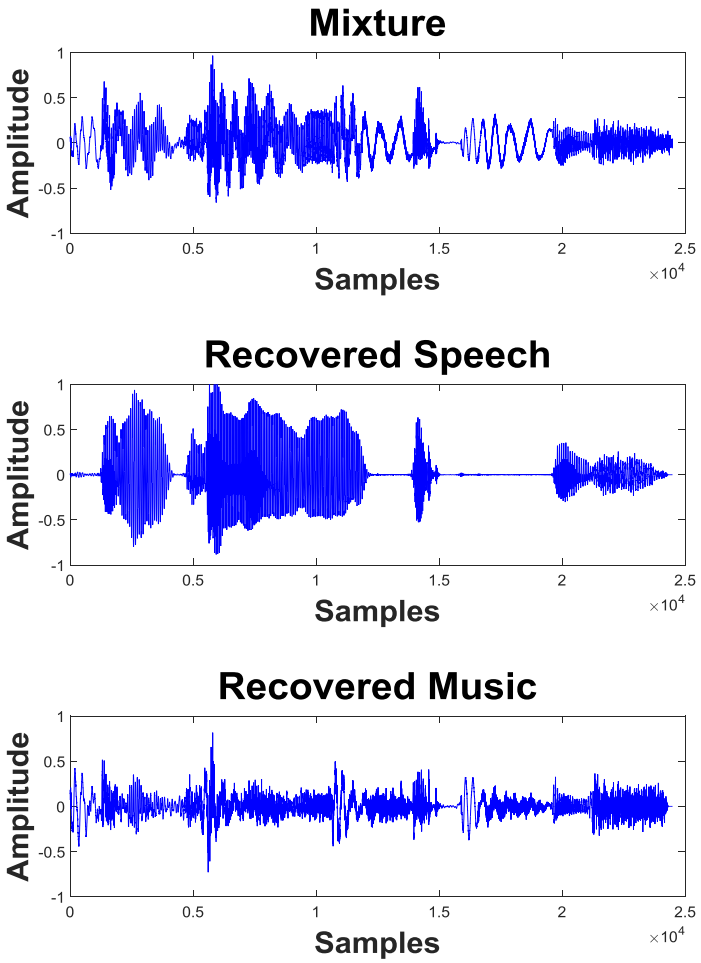


Fig. 5.12 Separation performance based on different input music

female mixture delivers 10.67 dB. Also, it is noted that the proposed method performs significantly better than that of the ELM-based method. The reason stems from the deep architecture which is able to extract more separable features. The proposed method also outperforms the DNN-based method indicating that the DNNs ensemble and stacking are able to provide more detailed information than single network. Although the IBM gives the overall best results, the proposed method has delivered almost as good results as the IBM. On average in terms of the SDR performance, the proposed approach gives 10.12 dB, ELM obtains 5.23 dB, DNN delivers 7.06 dB while IBM gives 12.67 dB. The time domain results for the mixture, recovered speech and recovered bass music are plotted in Fig. 5.12.

5.4.6 *Generalization to different speaker*

To further evaluate the effectiveness of the proposed approach, we have conducted experiments with regard to different speakers. The training data is from one speaker but the testing data is from different speaker. The training set contains speech mixed with music and the system is tested on mixtures of speeches of different speaker mixed with same music. 10 utterances of a speaker are selected and mixed with guitar music at 0 dB to generate the training dataset while another 10 utterances of a different speaker are selected and mixed with the music at 0 dB to generate the testing dataset.

The SDR performance is shown in Fig. 5.13. It is observed that despite the proposed system is trained with different speeches, the separation performance remains robust with little

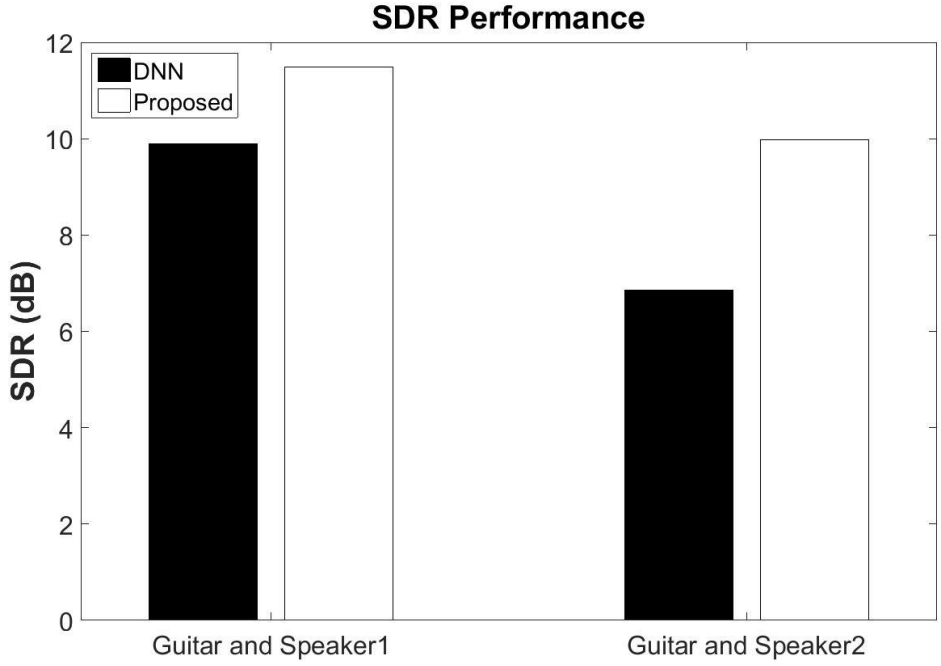


Fig. 5.13 SDR with unmatched music

fluctuation. The SDR performance for music and speaker2 gives 9.97 dB. In contrast, DNN delivers 6.85 dB.

5.4.7 *Separation performance of both proposed methods*

In this section, experiments are conducted to evaluate the effectiveness of the proposed pseudo-stereo mixture GEM-MU NMF-2D based DSELM approach and the DNN ensemble system under different SNR conditions. To create the training set, we randomly choose 50 utterances from the 'CHiME' database and guitar music from the RWC database. This training set is used to train the pseudo-stereo mixture GEM-MU NMF-2D based DSELM approach. For DNN ensemble system, the selected utterances are mixed with guitar music at 0 dB to create the training set. To generate the test mixtures under different SNR, we randomly select 25 utterances from the same speaker. The selected utterances are mixed with guitar music at SNR ranging from -6dB to 6dB with an increment of 3 dB. The comparison results are illustrated in Fig. 5.14.

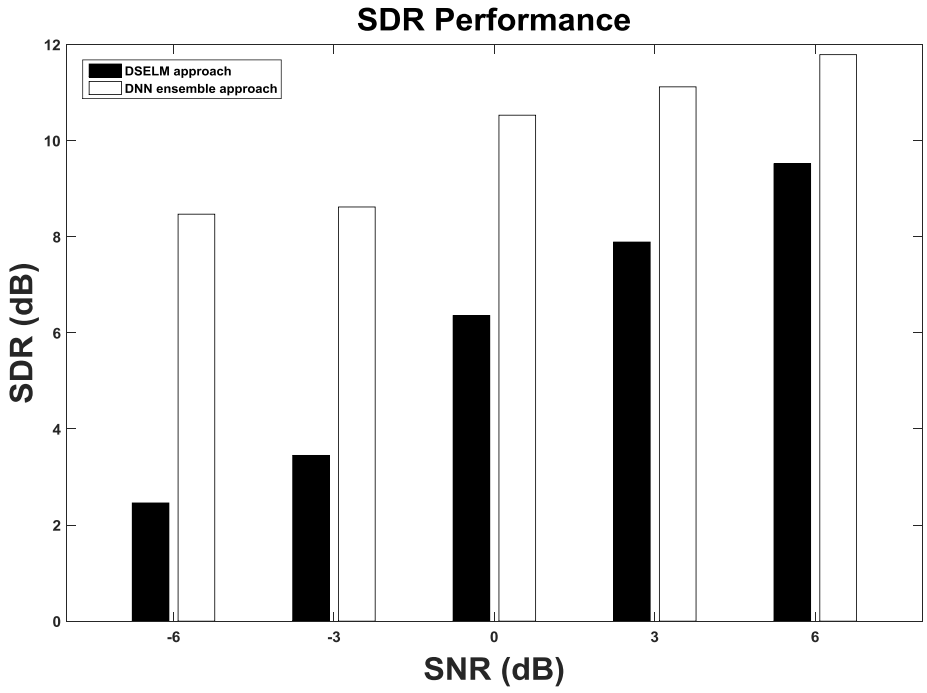


Fig. 5.14 Separation performance of DSELM approach and DNN ensemble approach

It is notable that the proposed DNN ensemble system outperforms the DSELM approach for all input in different SNR conditions range from -6dB to 6dB. It can be seen that the SDR performance of DNN ensemble approach changes only minutely. This is because the DNN ensembles with MSE is able to extract more complementary and robust features. In contrast, the SDR performance of DSELM approach changes dramatically. This maybe because the first stage of DSELM approach is coarse separation using pseudo stereo mixture GEM-MU based

NMF-2D. As mentioned before, the separation performance highly depends on the first stage. Although extra information is provided by the pseudo stereo mixture, for different SNR conditions, especially -6dB and -3dB SNR, NMF-2D is not able to separate the mixture accurately. Furthermore, the extract features using DSELM is not robust and discriminative enough to make accurate decisions. This lead to poor separation performance of the DSELM approach compared with DNN ensemble approach.

5.5 Conclusion

The impetus behind this research is that although the estimation of ideal binary mask based on machine learning approaches have acquired considerable success in tackling single channel audio separation problem, its performance level can be improved. In this chapter, a DNN ensemble system has been proposed. The proposed system extracts various features using different initializations. In addition, by exploring the complementary property of each DNN, the system is able to extract the most discriminative features, hence improving classification accuracy. Experiments have been conducted and shown that the proposed approach offers a considerable better separation performance compared with conventional methods.

Chapter 6. Conclusion of thesis

In this thesis, we have successfully established the effectiveness of using machine learning approach such as DSELM and DNN to tackle the SCSS problem.

6.1 Proposed Single Channel Source Separation approaches

In chapter 4, the DSELM with assistance of channel diversity realized by a pair of artificial stereo signals is proposed. The proposed approach consists of two main stages: A coarse separation and estimated sources refining. In the former stage, GEM-MU algorithm is developed to optimize the parameters of the NMF-2D to coarsely separate the pre-mixed signals. Furthermore, a pseudo stereo mixture is generated to increase the dimensionality of the mixing channel matrix and to reduce the ambiguity between estimating the mixing coefficients and the source signals. In the later stage, DSELM is developed to extract features of the coarsely separated sources to check for the validity followed by a joint energy minimization method to estimate a binary mask to improve the performance of the coarsely separated sources. The proposed approach yields superior performance compared with the existing SS methods.

The motivation of this research is that although matrix factorization-based approaches have acquired considerable success in tackling single channel audio separation problem, its performance level still can be improved. Therefore, DSELM with energy minimization function is developed to further improve the separation performance.

In chapter 5, a novel DNN based ensemble system is proposed. The proposed approach fully explores the complementary property of the raw acoustic features including AMS, MFCC and RASTA-PLP by using the ensemble system combined with MSE. The ensemble system is constructed in two steps: training a number of individual DNN and combining the component output. As for the training of a DNN, the layer-wise pre-training method is utilized. The trained DNN has ability to capture higher-order correlations between input data. To combine the components output, MSE is utilized to embed the extracted features into a low-dimensional representation. In addition, we treat the penultimate layer as the learned intermediate features. The ultimate goal is to thoroughly explore the complementary property of the learned features. Therefore, a second module is stacked at the top to extract more robust and discriminative features. Finally, ELM classifier is utilized to estimate a binary mask. The proposed approach outperforms well-known reliable model based approaches.

6.2 Future Works

Two novel approach has been proposed to address SCSS problem and the experimental results have demonstrated promising performances. The area of SCSS is an interesting and challenge topic that will continuously receive attention from the academic area as well as the commercial world. Although machine learning, especially DNN has made big strides toward the ultimate goal of solving the SCSS problem, various interesting research is still ahead of us. In the future, we will attempt to explore some researches list a few below:

6.2.1 *Informed source separation*

The rise of informed source separation has caught much attention in recently years. User-generated exemplar provides extra information that generated by user, who attempts to mimic one of the sources within the mixture both in terms of words and tone. DNN needs to be trained with labeled data, however, if there is no labeled data or even no training data (blind source separation), the use of DNN is impossible. One way to solve this problem is to train or fine-tune the DNN with user-generated exemplar source. This way of training or fine-tuning will enable DNN separate the mixture more precisely.

6.2.2 *Deep Reinforcement Learning*

Reinforcement Learning (RL) is an area of machine learning inspired by behaviorist psychology. It allows machines to automatically determine the ideal behavior within a specific context, in order to maximize its performance. RL stands out from other machine learning paradigms in that: no supervisor is needed, the machine learn its behavior based on feedback from the environment. RL has been applied to play Atari games, control robotic and cooling system. The key idea of using Deep Reinforcement Learning in source separation is to introduce a score, which can be SDR value. By comparing SDR, the sources can be re-estimated until some criteria are satisfied.

6.2.3 *Transfer Learning*

Human can recognize and apply relevant knowledge from previous learning experience when we encounter new tasks. Similarly, transfer learning focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. Transfer learning tend to be highly dependent on the machine learning algorithms being used to learn the tasks, and can be considered as the extension of those algorithms, such as neural network, Bayesian networks, and reinforcement learning such as Q-learning algorithm. For source separation, we can use the transfer learning concept that DNN is trained and fine-tuned using a certain database

and the trained network can be further refined to separate different speakers by providing limited data or exemplar of the target speakers.

Reference

[1] R. Vigario, V. Joursmaki, M. Hamalainen *et al.*, ""Independent Component Analysis for identification of artifacts in magnetoencephalographic recordings," *Advances in Neural Information Processing Systems*, pp. 229-235, 1998.

[2] A. Taleb, and C. Jutte, "Source separation in post-nonlinear mixtures," *IEEE Trans. On Signal Processing,* vol. 47, no. 10, pp. 2807-2820, 1999.

[3] F. Acernese, A. Ciaramella, S. D. Martino *et al.*, "Neural Networks for Blind Source Separation of Stromboli Explosion Quakes," *IEEE Transactions on Neural Networks,* vol. 14, pp. 167-175, 2003.

[4] J. Koikkalainen, and J. Lotjonen, "Image Segmentation with the Combination of the PCA and ICA-Based Modes of Shape Variatio," *IEEE International Symposium on Biomedical Imaging: Nano to Macro,* vol. 1, pp. 149-152, 2004.

[5] R. J. Weiss, and D. P. W. Ellis, "Monaura Speech Separation Using Source-Adapted Models," *IEEE Workshop on Appl. Signal Processing to Audio and Acoustics (WASPAA)*, pp. 114-117, 2007.

[6] P. Mowlaee, R. Saeidi, and Z. H. Tan, "Joint single-channel speech separation and speaker identification," *IEEE International Conference on Acoustics Speech and Signal Processing*, pp. 237-240, 2010.

[7] C. Jutten, M. Babaie-Zadeh, and S. Hosseini, "Three Easy Ways for Separating Nonlinear Mixtures," *Signal Processing,* vol. 84, no. 2, pp. 217-229, 2004.

[8] D. L. Wang, "Time-frequency masking for speech separation and its potential for hearing aid design," *Trends in Amplification,* vol. 12, no. 4, pp. 332-353, 2008.

[9] A.F. Mindkoglu, and A. J. V. D. Veen, "Separation of overlapping RFID signals by antenna arrays," in Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on, Las Vegas, NV, USA, 2008.

[10] R.Romo Vázquez, H.Vélez-Pérez, R.Ranta *et al.*, "Blind source separation, wavelet denoising and discriminant analysis for EEG artefacts and noise cancelling," *Biomedical Signal Processing and Control,* vol. 7, no. 4, pp. 389-400, 2012.

[11] K. Drakakis, S. Rickard, R.D. Fr´ein *et al.*, "Analysis of Financial Data Using Non-Negative Matrix Factorization," *International Mathematical Forum*, no. 38, pp. 1853-1870, 2008.

[12] L.T. Duarte, S. Moussaoui, and C. Jutten, "Source Separation in Chemical Analysis : Recent achievements and perspectives," *IEEE Signal Processing Magazine,* vol. 31, no. 3, pp. 135-146, 2014.

[13] Y. Ephraim, and D. Malah, "Speech enhancement using a minimum mean square error

short-time spectral amplitudo estimator," *IEEE Transactions on Audio, Speech, and Language Processing,* vol. 32, no. 6, pp. 1109-1121, 1984.

[14] R. Weiss, and D. Ellis, "Speech separation using speaker-adapted eigenvoice speech models," *Comput. Speech Lang.,* vol. 24, pp. 16-29, 2010.

[15] S. Roweis, "One microphone source separation," *Adv. Neural Inf. Process. Syst.,* vol. 13, pp. 793-799, 2000.

[16] M. Wohlmayr, M. Stark, and F. Pernkopf, "A probabilistic interaction model for multipitch tracking with factorial hidden markov models," *IEEE Transactions on Audio Speech and Language Processing,* vol. 19, no. 4, pp. 799-810, May, 2011.

[17] Q. Wang, W. L. Woo, and S. S. Dlay, "Informed Single-Channel Speech Separatoin Using HMM-GMM User-Generated EWxemplar Source," *IEEE Transactions on Audio Speech and Language Processing,* vol. 22, pp. 2087-2100, Dec, 2014.

[18] D. Reynolds, "Gaussian Mixture Models," *Encyclopedia of Biometric Recognition, Springer*, 2008.

[19] S. T. Roweis, "One microphone source separation," *Advances in Neural Information Processing Systems,* vol. 13, pp. 793-799, 2001.

[20] L. Benaroya, and F. Bimbot, "Wiener based source separation with HMM/GMM using a single sensor," *Proc. Int. Conf. Ind. Compon. Anal. Blind Source Separation (ICA'03)*, pp. 957-961, 2003.

[21] A. Simon, O. Alexey, G. Remi *et al.*, "Blind Spectral-GMM Estimation for Underdetermined Instantaneous Audio Source Separation," *8th International Conference on Independent Component Analysis and Signal Separation,* vol. 5441, 2009.

[22] Z. Ghahramani, "An introduction to hidden Markov models and Bayesian networks," *International Journal of Pattern Recognition and Artificial Intelligence,* vol. 15, pp. 9-42, 2001.

[23] R. J. Weiss, and D. P. W. Ellis, "Speech separation using speaker-adapted eigenvoice speech models," *Computer Speech and Language*, 2008.

[24] R. J. Weiss, and D. P. W. Ellis, "Speech separation using speaker-adapted eigenvoice speech models," *Computer Speech and Language,* vol. 24, no. 1, pp. 16-29, 2010.

[25] B. Gao, W. L. Woo, and S. S. Dlay, "Adaptive Sparsity Nonnegative Matrix Factorization for Single Channel Source Separation," *IEEE Journal of Selected Topics in Signal Processing,* vol. 5, no. 5, pp. 989-1001, 2011.

[26] N. Bertin, R. Badeau, and E.Vincent, "Enforcing harmonicity and smoothness in Bayesian non-negative matrix factorization applied to polyphonic music transcription," *IEEE Transactions on Audio Speech and Language Processing,* vol. 18, no. 3, pp. 538-

549, 2010.

[27] E. Vincent, N. Bertin, and R. Badeau, "Adaptive harmonic spectral decomposition for multiple pitch estimation," *IEEE Transactions on Audio Speech and Language Processing,* vol. 18, no. 3, pp. 528-537, 2010.

[28] P. Smaragdis, and J. C. Brown, "Non-negative matrix factorization for polyphonic music transcription," *IEEE Workshop on Appl. Signal Processing to Audio and Acoustics (WASPAA)*, pp. 177-180, 2003.

[29] R. Zdunek, and A. Cichocki, "Nonnegative matrix factorization with constrained second-order optimization," *Signal Processing,* vol. 87, no. 8, pp. 1904-1916, Aug, 2007.

[30] P. Sajda, S. Du, T. Brown *et al.*, "Non-negative matrix factorization for rapid recovery of constituent spectra in magnetic resonance chemical shift imaging of the brain," *IEEE Transactions on Medical Imaging,* vol. 23, no. 12, pp. 1453-1465, 2004.

[31] T. Virtanen, "Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria," *IEEE Transactions on Audio Speech and Language Processing,* vol. 15, no. 3, pp. 1066-1074, 2007.

[32] M. Kim, J. Yoo, K. Kang *et al.*, "Nonnegative Matrix Partial Co-Factorization for Spectral and Temporal Drum Source Separation," *IEEE Journal of Selected Topics in Signal Processing,* vol. 5, no. 6, pp. 1192-1204, Oct, 2011.

[33] K. Kwon, J. W. Shin, and N. S. Kim, "Target source separation based on discriminative nonnegative matrix factorization incorporating cross-reconstruction error," *IEICE Trans. Information and Systems*, no. 11, pp. 2017-2020, 2015.

[34] F. Weninger, J. L. Roux, J. R. Hershey *et al.*, *Discriminative NMF and its application to single-channel source separation*, 2014.

[35] Z. Wang, and F. Sha, "Discrimivative non-negative matrix factorization for single-channel speech separation," *ICASSP*, 2014.

[36] J. L. Roux, F. Weninger, and J. R. Hershey, *Sparse NMF – half-baked or well done?*, 2015.

[37] T. Virtanen, and A. Klapuri, "Analysis of polyphonic audio using source-filter model and non-negative matrix factorization," *Proc. Advances in Models for Acoustic Proc., Neural Info. Proc. Syst.*, 2006.

[38] A. Klapuri, "Analysis of musical instrument sounds by source-filter-decay model," *ICASSP,* vol. 1, pp. I53-I56, 2007.

[39] H. Kameoka, and K. Kashino, "Composite autoregressive system for sparse souece-filter representation of speech," *ISCAS*, pp. 2477-2480, 2009.

[40] D. D. Lee, and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature,* vol. 401, no. 6755, pp. 788-791, 1999.

[41] B. Gao, "Single Channel Blind Source Separation," Electrical, Electronic and Computer Engineering, Newcastle University, 2011.

[42] B. Gao, W. L. Woo, and S. S. Dlay, "Unsupervised single channel separation of nonstationary signals using gammatone filterbank and Itakura-Saito nonnegative matrix two-dimensional factorizations," *IEEE Trans. on Circuits and Systems I,* vol. 60, pp. 662-675, Mar, 2013.

[43] M. N. Schmidt, and M. Mørup, "Nonnegative matrix factor 2D deconvolution for blind single channel source separation," *6th International Conference on Independent Component Analysis and Blind Source Separation, Chareston, SC*, pp. 700-707, 2006.

[44] A. AI-Tmeme, W. L. Woo, S. S. Dlay *et al.*, "Underdetermined Convolutive Source Separation using GEM-MU with Variational Approximated Optimum Model Order NMF2D," *IEEE Transactions on Audio Speech and Language Processing,* vol. 25, no. 1, pp. 35-49, 2017.

[45] M. N. Schmidt, and M. Morup, "Nonnegative matrix factor 2-D deconvolution for blind single channel source separation," *Conf. on Independent Component Analysis and Signal Separation (ICA '06)*, pp. 700-707, 2006.

[46] A. S. Bregman, *Auditory scene analysis: The perceptual organization of sound*: The MIT Press, 1994.

[47] D. L. Wang, "On ideal binary mask as the computational goal of auditory scene analysis," *Speech Sep. Humans Mach.,* vol. 60, pp. 63-64, 2005.

[48] D. L. Wang, and G. J. Brown, *Computational Auditory Scene Analysis: Principles, ALgorithms and Applications*, Hoboken, NJ: Wiley-IEEE Press, 2006.

[49] N. Li, and P. Loizou, "Factors influencing intelligibility of ideal binary-masked speech: Implications for noise reduction," *Journal of the Acoustical Society of America,* vol. 123, no. 3, pp. 1673-1682, 2008.

[50] D. Brungart, P. Chang, B. Simpson *et al.*, "Isolating the energetic component of speech-to-speech masking with ideal time-frequency segregation," *Journal of the Acoustical Society of America,* vol. 120, pp. 4007-4018, 2006.

[51] M. Anzalone, L. Calandruccio, K. Doherty *et al.*, "Determination of the potential benefit of time-frequency gain manipulation," *Ear and Hearing,* vol. 27, no. 5, pp. 480-492, 2006.

[52] Y. Bengio, P. Lamblin, D. Popovici *et al.*, "Greedy layer-wise training of deep networks," *Advances in Neural Information Processing Systems 19 (NIPS'06)*, pp. 153-160, 2007.

[53] D. Erhan, P. A. Manzagol, Y. Bengio *et al.*, "The difficulty of training deep architectures and the effect of unsupervised pretraining," *Proceedings of The Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS'09)*, pp. 153-160, 2009.

[54] G. E. Hinton, and R. R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks," *Science Magazine,* vol. 313, pp. 504-507, 2006.

[55] G. E. Hinton, S. Osindero, and Y. W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Computation,* vol. 18, pp. 28, July 2006, 2006.

[56] G. E. Hinton, *To recognize shapes, first learn to generate images*, University of Toronto, 2006.

[57] N. L. Roux, and Y. Bengio, "Representational power of restricted boltzmann machines and deep belief networks," *Neural Computation,* vol. 20, no. 6, pp. 1631-1649, 2008.

[58] Y. Wang, and D. L. Wang, "Towards scaling up classification-based speech separation," *IEEE Transactions on Audio Speech and Language Processing,* vol. 21, no. 7, pp. 1381-1390, Jul, 2013.

[59] Y. Wang, and D. L. Wang, "Cocktail party processing via structured prediction," *In Advances in Neural Information Processing Systems*, pp. 224-232, 2012.

[60] A. Narayanan, and D. L. Wang, "Ideal ratio mask estimation using deep neural networks for robust speech recognition," *Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing*, pp. 7092-7096, 2013.

[61] Y. Xu, J. Du, L. Dai *et al.*, "An experimental study on speech enhancement based on deep neural networks," *IEEE Signal Processing Letters*, pp. 66-68, 2014.

[62] Y. Xu, J. Du, L. R. Dai *et al.*, "A Regression Approach to Speech Enhancement Based on Deep Neural Networks," *IEEE Transactions on Audio, Speech, and Language Processing,* vol. 23, no. 1, pp. 7-19, Jan, 2015.

[63] P. S. Huang, M. Kim, M. H. Johnson *et al.*, "Deep learning for monaural speech separation," *IEEE International Conference on Acoustics Speech and Signal Processing*, pp. 1562-1566, 2014.

[64] P. S. Huang, M. Kim, M. H. Johnson *et al.*, "Singing-voice separation from monaural recordings using deep recurrent neural networks," *International Society for Music Information Retrieval (IS-MIR)*, 2014.

[65] Y. Tu, J. Du, Y. Xu *et al.*, "Speech separation based on improved deep neural networks with dual outputs of speech features for both target and interfering speakers," *Chinese Spoken Language Processing (ISCSLP), 2014 9th International Symposium on*, 2014.

[66] A. A. Nugraha, A. Liutkus, and E. Vincent, "Multichannel audio source separation with deep neural networks," *IEEE Transactions on Audio, Speech, and Language Processing,* vol. 24, no. 9, pp. 1652-1664, Jun, 2016.

[67] Z. Jin, and D. L. Wang, "A supervised learning approach to monaural segregation of reverberant speech," *IEEE Transactions on Audio Speech and Language Processing,* vol. 17, no. 4, pp. 625-638, May, 2009.

[68] D. Wu, W. L. Woo, and S. S. Dlay, “NMF-2D-based source separation using extreme learning machine,” *Proc. of 2nd IET Int. Conf. in Intelligent Signal Processing*, pp. 1-5, 2015.

[69] G. B. Huang, Q. Y. Zhu, and C. K. Siew, “Extreme learning machine: Theory and applications,” *Neurocomputing,* vol. 70, pp. 489-501, 2006.

[70] J. Cao, Z. Lin, G. B. Huang *et al.*, “Voting based extreme learning machine,” *Information Sciences,* vol. 185, no. 1, pp. 66-77, 2012.

[71] L. L. C. Kasun, H. Zhou, G. B. Huang *et al.*, “Representational Learning with Extreme Learning Machine for Big Data,” *IEEE Intelligent Systems,* vol. 28, no. 6, 2013.

[72] G. B. Huang, L. Chen, and C. K. Siew, “Universal approximation using incremental constructive feedforward networks with random hidden nodes,” *IEEE Trans. Neural Network,* vol. 17, no. 4, pp. 879-892, 2006.

[73] G. B. Huang, M. B. Li, L. Chen *et al.*, “Incremental extreme learning machine with fully complex hidden nodes,” *Neurocomputing,* vol. 71, no. 4, pp. 576-583, 2008.

[74] G. B. Huang, “An insight into extreme learning machines: Random neurons, random features and kernels,” *Cognit. Comput,* vol. 6, no. 3, pp. 376-390, 2014.

[75] J. X. Tan, W. W. Deng, and G. B. Huang, “Extreme Learning Machine for Multiplayer Perceptron,” *IEEE Transactions on Neural Network and Learning System,* vol. 7, no. 4, pp. 809-821, Apr, 2015.

[76] S. Ding, N. Zhang, X. Xu *et al.*, “Deep Extreme Learning Machine and Its Application in EEG Classification,” *Mathematical Problems in Engineering,* vol. 2015, 2015.

[77] H. Larochelle, Y. Bengio, J. Louradour *et al.*, “Exploring strategies for training deep neural networks,” *Journal of Machine Learning Research,* vol. 10, pp. 1-40, 2009.

[78] Y. Bengio, *Learning deep architectures for AI*, Universit ´e de Montr´eal, 2007.

[79] G. E. Hinton, S. Osindero, and Y. W. Teh, “A fast learning algorithm for deep belief nets,” *Neural Computation,* vol. 18, no. 7, pp. 1527-1554, 2006.

[80] L. Deng, and D. Yu, *Deep Learning: Methods and Applications*, 2014.

[81] H. Rue, and L. Held, *Gaussian Markov random fields : theory and applications*, 2005.

[82] D. Wu, “Human Action Recognition Using Deep Probabilistic Graphical Models,” Department of Electronic and Electrical Engineering, The University of Sheffield, 2016.

[83] G. Hinton, *A practical guide to training restricted Boltzmann machines*, Univ. of Toronto, Toronto, ON, Canada, Tech. Rep. 2010-003, 2010, 2010.

[84] K. Yu, W. Xu, and Y. Gong, “Deep learning with kernel regularization for visual recognition,” in Neural Information Processing Systems - NIPS, Hyatt Regency Vancouver, in Vancouver, B.C., Canada, 2008, pp. 1889-1896.

[85] B. Ribeiro, and N. Lopes, "Deep Belief Networks for Financial Prediction," *Lecture*

Notes in Computer Science pp. 766-773.

[86] S. Haykin, *Neural Networks and Learning Machines*.

[87] Y. Bengio, A. Courville, and P. Vincent, “Representation Learning: A Review and New Perspectives,” *Pattern Analysis and Machine Intelligence, IEEE Transactions* vol. 35, no. 8, pp. 1798-1828, Mar. 2013, 2013.

[88] Y. Bengio, "Deep Learning of Representations: Looking Forward." p. 24.

[89] L. Arnold, S. Rebecchi, S. Chevallier *et al.*, “An introduction to Deep Learning,” in Advances in Computational Intelligence and Machine Learning, ESANN'2011, Bruges, Belgium, 2011, pp. 477-488.

[90] I. Arel, D. C. Rose, and T. P. Karnowski, “Deep Machine Learning-A New Frontier in Artificial Intelligence Research,” *Computational Intelligence Magazine,* vol. 5, no. 4, pp. 6, Nov. 2010, 2010.

[91] N. Tengtrairat, and W. L. Woo, “Single-Channel Separation using Underdetermined Blind Method and Least Absolute Deviation,” *Neurocomputing,* vol. 147, pp. 412-425, 2015.

[92] Y. Xiang, S. K. Ng, and V. Nguyen, “Blind Separation of Mutually Correlated Sources Using Precoders,” *IEEE Transactions on Neural Networks,* vol. 21, no. 1, pp. 82-90, Jan, 2010.

[93] G. B. Huang, H. Zhou, and R. Zhang, “Extreme Learning Machine for Regression and Multiclass Classification,” *IEEE Trans. Systems, Man, and Cybernetics,* vol. 42, pp. 513-529, 2011.

[94] G. B. Huang, S. Song, J. N. D. Gupta *et al.*, “Semi-Supervised and Unsupervised Extreme Learning Machines,” *IEEE Trans., Cybernetics,* vol. 44, pp. 2405-2417, 2014.

[95] D. Serre, “Matrices: Theory and Applications,” *SpringerVerlag*, 2002.

[96] A. Ozerov, C. Fevotte, R. Blouet *et al.*, “Multichannel Nonnegative Tensor Factorization with Structured Constraints for User-Guided Audio Source Separation,” *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 257-260, 2011.

[97] F. D. Neeser, and J. L. Massey, “Proper Complex Random-Processes with Applications to Information-Theory,” *IEEE Transactions on Information Theory,* vol. 39, no. 4, pp. 1293-1302, 1993.

[98] A. Ozerov, and C. Fevotte, “Multichannel Nonnegative Matrix Factoruzation in Convolutive Mixtures for Audio Source Separatioin,” *IEEE Transactions on Audio, Speech, and Language Processing,* vol. 18, no. 3, pp. 550-563, 2010.

[99] F. D. Neeser, and J. L. Massey, “Proper complex random-processes with applications to information-theory,” *IEEE Trans. Inf. Theory,* vol. 39, no. 4, pp. 1293-132, 1993.

[100] J. Tang, C. Deng, and G. B. Huang, "Extreme Learning Machine for Multilayer Perceptron," *IEEE Trans., Neural Networks and Learning Systems*, 2015.

[101] A. Beck, and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal of Imaging Sciences,* vol. 2, no. 1, pp. 183-202, 2009.

[102] A. Beck, and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm with application to wavelet-based image deblurring," *IEEE Intel. Conf. on Acoustics, Speech and Signal Processing*, pp. 693-696, 2009.

[103] M. Goodwin, "The STFT, sinusoidal models, and speech modification," *Springer Handbook of Speech Processing*, pp. 229-258, 2008.

[104] C. Fevotte, N. Bertin, and J. L. Durrieu, "Nonnegative matrix factor- ization with the Itakura-Saito divergence With application to music analysis," *Neural Computation,* vol. 21, no. 3, pp. 793-830, 2009.

[105] J. Barker, E. Vincent, N. Ma *et al.*, "The PASCAL CHiME speech separation and recognition challenge," *Computer Speech and Language,* vol. 27, pp. 621-633, 2013.

[106] M. Goto, H. Hashiguchi, T. Nishimura *et al.*, "RWC music database: Music genre database and musical instrument sound database," *Proc. Int. Symp. Music Inf. Retrieval (ISMIR)*, pp. 229-230, 2003.

[107] Y. Bengio, "Learning Deep Architectures for AI," *Foundations and Trends in Machine Learning,* vol. 2, no. 1, pp. 1-127, 2009.

[108] C. Fevotte, N. Bertin, and J. L. Durrieu, "Nonnegative matrix factor- ization with the Itakura-Saito divergence. With application to music analysis," *Neural Computation,* vol. 21, no. 3, pp. 793-830, 2009.

[109] "Mel Frequency Cepstral Coefficient (MFCC) tutorial," http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/

[110] R. Polikar, "Ensemble learning," *Scholarpedia,* vol. 4, no. 1, pp. 2776, 2009.

[111] T. Xia, D. C. Tao, T. Mei *et al.*, "Multiview Spectral Embedding," *IEEE Transactions on Systems, Man, and Cybernetics,* vol. 40, no. 6, pp. 1438-1446, Dec, 2010.

[112] W. Kim, and R. Stern, "Mask classification for missing-feature reconstruction for robust speech recognition with unknown background noise," *Speech Commun.,* vol. 53, no. 1, pp. 1-11, 2011.

[113] G. Hu, and D. L. Wang, "Monaural speech segregation based on pitch tracking and amplitude modulation," *IEEE Transactions on Neural Network,* vol. 15, no. 5, pp. 1135-1150, Sep, 2004.

[114] G. B. Huang, Q. Y. Zhu, and C. K. Siew, "Extreme learning machine: Theory and applications," *Neural Computing,* vol. 70, pp. 489-501, Dec, 2006.

[115] Y. M. Yang, and Q. M. J. Wu, “Extreme Learning Machine with subnetwork hidden nodes for reression and classification,” *IEEE Transactions on Cybernetics,* vol. 46, no. 12, pp. 2885-2898, Dec, 2016.

[116] G. B. Huang, H. Zhou, X. Ding *et al.*, “Extreme learning machine for regression and multiclass classification,” *IEEE Transactions on Systems, Man, and Cybernetics,* vol. 42, no. 2, pp. 513-529, Apr, 2012.

[117] D. Ellis, "PLP and RASTA (and MFCC, and Inversion) in Matlab," 2005.

[118] G. Kim, Y. Lu, Y. Hu *et al.*, “An algorithm that improves speech intelligibility in noise for normal-hearing listeners,” *Acoust. Soc. Amer.,* vol. 126, no. 3, pp. 1486-1494, 2009.

[119] H. Hermansky, and N. Morgan, “RASTA processing of speech,” *IEEE Transactions on Audio Speech and Language Processing,* vol. 2, no. 4, pp. 578-589, Oct, 1994.

[120] Y. Wang, and D. L. Wang, “Exploring Monaural Features for Classification-Based Speech Segregation,” *IEEE Transactions on Audio Speech and Language Processing,* vol. 21, no. 2, pp. 270-279, Feb, 2013.

[121] L. Shao, D. Wu, and X. Li, “Learning deep and wide: A spectral method for learning deep networks,” *IEEE Transactions on Neural Network and Learning System,* vol. 25, no. 12, pp. 2303-2308, Dec, 2014.

[122] G. Garau, and S. Renals, “Combining Spectral Representations for Large-Vocabulary Continuous Speech Recognition,” *IEEE Transactions on Audio, Speech, and Language Processing,* vol. 16, no. 3, pp. 508-518, Mar, 2008.

[123] J. C. Bezdek, and R. J. Hathaway, “Some notes on alternating optimization,” *Proc. AFSS Int. Conf. Fuzzy Syst.,* vol. 2275, pp. 288-300, 2002.

[124] R. Bhatia, *Matrix Analysis*: New York: Springer-Cerlag, 1997.

[125] C. H. Taal, R. C. Hendriks, R. Heusdens *et al.*, “An algorithm for intelligibility prediction of time-frequency weighted noisy speech,” *IEEE Transactions on Audio Speech and Language Processing,* vol. 19, no. 7, pp. 2125-2136, Sep, 2011.

[126] " Perceptual Evaluation of Speech Quality (PESQ) Measurement Description," http://rfmw.em.keysight.com/rfcomms/refdocs/cdma2k/cdma2000_meas_pesq_desc.html.