Copula Models for Multivariate Density Estimation, Classification and Robust Speech Recognition

Alireza Bayestehtashk

M. S. Electrical Engineering
Amirkabir University of Technology-Tehran Polytechnic, 2008

Presented to the
Center for Spoken Language Understanding
within the Oregon Health & Science University
School of Medicine
in partial fulfillment of
the requirements for the degree
Doctor of Philosophy
in
Computer Science & Engineering

Center for Spoken Language Understanding
School of Medicine
Oregon Health & Science University

CERTIFICATE OF APPROVAL

This is to certify that the Ph. D. dissertation of

Alireza Bayestehtashk

has been approved.

Izhak Shafran, Thesis Advisor
Research Scientist, Google Inc.

Alexander Kain
Associate Professor

Xubo Song
Professor

Peter A. Heeman
Associate Professor

Andrew W. Senior
Research Scientist, Google Inc.

Dedication

To my family, specially my mother and father, for their endless support.

Acknowledgements

I would like to express my deepest gratitude to my advisor, Prof. Izhak Shafran, for his insightful guidance during all these years. Special thanks to the members of my thesis committee, Prof. Peter Heeman, Prof. Alexander Kain, Dr. Andrew W. Senior and Prof. Xubo Song for their constructive and invaluable comments. I also want to thank Dr. Amir Babaeian for his comments on KL mapping. I would like to say thanks to all of the faculty members at CSLU specially Prof. Meysam Asgari, Prof. Steven Bedrick, Prof. Steven Wu, and Prof. Jan van Santen for their enormous help during my PhD studies. Thanks to the staff of CSLU, Patricia Dickerson, Robert Stites and Ethan VanMatre for their continued support. Finally, it was impossible to finish this work without the support of my family who have endured my seven years absence from home with patience.

Contents

List of Tables

List of Figures

Abstract

Copula Models for Multivariate Density Estimation, Classification and Robust Speech Recognition

Alireza Bayestehtashk

Doctor of Philosophy
Center for Spoken Language Understanding
within the Oregon Health & Science University
School of Medicine

Thesis Advisor: Izhak Shafran

Abstract

Univariate distributions can be modeled accurately and efficiently using nonparametric kernel density estimators, which unfortunately cannot be easily extended to multivariate distributions. As an alternative, Gaussian mixture models are used to approximate multivariate distributions, especially because their estimation is relatively straight forward through the Expectation Maximization (EM) algorithm. Multivariate Gaussian mixture models implicitly impose a Gaussian mixture distribution on the marginal distributions. This is a strong assumption on the marginal distributions and is violated in many practical applications. Copula models disentangle the choice of marginal distributions from the dependency structure, making them powerful models for multivariate density estimation. According to copula theory, any distribution can be described by a set of univariate functions and a multivariate function, which is also known as a copula function. The univariate functions shape the marginal distributions and the copula functions represent the dependency among the random variables. In this thesis, we investigate how to harness this decoupling property of copula models to improve the density estimation and build better classifiers and sequence recognizers.

Multivariate Density Estimation and Classification: We introduce two copula models for multivariate density estimation, one of them addresses the data scarcity issue in estimating high dimensional distributions and the other utilizes the decoupling property to improve an already estimated multivariate distribution.

In the copula literature, one of the main challenges is to construct an appropriate copula function, particularly for high dimensional distributions. To address this problem, we propose a relatively straightforward modification to Gaussian copula models, one of the most popular multivariate copula functions. The Gaussian copula models require estimation of full correlation matrices, which are prone to data scarcity in many practical applications, especially in high dimensional distributions. We address this limitation by constraining the correlation matrices to be Toeplitz matrices and offset the loss of modeling capacity by introducing mixtures in a Gaussian mixture copula model.

In certain applications, such as modeling sequences, Gaussian mixture models are

convenient observation models in, for example, Hidden Markov Models, since all the parameters of the model can be easily estimated jointly using the EM algorithm. In such cases, we can replace the Gaussian mixture model with a copula model, whose copula function represents the dependencies already captured in the Gaussian mixture models and improves it further by augmenting the copula function with univariate functions to eliminate the error in the estimation of marginals distributions.

We evaluate the performance of both of the proposed methods on several density estimation tasks from the UCI Repository as well as our corpus of Monkey vocalizations, recorded at the Oregon National Primate Research Center. We find that our methods represent the data consistently better than Gaussian mixture models with equivalent number of parameters. We also evaluate our proposed methods on building generative classifiers for a number of classification tasks from the UCI Repository. We find that these generative models perform as well or better than discriminative classifiers such as a Support Vector Machine (SVM).

Sequence Recognition: One of the key challenges in recognizing sequences is the mismatch in training and testing conditions. In most practical applications, recognition systems utilize supervised training and as a result are privy only to a finite amount of training data. Thus, they fail to contain representative samples of all the conditions under which the model might be deployed in real-world applications.

We propose a novel copula-based feature enhancement method to address the mismatch between the multivariate distribution of features in any test utterance and the corresponding distribution in the training utterances. The method takes advantage of the decoupling property of the marginal univariate functions. Specifically, we estimate an optimal non-linear transformation of the test utterance to reduce the Kullback-Liebler divergence between the two distributions as parameterized by Gaussian copula models.

We report results on the Aurora 4 Automatic Speech Recognition (ASR) task, which contains utterances with a wide range of background noises that are not well represented in the training data. Our results show that the proposed copula-based model improves the accuracy by about 7% over current best results in the literature. In addition to Aurora 4, we use the proposed enhanced features for the 4th CHIME Speech Recognition task.

These features improve the performance of the baseline system by 4.3%, 1.4%, and 0.5% (absolute) for 1-channel, 2-channel and, 6-channel tracks, respectively. Furthermore, we formulate the copula-based transformation as a parametric model and integrate it into a GMM-HMM acoustic model. We utilize a new method to jointly learn the copula-based transformation and the acoustic model. Our results show that the integration of copula-based transformation into the acoustic model leads to further improvements in recognition accuracy on different tracks of CHIME 4.

Chapter 1

Introduction

Multivariate density estimation is a problem that is encountered in a wide variety of disciplines, including but not limited to machine learning, civil engineering, and financial analysis. Conventional approaches for density estimation can be broadly categorized into parametric and nonparametric methods. For parametric methods, we typically assume that the data is generated by a model with some unknown parameters, and our goal is to find a model and its parameters that is the best fit for the data [35]. The choice of the model plays a crucial role for this category. Typically, based on intuition or domain-knowledge, an expert identifies a limited set of predefined parametric models (e.g., Gaussian or Gaussian mixture models with a given number of components) and empirically evaluates their fit on a given training data set to pick the best one. The major drawback of parametric models is that any mismatch (bias) between the true distribution of data and selected model can not be compensated even by increasing the amount of data to infinity. In contrast to parametric methods, nonparametric methods allow the data to determine the complexity of the model; increasing the amount of data results in a more complicated model. The drawback of these methods is that they can not be extended to higher dimensions because of the curse of dimensionality. That is, the amount of training data required to provide the same performance increases exponentially with increases in dimension.

In both parametric and nonparametric estimation of the joint distribution of random variables, ultimately one form of distribution is chosen, often to maximize likelihood. This choice automatically dictates a specific form for univariate marginal distributions which is typically a poor fit for the true marginals. In contrast, when marginals are estimated in isolation, they can be estimated with very high accuracy using, for example, histogram and k-nearest neighbor density estimation. Thus, both conventional parametric and nonparametric approaches do not have enough flexibility to address this mismatch problem.

Copula models provide a novel paradigm to model multivariate distributions that solves the above mentioned mismatch. This model was introduced by Sklar in 1959 [86] and it is mainly recognized because of its success in financial risk prediction [58]. With a copula model, any distribution is comprised of a set of univariate functions and a multivariate function, which is also knows as a copula function. The univariate functions control the

shape of the marginal distributions and the copula function captures the dependency among the random variables. In this paradigm, a multivariate distribution is estimated by fitting a set of univariate marginal distributions and a copula function to data. The main obstacle in copula-based density estimation is the construction of copula function. While there are several bivariate copula functions in the literature [39], the number of copula functions for higher dimension barely exceeds a handful of functions.

In this thesis, we first introduces two multivariate copula functions and demonstrate their effectiveness on density estimation and classification tasks. Second, we propose a non-linear feature transformation to reduce the distributional mismatch between training and test utterances, and demonstrate its effectiveness on a couple of challenging automatic speech recognition tasks such as Aurora 4 [73] and CHIME 4 [92].

1.1 Contributions of the thesis

Mixture of Gaussian copula with Toeplitz structure

In Chapter 3, we investigate a popular copula model – the Gaussian copula model – for high dimensional settings. The standard Gaussian copula functions require estimation of a full correlation matrix, which can cause data scarcity in some settings. One approach to address this problem is to impose constraints on the parameter space. We present Toeplitz correlation structures to reduce the number of Gaussian Copula parameters. To increase the flexibility of our model, we also introduce mixtures of Toeplitz Gaussian Copula as a natural extension of the Gaussian Copula model.

Gaussian mixture model with marginal modification

We also propose a computationally simple method in Chapter 3 to modify the marginal distributions for the conventional density estimators, in particular a previously estimated GMM. This is particularly useful when the GMMs are estimated as a component of a larger model such as a hidden Markov model (HMM).

In addition, we propose a simple generative classification model based on the copula

model that takes advantage of the accuracy of the nonparametric univariate density estimator and the multivariate dependencies captured in the Gaussian mixture model, thus alleviating the aforementioned limitations.

Investigation of proposed copula functions on density estimation and classification

Through empirical evaluation of likelihood on held-out data, we study the trade-off between correlation constraints and mixture flexibility, and report results on the Wine data set from the UCI Repository as well as our corpus of Monkeys' vocalization, recorded at the Oregon National Primate Research Center with the goal of developing automatic methods for recognizing social behaviors of individuals. We find that the mixture of Gaussian Copula with Toeplitz correlation structure and GMM with marginal modification model the data consistently better than Gaussian mixture models with equivalent number of parameters. We compare the performance of our generative classifier with previous classification benchmarks from the UCI repository and show that for the same number of parameters the proposed models consistently outperforms Gaussian mixture models. We find that these generative models perform as well or better than Support Vector Machines (SVM). We also apply the generative classifier to track natural behavior of animals in captivity using continous audio recording, which is often corrupted by unpredictable background noise. This application highlights how the copula model can be used in an application where deployment is likely to encounter noise types that can never be fully represented in a training set.

Robust speech recognition using multivariate copula models

In Chapter 4, we address the mismatch between training and testing distributions which significantly degrades the performance of ASR systems. In this work, we formulate the mismatch in term of the difference between distributions of training and test data, and propose a transformation to make test data similar to training data. Proposed copula based methods provide a generalization to Gaussianization and histogram based approach. While the others are ad hoc, our methods are theoretically motivated. We prove that if the

distributions are modeled using a Gaussian copula model, then there is an analytic form for the transformation. More precisely, we estimate the distribution of feature vectors for each utterance and the entire training data set using the Gaussian Copula Model (GCM). We then find a nonlinear transformation for any utterance in training and testing sets to match the distribution of utterances with the distribution of the entire training data. Furthermore, we formulate the copula-based transformation as a parametric model and integrate it into GMM-HMM acoustic model. We also propose a new method to jointly learn the copula-based transformation and the acoustic model.

1.2 Thesis Overview

Chapter 2 starts with an introduction to the copula model and different multivariate copula functions. Then, we propose two computationally simple approaches to built multivariate copula functions in Chapter 3. Finally, we evaluate these approaches in different density estimation and classification tasks. In chapter 4, we propose two simple methods to address the mismatch between training and testing conditions, which is a major bottleneck in everyday applications of ASR systems. First, we introduce a simple copula-based feature enhancement method independent of the ASR backend. Later in this chapter, we show that this enhancement can be further optimized by integrating it into the acoustic model. For evaluation, we use large vocabulary speech recognition tasks on Aurora 4 and Chime4. Chapter 5 gives the conclusions and future work.

Chapter 2

Related Work

2.1 Copula Model In A Nutshell

Multivariate density estimation is a crucial task especially in generative models as needed in many practical tasks, such as anomaly detection and classification in machine learning [63], analysis of highway bridge traffic loading in civil engineering [23], risk management in finance [64], analysis of drought in climate research [24], etc. Conventional density estimation methods, such as mixture models and kernel-based methods, typically assume a single parametric form for a joint density function. And the choice of joint density function automatically dictates a specific form for marginal distributions, which is often too simple to capture the marginal distributions for practical applications. However, the marginal distribution of each variable can be estimated more efficiently using non parametric approaches such as kernel density estimation [84]. A common problem in conventional techniques is that there is no flexibility in choosing the form of the marginal distributions even when such a misfit is known a priori. Except for the mathematical convenience, there is no widely accepted reason to couple joint density functions and marginal distributions in the literature. The problem can be overcome by decoupling the choice of marginal distributions from joint density function. Sklar's theorem provides the necessary theoretical foundation to decouple these choices [85].

Sklar's theorem states that any continuous Cumulative Distribution Function (CDF) $F(\mathbf{x})$ can be uniquely written in terms of a copula function $C(\mathbf{u})$ and a set of univariate marginal CDF $\{F_i(x_i)\}_{i=1}^n$ as follows:

$$F(x_1, x_2, \ldots, x_n) = C(F_1(x_1), F_2(x_2), ..., F_2(x_n)) \tag{2.1}$$

where $F(\mathbf{x})$ is an n-dimensional CDF and $C(\mathbf{u})$ is a special function known as a copula CDF in which the domain of $\mathbf{u}$ is bound to the unit hypercube $[0, 1]^n$. If $F(\mathbf{x})$ is differentiable, the Probability Density Function (PDF) $f(\mathbf{x})$ can be computed by taking derivatives of Equation (2.1) with respect to $\mathbf{x}$ as follows :

$$f(\mathbf{x}) = \frac{\partial^n C(F_1(x_1), \ldots, F_n(x_n))}{\partial x_1 \cdots \partial x_n} \tag{2.2}$$

where $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$. By applying the chain rule of calculus to Equation (2.2):

$$\begin{aligned} f(\mathbf{x}) &= \frac{\partial^n C(F_1(x_1), \ldots, F_n(x_n))}{\partial F_1(x_1) \cdots \partial F_n(x_n)} \\ &\quad \times \Pi_{i=1}^n \frac{dF_{x_i}(x_i)}{dx_i} \\ &= c(F_1(x_1), F_2(x_2), \ldots, F_n(n)) \Pi_{i=1}^n f_i(x_i) \end{aligned} \tag{2.3}$$

where $\{f_i(x_i)\}_{i=1}^n$ are univariate marginal PDFs of $f(\mathbf{x})$ and $c(\mathbf{u})$ is the copula PDF. Equation (2.3) shows that any continuous PDF can be factorized into the product of a set of univariate marginal PDFs and a copula function. The role of the copula function is to bind the marginal distributions and shapes the dependency structure among the random variables for a multivariate distribution. That also shows why the name 'copula', which means link in Latin, is a descriptive term for this function.

From a generative perspective, Equation (2.3) indicates that any continuous PDF can be constructed by choosing a copula function and a set of marginal distributions. Furthermore, the choice of copula function can be independent of the marginal distribution.

In Figure 2.1, we generate two toy distributions by choosing similar marginal distributions and different copula functions and in Figure 2.2: (a) and (b), we show two different distributions with the same copula function while their univariate marginal distributions are different.

Equation (2.3) also provides a framework for estimating multivariate distributions. Instead of estimating the PDF using a single function, we can estimate the PDF by estimating univariate marginal distributions and copula function. We choose proper parametric forms for univariate marginal CDFs and copula function. Then, we optimize the log likelihood function with respect to all of the parameters. There are several well-known two dimensional Copula function [39] but constructing an appropriate multivariate copula function is still a challenging task.

2.2 Multivariate Copula Models

Generally, the construction of multivariate copula functions can be categorized into three different classes : Archimedean, Vine and Gaussian Models [72].

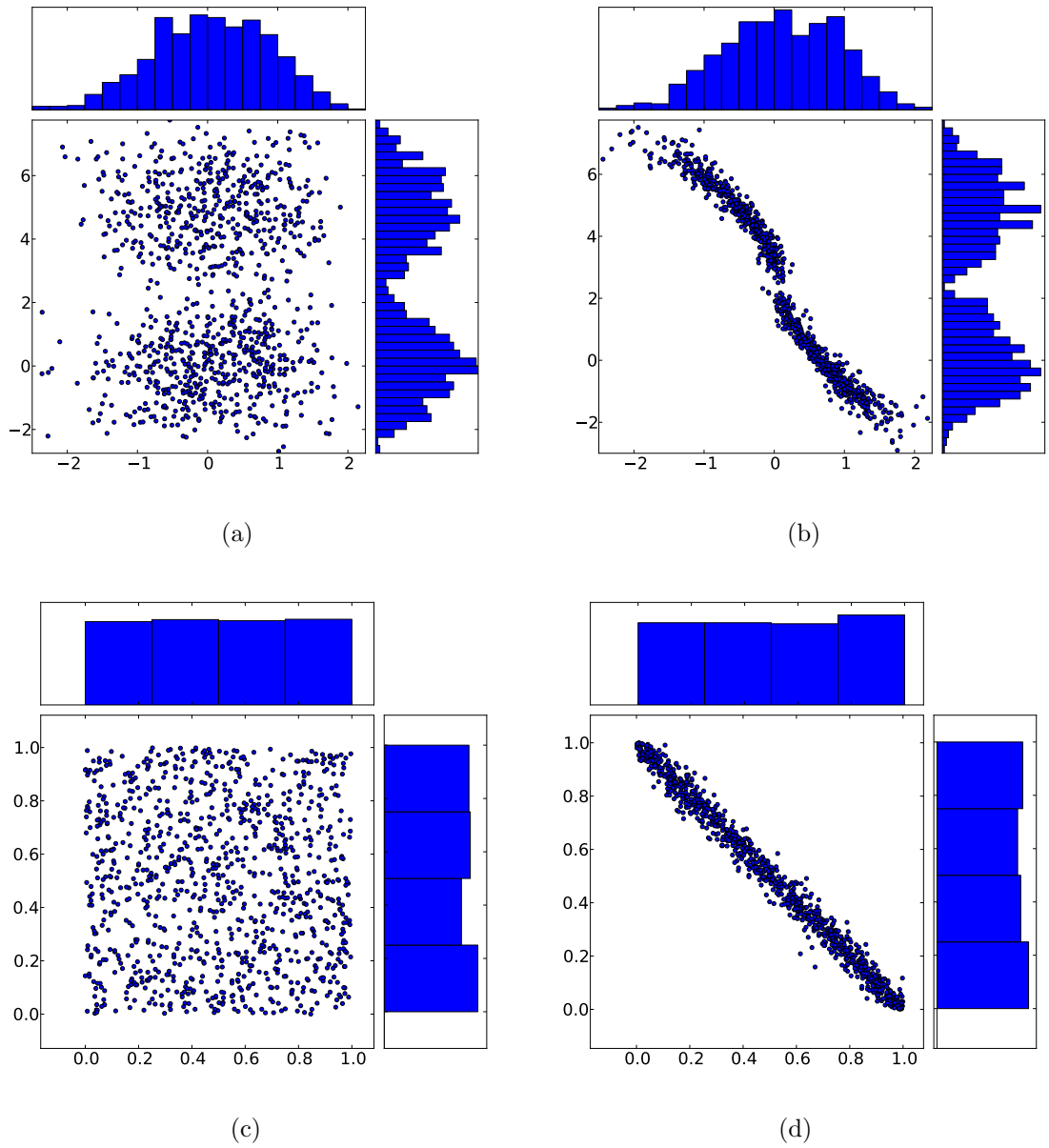
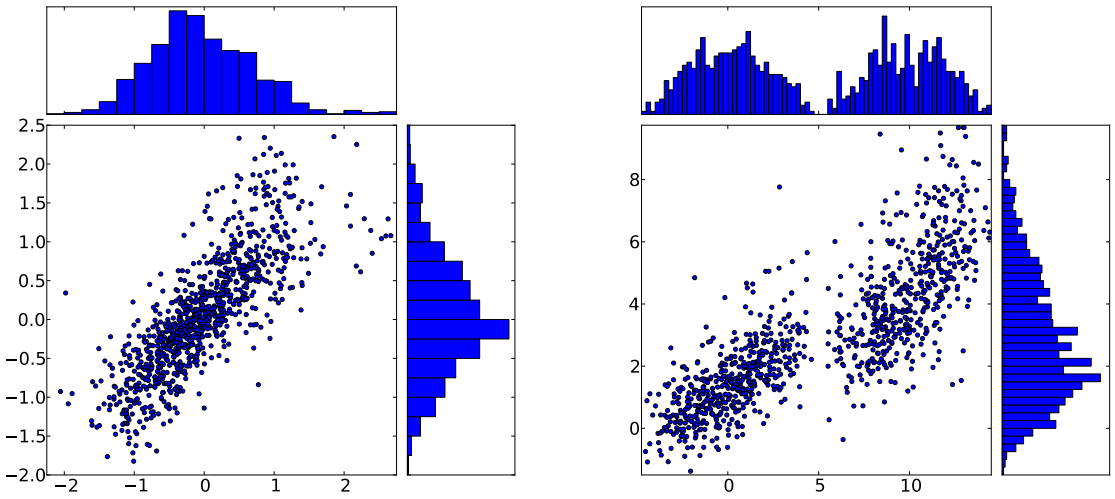
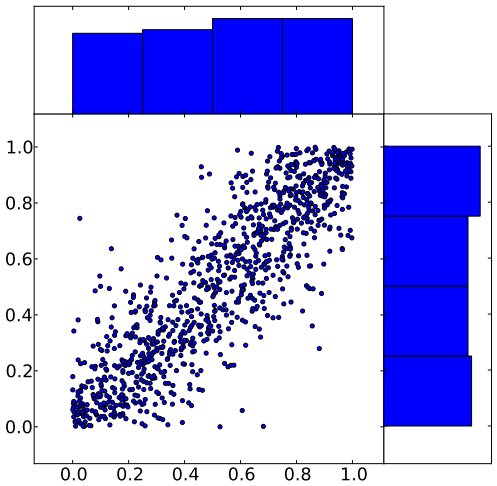


Figure 2.1: (a) and (b) are two different distributions generated by (c) and (d) copula functions respectively. As depicted, the marginal distributions of (a) and (b) along each axis are similar. Histogram along each axis represents an estimation of the marginal distribution along that axis. As depicted, copula distributions always have Uniform marginal distributions.



(a) (b)



(c)

Figure 2.2: (a) and (b) have been generated by different marginals and a single copula density in (c). As depicted, copula distribution always has Uniform marginal distributions.

2.2.1 Archimedean Copula

Perhaps the most common class is Archimedean copula [72], which is defined as:

$$c(\mathbf{u};\theta) = \varphi(\varphi^{-1}(u_1;\theta) + \ldots + \varphi^{-1}(u_n;\theta);\theta) \tag{2.4}$$

where φ is a generator function. There are multiples ways to define a generator function. Gumbel–Hougaard, Mardia–Takahasi–Clayton and Frank families are some of well-known Archimedean generators. See [72] for details. The major drawback of the Archimedean copula is that generator functions typically have one free parameter, regardless of dimension, which limits their applications for high dimensional problems. However, there has been some progress on increasing the number of free parameters of generator functions in Archimedean copula but they are computationally expensive to estimate [46].

2.2.2 Vine Copula

Pair-copula construction, also known as vine copula, is another approach to construct multivariate copula function. The idea is to use graphical models to decompose multivariate copula function into a set of bivariate copula functions. A tree average copula function [50] uses a tree structure graphical model for random variables. This method shows that a multivariate copula function can be factorized into several bivariate copula functions. More precisely, let $\mathcal{T} = (\mathcal{X}, \mathcal{E})$ be an undirected graph model with a tree structure. Each element in the node set $\mathcal{X}$ is a random variable and $\mathcal{E}$ is a set of pairs that encodes local dependencies in the graph $\mathcal{T}$. Its corresponding joint density can be written:

$$f(X) = \prod_{i=1}^{n} f_i(x_i) \prod_{(i,j)\in\mathcal{E}} \frac{f_{i,j}(x_i, x_j)}{f_i(x_i) f_j(x_j)} \tag{2.5}$$

where $f_{i,j}(x_i, x_j)$ is the bivariate density function between x_i and x_j. Then we rewrite the bivariate density function based on the Copula model using Equation (2.3) giving:

$$f_{i,j}(x_i, x_j) = c_{i,j}(F_i(x_i), F_j(x_j)) f_i(x_i) f_j(x_j) \tag{2.6}$$

where $c_{i,j}(\cdot,\cdot)$ is the bivariate Copula density function between x_i and x_j. The Copula function for a tree-structured density can be computed by combining Equations (2.5) and

(2.6) :

$$c_{\mathcal{E}}(F_1(x_1), F_2(x_2), \ldots, F_n(x_n)) = \prod_{(i,j)\in\mathcal{E}} c_{i,j}(F_i(x_i), F_j(x_j)) \tag{2.7}$$

Equation (2.7) shows that a n-dimensional Copula density function $c_{\mathcal{T}}(\cdot)$ can be factorized into the bivariate Copula density functions by imposing a tree structure over the n-dimensional destiny function. Equation (2.7) also makes the construction of the n-dimensional Copula density function trackable because there are several well-studied bivariate Copula density functions available to choose from. However, the assumption of the tree structure is too restrictive for real data sets and it does not have the flexibility to model complicated density functions accurately. Bayesian mixture of all possible spanning trees has been proposed to alleviate this limitation. It uses a prior distribution over all possible spanning trees for a graph with n nodes, as shown in Equation (2.8).

$$P(\mathcal{E}|\beta) = \frac{1}{Z} \prod_{(i,j)\in\mathcal{E}} \beta_{i,j} \tag{2.8}$$

where $Z = \sum_{\mathcal{E}} \Big[\prod_{(i,j)\in\mathcal{E}} \beta_{i,j}) \Big]$ and β is a edge weight matrix. The average of all the tree-structured Copula density functions $c_{avg}(\mathbf{a}; \beta)$ is still a valid Copula density function and can be obtained by combining Equations (2.7) and (2.8) as shown in Equation (2.9).

$$\begin{aligned} c_{avg}(U;\beta) &\equiv \sum_{\mathcal{E}} P(\mathcal{E}|\beta) c_{\mathcal{E}}(U) \\ &= \sum_{\mathcal{E}} \Big[\frac{1}{Z} \prod_{(i,j)\in\mathcal{E}} \beta_{i,j} c_{i,j}(u_i, u_j) \Big] \end{aligned} \tag{2.9}$$

In this case, the estimation of the Copula model parameters requires a computationally expensive iterative method based on the expectation maximization algorithm.

Copula Bayesian Networks is another method for constructing the multivariate Copula function [31]. It uses the Bayesian Network (BN) to factorize the Copula density function into smaller Copula functions. The BN is a directed acyclic graphical model $G = (\mathcal{X}, \mathcal{E})$ that represents a joint density where $\mathcal{X}$ is a set of nodes and $\mathcal{E}$ is a set of directed edges between two nodes. Using the graph, a joint density $f(X)$ with n random variables can be factorized into a product of simpler conditional density functions $f(\cdot|\cdot)$ that have fewer

variables than n:

$$f_G(X) = \prod_{i=1}^{n} f(x_i|\mathbf{pa}(x_i)) \tag{2.10}$$

where $\mathbf{pa}(x_i)$ is a subset of $\mathcal{X}$. The node x_j is in $\mathbf{pa}(x_i)$ if there is a directed edge in $\mathcal{E}$ from x_j to x_i. The main contribution of Copula BN is to rewrite the conditional density based on the Copula model. Consider the simplest case of a conditional probability density where $\mathbf{pa}(x)$ consists of just one random variable y. By using Equation (2.3), the conditional probability density is

$$f(x|y) = \frac{f(x,y)}{f(y)} = \frac{c(F_x(x), F_y(y))f_x(x)f_y(y)}{f_y(y)} = c(F_x(x), F_y(y))f_x(x) \tag{2.11}$$

Equation (2.11) shows that the conditional probability density for a node with just one parent can be written based on the Copula model. The conditional probability density for a node with more than one parent has the following from:

$$p(x|\mathbf{Y}) = \frac{c(F_x(x), F_{y_1}(y_1), \ldots, F_{y_m}(y_m))}{\frac{\partial^m C(1, F_{y_1}(y_1), \ldots, F_{y_m}(y_m))}{\partial F_{y_1}(y_1), \ldots, F_{y_m}(y_m)}} f_x(x) \tag{2.12}$$

$$\equiv R(F_x(x), F_{y_1}(y_1), \ldots, F_{y_m}(y_m))f_x(x)$$

$$= R(F_x(x), F_{\mathbf{Y}}(\mathbf{Y}))f_x(x) \tag{2.13}$$

where C is a cumulative Copula function and $R(\cdot)$ is the ratio of two Copula densities. The computation of the denominator in Equation (2.13) is as easy as the computation of a standard Copula density function. Combining Equations (2.3) , (2.10) and (2.13) gives:

$$c(F_1(x_1), F_2(x_2), \ldots, F_m(x_m)) =$$

$$\Pi_{i=1}^{n} R_i(F_i(x_i), F_{\mathbf{pa}(x_i)}(\mathbf{pa}(x_i))) \tag{2.14}$$

where R_i is the ratio term of the i-th conditional density in Equation (2.14). Since the number of random variables in each conditional density is limited, the computation of the ratio term R_i is easy. Equation (2.13) provides a way to define a parametric model for high dimensional Copula density function. Copula BN uses a Bayesian information criterion

and greedy forward search to find the best model and its parameters. This method is computationally expensive since the method involves solving a structure learning problem.

As a less expensive alternative, we investigate Gaussian copula, which has a natural extension for high dimensional domain. In certain applications, such as estimating the multivariate Gaussian copula distribution for a speech utterance at test time, the available data may be insufficient for robust estimation of all the parameters of the model. For such cases, we introduce a version of the Gaussian copula with constrained parameterization.

2.2.3 Gaussian Copula

2.2.3.1 Definition

Gaussian Copula density is the most well-known multivariate Copula function and it can be obtained by applying the method of inversion to standard multivariate Gaussian [90]. The combination of Gaussian Copula with arbitrary univariate marginal distributions can represent more complicated densities such as non-elliptical and heavy-tailed densities. Multivariate Gaussian density can be written in a form that separates the marginal from the copula function. Using this decomposition, the derivation below shows how the Gaussian copula can be derived from the Gaussian density.

Definition 1 *An n-dimensional Gaussian density g with the mean vector μ, and the covariance matrix Σ has the following parametric form:*

$$g(X) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|^{\frac{1}{2}}}|\exp\{-\frac{1}{2}(X-\mu)^{\mathsf{T}}\Sigma^{-1}(X-\mu)\} \tag{2.15}$$

where T stands for the transpose. The covariance matrix Σ can be rewritten as:

$$\Sigma = DRD \tag{2.16}$$

where R is the correlation matrix[1], and D is the diagonal matrix of Standard Deviations (SD):

$$D = \begin{bmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \sigma_n \end{bmatrix} \tag{2.17}$$

where σ_i is the SD of x_i. By plugging Equation (2.16) into Equation (2.15) :

$$g(X) = \frac{1}{(2\pi)^{\frac{n}{2}} |R|^{\frac{1}{2}} \prod_{i=1}^{n} \sigma_i} \exp\{-\frac{1}{2} U^T R^{-1} U\} \tag{2.18}$$

where $U = [u_1, \dots, u_n]^T$, $u_i = \frac{x_i - \mu_i}{\sigma_i}$. Equation (2.18) can be rewritten as :

$$\begin{aligned} g(X) &= \Big[\frac{1}{(2\pi)^{\frac{n}{2}} \prod_{i=1}^{n} \sigma_i} \exp\{-\frac{1}{2} U^T U\}\Big] \Big[\frac{1}{|R|^{\frac{1}{2}}} \exp\{-\frac{1}{2} U^T (R^{-1} - I) U\}\Big] \\ &\equiv g_m(U) g_c(U) \end{aligned}$$

where $g_m(Y)$ is :

$$g_m(U) = \frac{1}{(2\pi)^{\frac{n}{2}} \prod_{i=1}^{n} \sigma_i} \exp\{-\frac{1}{2} U^T U\} \tag{2.19}$$

and $g_c(Y)$ is :

$$g_c(U) = \frac{1}{|R|^{\frac{1}{2}}} \exp\{-\frac{1}{2} U^T (R^{-1} - I) U\} \tag{2.20}$$

By substituting $y_i = \frac{x_i - \mu_i}{\sigma_i}$ into Equation (2.19), $g_m(Y)$ can be rewritten as the product of the marginal densities of g :

$$\begin{aligned} g_m(U) &= \prod_{i=1}^{n} \frac{1}{\sigma_i \sqrt{2\pi}} \exp\{-\frac{1}{2} (\frac{x_i - \mu_i}{\sigma_i})^2\} \\ &= \prod_{i=1}^{n} g_i(x_i; \mu_i, \sigma_i) \end{aligned} \tag{2.21}$$

[1] $R_{ij} = \frac{cov(x_i, x_j)}{\sqrt{var(x_i) var(x_j)}}$ is the correlation between x_i and x_j

where g_i is the marginal distribution of x_i. The variable u_i can be rewritten in terms of the MDF:

$$\begin{aligned} G_i(x_i) &= \int_{-\infty}^{x_i} \frac{1}{\sigma_i\sqrt{2\pi}} \exp\{-\frac{1}{2}(\frac{t-\mu_i}{\sigma_i})^2\}\mathrm{d}t \\ &= \int_{-\infty}^{\frac{x_i-\mu_i}{\sigma_i}} \frac{1}{\sqrt{2\pi}} \exp\{-\frac{1}{2}t^2\}\mathrm{d}t \\ &= \int_{-\infty}^{y_i} \frac{1}{\sqrt{2\pi}} \exp\{-\frac{1}{2}t^2\}\mathrm{d}t = \Phi(u_i) \end{aligned} \tag{2.22}$$

where Φ is the cumulative distribution function of the standard normal distribution. By substituting $u_i = \Phi^{-1}(G_i(x_i))$ into Equation (2.20), $g_c(U)$ can be written as :

$$g_c(U) = g_c(\Phi^{-1}(G_1(x_1)), \Phi^{-1}(G_2(x_2)), \ldots, \Phi^{-1}(G_n(x_n))) \equiv c_{gauss}(U; R) \tag{2.23}$$

where Φ^{-1} is the quantile function of standard normal distribution. By substituting Equation (2.21) and (2.23) into Equation (2.17):

$$g(X) = c_{gauss}(U; R) \prod_{i=1}^{n} g_i(x_i; \mu_i, \sigma_i) \tag{2.24}$$

Equation (2.24) shows that an n-dimensional Gaussian density g can be factorized into the product of its marginal, and another term $c_{gauss}(U; R)$. Based on Equation (2.3), $c_{gauss}(U; R)$ provides us a valid parametric form of the Copula density function. Since it is derived from the Gaussian distribution, it is called Gaussian Copula density.

In the general case, the marginal densities do not need to be Gaussian. The Gaussian Copula model can be constructed by substituting the Gaussian Copula density function into Equation (2.3):

$$f(X; R, \Lambda) = c_{gauss}(U; R) \prod_{i=1}^{n} f_i(x_i; \lambda_i) \tag{2.25}$$

Lemma 1 *The main difference between the Gaussian Copula model in Equation* (2.25)*, and standard Gaussian distribution is that the marginal density functions in the Gaussian distribution are necessarily Gaussian while the marginal density functions of the Gaussian Copula model can by any continuous density and this capability makes the Gaussian Copula model more flexible than the Gaussian distribution.*

2.2.3.2 Estimation

There are three methods to estimate the parameters of the Gaussian Copula model: Full Maximum Likelihood (FML), sequential 2-Step Maximum Likelihood (TSML) and Generalized Method of Moments [90]. Since the TSML is more straightforward, we adopt this approach in this thesis. It consists of two steps. The first step is to estimate the marginal (univariate) cumulative functions $\{\hat{F}_i(\cdot)\}_{i=1}^n$ using nonparametric kernel density estimation and map all data points into a new space, the Copula space.

$$U = [\Phi^{-1}(\hat{F}_1(x_1)), \ldots, \Phi^{-1}(\hat{F}_1(x_n))] \tag{2.26}$$

The second step is estimating the parameter of the Gaussian Copula density function R. The correlation matrix R can be computed using maximum likelihood in Copula space.

$$\hat{\boldsymbol{R}} = \underset{\boldsymbol{R}}{\operatorname{argmax}} \sum_{i=1}^{n} \big[- log|R| - U_i^T (R^{-1} - I) U_i \big] \tag{2.27}$$

where n is the number of data points. Equation (2.27) has a closed-from solution.

$$\hat{\boldsymbol{R}} = \frac{1}{n} \sum_{i=1}^{n} U_i U_i^T \tag{2.28}$$

The full correlation matrix has $O(n^2)$ parameters and not appropriate when n is large or for moderate-size data set. Liu et el. [62] address this problem by adding an L1 sparsity constraint to equation (2.27). Zezula [96] alsoproposes two special structures for correlation matrices to reduce the number of parameters. He uses uniform and serial correlation structures and estimates their parameters based on the Kendall rank correlation coefficient [49]. The uniform structure assumes that all entries in correlation matrix are equal ($R_{ij} = \rho$) while in serial correlation matrix, the entries are $R_{ij} = \rho^{|i-j|}$. Sample correlation estimator is the most common method to compute the correlation coefficient ρ but this estimator is not invariant through the transformation in Equation (2.26). The Kendall method is a rank-based method for estimating the correlation parameter and it is not sensitive to the strictly increasing transformation like in Equation (2.26). This property makes this method useful for the Copula model. Since these structures both have only one free parameter to estimate, they have poor representational power to model

real data. Toeplitz structure is a common way to increase the degree of freedom in a correlation matrix while keeping the number of free parameters limited. In this thesis, we use the Toeplitz structure as an extension to Zezula's work and show its combination with a mixture model can provide a richer Copula model.

Chapter 3

Grafted GMM copula and its applications in density estimation and classification

As mentioned in the previous chapter, Gaussian copula model provides a powerful model for estimating mutivariare densities. However, similar to the conventional Gaussian model, this model also has a limited ability to model multi-modal distributions due to the structure of the correlation matrix and the high number of free parameters. In this chapter, we present two modifications to address these problems. We use Toeplitz correlation structure to reduce the number of Gaussian Copula parameters. We also introduce a mixture of Gaussian Copula as a natural extension of the Gaussian Copula model to increase the flexibility of our model.

3.1 Gaussian Copula with Toeplitz Correlation Structure

The covariance estimation is a challenging task specially when the matrix has a special structure such as circular or Toeplitz. The conventional methods for estimating the covariance matrix such as maximum likelihood, don't have a good performance for high-dimensional data [21]. Cai et al. [22] have shown that Toeplitz covariance matrix can be approximated effectively for standard multivariate Gaussian distribution using tapering and banding approaches. They also have proven a minmax risk of convergence for their estimator.

In this section, we use tapering and banding approaches for estimating the correlation matrix in Gaussian copula model when the correlation matrix R in (4.1) is a Toeplitz matrix.The tapering method consists of three steps. First, the sample full correlation matrix is computed as in Equation (2.27). The second step is to average across each diagonal:

$$\tilde{\boldsymbol{R}}_m = \frac{1}{i-j} \sum_{i,j} \hat{\boldsymbol{R}}_{i,j} \qquad m = i - j \tag{3.1}$$

Finally, the entries that are far from the main diagonal are tapered with a function.

$$\boldsymbol{R}_{i,j}^{taper} = a_{|i-j|}\tilde{\boldsymbol{R}}_{|i-j|} \tag{3.2}$$

$$a_k = \begin{cases} 1 & \text{for } k \leq P/2 \\ 2 - \frac{2k}{n} & \text{for } n/2 < k \leq P \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

Where $P \leq \frac{n}{2}$. The banding version also can be computed from the tapered matrices as shown below where I is the indicator function and K is the bandwidth of band.

$$\boldsymbol{R}_{i,j}^{Band} = \boldsymbol{R}_{i,j}^{taper} \times I(|i-j| \leq K) \tag{3.4}$$

Through empirical evaluation, we found that the tapering method works better than the banding method, so we use the tapering method in the rest of this chapter.

3.2 Mixture of Copula Model

Drawing parallels to the use of mixtures to improve the flexibility and the capacity of the standard Gaussian models, we propose a similar extension, the Gaussian Mixture Copula.

Lemma 2 *Since the Copula function is a valid density function, the convex mixture of Copula functions is still a valid Copula density:*

$$c(U) = \sum_{i=1}^{M} w_i c_{gauss}(U; R_i), \quad \sum_{i=1}^{M} w_i = 1 \tag{3.5}$$

The parameters of the mixture model can be computed using EM algorithm. To train a mixture model with M components, we use a heuristic strategy in which we first randomly initialize $3M$ components. And after a few iterations of EM, we discard the M components with the smallest weights. For each of the remaining $2M$ components, we compute the average distance between this component and the other components. We use Frobenius distance between correlation matrices to measure the pairwise distance between two components. Then, we choose the component with highest average distance. We repeat this process to select M components. Finally, we use these M component as initial points and run a few iterations of EM to compute the ultimate mixture model.

3.3 GMM with marginal modification

In practical applications, it is common to encounter datasets where some features are bounded to a specific range, or have a heavy-tailed distributions such as log-normal and Cauchy distributions. The conventional density estimation models, such as GMM, typically do not have any mechanism to adapt their marginal distributions to the data, so there is a mismatch between the marginal distributions of data and the model for these applications. In Figure 3.1, we illustrate this mismatch between marginal distributions data and GMM for a typical bivariate dataset.

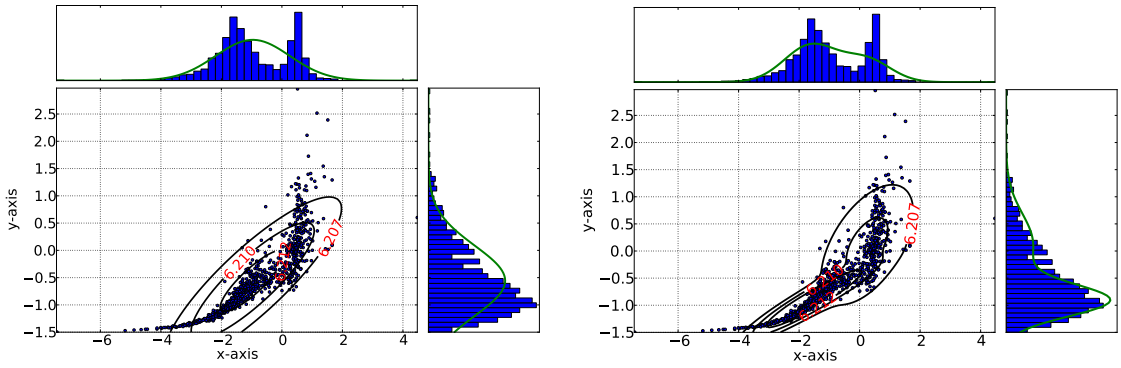


Figure 3.1: Illustration of the difference between the marginal distributions of data (green curves) and the marginal distributions of estimated GMMs (blue histograms) with (a) one and (b) two component mixtures.

In this section, we propose a simple method based on copula model to address the mismatch problem for GMMs. We decompose the estimated GMM into its marginals and copula function. Assuming that the copula function fully captures the multivariate interaction, we replace the marginals with that from the target domain. The intuition is that the marginals are univariate and can be estimated reliably with very little data from the new domain.

According to the copula model in Equation (2.3), any joint distribution, including GMM, can be factored into a copula function $c(\cdot)$ and a set of marginals $\{f_j(x_j)\}$ as follows:

$$\begin{aligned}\log \sum_{i=1}^{M} w^i N(\mathbf{x}; \mu^i, \Sigma^i) &= \log c_{gm}(u_1, u_2, ..., u_n) \\ &+ \sum_{j=1}^{n} \log f_j(x_j) \end{aligned} \tag{3.6}$$

where u_j is j-th cumulative cumulative function $F_j(x_j)$. In GMM, each univariate marginal density function $f_j(x_j)$ can be computed by integrating out $\mathbf{x} \setminus x_j$ as: Σ^i_{jj}, respectively:

$$f_j(x_j) = \sum_{i=1}^{M} w^i N(x_j; \mu^i_j, \Sigma^i_{jj})$$

where μ^i_j is the j-th component of the i-th mean vector μ^i, and Σ^i_{jj} is the j-th diagonal entry of the i-th covariance matrix Σ^i. Since marginal density functions are closed-form, the copula function also has an analytic form as follows:

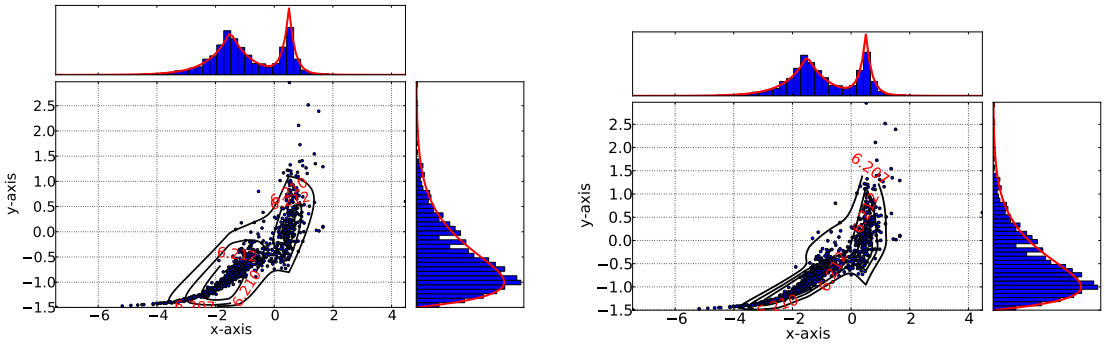
$$\begin{aligned}\log c_{gm}(u_1, \ldots, u_n) &= \log \sum_{i=1}^{M} w^i N(\mathbf{x}; \mu^i, \Sigma^i) \\ &- \sum_{j=1}^{n} \log \sum_{i=1}^{M} w^i N(x_j; \mu^i_j, \Sigma^i_{jj}) \end{aligned} \tag{3.7}$$

Thus, the copula density can be computed easily from the estimated joint GMM by the associated univariate marginal distributions. Now, we can easily construct a new joint distribution with new univariate marginals $u_j = \hat{F}_j(x_j)$ as follows:

$$\log f_{new}(\mathbf{x}) = \log c_{gm}(u_1, u_2, \ldots, u_n) + \sum_{j=1}^{n} \log \hat{f}_j(x_j) \tag{3.8}$$

Note that we estimate new marginal distributions separately using non-parametric density estimation but they can also be obtained based on the prior knowledge. Figure 3.2 shows the result of modifying the marginal distributions of GMM for the previous bivariate dataset. We use the true marginal synthetic distributions for this example since we know them a priori. Clearly, the resultant distributions are significantly better than GMMs.

The implementation of this method is simple but tricky. To evaluate the log likelihood function at a given point, we first map each feature x to a value of u in the unit interval $[0, 1]$ using new cumulative distribution $\hat{F}(x)$. Next, we transform u using the inverse of marginal cumulative function of GMM $x^{test} = F^{-1}(u)$. In theory, the value of x^{test} can



(a) (b)

Figure 3.2: Bivariate joint densities for GMMs with marginal modifications. The GMMs have (a) one and (b) two component.

be any value in $(-\infty, \infty)$. To avoid numerical instability in practice, we truncate the value of u at 0.05 and 0.95. We then plug $\mathbf{x}^{test} = [x_1^{test}, \ldots, x_n^{test}]$ in to the Equation 3.7 to compute the copula function. Finally, we add the log likelihood of the copula function and marginal distributions, according to Equation 3.8, to compute the log likelihood function. This computationally simple approach provides a framework to improve the performance of already trained GMM by combining the dependencies modeled in the copula function of the GMM with the more accurate estimation of marginal distributions.

3.3.1 Experimental results on UCI databases

In this section, we compare the performance of the proposed methods with two other models – Naive non-parametric estimator and Gaussian mixture models. We evaluate the models in terms of average log likelihood over many held-out sets, which is a standard practice for comparing models for density estimation [82]. The naive models assumes the variables are independent, and hence the joint probability is the product of the marginal density functions :

$$\hat{f}(X) = \prod_{j=1}^{M} \hat{f}_j(x_j) \tag{3.9}$$

We use Gaussian KDE to estimate univariate marginal densities as follows :

$$\hat{f}(x) = \frac{1}{Nh} \sum_{i=1}^{N} k(\frac{x - x_i}{h}) \tag{3.10}$$

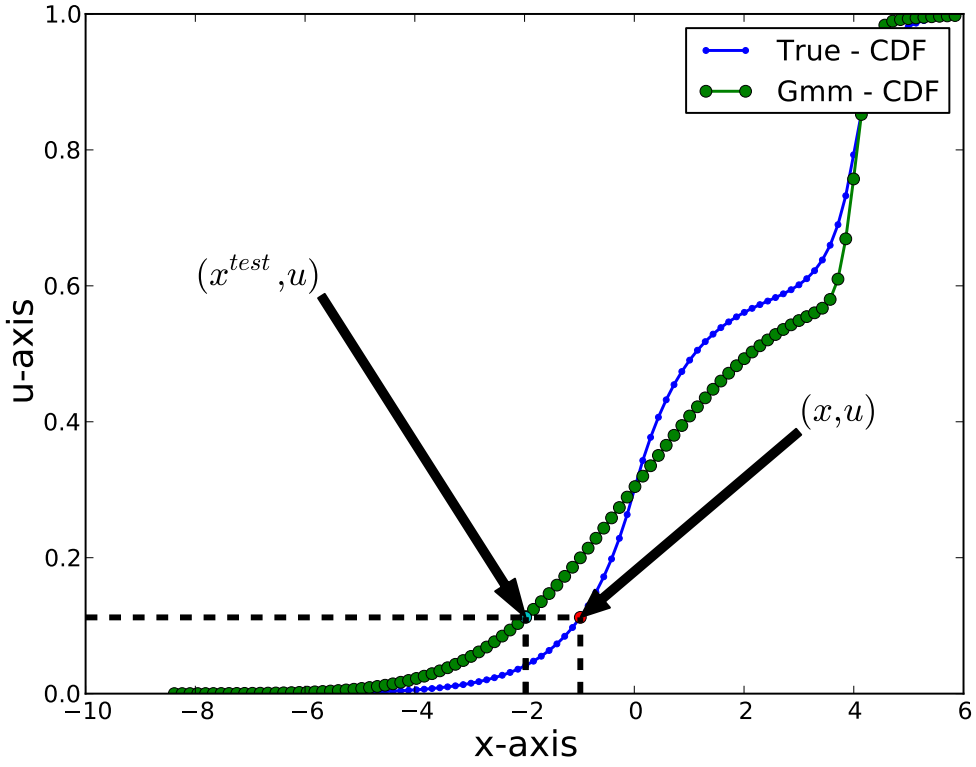


Figure 3.3: Schematic for modifying marginal distributions based on the copula model. The true CDFs are computed using the non-parametric method and utilized to map a x^{test} to the copula domain u. Then, the inverse CDFs of GMM are applied to u in order to obtain the x that is used for the evaluation of copula density function.

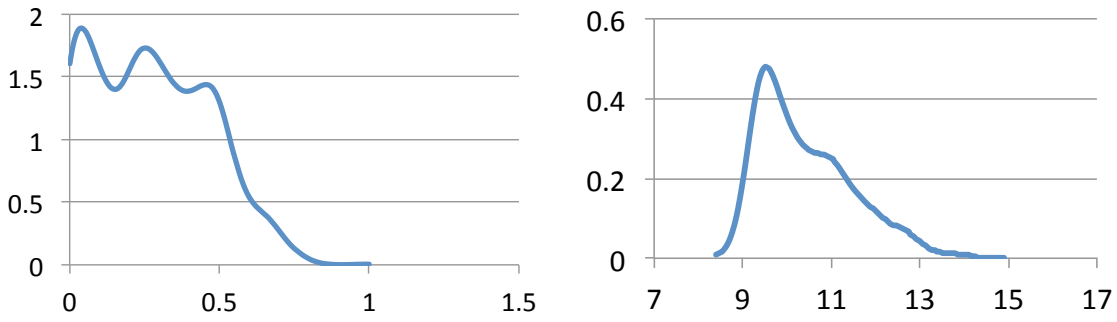
where k is the Gaussian kernel and h is its bandwidth. The bandwidth can be computed based on the empirical standard deviation $\hat{\sigma}$ [84]:

$$h = (\frac{4\hat{\sigma}^5}{3N})^{0.2} \tag{3.11}$$

3.3.1.1 Density estimation on Wine Quality Data Set

In this section, we evaluate the performance of the proposed methods on mutivariare density estimation task. We use the red wine data set [28], which is comprised of 1599 samples. Each sample has 11 attributes relevant for predicting the quality of wine. This data is a good representative of many practical applications where marginal distributions differ considerably across feature components. Figure 3.4 shows two marginal densities

for the wine data set. As shown in the figure, the marginal distributions are multimodal and bounded. Clearly, models, such as GMMs with a finite number of components, are not well-suited for this data.



(a) Feat 3 (b) Feat 11

Figure 3.4: Two marginal distributions of Wine data set

Now, we randomly split the data set into two equal sets and use one as a training set and the other one as a test set. We repeat this experiment 100 times. This approach has been used previously in the literature to evaluate the performance of density estimation methods [31]. In Figure 3.5, we compare the performance of mixture of Gaussian copula, GMM with modified marginals and standard GMM with different configurations. We use the averaged log likelihood over test set as the performance metric.

The results show that the marginal modification noticeably improves the performance of the GMM, which also justifies that the marginal distributions of data are far from the Gaussian assumption. However, the standard deviation of GMMs with modified marginal distributions are slightly higher than the standard GMMs. Mixture of copula with Toeplitz correlation matrix and diagonal GMM with marginal modification have $O(n)$ parameters. Note, both of them fit the data significantly better than the GMM with diagonal covariance matrix, which has same number of parameters.

3.3.1.2 Classification task

As mentioned earlier, one common application of density estimation methods is to build generative classifiers. To construct a generative classifier, we estimate the class-conditional distribution of each class $p(x|c)$ independently. Then, we compute the posterior probability

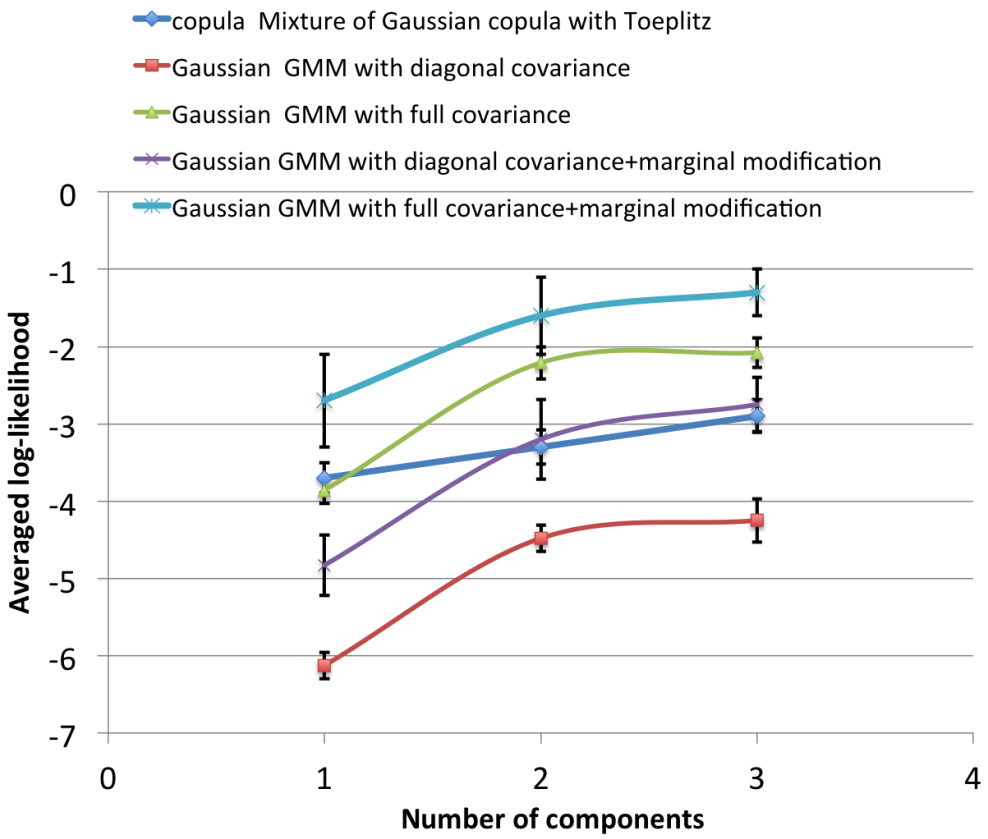


Figure 3.5: Averaged log-likelihood on the Wine data set.

of each class using Bayes rule as follows :

$$p(c|x) \propto p(x|c) \times p(c) \tag{3.12}$$

where the prior $p(c)$ is the proportion of class c in the training data. Finally, we use the maximum of the posterior probabilities to assign a class label to an input.

In this section, we utilize our proposed density estimators, including mixture of copula and GMM with modified marginal distributions, to approximate class-conditional distributions for the classification task. We compare the resultant classifiers with naive non-parametric classifier, Gaussian mixture classifier, support vector machine and Copula Network Classifiers [32].

The **naive classifier** assumes that the variables for each class-conditional density are

independent and hence the joint probability is simply the product of the marginals. In the case of naive non-parametric model, the univariate marginal densities are modeled by Gaussian kernel density estimation. The bandwidth of the Gaussian kernel h was set using the empirical standard deviation $\hat{\sigma}$.

In **Gaussian mixture classifier**, the class-conditional densities were modeled using GMMs with full or diagonal covariances. The parameters of the GMMs were estimated by EM algorithm. The number of components were set using Akaike information criterion (AIC), measured on the training set [2].

For the **support vector machine**, we used scikit-learn toolkit to obtain SVM classifers [76] with radial and polynomial kernel functions. The optimal parameters of the SVMs were set using a grid search on a 5-Fold cross validation over training set.

The performance of the classifiers were evaluated using classification accuracy on 5-fold cross validation. In addition, we use one-against-one strategy to construct classifiers with more than two classes. Table 3.1 shows the results of different methods on 4 data sets from the UCI repository [11]– Red Wine, Pima, Magic and Glass. For the Wine

Table 3.1: Average classification accuracy on 4 UCI classification tasks with their standard deviations, where † denotes use of Ledoit-Wolf method for estimating covariances and * denotes [max:min]. Note, one class of Glass data does not have enough samples for fitting GMM.

Method	Wine	Glass	Pima	Magic
non-Param	57.0(3.3)	92.2(4.4)	76.0(4.0)	75.7(0.3)
GMM+Diag	53.3(4.3)	91.2 (5.9)†	74.9(3.0)	79.3(0.8)
GMM+Full	53.3(4.7)	NA	74.5(2.4)	84.5(0.6)
SVM+Poly	57.1(3.1)	78.7(6.5)	75.3(1.8)	82.3(0.6)
SVM+RBF	**62.0(2.0)**	91.7(3.9)	77.1(2.8)	85.0(0.5)
CNC [32]	59[56:61]*	70[52:86]*	76[73:79]*	81[80:82]*
mixture of copula	61.3(1.9)	90.1(3.1)	76.9(3.2)	**87.0(0.8)**
best GMM+marginal modification	58.7(1.4)	**94.4(3.6)**	**77.3(3.7)**	85.8(0.6)

data set, as reported in Table 3.1, the non-parametric naive classifier performs better than GMMs with diagonal or full covariance matrices. The proposed mixture of Gaussian model significantly outperforms all the classifiers except for SVMs with radial basis function. But, the performance gain of SVM is not statistically significant. For the Glass data set,

since the number of samples for some classes is insufficient to estimate the covariance robustly, we use GMM with one component and use Ledoit-Wolf method [56] to estimate the GMM with modified marginal distributions. This outperforms all the other classifiers significantly. In the case of the Pima task, as reported in forth column of Table 3.1, the performance of GMM with modified marginal distributions method is comparable to the performance of the SVMs, and significantly better than others. For the Magic task, the results show that the GMMs outperform non-parametric naive classifier, implying that the effect of dependencies is more important than the marginals. The interesting point is that the combination of non-parametric marginal (naive) and the GMM, through the GMM with marginal modification, performs better than each one by itself. For this task, the mixture of copula is siginificaly better than others.

In the next experiment, we evaluate the performance of the proposed methods on diagnosing Parkinson's disease using the Parkinson Speech Dataset [79]. The goal is to classify healthy individuals from PD patients using speech related features extracted from their voice samples. This dataset consists of multiple samples for 40 subjects where 20 are healthy and 20 are PD patients. The results in Table 3.2 are average classification accuracies for the leave-one-subject-out cross-validation and show that our GMM with marginal modification outperforms significantly the K-Nearest Neighbor (KNN) and SVM with linear and radial basis functions.

Table 3.2: Average classification accuracy on Parkinson Speech Dataset

KNN-7	SVM-lin	SVM-rbf	GMM	GMM with marginal modification	mixture of copula
57.5	52.5	55	57.5	**67.5**	62.5

3.4 Experiments on OHSU Monkeys' vocalization corpus

This section describes the application of the copula models for classifying vocalizations from rhesus macaques that was collected at OHSU's Oregon Primate Research Center. This task has large amounts of background noise and we wish to study and evalute the robustness of copula model.

3.4.1 Motivation for the Task

Current approaches for observing the animal behaviors completely depend on human observation. A highly trained observer watches the animals in the group and records the occurrence or duration of the behaviors listed on an ethogram (a set of behaviors with their quantitative descriptions) [3]. There is a wide range of behaviors such as aggression, displacement, fear grimace, lipsmack, scream, grunting etc. that can be used in studies of social behaviors [65]. Human observation has two major limitations: First, feasible ethograms are limited to a small subset of behaviors since the rate of analyzing the data and its accuracies drop when an observer annotates more behaviors. Second, it is impossible to annotate all behaviors of every animal in a group in a single pass. In practice, the observer is forced to go through the data multiple times and in each pass, annotate a specific behavior of all animals or a particular individuals' activities. In addition, the behaviors with auditory modality, such as barking, cooing and grunting, are difficult and time consuming for human observers to annotate.

Having an automated method for observing and modeling the social activities could lead to a better understanding of behaviors of social animals and open up new directions for researchers in behavioral ecology, anthropology, evolutionary psychology, conservation biology, and neuroscience.

3.5 The Corpus of Rhesus Macaque Vocalizations

Our corpus consists of audio and video recordings of social behaviors of groups of rhesus macaques. The study and the data collection was approved by OHSU's Institutional Animal Care and Use Committee. Groups of 4-6 animals were formed, introduced into the pen, which is about 12 ft long, 7 ft deep and 7 ft tall as shown in Figure 3.6, and observed over a period of about 2 months. We recorded behavior as the group settled into their stable social hierarchy. After approximately two weeks, we perturbed the social hierarchy of the groups using standard procedures such as presence of an unfamiliar human (outside the cage), and introduction of toys and desirable food. The observations were performed to minimize the disruption of animal care and husbandry. This meant swapping

Figure 3.6: A group of monkeys in a Pen.

the spent audio recorder, housed in their collars, with a fully charged one on a specified day of the week. The recordings were performed till about 7pm on the same day and between about 7am and about 7pm the subsequent day, corresponding to the hours when the lights remained on. In all, 80 such sessions were recorded from 5 different groups.

Video recordings were captured by three cameras mounted on three different corners of the pen and one fisheye-lens camera mounted on the ceiling. All four cameras were fully synchronized in the frame level and their frame rate was controlled by an external trigger to be exactly 12 fps. The mounting locations of cameras were carefully chosen to support 3D reconstruction of the observation sessions and maximize the coverage of the visible space in the cage.

Audio was recorded using tiny recorders, EDIC B21, which is about 40 x 15 x 10 mm in dimension, 8g in weight, and has a battery life of 2-3 days. These recorders were placed in a custom housing that was attached to a standard collar, as shown in Figure 3.7. Each

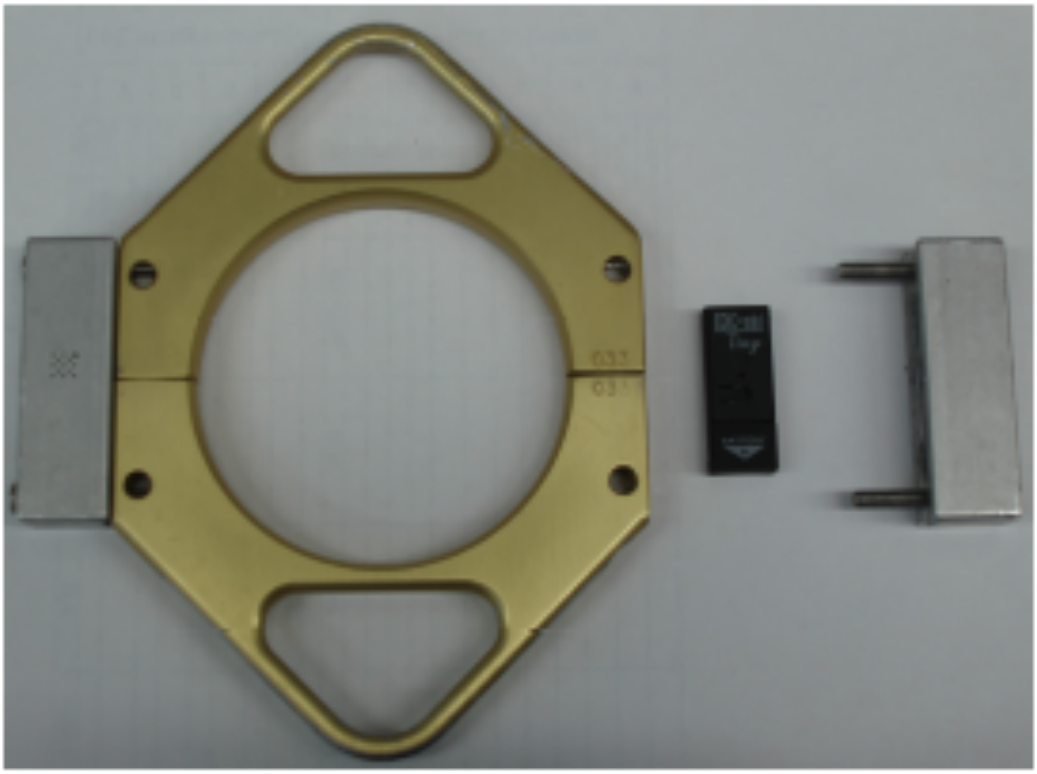


Figure 3.7: Tiny low-power audio recorder along with its housing that attaches to the monkey's collar

recorder was programed to record 12 hours at 8 kHz sampling rate for each session. Unlike the video recordings, the audio recordings could not be synchronized via hardware or other means. Our calibration attempts using chirp signals show that the asynchrony is erratic and not easily predictable such as a constant offset or a linear drift. In all, we have about 3800 hours of audio recordings.

3.5.1 Challenges

There are several challenges in processing the above mentioned audio recordings and this section addresses them.

1. High background noise: Monkeys move about such that their collars hit the walls and metal mesh of the pen. In addition, the recordings also contain the conversations of human caretakers.

2. Multiple speakers: Even though each monkey has a separate collar-mounted recorder, the recordings contain vocalizations from its neighbors. So, attributing which monkey spoke when is non-trivial.

3. Sample dropout: The recorders appear to lose sample randomly over the course of the long 12 hour recording sessions. This is similar to the problem that occurs in unreliable low-power sensor networks and complicates the problem of aligning the recordings which is necessary for identifying which monkey vocalized when.

4. Length of recordings: The sessions are about 12 hours long, which makes it infeasible to apply conventional solutions such as dynamic programming to align waveforms.

Given the amounts of data, the first step was clearly eliminating the segments with very low probability of vocalizations. From listening to several random examples and from preliminary experiments, simple methods based on energy or spectral entropy were confounded by large amounts of background noise. So, before we could eliminate segments without vocalization, we had to improve the signal to noise ratio. After enhancing the signal and removing unvocalized segments, we were able to achieve high accuracies in detecting vocalizations using a supervised classifier fairly easily. In contrast, without the signal enhancement, the supervised classifier was unusable. Having identified the vocalized segments, we aligned the recordings by just focusing on these segments containing high signal-to-noise ratio. This improved the quality of alignment compared to aligning with portions that included background noise without vocalizations. Below, we describe each of the steps in more detail.

3.5.2 Filtering Stationary Background Noise

The pen housing used for collecting the corpus is part of a bigger laboratory, which was not designed for high quality audio recordings. The infrastructure including ventilation and lighting introduced a significant amount of background noise. The walls are acoustically reflective and not dampened in any way, causing significant reverberations.

The recordings contain two sources of additive noise – a significant amount of background noise that was largely constant in nature, on top of which there were bursts of

metallic clangs from different distances. Knowing that the first component is a good candidate for signal enhancement techniques, we applied noise spectral subtraction.

Noise spectral subtraction is a simple and computationally efficient method for reducing the background noise and enhancing the audio. It is a nonparametric method and has two major steps. The first and the more important step is to estimate the background noise. The more sophisticated techniques locate a segment in the recording which contains only noise. Simpler approaches typically assume the initial few milliseconds are noise and estimate the background from it. We were interested in quickly characterizing the potential benefit of this simple technique, so we resorted to the implementation in Audacity [67], where the user needs to manually choose an appropriate segment containing noise, from which a noise profile is created. The noise profile simply consists of a set of statistics like maximum for each frequency bin in Discrete Fourier transform (DFT) computed across all the frame of noise segment. The second step uses the noise profile to attenuate the power spectrum of the parts of signal that are similar to the noise and leave the rest unchanged. Finally frequency-smoothing and time-smoothing are applied to produce a natural sound and prevent rapid changes in the gain of the output signal.

This simple approach was remarkably effective. After signal enhancement with spectral subtraction, it was relatively easy to filter out unwanted segments which contained only silence or background noise and no vocalizations. This was useful in reducing the size of the data significantly. Energy-based segmentation is simple and computationally efficient method for removing such segments. All segments below -35 db were removed. This reduced the corpus by a factor of 10 and made it feasible to process the data using the next few steps.

3.5.3 The Task of Detecting Vocalizations

The candidate segments extracted from the previous step contains three types of audio – vocalizations from monkeys, bursty noises such as metal clangs, and human conversations. Human interference is unavoidable in the standard animal laboratory setting since animal husbandry requires mandatory routine checks, multiple times a day, by the staff to feed and monitor them.

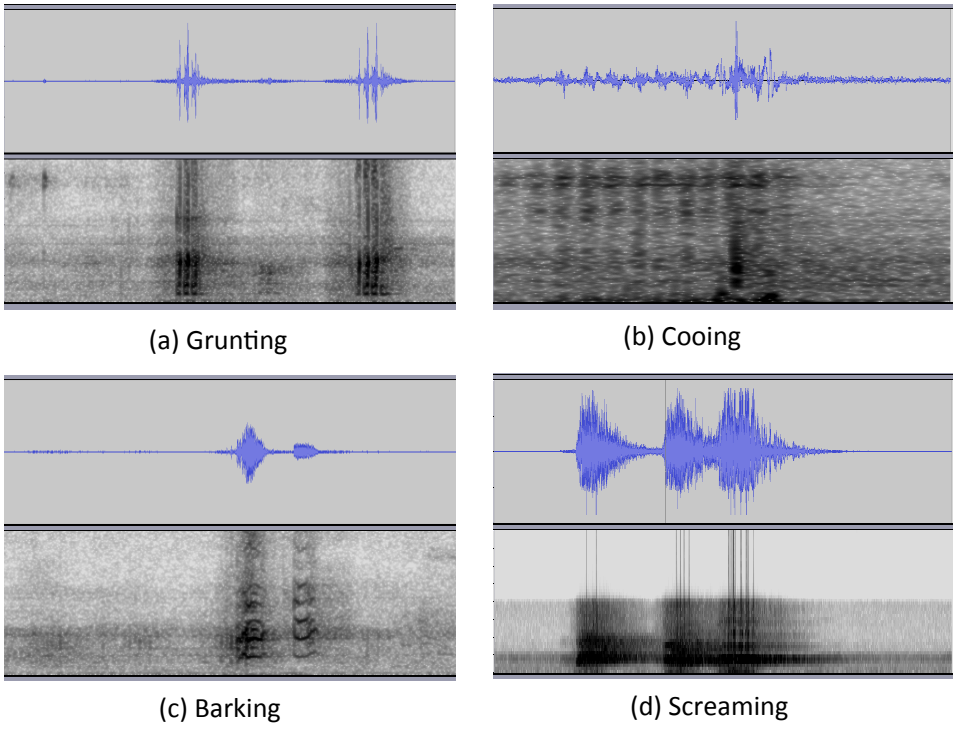


Figure 3.8: Different types of monkey vocalizations

The difficulty in isolating the human conversations is that monkey vocalizations vary largely depending on the type (e.g., grunting, cooing, barking and screaming), as illustrated in Figure 3.8. We manually checked a 12-hour recording from one monkey and carefully annotated all segments as belonging to the monkey and non-monkey. The data set consists of 1147 segments where 625 of them are monkey vocalization segments.

For each segment, we extracted a fixed dimension feature vector using OpenSmile [34], a standard feature extraction tool that extracts a rich set of features for each segment. Briefly, the toolkit extracts features in two steps. First step is extraction of 25 msec long frames using a Hanning window at a rate of 100 frames/sec and computation of frame-level features such as root mean square, MFCCs, Zero-crossing rate, voicing probability, F0 and their deltas. The second step is aggregating frame-level features into the segmental feature vector by applying statistical functions such as mean, median, variance, minimum

and maximum across all frame-level features of a segment. We extract about 400 features for each segment. Then, we utilize principle component analysis to reduce the number of dimensions to 17, which captures 90% of the variance.

In this section, we use our proposed methods to estimate the distribution of monkey vocalization segments. Due to the insufficiency of data for GMM with full covariance, we limit our experiments to the diagonal covariance. Similar to the previous experiments, we randomly divide the data into training and test sets and use them for training and evaluation. We repeat this experiment 100 times. Figure 3.9 shows the performance of several methods in terms of log likelihood for the monkey vocalization data set. The results demonstrate that the Toeplitz assumption for speech-like data is more natural where the correlation between feature components tapers off naturally when the components are further apart from each other. Thus, they provide nearly the same benefits as a full correlation Copula model but with fewer parameters.

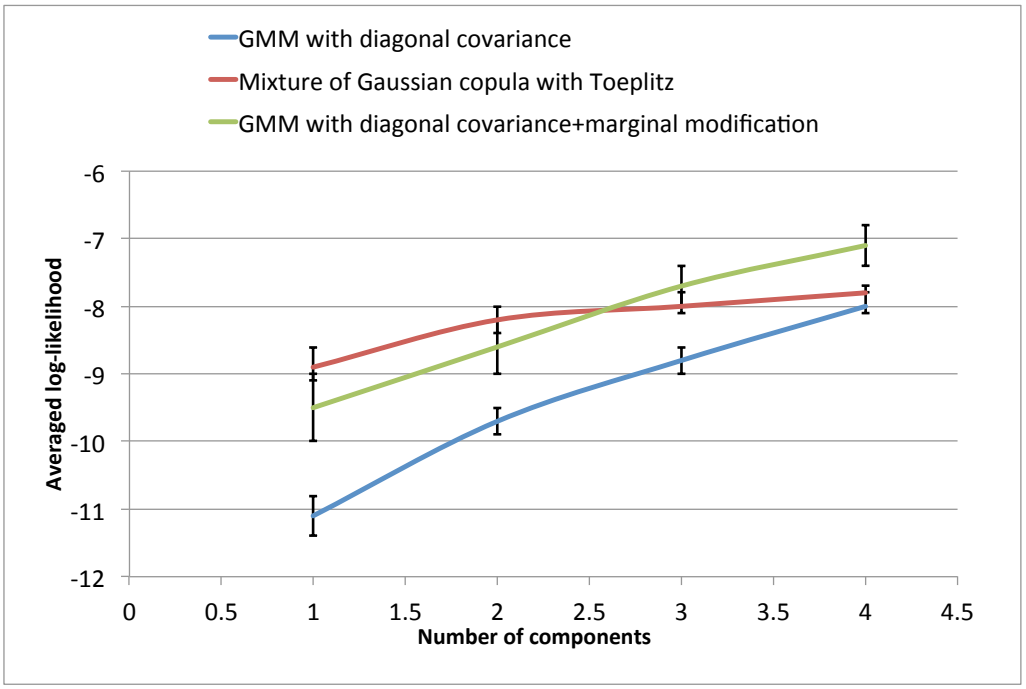


Figure 3.9: Averaged log-likelihood on Monkeys' vocalization data set.

For the classification task, we use our proposed methods to build generative classifiers

for detecting monkey vocalizations. In this experiment, we compare our proposed classifiers with K-nearest neighbor classifier, GMM-based classifier and SVMs. For the SVM, we investigated two different types of kernel, namely, the radial and the polynomial basis functions. The parameters of the classifiers were tuned using a grid search on a 5-Fold cross validation over training set and evaluated over the held-out set. Results, reported in Table 3.3, show that GMM with marginal modification works best for this task.

Table 3.3: The performance (accuracy) of different classifiers in detecting segments with vocalization from the monkeys.

Method	Ave. Accuracy	Std. Accuracy
K-NN	80.3	3.6
GMM	86.5	2.1
SVM-poly	87.9	1.8
SVM-rbf	91.3	1.5
Mixture of copula	90.1	2.5
GMM with marginal modification	92.9	1.4

Table 3.4 shows the results of 5-fold cross-validated paired t-test between GMM with modified marginal distributions and two other best classifiers, mixture of copula and SVM. According to these results, our proposed method almost became significant.

3.6 Summary

In this chapter, we proposed two computationally simple methods to construct multivariate copula functions: Mixture of Gaussian copula model with Toeplitz correlation structure and GMM with modified marginal distributions. Both of these grafted GMM copulas involves the estimation of a set of marginal distributions and a mixture model, which makes them a more powerful alternative for GMMs. Note that the marginal modification method

Table 3.4: 5-fold cross-validated paired t-test between GMM with modified marginal distribution and two other best classifiers.

	t-value	p-value
SVM-rbf	1.82	0.08
Mixture of copula	2.48	0.04

provides a simple way to adjust the marginal distributions of already trained GMMs without retraining. These method can also be used to estimate the class-conditional densities in generative classifications. The resulting class-conditional multivariate distributions form better classifiers than their corresponding conditional GMM counterparts with the same number of parameters. Our proposed models perform consistently better than GMMs with different settings on different classification tasks. The performance of both models are comparable to SVM in many cases, even though it is a generative model.

Chapter 4

Application of Copula Models in Automatic Speech Recognition

4.1 Introduction

Generally, the mismatch between the training and testing conditions degrades the performance of machine learning tasks, including ASR. The mismatch in ASR systems can be at corpus, speaker or utterance level. The corpus-specific mismatch stems from factors that differ from one dataset to another. For example, if the training and testing datasets have been recorded by two microphones with different characteristics, the resulting mismatch is a coarse-grained corpus level. The speaker-specific mismatch mainly originates from acoustical differences that exist among different speakers. Factors such as speaking style, vocal characteristics and accent are responsible for variations in speech that are unique for each speaker. As an example, the conventional ASR systems trained by adult speakers have a poor performance on kids' speech mainly due to the mismatch between the acoustical features of adults and kids. Mismatch can even occur at an utterance-level. This type of mismatch is related to factors like background noise or reverberation distortion that can vary from one utterance to another. In Figure 4.1, we plotted the distributions of the first two MFCC features for an utterance from a speaker under different noise situations. We also use convex hulls to represent feature space boundaries. The convex hull is the smallest convex area that contains the data points and it provides a simple and effective way to visualize the boundary of feature space variations. As Figure 4.1 shows, the shape and location of the scatter plots of MFCC features change substantially under different noise conditions, including in street, cafe, pedestrian and bus conditions.

Real-world applications require ASR systems to properly respond under diverse noisy environments. In addition, the wide range of audio capture devices (e.g., smartphones and tablets) with different channel characteristics highlight the challenges that ASR systems might face in real applications. The aforementioned challenges in real-world applications are reflected in input speech in forms of additive and convolutional noises. These variations, to some extent, can be modeled by ASR acoustic models. However, it is impractical to collect training data that represents a wide range of background noises and reverberations.

DNNs are currently the most popular and effective models for acoustic modeling in

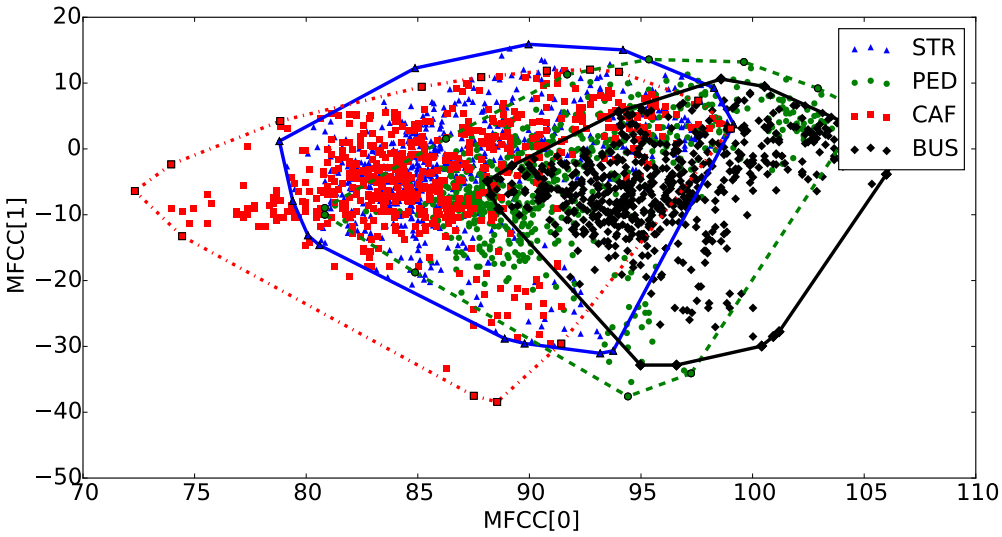


Figure 4.1: Scatter plots and their corresponding convex hulls of the first two MFCC features for a phrase uttered by a female speaker under four different noisy conditions: street junction (STR), pedestrian area (PED), cafe (CAF) and bus (BUS).

ASR systems, after researchers displaced GMMs, which were popular for several decades before then [45]. DNNs are particularly effective in large vocabulary tasks with large amounts of training data. GMMs, on the other hand, are simpler and faster to train. As such, they are still employed in small tasks with limited training data. Both these models are capable of representing real-valued multivariate stochastic processes and have relatively simple estimation algorithms for learning the optimal parameters for a recognition task from labeled training data. With sufficient parameters, both GMMs and DNNs have enough capacity to easily overfit the training data. Therefore, for good generalization, one has to cautiously choose the optimal model size by empirically evaluating the performance on a held-out data set. In practice, the learning process is effective if the model parameters are learned on a training data that has a minimal mismatch with the testing data.

For GMMs and DNNs, features are computed from raw waveforms in terms of the logarithm of the mel-warped frequencies, and mel-warped cepstral coefficients. These features do not explicitly factor the observed signal into the additive and convolutional components present in the input. Both these features have homomorphic properties where

convolutional noise becomes additive but the additive noise interacts with the speech signal in non-linear manner.

The strategies adopted to disentangle the additive and convolutional noises can be broadly categorized into model-based and feature-based methods. **Feature-based methods** transform features into a new feature space representation such that the effect of additive and convolutional noises are minimized while speech-related variations remain unchanged [27] [33]. Figure 4.2 illustrates the main concept behind feature-based methods through a toy example.

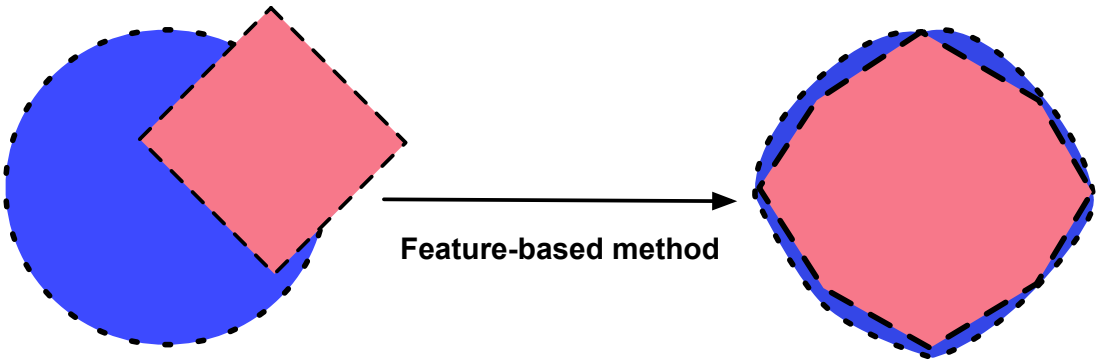


Figure 4.2: On the left,the circle and diamond are two noisy versions of a single hypothetical distribution in the original feature space. The right part shows these distributions after feature-based transformation, where they are now more similar in the new feature space representation.

The simplest version of such a transformation is the well-known Cepstral Mean-Variance Normalization (CMVN), which removes the convolutional channel noise in the homomorphic cepstral domain [69]. This method assumes that the channel noise varies slowly—a mild assumption that is often true. The key advantage of this feature-based method is its ability to generalize to noises that have never been seen in the training data. There are several feature-based transformation methods in the literature [42, 43, 30, 29, 37]. Here we limit our review to the most relevant methods. One of the earliest proposed methods is Histogram Equalization (HE), which share similar motivation with our

proposed method given in Section 4.2. The core of this method is learning a coarse transformation such that the histogram of test features are the same with those in the training set. In the same vein, Gaussianization method [25] learns a transformation with respect to a constrain such that the distribution of transformed features are Gaussian. Both these approaches are *ad hoc* as they do not consider an accurate estimation of training and testing distributions in the computation of a feature-based transformation. In contrast, our method, as we describe in Section 4.2, provides a principled mechanism based on the Copula model to take training and test distributions into account.

Model-based methods, on the other hand, transfer the acoustic model such that the transformed model is well-fitted for noisy data. In general, it is hard to modify the acoustic model to accurately model the noise characteristics, since the interaction between the speech and noise is highly nonlinear in feature space. For example, Parallel Model Combination (PMC) estimates two acoustic models for clean speech and noise. Then, it uses Taylor series approximation to combine two models in log-spectral domain to model noisy speech data [70]. PMC has shown performance gains in certain tasks for GMM-HMM based acoustic models. However, this method is not effective in large vocabulary speech recognition. For a more comprehensive review of Model-based methods explored in the literature, see [36].

One brute force approach that has been shown to be remarkably effective is increasing the diversity of the training data by artificially distorting the input signal with different noise types. This technique, often referred to as multi-style training (MTR) in the literature [60], has been shown to be particularly effective in deep neural networks where the network has sufficient representational power to model more diverse datasets implicitly. The effectiveness of MTR depends entirely on the diversity of the simulated distortions of the input signal and it is a non-trivial task to generate all combinations of potential sources of input distortions. The two common distortions employed for this purpose are reverberation and additive background noise. In the case of reverberation, the distortion is generated by convolving the input signal with the impulse response of a room whose dimensions, and the location of the source and the microphone, have been specified. The resultant signal is further distorted with appropriately amplified or diminished background

noise of a specified type. Since the distortion has a number of free parameters, each of which belongs to an open set, it is impossible to represent all potential distortions that may exist in a real-world utterance.

To summarize, the DNN and GMM models do not have an inherent mechanism to factor out the additive and convolutional noise components from input features. This severely limits the ability of current ASR systems to explicitly represent the model and noise components in real-world conditions.

In this chapter, we focus on addressing the distributional mismatch that appears between training and testing conditions similar to what we previously discussed in feature-based methods. In general, this mismatch is hard to model due to the nonlinear interaction between the speech and noise in feature space. We investigate the use of the copula model to reduce the effect of the mismatch between the training and testing set.

The Copula model is an effective method that allows decoupling of marginal distributions from the dependency model in distribution estimation. In a nutshell, we first estimate the distribution of the training and testing sets using the Copula model. Then, we find a nonlinear transformation that minimizes the Kullback-Leibler (KL) divergence between the training and testing distributions. Finally we apply the transformation to mitigate the mismatch between the testing and training set. It can be shown that the Mean Variance Normalization (MVN) and Histogram Equalization (HE) are two special cases of our method.

The rest of the chapter is organized as follows. In Section 4.2, we discuss the Copula model and its Gaussian variants, and then describe the optimal transformation to convert one Gaussian copula model to another one. In Section 4.3, we propose a new normalization method based on the couple-based transformation for ASR. Then, we employ the above normalization in different acoustic models and evaluate their recognition results on Aurora 4 and CHIME 4 tasks. Next, we formulate a new approach to embed the copula-based transformation in the acoustic model, and modify the learning method for the GMM-HMM acoustic model to jointly learn the transformation as well as the acoustic model. Finally, we report the recognition result of the joint learning and compare it with our first proposed method, and conclude with a summary of our work in this chapter.

4.2 Optimal Transformation for Matching Two Gaussian Copula Models

A common approach to reduce the mismatch effect is transforming the features to a new space such that the PDF of the testing set is closer to the PDF of the training set. In order to measure how similar two PDFs are, KL divergence is often used and minimizing KL divergence results in the reduction of mismatch between two PDFs. The main drawback is finding distributions that properly represent training and testing sets. The empirical distribution of training and testing sets, particularly for speech data, are complex and multimodal which makes mixture models a good candidate to model these distributions [78] However, the computation of transformation for mixture models with too many parameters becomes almost analytically and computationally intractable.

In this section, we propose a method to estimate the transformation based on the Copula model. We show that there exists an optimal nonlinear transformation that minimizes the KL divergence between the training and transformed testing distributions if these distributions are modeled by Gaussian Copula model (GCM). GCM is a simple and powerful approach for estimating a multivariate distribution [13, 86, 15], derived from the standard Gaussian model by relaxing Gaussianity constraints on the marginal density functions. According to the GCM formulation, as discussed in Chapter 3, any multivariate distribution can be decomposed as follows:

$$f(\mathbf{x}; R, \Lambda) = c(\mathbf{u}; R) \prod_{i=1}^{n} f_i(x_i; \lambda_i) \tag{4.1}$$

where $f_i(x_i; \lambda_i)$ is i-th marginal density function and λ_i is the parameter of i-th marginal distribution. $c(\mathbf{u}; R)$ is Gaussian copula density function as follows:

$$c(\mathbf{u}; R) = \frac{1}{|R|^{\frac{1}{2}}} \exp\{-\frac{1}{2}\mathbf{u}^T(R^{-1} - I)\mathbf{u}\} \tag{4.2}$$

where R is a correlation matrix[1]. The i-th component of vector $\mathbf{u}$ is as follows:

[1]Correlation matrix is a special covariance matrix where the diagonal elements are equal to one.

$$u_i = \Phi^{-1}(F_i(x_i))$$

where Φ^{-1} is the quantile function of standard normal distribution and F_i is i-th marginal cumulative function.

Now let $\mathbf{x} \sim f(\mathbf{x})$ and $\mathbf{y} \sim g(\mathbf{y})$ be the distribution of test and training sets where both $\mathbf{x}$ and $\mathbf{y}$ are random vectors of size n. The following proposition describes the optimal transformation when KL divergence is used to measure the mismatch.

Proposition 1 *The optimal transformation converts the test distribution to the train distribution when the difference is by KL divergence.*

Proof 1 *The KL divergence is always non-negative, $D_{KL}(f||g) \geq 0$, which is also known as Gibbs' inequality, with $D_{KL}(f||g) = 0$ if and only if $f = g$ every where. So if the transformed test set has the same distribution as the training set, the KL divergence reaches to its achievable minimum value, which is zero. For any distribution other than the training distribution, the KL divergence is greater than 0, so the optimal transformation is also unique.*

From a generative perspective, every distribution can be represented as a copula distribution and a set of univariate transformations. The Copula distribution is a marginal-free distribution bounded to the unit hypercube $[0 \;\; 1]^n$, and the set of univariate transformations, which are in the form of inverse of cumulation density functions, shapes the marginal distributions. Each dimension has its own univariate transformation. To draw a sample from a distribution based on the copula model, we first draw a sample from its copula distribution, which is a vector, and then apply the univariate transformation to each component. The same idea can be used to transform the testing distribution $\mathbf{x} \sim c(\mathbf{u}, R_f) \prod_{i=1}^{n} f_i(x_i)$ into the training distribution $\mathbf{y} \sim c(\mathbf{v}, R_g) \prod_{i=1}^{n} g_i(y_i)$. We first remove the marginals from the testing distribution by applying its marginal cumulative density functions $F_i(x_i)$ to each dimension. The resultant distribution has uniform marginals and it is equivalent to the copula distribution of the testing distribution. Then, we convert the Copula distribution into the Copula distribution of the training distribution using a pseudo-linear transformation $\mathbf{v}' = W\mathbf{u}'$ where $\mathbf{v} = \Phi(\mathbf{v}')$, $\mathbf{u} = \Phi(\mathbf{u}')$ and $W = R_g^{1/2} R_f^{-1/2}$. Finally, we

shape the marginals of the transformed distribution by applying the inverse of cumulative density functions $G_i^{-1}(x_i)$. Figure 4.3 shows the block diagram of the three stages of the copula-based transformation. In the following, we first investigate the optimality of copula-based transformation for two special cases and then show the optimality condition for the general case.

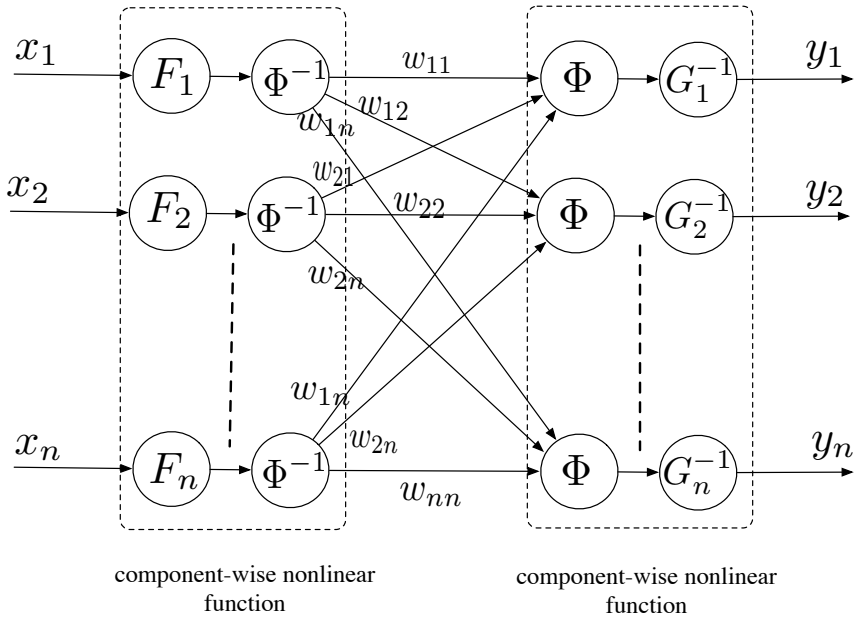


Figure 4.3: Three components of copula-based transformation: removing marginal distributions of test set, adjusting the Gaussian copula function and shaping the marginal distributions similar to the train set between the distribution of the training data and the transformed test data

Special case 1: Joint Multivariate Gaussian Distribution

Let $f(\mathbf{x}) \sim N(0, \Sigma_x)$ and $g(\mathbf{x}) \sim N(0, \Sigma_y)$ be two multivariate Gaussian distributions. From probability theory, we know that a linear transformation of a multivariate gaussian distribution is also a multivariate Gaussian distribution. The distribution of $\mathbf{x}$

under the linear transformation $T(\mathbf{x}) = \Sigma_y^{1/2}\Sigma_x^{-1/2}\mathbf{x}$ becomes a multivariate Gaussian distribution with covariance matrix $\Sigma = \left[\Sigma_y^{1/2}\Sigma_x^{-1/2}\right]\Sigma_x\left[\Sigma_y^{1/2}\Sigma_x^{-1/2}\right]^T = \Sigma_y$, which is equal to the distribution of $\mathbf{y}$. According to Copula theory, a multivariate Gaussian distribution is a special form of GCM where the marginals are Gaussian and the correlation matrix is $R_x = \left[diag(\Sigma_x)^{-1/2}\right]\Sigma_x\left[diag(\Sigma_x)^{-1/2}\right]$. Since the marginals are Gaussian, the i-th marginal cumulative function $F_i(x_i)$ in Eq. (4.1) is equal to $\Phi(\frac{x_i}{\sigma_{x_i}})$ where σ_{x_i} is the standard deviation of x_i. Similarly $G_i(y_i)$ is equal to $\Phi(\frac{y_i}{\sigma_{y_i}})$. By simplifying the component-wise functions in Figure 4.3 using $\Phi^{-1}(F_i(x_i)) = \frac{x_i}{\sigma_{x_i}}$ and $G_i^{-1}(\Phi(y_i)) = y_i\sigma_{y_i}$, the ultimate transformation represented as follows:

$$\begin{aligned} T(\mathbf{x}) &= diag(\Sigma_y)^{1/2}R_y^{1/2}R_x^{-1/2}diag(\Sigma_x)^{-1/2}\mathbf{x} \\ &= \Sigma_y^{1/2}\Sigma_x^{-1/2}\mathbf{x} \end{aligned}$$

which is equivalent to the optimal transformation, as described based on probability theory.

Special case 2: Multivariate Distribution with Independent Variables

For a multivariate distribution with independent variables, the probability density function $f(\mathbf{x}) = \prod_{i=1}^n f_i(x_i)$ (and $g(\mathbf{y}) = \prod_{i=1}^n g_i(y_i)$) is the product of marginal density functions. From probability theory, the optimal transformation is composed of n univariate transformations $x_i \to y_i : h_i = G_i^{-1}(F_i)$ where F_i and G_i are the i-th marginal cumulative distribution functions for $f(\mathbf{x})$ and $g(\mathbf{y})$. According to the copula model in Eq. (4.1), correlation matrices R_f and R_g are identity matrices and the weights are $w_{ii} = 1$ and $w_{ij} = 0$ for $i \neq j$. As a result, all Φ and Φ^{-1} terms in Figure 4.3 cancel out and result in the optimal transformation.

General Case: Gaussian Copula Model

Based on the copula model, two distributions are equivalent if both their copula distributions and marginals are identical. According to the aforementioned argument presented

for Special Case 2, the conversion of the marginals from one distribution into another distribution is straightforward using the cumulative density functions. Thus, the KL divergence between the training and transformed test distributions reaches its minimum value if the KL divergence between their corresponding copula distributions are at the minimum value.

Applying a pseudo linear transformation $\mathbf{v}' = W\mathbf{u}'$ to a Gaussian copula distribution $c(\mathbf{u}', R)$ results in another Gaussian copula distribution with a different correlation matrix $R_{new} = WRW^T$. Therefore, the KL divergence between the transformed testing and training gaussian copula density functions c_t and c_g is as follows:

$$D_{KL}(c_t||c_g) = \int \cdots \int_0^1 c_f(\mathbf{u}'; R_t) ln \frac{c_f(\mathbf{u}'; R_t)}{c_g(\mathbf{u}'; R_g)} d\mathbf{u}$$

where $\mathbf{u}' = \left[\Phi^{-1}(u_1) \ldots \Phi^{-1}(u_n)\right]^T$. By plugging this into copula density function from Eq. (4.1), the KL divergence can be written as:

$$\begin{aligned} D_{KL}(c_t||c_g) &= \int \cdots \int_0^1 \frac{1}{|R_t|^{\frac{1}{2}}} exp\{-\frac{1}{2}\mathbf{u'}^T(R_t^{-1} - I)\mathbf{u}'\} \\ &\times ln \frac{\frac{1}{|R_f|^{\frac{1}{2}}} exp\{-\frac{1}{2}\mathbf{u'}^T(R_t^{-1} - I)\mathbf{u}'\}}{\frac{1}{|R_g|^{\frac{1}{2}}} exp\{-\frac{1}{2}\mathbf{u'}^T(R_g^{-1} - I)\mathbf{u}'\}} d\mathbf{u} \end{aligned}$$

The derivative of $u = \Phi(u') = \int_{-\infty}^{u'} e^{-\frac{t^2}{2}} dt$ with respect to u' is $\frac{du}{du'} = e^{-\frac{u'^2}{2}}$, and thus,the KL divergence can be simplified further as follows:

$$\begin{aligned} D_{KL}(c_t||c_g) &= \int \cdots \int_{-\infty}^{\infty} \frac{1}{|R_t|^{\frac{1}{2}}} exp\{-\frac{1}{2}\mathbf{u'}^T R_t^{-1}\mathbf{u}'\} \\ &\times \frac{1}{2}\left[ln\frac{|R_g|}{|R_t|} - \mathbf{u'}^T R_t^{-1}\mathbf{u}' + \mathbf{u'}^T R_g^{-1}\mathbf{u}'\right] d\mathbf{u}' \\ &= \frac{1}{2}\left[ln\frac{|R_g|}{|R_t|} - n + \mathrm{tr}(R_g^{-1}R_t)\right] \\ &= \frac{1}{2}\left[ln\frac{|R_g|}{|WR_fW^T|} - n + \mathrm{tr}(R_g^{-1}WR_fW^T)\right] \end{aligned}$$

Where n is the dimension of testing and training distributions. By minimizing the KL divergence [2] with respect to W, we find the optimal transformation $W = R_g^{1/2} R_f^{-1/2}$.

4.3 Proposed Copula-Based Feature Enhancement for ASR

In this section, we propose a feature enhancement method for ASR systems based on the copula model. Simply, we estimate the distribution of feature vectors in the entire training set using a Gaussian copula model (GCM). We then find a nonlinear transformation for every utterance in the training and testing sets to match the distribution of each utterance with the distribution of the entire train set.

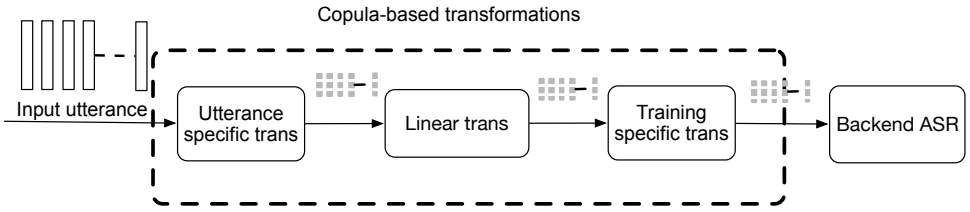


Figure 4.4: Block diagram of Copula-based feature enhancement method when the enhancement method is independent of the backend ASR

Using GCM to model distributions, the nonlinear transformation can be decomposed into three distinct blocks as depicted in Figure 4.4. The first block, which is the utterance-specific nonlinear transformation, is computed using the marginal CDFs of the input utterance and now it varies from one utterance to another. By utterance in this section, we mean a segment of speech represented as an ordered sequence of feature vectors for each 10 ms frames. The first block removes the marginal CDFs from the input utterance by replacing the real-valued input features with their normalized ranks in each utterance, so the output of the first block has uniform marginal distributions. The linear transformation block is a function of the correlation matrices of the input utterance and the training set as descried in Section 4.2. The aim of the linear transformation is to adjust the correlation

[2] At the optimum, $D_{KL}(c_t||c_g)$ reduces to zero.

of each utterance to be similar to the training set. The third block, which is a training specific transformation, is computed using the entire training set and doesn't vary across the utterances. The training specific transformation shapes the marginal distributions of the enhanced features. The marginal CDFs of the enhanced features are dictated by the type of nonlinearity in the third block. For example, if the aim is to have Gaussian marginal distributions for enhanced features, the transformations for the third block have to be quantile functions of the normal distribution. Previous works [44, 59] and our empirical experiments have shown that the inverse of marginal CDFs of the training set are a good choice for the training-specific nonlinear transformations, and demonstrated an acceptable performance improvement for ASR systems under different noise conditions.

4.4 Experimental Results on Aurora4 Dataset

4.4.1 Dataset and Baseline System

The Aurora 4 data set [95, 74] is an extension of the Wall Street Journal (WSJ0) dataset [38] that also contains noisy speech data. WSJ0 is a clean read-speech corpus with medium vocabulary size where the speech data was originally recorded using two microphones at 16 kHz. The primary microphone is a close-talking microphone, which is the same for all recordings. There is a secondary microphone, which is a desk mounted microphone, chosen randomly from a set of microphones. Although the standard WSJ0 provides a variety of training sets, the Aurora4 uses only WSJ0 SI-84 subset as the clean training set. The SI-84 set has 7138 utterances from 83 speakers and it only contains the data from the primary microphone. In addition to the clean training set, Aurora4 also has a multi-condition training set which consists of both clean and noisy speech data. The noisy utterances were obtained by manually adding six different types of noises (*street traffic*, *train station*, *car*, *babble*, *restaurant*, *airport*) to the clean data as depicted in Figure 4.5 where half of the utterances are from the secondary microphone to introduce channel distortion.

The level of the noise in the multi-condition training set has been chosen randomly such that SNR value ranges from 5 to 15 DB. The evaluation set in Aurora4 consists

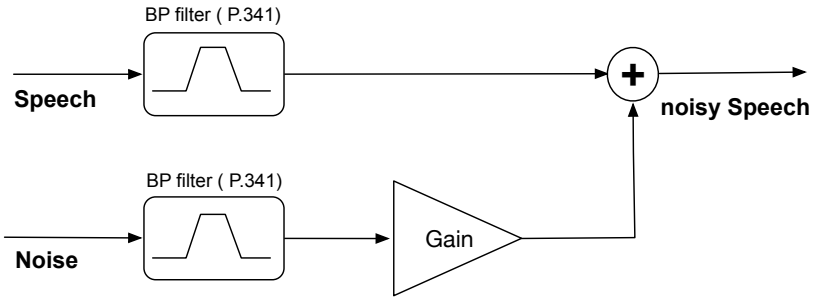


Figure 4.5: Block diagram of noise addition process for Aurora 4

of 14 subsets which includes the Nov92 evaluation set and its noisy variants from both primary and secondary microphones. The Nov92 evaluation set has 330 utterances from 8 speakers. Similar to the evaluation set, the development set is composed of 14 subsets derived from 330 utterances of the Nov92 development test set. In contrast to the training set, the level of SNR varies from 5 to 15 dB in evaluation and development sets.

We utilize Kaldi [77], an open source speech recognition toolkit, to implement a baseline ASR. The baseline for the Aurora4 dataset consists of both GMM-HMM and DNN-HMM based ASR systems. The GMM-HMM based ASR system consists of monophone and triphone models. For the monophone model, each phone is modeled by a GMM-HMM model making 350 HMM states with a total of 1k Gaussian components in the final monophone model.

The triphone model is obtained by replacing each single phone in the monophone model with a group of three consecutive phones and applying Linear Discriminative Analysis (LDA) and Maximum Likelihood Linear Transformation (MLLT). The final triphone model includes 2500 shared HMM states with a total of 15k Gaussian components. The DNN-HMM based ASR system utilizes DNN to implicitly model the output probability for each state in HMM. The input feature for DNN is the concatenation of filter bank features for a context window with length 11 frames and the target is the state level alignment obtained from the triphone model. We use a DNN with 7 hidden layers where each hidden layer has 2048 sigmoidal units. We train the first DNN using the alignment obtained from the GMM-HMM baseline. Then, we use the alignment obtained from the first DNN to

Table 4.1: Details on dataset and baseline ASR system for Aurora 4

Dataset	Train	2mics × [890 clean + 6 × 440 noisy] Utts 83 Spks 2mics ×14 hours	
	Dev	330 × 14 utts 10 Spks	
	Eval	330 × 14 utts 8 Spks	
Baseline	GMM-HMM	Monephone	39 MFCC+Δ +ΔΔ 350 HMM states 1000 Gaussians
		Triphone	39×3 input features LDA 91 to 40 2.5k HMM states 15k Gaussians
	DNN-HMM	11x40 input featuers 7 hiddens layers with 2048 units 2.5k output Cross entropy training	

train the second DNN.

4.4.2 Effect of Marginal Estimation

Accurate estimation of marginal distribution play a crucial role in the Copula model as described in Equation 4.1. In this section, the aim is to investigate the effect of different marginal estimators on the performance of copula-based feature transformation for an ASR system. We assume that the transformation only consists of marginal distributions by $W = I$ in Figure 4.3. For simplicity, we limit our experiments to a monophone system for Aurora 4. In order to equalize the effect of quantization for all methods, we represent each estimated CDF with a lookup table where the size of the table is equal to the number of data points in training set.

We use three different methods to estimate marginal CDFs. Then, we apply the estimated CDF of the each utterance to convert the of original features into a unit interval and use the inverse of the estimated CDF of the entire training data, which is also a lookup table, to covert back. We use a linear interpolation for missing values in the lookup table.

For the first method, we use cumulative histogram approach where the CDF of x is

equal to the fraction of data points that is less x.

Our second method is Gaussian kernel density estimation (GKDE) [75] which consists of the summation of several gaussian kernel functions centered on training data points $\{x_i\}_{i=1}^{N}$ as follows:

$$\hat{f}(x) = \frac{1}{Nh} \sum_{i=1}^{N} k(\frac{x - x_i}{h}) \tag{4.3}$$

where h is the bandwidth of the gaussian kernel and can be computed based on the empirical standard deviation $\hat{\sigma}$ [84]

$$h = (\frac{4\hat{\sigma}^5}{3N})^{0.2} \tag{4.4}$$

The third method is kernel density estimation via diffusion [20, 19] which has been shown better performance than the other two methods for estimating multimodal and bounded distributions. This method provides a computationally inexpensive way to estimate a univariate PDF even though the theoretical justification is somewhat complicated. To estimate a PDF using the kernel density estimation via diffusion, we first compute the histogram of the input data. We then apply the discrete cosine transform to the histogram and multiply the result with a constant complex vector.[3] Finally, we apply the inverse of the discrete cosine transform to obtain the PDF. We can also approximate the CDF by computing the cumulative sum of the PDF.

Table 4.2 presents the monophone results on Aurora4 with enhanced features for training and testing obtained by different marginal estimators. The monophone system is trained using the multi conditions training set and $W = I$.

Table 4.2: Monophone WERs on Aurora 4 eval set trained and tested with enhanced features. The enhanced features are obtained using different marginal estimators, multi conditions training set and $W = I$.

Marginal estimator	WER[%]
Histogram	35.1
Gaussian KDE	36.3
KDE via diffusion	**34.6**

[3]Constant complex vector is pseudo delay in frequency domain $e^{-jk^2\pi^2/2t}$ and controls the bandwidth of KDE via diffusion.

From the results, the KDE via diffusion outperforms mainly due to its discriminatory power to deal with multimodality and boundary problems in estimating univariate distributions. The Gaussian KDE has the worst WER because its performance depends on the bandwidth, which is hard to estimate accurately.

In the previous experiment, we represented the CDFs of the entire training set using a lookup table where the number of entries is equal to the total number of feature vectors in the training set. In practice, it is not feasible because the number of feature vectors in the training set are huge even for a moderate-size data set. A simple way to address this problem is to uniformly quantize the inverse of CDF, which is called quantile function. The quantization for quantile functions is fairly straightforward and simple because their domain are always limited to the unit interval [0 1].

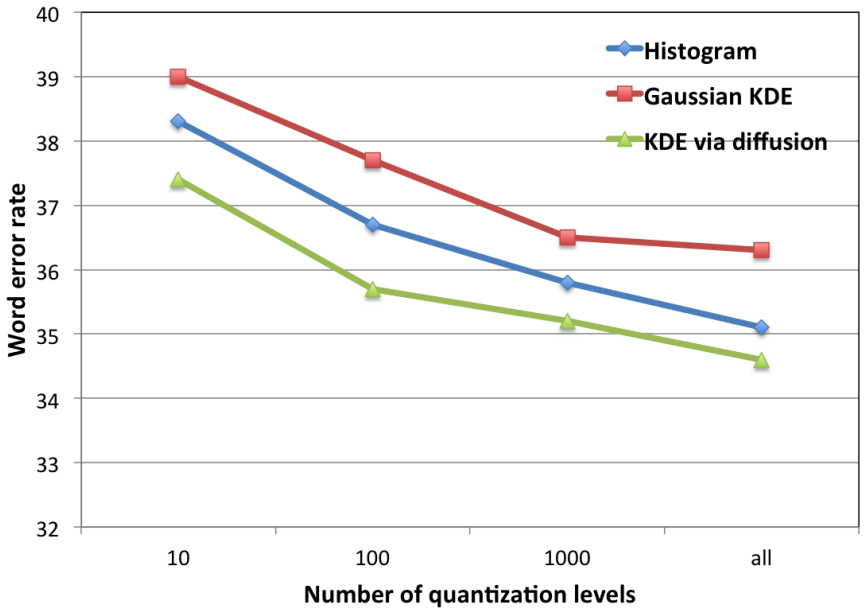
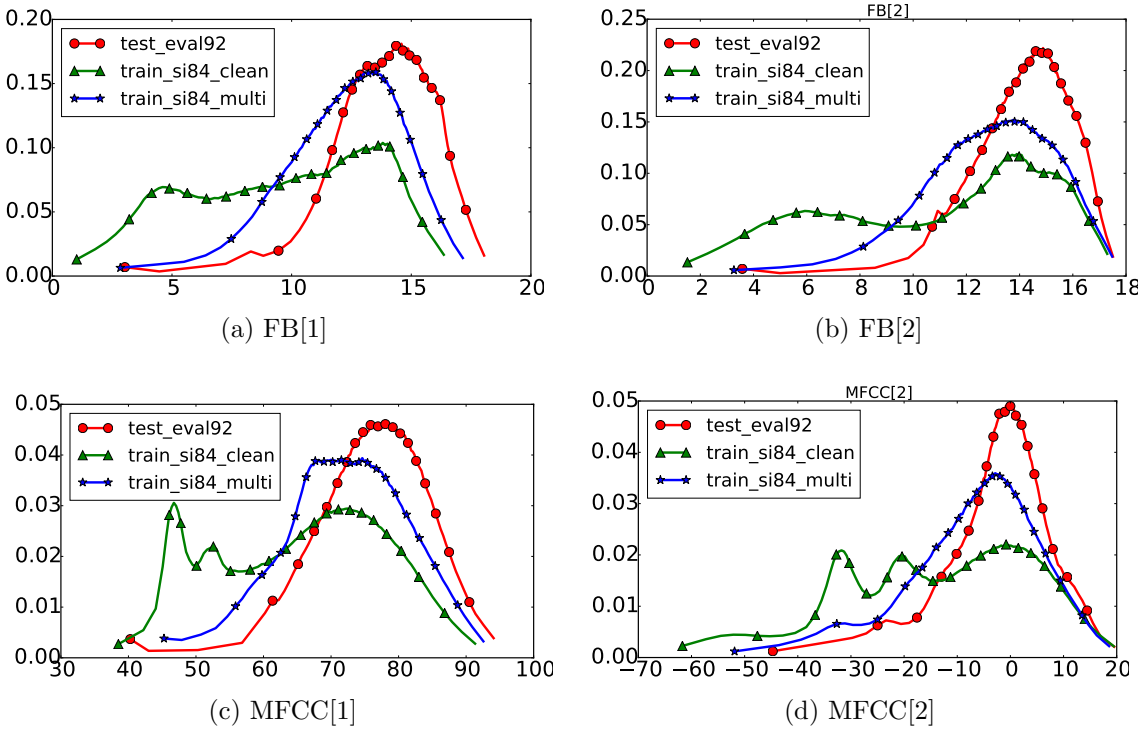


Figure 4.6: The effect of quantization level of the quantile functions (inverse of the CDFs) on the performance of monophone system for Auroa4. The monophone systems are trained and tested with enhanced features. And the enhanced features are obtained using different marginal estimators, multi conditions training set and W = I.

From the graph in Figure 4.6, we see that the performance of the monophone system improve when we increase the number of quantization levels. This is also noticeable that KDE via diffusion with the quantization level of 100 is an acceptable compromise between



(a) FB[1] (b) FB[2]

(c) MFCC[1] (d) MFCC[2]

Figure 4.7: Empirically computed marginals on the training set – clean and multi-conditional data – and the test set for first two filter bank and MFCC features.

the computational complexity and the performance, so we use KDE via diffusion with the quantization level of 100 to estimate CDFs in the rest of this chapter.

4.4.3 Analysis of Marginal Distributions

A simple check for suitability of copula-based normalization is to measure the mismatch between the marginal distributions of training and test datasets. In Figure 4.7, we plot the marginals of the first two filter bank coefficients and MFCC features computed on the training set (clean and multi-condition) and test set. Compared to the clean training set, the multi-conditional training set is closer to the test set. This may explains the popularity of using multi-conditional training data for acoustic model training. However, even with that, there is still a significant difference compared to the test set.

4.4.4 Analysis of Normalization Style

There are two approaches to apply the copula-based normalization to ASR systems. The first approach is to train the backend ASR using the original features and use the copula-based normalization just to alleviate the mismatch between the training and test conditions during the test time. The main advantage of this approach is to provide a quick and simple way to add noise-robustness to already trained ASR systems without retraining them. The second approach is to normalize every utterance (in training and test set) so they are close to a template distribution (corresponding to that of entire unnormalized training data) and then use the normalized features both for training and testing. In Figure 4.8, we compare the performance of the monophone systems on Aurora 4 task with different strategies to normalize the training and test sets. For this experiment, we use clean and multi-condition training sets for training, and evaluation set for evaluating the monophone systems.

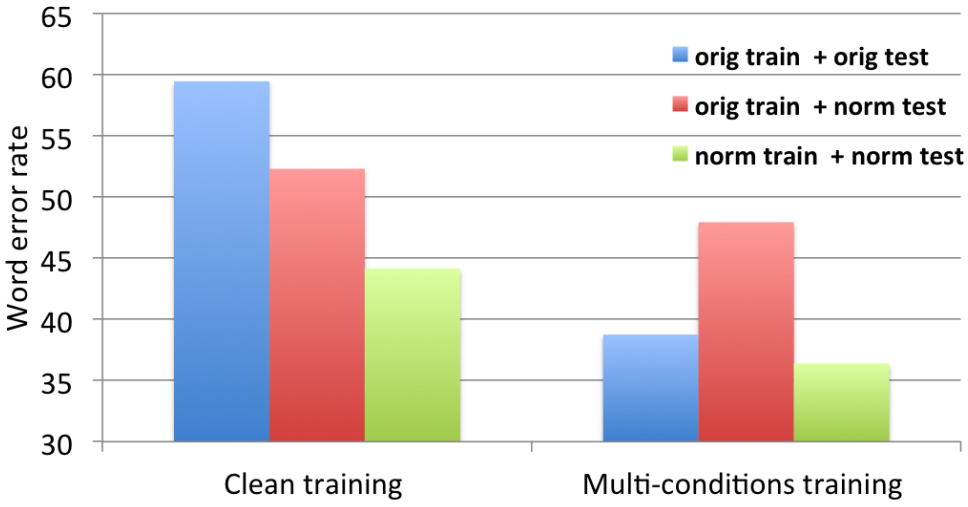


Figure 4.8: Monophone WERs on Aurora 4 evaluation set with different normalization configurations for clean and multi-conditions train sets.

Experimental results reveal that the normalization of the test set improves the performance of already trained ASR system with clean training set while only normalization of the test set degrades the performance of ASR system trained by multi-condition training set. The best performance is obtained by normalizing features for both training and testing, and using multi-condition training set.

The evaluation set of Auroa 4 can be decomposed into four subsets : *clean*, *distorted clean*, *noisy* and *distorted noisy*. Figure 4.9 plots the performance of monophone systems on each subset for different normalization strategies. Except for the clean set, apply copula normalization at both test and training time in conjunction with multi-conditional training provides the best performance. From the results of the clean subset in Figure 4.9, it is noticeable that the normalization of features for the clean training improves the performance of ASR on clean subset.

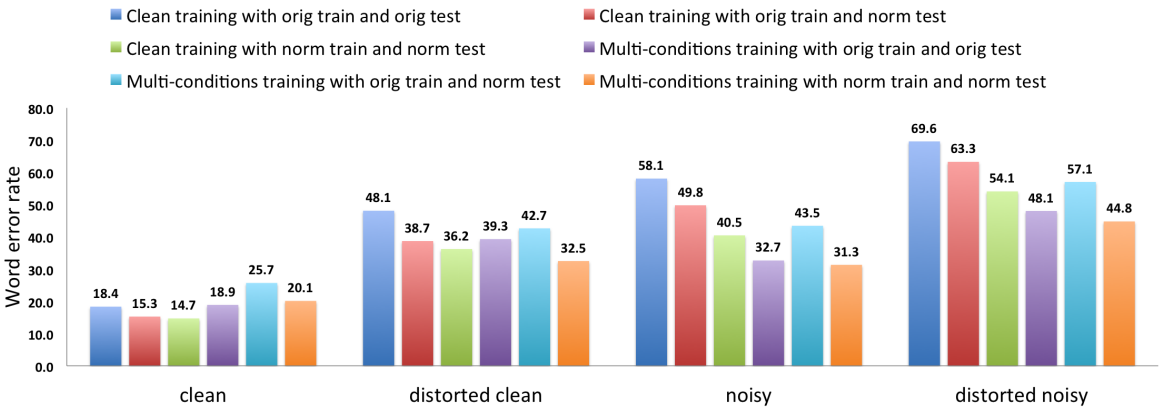


Figure 4.9: Monophone WERs on different subsets of Auroa 4 eval set with different normalization configurations for clean and multi-conditions training sets.

In the rest of this chapter, we limit our experiments to the best normalization style.

4.4.5 Effect of Normalization on Triphone and DNN Based Models

In this section, we investigate the effect of the copula based normalization on the performance of triphone and DNN-based models, which are more complicated than monophone system. In Table 4.3, we compare the WER of the triphone model with original MFCC, normalized MFCC without correlation correction $W = I$ and normalized MFCC with correlation correction $W = R_g^{1/2} R_f^{-1/2}$ where R_g is a global correlation matrix computed over the entire train set and R_f is per utterance correlation matrix. R_g and R_f are estimated using full and Toeplitz matrix structures respectively [13]. The results in Table 4.3 show that the copula based normalization with correlation correction gives 9% relative WER improvement over the original MFCC features.

Table 4.3: Average WER of clean, noisy, distorted clean and distorted noisy conditions for triphone model on Aurora 4 task with different features: original MFCC, normalized MFCC without correlation correction $W = I$ and normalized MFCC with correlation correction. The model is trained by multi conditions training set

	original MFCC	**normalized MFCC with $W = I$**	**normalized MFCC with $W = R_g^{1/2} R_f^{-1/2}$**
Triphone WER	19.4	18.3	**17.7**

Table 4.4 reports the WER of DNN based models with original FB, normalized FB without correlation correction and normalized FB with correlation correction. For this experiment, we use the alignment of the triphone model, which is obtained either by original MFCC or by normalized MFCC with $W = R_g^{1/2} R_f^{-1/2}$. The results show the DNN based model with the proposed copula based normalization achieves 21% relative improvement over the DNN baseline with original features.

Table 4.4: Average WER of clean, noisy, distorted clean and distorted noisy conditions for DNN model on Aurora 4 with different features: original FB, normalized FB without correlation correction $W = I$ and normalized FB with correlation correction. The model is initialized by the alignment of triphone model, which is obtained either by original MFCC or by normalized MFCC with $W = R_g^{1/2} R_f^{-1/2}$.

	original FB	**normalized FB with $W = I$**	**normalized FB $W = R_g^{1/2} R_f^{-1/2}$**
DNN + alignment of original MFCC	13.4	11.8	11.6
DNN + alignment of normalized MFCC	12.7	11.1	**10.5**

In addition, our best result on Auroa 4 is comparable with the state of the art methods for this task as shown in Table 4.5.

4.5 Experimental Results on CHIME 4 Dataset

4.5.1 Dataset and Baseline System

In this section, we use the 4th CHIME speech recognition challenge [91], which provides read speech dataset and Kaldi-based baseline to train and evaluate the performance of

Table 4.5: Average WER of clean, noisy, distorted clean and distorted noisy conditions of our best model compared with other state of the art methods on Aurora 4.

	WER
Recurrent deep neural networks [93]	12.7
DNN noise-aware training [83]	12.4
Joint noise adaptive training [71]	11.1
our best	**10.5**

ASR systems under different noisy conditions for multichannel speech data. The corpus consists of real and simulated noisy speech data. Real noisy speech data have been recorded in real noisy environments such as *cafe*, *bus*, *street junction* and *pedestrian area* using a portable device with an array of six microphones as shown in Figure 4.10 and, a closed-talk microphone. The simulated noisy data were generated by artificially mixing clean data with different background noise. Background noises have been recorded using the same device without close-talking microphone. The clean data used for generating the training

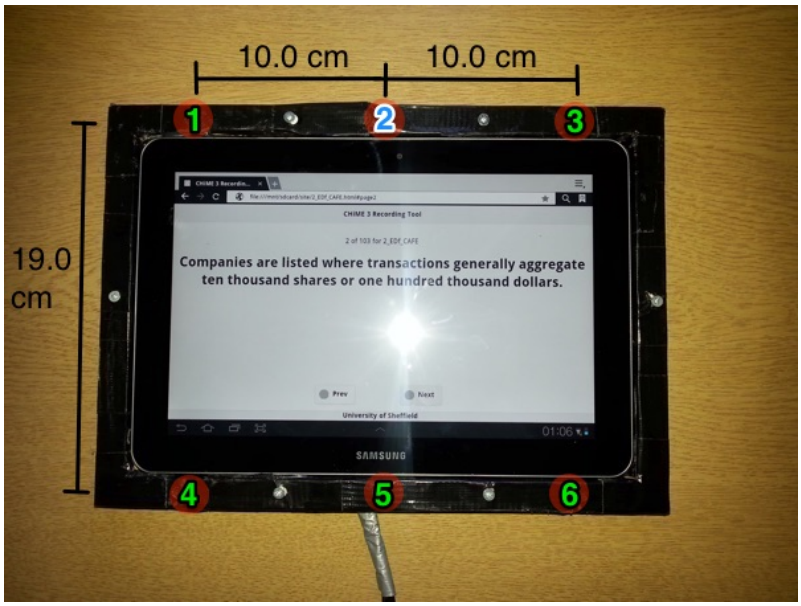


Figure 4.10: Recording device used to capture multi-channel audio for the 4th Chime challenge [92].

set is SI-84 subset from the Wall Street Journal (WSJ0) corpus that consists of 7138 clean recordings from 84 speakers with different dialects. For the development and test sets, a clean data were recorded in a quiet place, which is called booth, using the same portable

device. The mixing process for generating simulated noisy data for the development(or test) and train sets are slightly different. Both clean data and background noise are multi channel data for development (or test), and thus the mixing process is to simply add corresponding channels of the clean and noisy data. The clean data for generating the training, which is the SI-84 subset of WS0, is single channel. So it is necessary to convert the single channel data to multichannel data before adding noise. This conversion process consists of two steps. The first step is to randomly choose a segment from real noisy speech data and estimating a set of impulse responses in the form of IIR filters between the closed-talk channel and other channels. Then, this set of IIR filters are applied to single channel data from the WSJ0 to obtain multichannel clean data. The second step is to add multichannel noise to clean data to generate simulated multichannel noisy data. The overall process of generating simulated noisy data from single channel data is depicted in Figure 4.11.

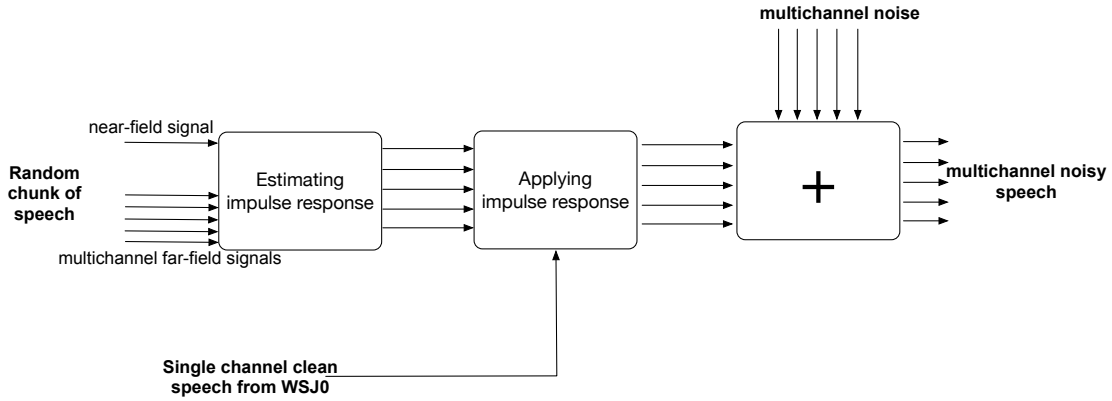


Figure 4.11: Block diagram of noise addition process for CHIME 4

To evaluate the effect of multichannel framework on the performance of ASR systems, the 4th CHIME defines three tracks, in which the number of available channels is limited at the test time. For the 1-channel (and 2-channel) track, each utterance consists of one channel (and two channels), which is randomly chosen. And for the 6ch-track, all six channels are available during the test time.

For our experiments in this section, we use single channel audio data, particularly

channel 5, to train the baseline. For 6ch and 2ch track, we use an open source tool, BeamFormIt, for converting the multichannel data into a single channel before testing [4].

Table 4.6: More details on dataset and baseline configurations for CHIME 4

Dataset	Train	8738 simulated noisy utts from 83 spks 400 × 4 real noisy utts from 4 spks 4 types of noise	
	Dev	410 × 4 real noisy utts 410 × 4 simulated noisy utts 4 types of noise 4 spks	
	Test	330 × 4 real noisy utts 330 × 4 simulated noisy utts 4 types of noise 4 spks	
Baseline	GMM-HMM	Monophone	39 MFCC+Δ +ΔΔ 350 HMM states 1000 Gaussians
		Triphone1	39 MFCC+Δ +ΔΔ 2.5k HMM states 15k Gaussians
		Triphone2	LDA 91 to 40 MLLR 40
		Triphone3	FMLLR
	DNN-HMM	DNN	11x40 input featuers 7 hiddens layers with 2048 units 2.5k output Cross entropy training
		DNN+SMBR	Sequence discriminative training State-level minimum Bayes risk
		LM rescoring	Recurrent neural network Smoothed 5-gram Knesser-Ney

Our baseline for GMM-HMM system builds gradually from scratch and consists of multiple GMM-HMM subsystems. We first train a monophone system using flat start initialization method. Then, we use the alignment of the monophone system to train the first triphone system. We use MFCC, its delta and delta-delta features as the input for the monophone and first triphone systems. We utilize the alignment of the the first triphone system to train the second triphone system using LDA and Maximum Likelihood Linear Transforms (MLLT) feature transformations. The input feature for the second triphone system is the context feature of 13 MFCC with window length 9, which is obtained by splicing four left and four right frames. We utilize Feature space Maximum Likelihood Linear Regression (FMLLR), which is a common method for speaker adaption, to obtain

the final triphone system. Next, we use FMLLR features as input for a DNN with 7 hidden layers where each layer has 2040 sigmoidal units. We initialize the DNN using RBM pretreating method and use the cross-entropy training to obtain the first DNN. Then, we use state-level minimum Bayes risk (smbr) training to compute the second DNN. Finally, we use Recurrent Neural Network (RNN) and Smoothed 5-gram Knesser-Ney (5gkn) language models for improving the performance further. Table 4.7 shows the recognition results of the baseline systems on different tracks of CHIME 4.

Table 4.7: Average WERs of the baseline systems trained on single channel data.

Track	System	Dev		Test	
		simu	real	simu	real
1ch	GMM	24.2	21.8	33.5	37.3
	DNN	17.4	16.5	26.0	30.0
	smbr	15.8	14.6	24.0	27.1
	smbr+5gkn	13.9	12.3	22.1	24.3
	smbr+rnn	12.8	11.5	20.8	22.9
2ch	GMM	18.7	16.3	27.3	28.7
	DNN	13.5	12.2	20.4	22.4
	smbr	12.1	10.8	18.8	20.0
	smbr+5gkn	10.7	9.6	16.4	17.6
	smbr+rnn	9.3	8.4	15.2	16.2
6ch	GMM	14.2	12.7	21.1	21.7
	DNN	10.1	9.5	15.9	16.6
	smbr	9.0	8.2	14.2	14.7
	smbr+5gkn	7.8	7.0	12.1	12.8
	smbr+rnn	6.7	6.0	10.9	11.3

4.5.2 Effect of Copula Based Normalization

Similar to our best configuration for the copula based normalization on Aurora4, we obtain our models in 3 stages: (a) estimating a canonical multivariate copula distribution of the 13-dim MFCC features using all the utterances in the single channel noisy training data; (b) transforming each utterance in the training data to reduce the KL distance distance between the multivariate distribution of the enhanced utterance and the canonical distribution; and (c) training a standard acoustic model using enhanced features. At the test time and before decoding, we use the same way to enhance the features of each

utterance.

Figure 4.12 illustrates WERs of smbr+rnn model, which is the best baseline model, for different tracks of 4th CHIME trained with original and copula-based enhanced features. The gains are particularly remarkable in a single channel test set where the mismatch problem between the train and test is more severe. The mismatch problem for two other cases (2ch and 6ch) is less problematic mainly because we apply beamforming before decoding, which reduces the effect of noise.

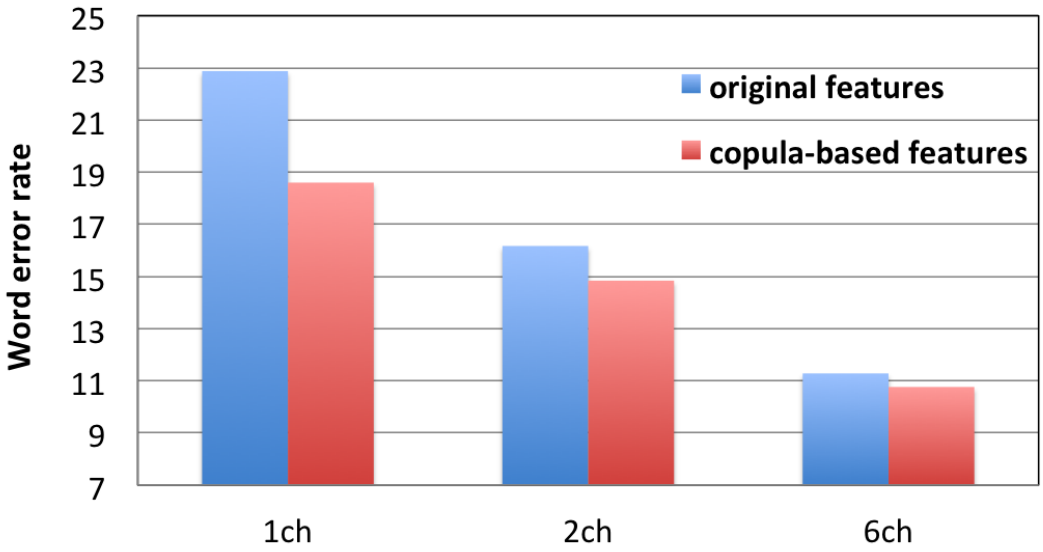


Figure 4.12: Recognition results of the best baseline model, which is smbr+rnn, on different tracks of 4th CHIME when the model is trained with : original and copula-based enhanced features.

In Figure 4.13, we report the performance of different systems on 1-ch track trained either by the original features or by the copula-based enhanced features. The results show that the copula-based features consistently improve the performance of all the models listed in the baseline.

Up until now, we have used the recognition results on the test set, which consists of both simulated and real noisy data, to evaluate the performance of ASR models on the 4th CHIME task. However, the ultimate goal is to improve the performance of ASR systems on real noisy data, which is a more realistic case. Figure 4.14 shows the WERs of smbr+rnn system on real and simulated subsets of 1-ch track. It is noticeable that

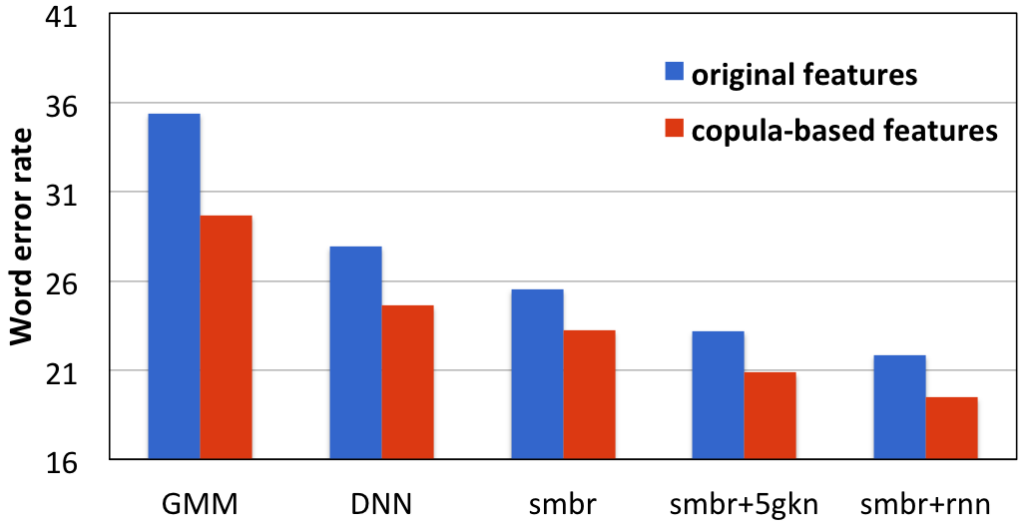


Figure 4.13: WERs of baseline models on 1-ch track of the 4th CHIME task. Note, 5gkn and rnn stand for 5-gram Knesser-Ney and recurrent neural network language models respectively.

the copula-based features improves the recognition result on real noisy data while the improvement for simulated noisy data is negligible.

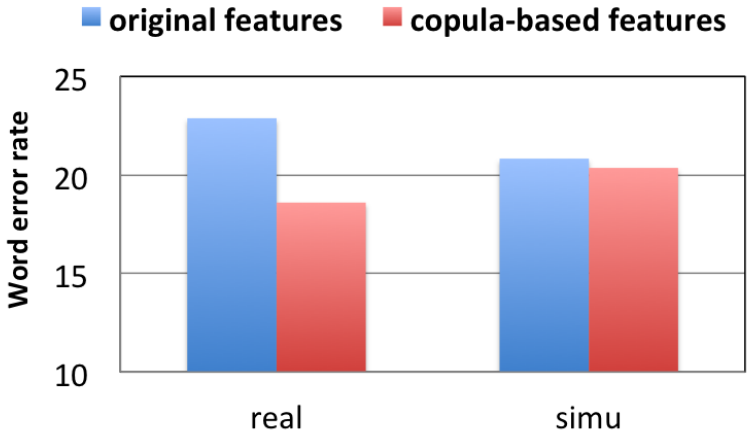


Figure 4.14: WERs of smbr+rnn system on real and simulated subsets of 1-ch track when the model is trained with : original and copula-based enhanced features.

In Figure 4.15, a comparison between the original and copula-based features on the performance of smbr+RNN model shows that the gain is highest in *bus* background noise.

Our hypothesis is that there is more structure and correlation in the noise in this case for which the multivariate copula is an appropriate representation.

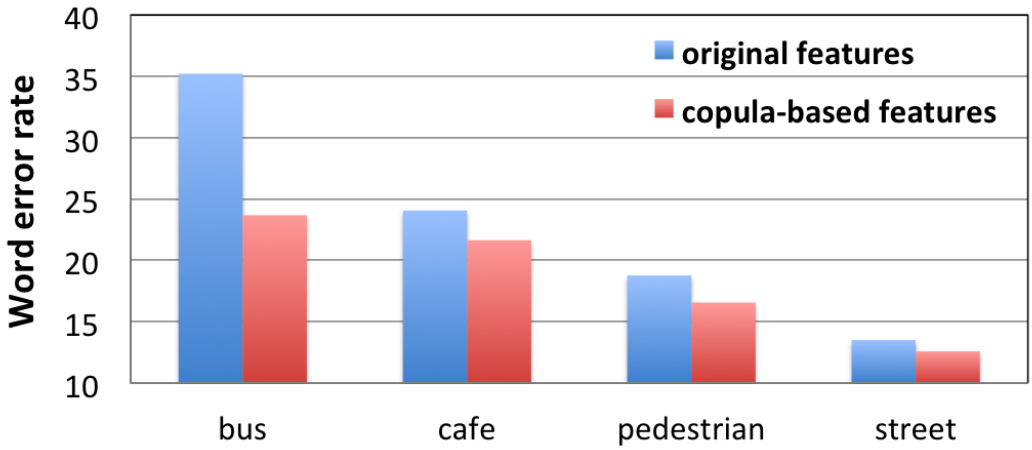


Figure 4.15: Recognition results of smbr+RNN model on different real noisy subsets of 1-ch track eval set. when the model is trained with : original features and copula-based enhanced features.

Finally, our copula based system is sufficiently different from the baseline system that we are able to obtain additional gain through system combination using minimum Bayes risk [94], as reported in Table 4.8. Table 4.9 lists the WERs of the combined model on different tracks for different noises.

4.5.3 Analysis of Channel and Beam Forming Distortions

In this section, we use copula-based normalization to address two other mismatch problems in the 4th CHIME task. Perviously, we trained our models using single channel data (channel 5). For 6ch and 2ch tracks, we first apply beam forming to convert multichannel data into a single channel data, and then use the resultant single channel data for decoding. For 1ch track, we directly use single channel data without any further preprocessing for decoding. The key point of the above configuration is to provide a single ASR system that works with single and multi channel data. However, there are still two mismatch problems that we are not able to address using the above configuration. For 1ch track, the above configuration suffers from channel mismatch problem because the test set consists of single channel audio data from different channels and the training set comes from a

Table 4.8: Average WERs of the baseline systems trained on single channel features after copula-based transformation.

Track	System	Dev		Test	
		simu	real	simu	real
1ch	GMM	23.0	19.8	30.0	29.4
	DNN	17.6	15.4	24.9	24.4
	smbr	16.5	13.9	23.5	23.1
	smbr+5gkn	14.7	12.1	21.7	20.1
	smbr+rnn	13.2	10.7	20.4	18.6
	copula+baseline	**12.1**	**9.8**	**19.2**	**18.6**
2ch	GMM	18.1	15.2	24.9	24.4
	DNN	13.9	12.1	20.4	19.8
	smbr	12.7	10.7	19.1	18.2
	smbr+5gkn	10.9	9.1	17.2	16.4
	smbr+rnn	9.6	8.0	15.6	14.8
	copula+baseline	**8.8**	**7.3**	**13.9**	**13.8**
6ch	GMM	14.4	12.5	19.7	19.3
	DNN	10.8	9.6	16.0	15.4
	smbr	9.8	8.2	15.2	14.5
	smbr+5gkn	8.2	7.1	13.0	12.2
	smbr+rnn	7.1	6.1	11.7	10.8
	copula+baseline	**6.3**	**5.4**	**10.1**	**10.1**

Table 4.9: Average WERs after combining the baseline and copula-based system using MBR decoding.

Track	Envir.	Dev		Test	
		simu	real	simu	real
1ch	BUS	10.3	12.6	13.8	26.0
	CAF	15.7	10.5	23.5	20.8
	PED	9.3	6.6	18.8	15.7
	STR	12.9	9.6	20.6	11.9
2ch	bus	7.2	9.2	10.0	19.4
	CAF	11.8	7.5	16.2	14.1
	PED	6.9	4.9	14.2	12.0
	STR	9.1	7.7	15.2	9.7
6ch	bus	5.3	6.8	6.7	13.3
	CAF	7.7	5.1	11.2	9.5
	PED	5.1	3.9	10.0	8.5
	STR	7.2	5.7	12.5	9.1

specific channel, which is channel 5. A simple solution for the channel mismatch problem is to use a new train set, which has all the channels. Table 4.10 lists the recognition results for the 1ch track. The results show that the new training set significantly improves the performance and further gain is achieved by combining the new training set with the copula based normalization.

Table 4.10: WERs of smbr+RNN system on 1-ch track for different training configurations.

Training configuration	Dev		Eval	
	simu	real	simu	real
ch5 + original feat	12.8	11.5	20.8	22.9
ch5 + copula-based feat	13.2	10.7	20.3	18.6
all channels + original feat	**11.3**	9.4	16.9	17.7
all channels + copula-based feat	11.7	**8.2**	**16.6**	**16.3**

Regardless of the type, the aim of beam forming is to use multiple channel data to reduce the effect of noise, and it doesn't necessarily produce a single channel data with the same characteristic as to a specific channel. So, it generally causes a systematic mismatch between the training and test conditions if we use a specific channel for training. To address this problem, we can augment the original training set by an additional set such that the mismatch between the augmented training and test sets is minimized. We construct an additional set for 2-ch track by choosing all combinations of two channels for each utterance in the training set, and applying beam forming. Table 4.11 shows that the augmented training set with copula based normalization outperforms on the test set of 2ch track.

Table 4.11: WERs of smbr+rnn system on 2-ch track for different training configurations.

Training configuration	Dev		Eval	
	simu	real	simu	real
ch5 + original feat	9.3	8.4	15.2	16.2
ch5 + copula-based feat	9.6	8.0	15.6	14.8
augmented data + original feat	**8.7**	**7.5**	14.4	14.9
augmented data + copula-based feat	9.1	7.9	**14.3**	**14.1**

Similar to 2ch track, we construct an additional set using all the channels for 6-ch track. Table 4.12 reports the recognition results for smbr+RNN system. The results show that

the augmentation method improves the recognition results over the single channel training. In addition, the combination of the augmentation and the copula based normalization significantly improves the performance of the baseline system.

Table 4.12: WERs of smbr+RNN system on 6-ch track when the training set is: channel 5, channel 5 with copula-based feat, augmented data and augmented data with copula-based feat

Training configuration	Dev		Eval	
	simu	real	simu	real
ch5 + original feat	**6.7**	6.0	10.9	11.3
ch5 + copula-based feat	7.1	6.1	11.7	10.8
augmented data + original feat	7.2	**5.9**	**10.1**	10.5
augmented data + copula-based feat	7.3	6.7	10.3	**9.6**

4.6 Integration into Acoustic Model

4.6.1 Motivation

The choice of the training-specific transformation, presented in section 4.3, has a major effect on the overall performance of the enhancement method for ASR systems. Table 4.13 shows the WERs of GMM-HMM system on the 1ch track of the 4th CHIME task for two different training-specific transformations. For this experiment, we impose the enhanced features to have either uniform marginal distributions or a set of marginal distributions similar to the entire training set by choosing proper training-specific transformations. For the uniform distribution, we extract the range [xmin, xmax] for each dimension where the xmin and xmax are the minimum and maximum values across the entire training set. Then, we use the inverse CDF of uniform distribution over [xmin, xmax] as training-specific transformation. For the latter one, we utilize the inverse of marginal CDF of the entire train set as training-specific transformation. Results in Table 4.13 show that WERs for two training-specific transformations are significantly different. This difference motivates us to propose a method to learn the training-specific transformation automatically during the training of the ASR system rather than computing it independently.

Table 4.13: Recognition results of GMM-HMM system on 1-ch track of CHIME 4 for two different training-specific transformation. For this experiment, we assume $W = I$

	WER[%]
Uniform marginals	33.6
Marginals similar to entire training set	31.3

4.6.2 Problem Definition

As mentioned before, the performance of the ASR system with the proposed feature enhancement heavily depends on the choice of the training-specific transformation. Previously, we performed the feature enhancement method on data, which was completely independent of the backend ASR. Then, we computed the the acoustic model using the enhanced features [16, 17]. In this section, we present a new method to compute the training-specific transformation automatically as a part of the backend ASR as depicted in Figure 4.16. In order to integrate the training-specific transformation into the backend ASR, we assume that the training-specific transformation is part of the acoustic model in the ASR system and has a parametric vector-valued function with unknown parameter C as follows :

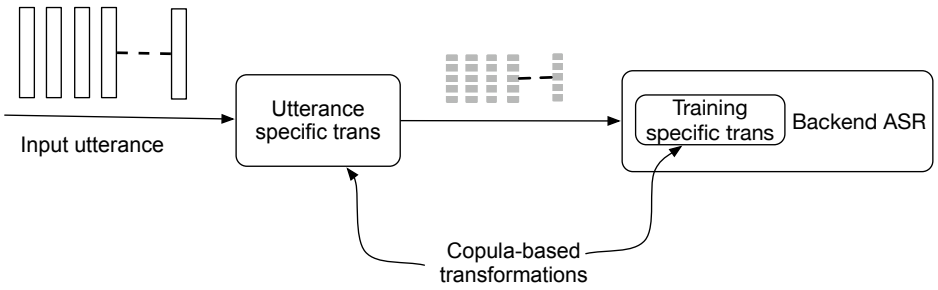


Figure 4.16: Block diagram of copula-based feature enhancement method when the enhancement method is a part of backend ASR

$$\mathbf{x} = \mathbf{f}(\mathbf{u}; C) = \begin{bmatrix} f_1(u_1) \\ f_2(u_2) \\ \vdots \\ f_n(u_n) \end{bmatrix} \tag{4.5}$$

where **u** and **x** are normalized rank and enhanced feature vectors respectively. We further assume that each coordinate function $f_i(u_i)$ is a polynomial function with two constraints to retain the range of the enhanced features similar to the training set:

$$f_j(u) = \sum_{k=0}^{K} c_{kj} u^k$$
$$\text{subject to} \quad f_j(0) = x_{min,j}, f_j(1) = x_{max,j}$$

where $[x_{min,j}, x_{max,j}]$ is the range of i^{th} feature in the training set. Since the normalized rank features are limited to the unit interval [0 1], the constraints for the coordinate functions guarantee that the enhanced feature can not have a range beyond the training set. By enforcing the range for each feature using the above constraints, we also prevent the model from learning the constant function, which is trivial. Regardless of the type of acoustic models, we generally can find the optimal acoustic model and training specific transformation by adding the parameters of the training specific transformation C to the parameters of the acoustic model Θ and optimizing the acoustic model with respect to all parameters. In this section, we consider GMM-HMM as the acoustic model and use a modified Expectation Maximization (EM) to jointly maximize the likelihood of the model with respect to all parameters.

4.6.3 Proposed Method

Maximum Likelihood (ML) is a simple framework for fitting a parametric density function $p(x|\Theta)$ to observation data $\mathcal{X} = \{x_i\}_{i=1}^n$. The goal of the ML approach is to find the parameters Θ that maximize the log-likelihood of the parameters given the observation data :

$$L(\Theta) = \log \prod_{i=1}^{N} p(x_i|\Theta) \tag{4.6}$$

In Equation 4.6, we assume that the observation data are independent and identically distributed samples **directly** drawn from the model $p(x|\Theta)$. Our problem in this section is different from the typical ML scenario, when we have access to direct samples of the model. Assume that our observation set $\mathcal{U}$ is an image of direct samples under a nonlinear transformation as illustrated in Figure 4.17. In addition to the parameters of the model Θ,

the parameters of the transformation C are also unknown. Our aim is to use the observed samples to jointly estimate all the parameters.

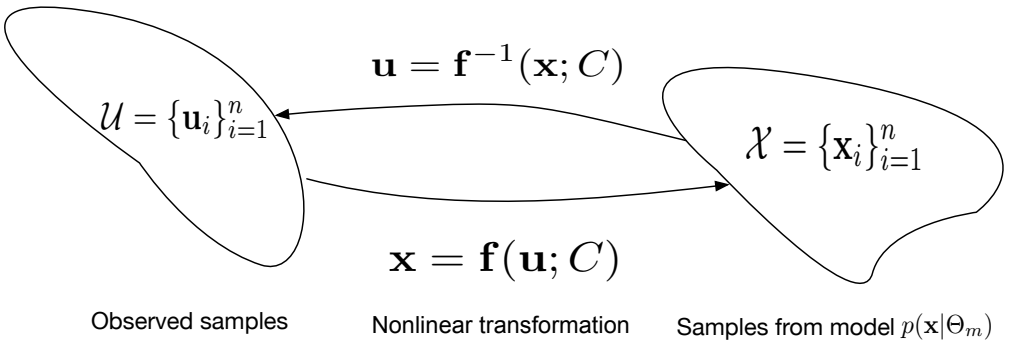


Figure 4.17: Schematic of the generation of observation data when the model and the observed data points reside in two different domains

In general, the log-likelihood function of the parameters with respect to the observation data is defined as :

$$L(\Theta, C) = \log \prod_{i=1}^{N} p(\mathbf{u}_i|\Theta, C) \tag{4.7}$$

since $\mathbf{x} = \mathbf{f}(\mathbf{u}; C)$, the probability in Equation 4.7, which is in $\mathcal{U}$ domain, can be written as the product of the probability in $\mathcal{X}$ domain and the Jacobian of the transformation J_f:

$$p(\mathbf{u}|\Theta, C) = p(\mathbf{x}|\Theta)|J_f(\mathbf{u}; C)| \tag{4.8}$$

where $|\quad|$ stands for the matrix determinant. Since each coordinate function in Equation 4.5 is a univariate function, the Jacobian matrix of the nonlinear transformation is diagonal and so, the determinant is the product of the diagonal entries:

$$|J_f(\mathbf{u}; C)| = \prod_j f_j'(u_j) = \prod_j \sum_{k=1}^{K} k c_{kj} u_j^{k-1} \tag{4.9}$$

Putting Equations 4.7, 4.6 and 4.9 together, the likelihood function can be simplified as:

$$
\begin{aligned}
L(\Theta, C) &= \log \prod_i p(\mathbf{x}_i|\Theta) + \log \prod_i |J_f(\mathbf{u}_i; C)| \\
&= \sum_i \log p(\mathbf{f}(\mathbf{u}_i; C)|\Theta) + \sum_{i,j} \log \sum_k k c_{kj} u_{ij}^{k-1}
\end{aligned}
\tag{4.10}
$$

Where u_{ij} is j^{th} element of i^{th} observation vector $\mathbf{u}_i$. Equation 4.10 provides a general formula to find the model and the nonlinear transformation together by maximizing the likelihood with respect to C and Θ simultaneously. By comparing the likelihood in Equation 4.6 and 4.10, the likelihood of C and Θ in $\mathcal{U}$ domain simplifies to a typical ML problem in $\mathcal{X}$ domain as described in Equation 4.6, if the nonlinearity transformation C is held fixed. The first term in Equation 4.10 encourages the model and the nonlinear transformation to attain higher likelihood in X domain. The second term in Equation 4.10, which is just a function of C, is a regularization term and penalizes nonlinear transformations that have low first derivatives at observed data points.

It is computationally expensive to use standard techniques such as gradient ascent for maximizing the likelihood with respect to C and Θ because the computation of gradient with respect to Θ typically involves the transformation of the observed data into $\mathcal{X}$ domain for every value of C, as shown in Algorithm 1. A common method to reduce the number of required transformations (hence, computational complexity) is to use the alternating optimization method [41]. Based on this method, we start with some initial values for the parameters. We iteratively consider one parameter set as our target and keep the other one fixed. Then, we update the target parameters by optimizing the objective with respect to the target. We continue the sequence of optimization with respect to one parameter to reach a local optima, as shown in Algorithm 2.

Algorithm 1 Gradient Ascent	**Algorithm 2** Alternating Maximization
Input: $\mathcal{U}$	**Input:** $\mathcal{U}$
Output: Θ and C	**Output:** Θ and C
Initialization Θ^0 and C^0	**Initialization** Θ^0 and C^0
Computing $\mathcal{X}$ using $\mathbf{x} = \mathbf{f}(\mathbf{u}; C^0)$	Computing $\mathcal{X}$ using $\mathbf{x} = \mathbf{f}(\mathbf{u}; C^0)$
while $i \leq k$ **do**	**while** $i \leq k$ **do**
$\Theta^i = \Theta^{i-1} + \alpha \frac{\partial L}{\partial \Theta}\|_{\Theta^{i-1}, C^{i-1}}$	$\Theta^i = \arg\max_\Theta L(C^{i-1}, \Theta)$
$C^i = C^{i-1} + \alpha \frac{\partial L}{\partial C}\|_{\Theta^{i-1}, C^{i-1}}$	$C^i = \arg\max_C L(C, \Theta^i)$
Updating $\mathcal{X}$ using $\mathbf{x} = \mathbf{f}(\mathbf{u}; C^i)$	Updating $\mathcal{X}$ using $\mathbf{x} = \mathbf{f}(\mathbf{u}; C^i)$
$i = i + 1$	$i = i + 1$
end while	**end while**

However, the optimization of the likelihood with respect to Θ itself, which only involves $p(\mathcal{X}|\Theta)$ in Equation 4.10, is somewhat challenging if the acoustic model is a GMM-HMM.

GMM-HMM is a partially observed probabilistic model to model temporal data. In general, partially observed models have two types of random variables: hidden and visible variables. The observation data, which is used to estimate the parameters of the model, only consists of the visible variables. The log-likelihood of parameters for partially observed model can be obtained by marginalizing over hidden variables as:

$$\begin{aligned} L(\Theta) &= \log \prod_i p(\mathbf{x}_i|\Theta) \\ &= \log \prod_{i=1}^{N} \sum_{\mathbf{h}} p(\mathbf{x}_i, \mathbf{h}|\Theta) \end{aligned} \tag{4.11}$$

where x and h are respectively observed and hidden variables. Unfortunately, the explicit maximization of the likelihood for partially observed data is typically infeasible due to the order of logarithm and summation in Equation 4.11.

Expectation Maximization (EM) is a common method to estimate the parameters for partially observed models. The idea is to approximate the log-likelihood function around Θ^{cur} using a lower bound as:

$$L(\Theta) \geq L(\Theta^{cur}) + Q(\Theta; \Theta^{cur}) - Q(\Theta^{cur}; \Theta^{cur}) \tag{4.12}$$

where the auxiliary function Q is the expectation of the joint log-probability of observed and hidden variables given the observation data $\mathcal{X}$ and current estimation of the parameters Θ^{cur}:

$$Q(\Theta;\Theta^{cur}) = E\left\{\log p(\mathbf{x},\mathbf{y}|\Theta)|\mathcal{X},\Theta^{cur}\right\} \tag{4.13}$$

And instead of maximizing $L(\Theta)$, we repeatedly maximize the lower bound with respect to Θ as :

$$\Theta^{new} = \arg\max_{\Theta} Q(\Theta;\Theta^{cur}) \tag{4.14}$$

The auxiliary function for GMM-HMM can be decomposed into GMM and HMM parts as described by [18] as follows :

$$Q(\Theta;\Theta^{old}) = \underbrace{\sum_{i,j,t}\xi_{ij}(t)\log a_{ij}}_{\text{hmm}} + \underbrace{\sum_{i,t}\gamma_i(t)\log b_i(x_t)}_{\text{gmm}} \tag{4.15}$$

where the HMM part only consist of the transition probability matrix $A = [a_{ij}]$ and the GMM part includes the parameters of the output probabilities $b_i(x)$ which are means, covariance matrices and component weights. In addition to the parameters, Equation 4.15 also consists of two posterior probabilities as:

$$\xi_{ij}(t) = P(s_t = i, s_{t+1} = j|\mathcal{X},\Theta^{old}), \gamma_i(t) = P(s_t = i|\mathcal{X},\Theta^{old}) \tag{4.16}$$

where $\xi_{ij}(t)$ is the probability of being in the state i at time t and j at time t+1 given the observation and the current parameters. $\gamma_i(t)$ is the probability of being in the state i given the observation and the current parameters. Based on Equation 4.15, the transition probability matrix A and the parameters of output probabilities can be updated independently for each step of the maximization. However, dealing with the GMM part is somewhat tricky, since it consists of $\mathbf{x}$ which depends on the nonlinear transformation C.

By slightly abusing the notation of the index i to represent hidden state and mixture component together, we can further simplify the GMM part of the auxiliary function as follows :

$$Q_{gmm}() = \frac{1}{2}\sum_{i,t}\left[(\mathbf{x_t}-\mu_\mathbf{i})^\mathbf{T}\mathbf{\Sigma_i^{-1}}(\mathbf{x_t}-\mu_\mathbf{i}) + |\mathbf{\Sigma_i}|\right]\gamma_i(t) \tag{4.17}$$

where μ_i and Σ_i are the mean and covariance for i^{th} gaussian component in the model. By plugging 4.17 into 4.10 and replacing $\mathbf{x}$ with $\mathbf{f}(\mathbf{u}; C)$, we can construct an auxiliary function for the joint log-likelihood $L(\Theta, C)$ as follows:

$$\begin{aligned} Q(\Theta, C; \Theta^{cur}, C^{cur}) = \frac{1}{2} \sum_{i,t} \left[(\mathbf{f}(\mathbf{u}_t; C) - \mu_\mathbf{i})^\mathbf{T} \mathbf{\Sigma_i^{-1}} (\mathbf{f}(\mathbf{u_t}; \mathbf{C}) - \mu_\mathbf{i}) + |\mathbf{\Sigma_i}| \right] \gamma_i(t) \\ + \sum_{i,j} \log \sum_k k c_{kj} u_{ij}^{k-1} \end{aligned} \tag{4.18}$$

Another way to obtain the auxiliary function for the joint log-likelihood is through the definition of the lower bound similar to Equation 4.13. Again, we can use alternating maximization method to maximize the auxiliary function. We start with some initial values for the parameters. We apply the nonlinear transformation with current value of C to map observed samples into $\mathcal{X}$ domain. We update the parameters of the model Θ using multiple iterations of Viterbi algorithm (or Baum-Welch algorithm). This is equivalent to the maximization of the auxiliary function with respect to Θ by holding C fixed. We use Viterbi algorithm (or Baum-Welch algorithm) to estimate the posterior probability $\gamma_i(t)$ using the current value of Θ. Then, we hold the parameter of the model fixed and maximize the auxiliary function with respect to C, which is a weighted curve fitting with a regularization term. We repeat the update of C and Θ until convergence. The algorithm is summarized in Algorithm 3.

Algorithm 3 Alternating Maximization Algorithm for maximizing $Q(\Theta, C)$

Input: $\mathcal{U}$ {obtained by applying utterance-specific transformation.}
Output: Acoustic model Θ and training-specific transformation C

1: **Initialization** C^0
2: **while** $i \leq K$ **do**
3: Obtaining $\mathcal{X}$ by transforming the observation using $\mathbf{x} = \mathbf{f}(\mathbf{u}; C^0)$
4: M iterations of Viterbi (or Baum-Welch) training on $\mathcal{X}$ to compute Θ^{i-1}
5: Viterbi alignment on $\mathcal{X}$ to compute $\gamma_i(t)$
6: Computing C^i by maximizing $Q(C, \Theta^{i-1})$
7: $i = i + 1$
8: **end while**

4.6.4 Experimental Results

As mentioned in Section 4.5.1, the GMM-HMM baseline for CHIME 4 involves training several models including monophone, triphone with delta feature (tri1), triphone with MLLR+LDA feature transformation (tri2) and triphone with FMLLR feature (tri3). Each of these subsystems is a separate GMM-HMM with its own input and configuration.

In this section, we compare the performance of the final systems (tri3) when the copula-based normalization is combined with acoustic model training at different stage of the baseline (monophone and triphone systems). The training of the GMM-HMM models, regardless of their type, consists of 40 viterbi iterations for updating the parameter of the model, which is equal to $K \times M$ in Algorithm 3. For each of 5 iterations, we optimize once again the auxiliary function with respect to the parameters of the transformation and update the $\mathcal{X}$. We utilize Limited-memory BFGS method [61] for the optimization, and assume each marginal is a polynomial with degree 7. As shown in Table 4.14, the integration of the copula based normalization into the monophone model performs better than others. Apart from the improvement, the integration with the monophone is more computationally efficient mainly because the total number of parameters for the joint optimization is less. In addition, these results also reveal that the integration with more complicated models such as tri2 and tri3, in which the input feature is not MFCC, degrades the performance. Our experiments also show that the integration of copula-based normalization with more than one stage leads to instability in the training process.

Table 4.14: WER of tri3 on 1-ch track eval set when copula-based normalization is integrated into: monophone (mono), triphone with delta feature (tri1), triphone with MLLR+LDA feature transformation (tri2) and triphone with FMLLR feature (tri3)

Integration style	WER
Independent of backend ASR	29.7
mono	**28.3**
tri1	28.8
tri2	30.3
tri3	31.4

Table 4.15: WERs of smbr+rnn system on different tracks with integrated copula normalization and their relative improvements.

		WER%	Rel %
1-ch	augmented + copula	16.5	
	augmented + copula + integration	15.7	5.1
2-ch	augmented + copula	14.3	
	augmented + copula + integration	14.0	2.1
6-ch	augmented + copula	9.9	
	augmented + copula + integration	8.2	17.2

4.7 Results

For the next experiment, we integrate the copula-based transformation into monophone system and report WERs of the final smbr+rnn system on different tracks of CHIME 4. We use augmented data for this experiment because of its better performance on the independent variant of copula-based transformation. As shown in table 4.15, the integration method improves the overall performance for all track. However, the relative improvement for 6-ch track is more significant that others.

4.8 Summary

In this study, we use copula model to address the mismatch problem between the training and testing conditions in ASR systems. Simply, we formulate the mismatch in term of the difference between distributions of training and test data, and proposes a transformation to make test data similar to training data. We show that if the distributions are modeled using Gaussian copula model, then there is an analytic form for the transformation.

We explored three different strategies to use the proposed transformation in ASR system. For the first strategy, we trained the backend ASR using original features and just applied the transformation during the decoding phase. This approach provides a quick way to improve the noise-robustness of already trained ASRs without retraining them. Our results indicated that this strategy provides a moderate improvement for ASR models trained by clean training data and degrades the performance of ASR models trained by multi-condition training data. The second strategy is to transform each utterance

in training and test dataset such that the distribution of transformed utterance become similar to the distribution of the entire training set. We then use the transformed features for training and decoding. This method consistently improve the performance of different models over the original features. In contrast to the above two strategies, our third strategy is not independent of the backend ASR. We formulate the transformation method as a parametric model and plug it into the GMM-HMM acoustic model. We propose a computationally efficient optimization method to jointly find the parameter of the transformation and the acoustic model. Our results showed that the integration into the acoustic model improves the performance. However, this method is more computationally expensive than other two methods.

Chapter 5

Conclusion and future work

5.1 Conclusion

In Chapter 1, we presented two mutivariare density estimation methods based on the copula model : mixture of Gaussian copula model with Toeplitz correlation structure and grafted Gaussian mixture copula.

From a density estimation perspective, these methods consistently outperform GMMs with equivalent number of parameters. Our experiments shows that the standard deviation of density estimation for grafted Gaussian mixture copula is slightly higher than GMM and mixture of Gaussian copula model. From a computational complexity perspective, both of these methods only involves the estimation of a set of marginal distributions and a mixture model, which makes them a more powerful alternative for GMMs. Note that the grafted Gaussian mixture copula provides a simple way to adjust the marginal distributions of already trained GMMs without retraining.

In addition, these proposed methods have been used to estimate the class-conditional densities in generative classifications. The resulting class-conditional multivariate distributions form better classifiers than their corresponding conditional GMM counterparts with same number of parameters. Our proposed models perform consistently better than GMM classifier on different classification tasks. The classification performance of both model are also comparable to SVM in many cases, even though it is a generative model.

In Chapter 4, we proposed a computationally cheap distribution-based normalization method using copula model to address the mismatch between the training and test conditions in ASR systems. We demonstrated that the proposed method can improve the performance of ASR systems under noisy conditions, and in the presence of distortion. Our results reveal that the proposed method consistently benefits different ASR systems with a wide range of configurations such as monophone, triphone, and DNN-based models trained with MFCC and filter bank features. We also explored three different strategies to use the proposed method in ASR system. For the first strategy, we trained the backend ASR using original features and just applied the normalization during the decoding phase. This approach provides a quick way to improve the noise-robustness of already trained

ASR systems without retraining them. Our results indicated that that this strategy provides a moderate improvement for ASR models trained by clean training and degrades the performance of models trained with multi-condition training style. The second strategy is to normalize each utterance in train and test datasets to have a distribution similar to the distribution of the entire train set. And then we use the normalized features for training and decoding. This consistently improves the performance of different models over the original features. In contrast to the above two strategies, our third strategy is not independent of the backend ASR. We formulate the normalization method as a parametric model and plug it into the GMM-HMM acoustic model. We proposed a computationally efficient optimization method to jointly find the parameters of the transformation and the acoustic model. Our results showed that the integration into the acoustic model improves the performance. However, this method is more computationally expensive than other strategies.

5.2 Future work

In this thesis, we have introduced new methods based on copula model to accurately estimate multivariate density estimation. We have investigated the application of our proposed methods in different classification and ASR tasks. In general, our experiments revealed that copula model based density estimation is an effective and powerful alternative for standard methods, such as Gaussian model. Here, we list some directions for the future works.

In Chapter 3, we have shown that the performance of generative classifier significantly improves for different tasks by estimating class conditional densities using our copula-based density estimators, instead of a GMM . There are many other tasks in machine learning, which heavily depends on gaussian distribution, that can benefit from this replacement. For example, we use a decision tree-based clustering method to build phonetic decision tree for ASR systems. This method is top-down node splitting algorithm to construct the tree where each node is associated with some data points. For each node, we use a set of predefined questions to divide data points into some partitions and choose the

best split based on the log-likelihood improvement as goodness of split criterion. The log-likelihood improvement is the sum of log-likelihood of the partitions where each portion is modeled by a single Gaussian model. In practice, the distribution of each partition is far from Gaussian distributions. Similarly, we can use Gaussian copula model to compute the log-likelihood improvement.

In Chapter 4, we proposed a copula-based transformation to address the mismatch between the training and testing conditions. The original transformation was independent of the ASR backend. We have shown that a further improvement can be obtained by integrating this transformation with GMM-HMM acoustic model. This transformation can be combined with more sophisticated acoustic models, such as DNN or long short-term memory neural network. Simply, we can embed this transformation as the first layers of these models and use backpropagation method to find the transformation besides the acoustic model.

Bibliography

[1] A. Acero. *Acoustical and environmental robustness in automatic speech recognition*, volume 201. Springer Science & Business Media, 2012.

[2] Aho, Ken and Derryberry, DeWayne and Peterson, Teri. Model selection for ecologists: the worldviews of AIC and BIC. *Ecology*, 95(3):631–636, 2014.

[3] J. Altmann. Observational study of behavior: Sampling methods. *Behaviour*, 49(3/4):227–267, 1974.

[4] X. Anguera, C. Wooters, and J. Hernando. Acoustic beamforming for speaker diarization of meetings. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(7):2011–2022, 2007.

[5] Anguera Miró, Xavier. Robust speaker diarization for meetings. 2006.

[6] M. Asgari, A. Bayestehtashk, and I. Shafran. Robust and accurate features for detecting and diagnosing autism spectrum disorders. In *Proc. Interspeech*, 2013.

[7] M. Asgari, I. Shafran, and A. Bayestehtashk. Inferring social contexts from audio recordings using deep neural networks. In *Proc. IEEE Machine Learning for Signal Processing (MLSP)*, pages 1–6, 2014.

[8] A. Babaeean, A. Tashk, M. Bandarabadi, and S. Rastegar. Target tracking using wavelet features and RVM classifier. In *Proc. International Conference on Natural Computation*, volume 4, pages 569–572, Oct 2008.

[9] A. Babaeian, S. Rastegar, M. Bandarabadi, and M. Erza. Modify kernel tracking using an efficient color model and active contour. In *Proc. Southeastern Symposium onSystem Theory*, pages 59–63, March 2009.

[10] A. Babaeian, S. Rastegar, M. Bandarabadi, and M. Rezaei. Mean shift-based object tracking with multiple features. In *Proc. Southeastern Symposium on System Theory*, pages 68–72, March 2009.

[11] K. Bache and M. Lichman. UCI machine learning repository, 2013.

[12] A. Bayestehtashk, M. Asgari, I. Shafran, and J. McNames. Fully automated assessment of the severity of Parkinson's disease from speech. *Computer speech & language*, 29(1):172–185, 2015.

[13] A. Bayestehtashk and I. Shafran. Parsimonious multivariate copula model for density estimation. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5750–5754. IEEE, 2013.

[14] A. Bayestehtashk and I. Shafran. Parsimonious multivariate copula model for density estimation. In *Proc. IEEE ICASSP*, pages 5750–5754, May 2013.

[15] A. Bayestehtashk and I. Shafran. Efficient and accurate multivariate class conditional densities using copula. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 3936–3940, April 2015.

[16] A. Bayestehtashk and I. Shafran. Robust Automatic Speech Recognition for the 4th CHiME Challenge Using Copula-based Feature Enhancement. *The 4th CHiME Speech Separation and Recognition Challenge*, 2016.

[17] A. Bayestehtashk, I. Shafran, and A. Babaeian. Robust speech recognition using multivariate copula models. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5890–5894. IEEE, 2016.

[18] J. Bilmes. A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. Technical report, 1998.

[19] Z. Botev. Nonparametric density estimation via diffusion mixing. 2007.

[20] Z. I. Botev, J. F. Grotowski, D. P. Kroese, et al. Kernel density estimation via diffusion. *The Annals of Statistics*, 38(5):2916–2957, 2010.

[21] J. P. Burg, D. G. Luenberger, and D. L. Wenger. Estimation of structured covariance matrices. *Proceedings of the IEEE*, 70(9):963–974, 1982.

[22] T. Cai, C.-H. Zhang, and H. H. Zhou. Optimal rates of convergence for covariance matrix estimation. *The Annals of Statistics*, 38:2118–2144, 2010.

[23] C. C. Caprani. Probalistic analysis of highway bridge traffic loading. 2005.

[24] L. Chen, V. P. Singh, S. Guo, A. K. Mishra, and J. Guo. Drought analysis using copulas. *Journal of Hydrologic Engineering*, 18(7):797–808, 2012.

[25] S. S. Chen and R. A. Gopinath. Gaussianization. 2000.

[26] S. S. Chen, E. Setauket, and R. A. Gopinath. Gaussianization. *Proc. Neural Information Processing Systems*, 2001.

[27] I. Cohen and B. Berdugo. Speech enhancement for non-stationary noise environments. *Signal Processing*, 81(11):2403 – 2418, 2001.

[28] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4):547 – 553, 2009.

[29] A. De La Torre, A. M. Peinado, J. C. Segura, J. L. Pérez-Córdoba, M. C. Benítez, and A. J. Rubio. Histogram equalization of speech representation for robust speech recognition. *IEEE Transactions on Speech and Audio Processing*, 13(3):355–366, 2005.

[30] S. Dharanipragada and M. Padmanabhan. A nonlinear unsupervised adaptation technique for speech recognition. In *INTERSPEECH*, pages 556–559, 2000.

[31] G. Elidan. Copula Bayesian Networks. *Neural Information Processing Systems*, 2010.

[32] G. Elidan. Copula network classifiers. *The International Conference on Artificial Intelligence and Statistics*, 2012.

[33] Y. Ephraim and D. Malah. Speech enhancement using a minimum mean-square error log-spectral amplitude estimator. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 33(2):443–445, Apr 1985.

[34] F. Eyben, M. Wöllmer, and B. Schuller. Opensmile: The Munich versatile and fast open-source audio feature extractor. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1459–1462. ACM, 2010.

[35] D. A. Freedman. *Statistical models: theory and practice*. Cambridge University press, 2009.

[36] M. J. F. Gales. Model-based approaches to handling uncertainty. *Robust Speech Recognition of Uncertain or Missing Data*, 2011.

[37] L. Garcia, J. C. Segura, J. Ramurez, A. de la Torre, and C. Benitez. Parametric non-linear feature equalization for robust speech recognition. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 1, pages I–I. IEEE, 2006.

[38] J. Garofalo, D. Graff, D. Paul, and D. Pallett. CSR-i (WSJ0) complete. *Linguistic Data Consortium, Philadelphia*, 2007.

[39] C. Genest and A.-C. Favre. Everything you always wanted to know about copula modeling but were afraid to ask. *Journal of Hydrologic Engineering*, pages 347–368, 2007.

[40] C. Genest and L.-P. Rivest. Statistical inference procedures for bivariate archimedean copulas. *Journal of the American statistical Association*, 88(423):1034–1043, 1993.

[41] T. Goldstein, B. O'Donoghue, S. Setzer, and R. Baraniuk. Fast alternating direction optimization methods. *SIAM Journal on Imaging Sciences*, 7(3):1588–1623, 2014.

[42] F. Hilger and H. Ney. Quantile based histogram equalization for noise robust large vocabulary speech recognition. *Audio, Speech, and Language Processing, IEEE Transactions on*, 14(3):845–854, 2006.

[43] F. Hilger, H. Ney, et al. Quantile based histogram equalization for noise robust speech recognition. In *INTERSPEECH*, pages 1135–1138, 2001.

[44] F. E. Hilger. *Quantile based histogram equalization for noise robust speech recognition.* PhD thesis, Bibliothek der RWTH Aachen, 2004.

[45] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, Nov 2012.

[46] M. Hofert, M. Mächler, and A. J. Mcneil. Likelihood inference for archimedean copulas in high dimensions under known margins. *Journal of Multivariate Analysis*, 110:133–150, 2012.

[47] N. L. Johnson, S. Kotz, and N. Balakrishnan. *Continuous Univariate Distributions*, volume 1. Wiley, 2nd edition, 1994.

[48] N. L. Johnson, S. Kotz, and N. Balakrishnan. *Continuous Univariate Distributions*, volume 2. Wiley, 2nd edition, 1995.

[49] M. G. Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

[50] S. Kirshner. Learning with tree-averaged densities and distributions. *Neural Information Processing Systems*, 2007.

[51] S. Kirshner. Learning with tree-averaged densities and distributions. In *Proc. NIPS*, 2007.

[52] S. Kirshner. Learning with tree-averaged densities and distributions. *Proc. Neural Information Processing Systems*, 2007.

[53] S. Kirshner. Latent tree copulas. *Proc. Workshop on Probabilistic Graphical Models*, 2012.

[54] M. S. E. Langarani and J. P. van Santen. Modeling fundamental frequency dynamics in hypokinetic dysarthria. In *Spoken Language Technology (SLT), 2014 IEEE International Workshop on*. IEEE, 2014.

[55] O. Ledoit and M. Wolf. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, pages 365 – 411, 2004.

[56] Ledoit, Olivier and Wolf, Michael. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of multivariate analysis*, 88(2):365–411, 2004.

[57] B. Li and K. C. Sim. A spectral masking approach to noise-robust speech recognition using deep neural networks. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, 22(8):1296–1305, Aug 2014.

[58] D. X. Li. On default correlation: A copula function approach. 1999.

[59] S.-H. Lin, Y.-M. Yeh, and B. Chen. A comparative study of histogram equalization (HEQ) for robust speech recognition. *Computational Linguistics and Chinese Language Processing*, 12(2):217–238, 2007.

[60] R. Lippmann, E. Martin, and D. Paul. Multi-style training for robust isolated-word speech recognition. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'87.*, volume 12, pages 705–708. IEEE, 1987.

[61] D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

[62] H. Liu, J. Lafferty, and L. Wasserman. The nonparanormal: Semiparametric estimation of high dimensional undirected graphs. *The Journal of Machine Learning Research*, 10:2295–2328, 2009.

[63] Liu, Linxi and Wong, Wing Hung. Multivariate density estimation via adaptive partitioning (ii): posterior concentration. *arXiv preprint arXiv:1508.04812*, 2015.

[64] D. MacKenzie and T. Spears. 'The Formula That Killed Wall Street'? The Gaussian Copula and the Material Cultures of Modelling. *Working Paper*, 2012.

[65] D. Maestripieri and K. Wallen. Affiliative and submissive communication in rhesus macaques. *Primates*, 38(2):127–138, 1997.

[66] R. Mallik. On multivariate Rayleigh and exponential distributions. *IEEE Transactions on Information Theory*, 49(6):1499 – 1515, June 2003.

[67] D. Mazzoni and R. Dannenberg. Audacity [software]. Pittsburgh, 2000.

[68] S. H. Mohammadi, A. Kain, and J. P. van Santen. Making conversational vowels more clear. In *Interspeech*, 2012.

[69] S. Molau. *Normalization in the acoustic feature space for improved speech recognition*. PhD thesis, Bibliothek der RWTH Aachen, 2003.

[70] P. J. Moreno. *Speech recognition in noisy environments*. PhD thesis, Carnegie Mellon University Pittsburgh, 1996.

[71] A. Narayanan and D. Wang. Joint noise adaptive training for robust automatic speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 2504–2508, May 2014.

[72] R. B. Nelsen. *An introduction to copulas*. Springer Science & Business Media, 2007.

[73] N. Parihar and J. Picone. Aurora working group: DSR front end LVCSR evaluation AU/384/02. *Inst. for Signal and Information Process, Mississippi State University, Tech. Rep*, 40:94, 2002.

[74] N. Parihar, J. Picone, D. Pearce, and H.-G. Hirsch. Performance analysis of the Aurora large vocabulary baseline system. In *Signal Processing Conference, 2004 12th European*, pages 553–556. IEEE, 2004.

[75] E. Parzen. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076, 1962.

[76] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[77] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, et al. The Kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, number EPFL-CONF-192584. IEEE Signal Processing Society, 2011.

[78] Printz, Harry and Olsen, Peder A. Theory and practice of acoustic confusability. *Computer Speech & Language*, 16(1):131–164, 2002.

[79] B. Sakar, M. Isenkul, C. Sakar, A. Sertbas, F. Gurgen, S. Delil, H. Apaydin, and O. Kursun. Collection and analysis of a parkinson speech dataset with multiple types of sound recordings. *IEEE Journal of Biomedical and Health Informatics*, 17(4):828–834, July 2013.

[80] N. L. J. Samuel Kotz, N. Balakrishnan. *Continuous Multivariate Distributions: Models and Applications*, volume 1. Wiley, 2nd edition, 2000.

[81] G. Saon, S. Dharanipragada, and D. Povey. Feature space gaussianization. In *Proc. IEEE ICASSP*, volume 1, pages 329–32, 2004.

[82] D. W. Scott. *Multivariate density estimation: theory, practice, and visualization.* John Wiley & Sons, 2015.

[83] M. Seltzer, D. Yu, and Y. Wang. An investigation of deep neural networks for noise robust speech recognition. In *ICASSP 2013*. IEEE International Conference on Acoustics, Speech, and Signal Processing(ICASSP), 2013.

[84] B. W. Silverman. *Density estimation for statistics and data analysis*, volume 26. CRC press, 1986.

[85] A. Sklar. Fonctions de repartition a n dimensions et leurs marges. *Publ. Inst. Stat. Univ. Paris 8*, pages 229—231, 1959.

[86] A. Sklar. Random variables, distribution functions, and copulas: a personal look backward and forward. *Lecture notes-monograph series*, pages 1–14, 1996.

[87] A. Stark, A. Bayestehtashk, M. Asgari, and I. Shafran. Interspeech pathology challenge: Investigations into speaker and sentence specific effects. In *Proc. Annual Conference of the International Speech Communication Association*, 2012.

[88] A. Tashk, A. Sayadiyan, P. Mahale, and M. Nazari. Pattern classiffication using svm with gmm data selection training methods. In *Proc. IEEE Signal Processing and Communications*, pages 1023–1026, Nov 2007.

[89] A. Tewari, M. J. Giering, and A. Raghunathan. Parametric characterization of multimodal distributions with non-gaussian modes. *Proc. IEEE International Conference on Data Mining Workshops*, pages 286–292, 2011.

[90] P. Trivedi and D. Zimmer. Copula modeling: An introduction for practitioners. 1:1–111, 2005.

[91] E. Vincent, S. Watanabe, A. Nugraha, J. Barker, and R. Marxer. An analysis of environment, microphone and data simulation mismatches in robust speech recognition. *Computer Speech and Language*, 2016.

[92] Vincent, Emmanuel and Watanabe, S and Barker, Jon and Marxer, Ricard. The 4th CHiME speech separation and recognition challenge.

[93] C. Weng, D. Yu, S. Watanabe, and B.-H. Juang. Recurrent deep neural networks for robust speech recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing(ICASSP)*, pages 5532–5536, May 2014.

[94] H. Xu, D. Povey, L. Mangu, and J. Zhu. Minimum bayes risk decoding and system combination based on a recursion for edit distance. *Computer Speech & Language*, 25(4):802–828, 2011.

[95] S.-K. A. Yeung and M.-H. Siu. Improved performance of Aurora-4 using HTK and unsupervised MLLR adaptation. In *Proceedings of the Int. Conference on Spoken Language Processing, Jeju, Korea*, 2004.

[96] I. Zezula. On multivariate gaussian copulas. *Journal of Statistical Planning and Inference*, 139(11):3942 – 3946, 2009.