Republic of Tunisia
Ministry of Higher Education and Scientific Research
Manouba University
National School of Computer Sciences

THESIS

Submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Computer Sciences

by

Sarra FALLEH

Analysis of hyperspectral images by content using a Geometric Deep Learning approach

Conducted within the CRISTAL laboratory, GRIFT research group

Defended on jj/mm/aaaa, in front of the committee composed of:

President:	**Pr.**	
Reviewer:	**Pr.**	Slim MHIRI
Reviewer:	**Pr.**	Olfa MARRAKCHI
Thesis Director:	**Pr.**	Faouzi GHORBEL
Thesis Co-Director:	**Dr.**	Molka TROUDI

République Tunisienne
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université de Manouba
École Nationale des Sciences de l'Informatique

THÈSE

Présentée en vue de l'obtention du diplôme de

Doctorat en Sciences Informatiques

par

Sarra FALLEh

Analyse d'images hyperspectrales par le contenu par une approche Deep Learning Géometrique

Réalisée au sein du laboratoire CRISTAL, pôle GRIFT

Soutenue le jj/mm/aaaa, devant le jury composé de :

Président :	**Pr.**	
Rapporteur :	**Pr.**	Slim MHIRI
Rapporteur :	**Pr.**	Olfa MARRAKCHI
Directeur de Thèse :	**Pr.**	Faouzi GHORBEL
Codirecteur de Thèse :	**Dr.**	Molka TROUDI

DEDICATIONS

To my beloved family, for their endless love, support, and patience.

To my father, who may no longer be with us in this world, but whose guidance and memory light my path every day. I carry your wisdom and love with me in every step I take.

To my mother, whose unwavering prayers and faith brought me to where I stand today. Your strength and love have been my constant source of courage.

To my big sister, who encourages me in her own unique ways, always by my side in spirit.

To my soul-mate sister, without whom I wouldn't be here. You have never stopped pushing me and believing in me.

To my brother, and to my sister- and brother-in-law, for their support and presence in this journey.

To my two nieces and two nephews, who bring so much joy, light, problem, worries and laughter into my life. You are the ones who give me the reason to dream bigger and love deeper.

To my friends, who have been there to encourage, uplift, and inspire me in countless ways.

To **All of you**,

I dedicate this work.

Falleh SARRA

ACKNOWLEDGEMENT

First, I express my deepest gratitude to my Ph.D. supervisor, Professor Faouzi GHORBEL, for his invaluable guidance, patience, and continuous support throughout the course of my research. His expertise and dedication have been crucial in shaping the direction and outcome of this thesis.

I am equally grateful to my thesis Co-Director, Dr. Molka TROUDI, for her consistent support, understanding, mentorship and insightful feedback, which have greatly enriched me and my research. Her thoughtful advice has been a guiding light through the challenges of this journey.

I also extend my sincere thanks to the jury members, Professor Olfa MARRAKCHI, Professor Slim MHIRI for their time, insightful comments, and valuable feedback on my work. Their contributions have helped elevate the quality of this thesis.

In addition, I would like to express my gratitude to the administrative team of ENSI. Their support, efficiency, and dedication have greatly contributed to creating a smooth and productive environment for my research.

Finally, to my family and friends, who have provided constant encouragement and support throughout this long journey, I owe more than words can express. Your belief in me has been my greatest source of strength.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

List of Abbreviations and Acronyms

AMSCR Affine Multi-Scale Curve Registration

AMSCR-EM Affine Multi-Scale Curve Registration Using EM Algorithm

CDTC Cumulative Distribution Target Criterion

EM The Expectation Maximization algorithm

EMD The Diffeomorphism Expectation Maximization Algorithm

FKDE Fast Kernel Density Estimate

GAT Graph Attention Networks

GCN Graph Convolutional Networks

GDL Geometric deep learning

GMM Gaussian Mixture Models

HSI Hyperspectral imaging

IoU Intersection over Union

KDE kernel density estimators

DKDE Diffeomorphic kernel Density Estimators

LSTM Long Short-Term Memory

MAE Mean Absolute Error

MAPE Mean Absolute Percentage Error

MASE Mean Absolute Scaled Error

MCEM Monte Carlo EM

MISE Mean Integrated Square Error

MLE Maximum Likelihood Estimation

MSE Mean Squared Error

MSLE Mean Squared Logarithmic Error

PCA Principal Component Analysis

PDF Probability Density Function

RMSE Root Mean Squared Error

GENERAL INTRODUCTION

With the increasing importance of data analysis across a wide range of fields, this thesis is dedicated to exploring and evaluating advanced classification and regression algorithms. In this thesis, we propose a set of original Expectation Maximization algorithms adapted to nonlinear data, which are defined by the fact that they do not belong to a vector space. The spaces of such data are represented by topological and differential manifolds. As an example, for dimensions greater than or equal to 2, we can mention the quarter plane, spheres, etc. In this research, the one-dimensional case is thoroughly studied. One-dimensional nonlinear data belong to semi-bounded domains (the half-real line) or bounded domains such as compact intervals. This new family of algorithms, which we propose in this thesis, can be identified as belonging to the new class called "geometric deep learning".

In this thesis, we address the challenges associated with the EM algorithms in unsupervised classification. A common issue arises when dealing with data on a bounded or semi-bounded support, especially with an accumulation of data near the finite endpoint. The boundary issues, a well-known problem in non-parametric density estimation [Malec and Schienle, 2014], can deeply impact the estimation quality near the finite endpoints of the support and can produce an overflow above the end of the support. To address this issue, several boundary correction methods have been proposed. The reflection method [Choi et al., 2022] utilizes the concept of reflecting observations beyond the boundaries to create a larger sample space, thus reducing bias at the edges. Boundary kernels [Marshall and Hazelton, 2010] specially designed kernel functions that diminish their influence as they approach the boundaries, ensuring smoother and more accurate density estimates near the support limits. Transformation methods [Marron and Ruppert, 1994] involve altering the original data to map the boundary points to a more central region, thus mitigating the distortion caused by the boundaries and improving the overall estimation quality, etc.

The EM algorithm can also be affected by boundary issues. The estimated parameters may not be appropriate for the data near the boundaries, especially in cases where there is

an agglomeration near the endpoint of the support. To address the boundary issues in the EM algorithm, we propose a novel approach that applies a diffemorphic transformation on the data that converts the bounded or semi-bounded support to an unbounded support. This approach involves performing an unsupervised estimation on the infinite support and then applying a reverse function to return to the original support. This method aims to improve the robustness and accuracy of the EM algorithm near the finite endpoints of the support.

Another part of the thesis focuses on the evaluation of regression algorithms, specifically deep learning models such as LSTM. We aim to evaluate the performance of LSTM models under various conditions, including technical analysis and dimensionality reduction.

Traditional evaluation metrics, while useful, often lack the precision required to detect nuanced performance differences, leading to potentially inaccurate estimation of an algorithm's true capabilities or limitations. So, in this thesis, we will base our algorithm's evaluation on their error distribution. As the error can be viewed as a realization of a continuous random variable, estimating its distribution can provide us with insights into the accuracy and stability of the prediction model.

To be able to estimate these PDFs, we first need to address the challenge of density estimation in regression. As we have no knowledge about the errors' distribution shape (multiple local minima, one local minima, etc.) or components (potential mixture distributions), we will utilize a non-parametric methods for probability density estimation. The Kernel Density Estimator (KDE) will be employed with the bandwidth optimized using the fast plug-in algorithm (the Fast Kernel Density Estimator (FKDE) [Troudi et al., 2008]). This algorithm identifies the distribution without assuming a specific parametric form.

A probabilistic criterion is also used to quantitatively assess the algorithm's performance. The Cumulative Target Distribution Criterion (CDTC) was introduced in [Ben Slimen et al., 2022]. The CDTC is specifically designed to address the limitations of existing metrics by providing a more robust evaluation of algorithms. It offers a new perspective on model evaluation based on the error cumulative distribution function.

The thesis dissertation is organized into four main chapters. The first chapter 1 begins with a detailed exploration of the theoretical foundations of the statistical methods used in the following chapters. First, we start with an overview of the non-parametric estimation methods, and we focus on the Kernel density estimator and the bandwidth optimizers. The second part of the first chapter will focus on the presentation of the EM algorithm and its popular variants. The last section of the first chapter provides an overview of the most commonly used traditional evaluation metrics in regression, offering a detailed but not exhaustive review.

The first chapter 2 will be dedicated to the introduction of the new variant of the EM algorithm. This new variant was specially developed to deal with the boundary issues with the EM algorithm. Here, we propose to apply a diffeomorphic transformation on the data to transform it from a bounded support to an infinite support, identify the mixture components on the infinite support and then re-transform the obtained parameters to the original space with the reverse diffeomorphic transformation. The EMD algorithm will be tested on simulated data and will be applied to real-data applications such as the ultrasound image segmentation. The EM effectiveness in identifying mixture's components is evaluated in the context of Multi-scale Contour Registration.

In the chapter 3, we concentrate on the evaluation of deep learning algorithms precisely LSTM. We will compare and evaluate the LSTM performance in the financial forecasting field. We also want to assess the impact of the dimensionality reduction with PCA on the accuracy and stability of the LSTM algorithm. To do so, we will use the probabilistic criterion CDTC to provide deeper insight into model performance compared to traditional metrics.

In the chapter 4, we focus on applying geometric deep learning (GDL) techniques to hyperspectral image segmentation. Given the complexity and high-dimensional nature of hyperspectral data, traditional algorithms face challenges in handling these images. GDL, which can manage non-Euclidean structures like graphs [Bronstein et al., 2017], is well-suited for this task. We compare three GDL-based segmentation modelsGCN, GAT, and a hybrid GCN-GATwith and without Principal Component Analysis (PCA), to evaluate their effectiveness in hyperspectral image analysis.

Fundamentals of Statistical Estimation and Model Evaluation

Contents

Chapter 1 Abstract

This chapter provides an overview of statistical estimation methods and model evaluation techniques, with a focus on both non-parametric estimators and regression metrics. It begins with an introduction to non-parametric estimators, including Kernel Density Estimation (KDE), and explores bandwidth optimization methods such as cross-validation, rule of thumb (ROT), and the plug-in method, as well as various KDE variants.

The chapter then delves into the Expectation-Maximization (EM) algorithm, discussing its foundations in Maximum Likelihood Estimation (MLE) and its various extensions, such as Stochastic EM, Monte Carlo EM, and Variational EM, among others. Key algorithmic steps including initialization, expectation, maximization, and convergence criteria are examined.

Lastly, a detailed analysis of regression evaluation metrics is provided. Different types of metrics, including error-based, percentage-based, and variance-based metrics, are reviewed, as well as additional measures such as bias metrics, relative measures, and averaging single-iteration metrics. This chapter lays the groundwork for evaluating algorithms with precision and accuracy, addressing both the strengths and limitations of each metric.

1.1 Introduction

This first chapter introduces the scientific basis of the research developed in the following chapters of this thesis. The first section 1.2, will provide a review of probability density function (PDF) estimation, a fundamental task in statistical data analysis. The PDF estimation is categorized into parametric and non-parametric approaches.

The Parametric PDF estimation supposes that the data follows a known probability distribution, characterized by a set of parameters. It is based on selecting a probability distribution (e.g., Gaussian, Poisson, exponential) and estimating its parameters using the observed data. This approach simplifies the estimation process by reducing it to determining a limited number of parameters.

In the literature, many methods were developed for parametric PDF estimation. For example, maximum likelihood estimation (MLE) is one of the most widely used. It calculates the best-fitting parameters for a given distribution to match the observed data. In other words, they adjust the parameters of the chosen distribution to maximize the likelihood of seeing the data we have. ([Myung, 2003] [Francos et al., 1995] [Van der Linden, 2016]). Another common method is the method of moments. This method matches the sample moments (e.g., mean, variance) with the theoretical moments of the assumed distribution [Sumair et al., 2022] [Barboza and Viens, 2017]. The Least Squares method minimizes the sum of squared differences between the observed data and the predicted values from the model [Ding, 2023] [Galrinho et al., 2014]. The Bayesian Estimation is another popular method that combines prior knowledge about the parameters with the observed data to obtain a posterior distribution [Alpert and Yuan, 2008] [Allison and Dunkley, 2014].

The major limitation of parametric density estimation is the assumption of the correct distribution. If the assumed distribution is incorrect, the estimates can be biased. Therefore, in cases where no prior assumptions about the data distribution can be made, non-parametric methods become more appropriate. Unlike parametric approaches, non-parametric PDF estimation does not assume that the data follows any specific family of distributions. Instead,

these methods focus directly on estimating the density function from the data itself. This makes non-parametric methods highly flexible and powerful when dealing with mixtures, complex, or unknowing distributions.

In the context of classification, where the data consists of a mixture of populations without any prior assumptions regarding the PDF of the underlying components, the parametric methods are not well suited for such cases. Therefore, we will utilize the non-parametric multi-modal methods. Given that non-parametric methods require bandwidth optimization, the first section 1.2 provides an overview of these methods, with particular emphasis on KDE.

In a second section 1.3, we focused on the Expectation-Maximization (EM) algorithm, a powerful tool for finding maximum likelihood estimates of parameters in statistical models, particularly in problems with incomplete data or latent variables [McLachlan and Krishnan, 2007] [Sammaknejad et al., 2019]. It is an iterative algorithm commonly used to estimate the parameters of a Gaussian Mixture Model (GMM) by alternating between estimating the missing data (Expectation step) and optimizing the likelihood (Maximization step).

In the third section 1.4, we will review various evaluation metrics employed in the literature. Given the large number of metrics available, this thesis will specifically concentrate on a limited number of regression evaluation metrics.

1.2 Non-Parametric Estimators

Non-parametric estimation is essential in machine learning, especially when the data distribution is unknown, either because it does not correspond to a known distribution or because the samples follow a mixture of different distributions (case of classification problems). Unlike parametric methods, which are based on the assumption that the data follow a specific distribution, non-parametric methods make no assumptions about the statistical distribution of the data.

1.2.1 Overview of Non-Parametric Estimators

Non-parametric estimators do not assume a predefined form for the underlying probability distribution. Instead, they use the data itself to estimate the distribution directly.

This approach is particularly useful when the data distribution is unknown or when the data consists of a mixture of different distributions with no prior information available about the component distributions.

Numerous non-parametric methods for PDF estimation are documented in the literature. This section will highlight the principal, most popular ones.

- **Histograms** are among the most straightforward non-parametric methods. They are One of the simplest, earliest non-parametric estimation approaches. They operate by partitioning the data into discrete intervals, or bins, and estimating density based on the frequency of observations within each bin. This simplicity makes histograms easy to implement and interpret. However, the histogram approach often suffers from discontinuities and lacks smoothness [Silverman, 1986]. Their effectiveness can be limited by the choice of bin width and boundaries, which may affect the smoothness and accuracy of the resulting density estimate [Coq et al., 2009]. The histogram density estimator $\hat{f}(x)$ can be expressed as:

$$\hat{f}(x) = \frac{1}{N \cdot h_N} \sum_{i=1}^{N} I_{[a_i, a_i + h_N]}(x),$$

  where N is the number of observations, h_N is the bin width, $I_{[a_i, a_i+h_N]}(x)$ is an indicator function that equals 1 if x is in the bin interval $[a_i, a_i + h_N]$ and 0 otherwise, and a_i represents the starting point of the i-th bin.

- **Spline** PDF estimator, introduced by [Wahba, 1981], divides the data range into intervals and applies polynomial functions within each interval, with smooth transitions at points called knots. Specifically, a spline function can be represented as:

$$S(x) = \sum_{i=1}^{N} c_i B_i(x), \tag{1.1}$$

where $B_i(x)$ are the basis functions, and c_i are the coefficients determined by the spline fitting process.

However, spline smoothing requires careful selection of the number and the placement of knots and polynomial degrees to balance between fitting the data well and avoiding overfitting [Kirkby et al., 2023].

- **Orthogonal series estimators** [Schwartz, 1967]. They express the unknown density function as a linear combination of orthogonal functions, such as Fourier, Hermite, or wavelet [Hall, 1982]. These functions form a basis for a function space, meaning that any function in the space can be represented as a linear combination of these basis functions. The density function $f(x)$ can be represented as:

$$\hat{f}(x) = \sum_{i=1}^{\infty} \alpha_i \phi_i(x), \tag{1.2}$$

where $\phi_i(x)$ are the orthogonal basis functions (e.g., Fourier or wavelet functions), and α_i are the coefficients determined from the data. Despite its advantages, it can be complex to implement and interpret.

- **self-consistent estimators** [Bernacchia and Pigolotti, 2011] defines an iterative procedure to refine density estimates. In this method, a candidate density function $\hat{f}(x)$ is updated iteratively based on previous estimates. The self-consistent estimator can be expressed as:

$$\hat{f}^{(t+1)}(x) = \frac{1}{N} \sum_{i=1}^{N} g\left(x; \hat{f}^{(t)}(x_i)\right), \tag{1.3}$$

where $\hat{f}^{(t)}(x)$ denotes the density estimate at iteration t, g is a function derived from the specific self-consistent method, and x_i are the data points. This approach iterates to refine density estimates, avoiding the need for parameters like bin sizes or bandwidths.

- **Neural network-based estimator** [Magdon-Ismail and Atiya, 1998] uses artificial neural networks to model and estimate the underlying probability distribution of data. The neural network is trained on a set of data points to learn the mapping between the input data and the probability density function. The density estimate $\hat{f}(x)$ at a given point x is obtained through the output of the neural network, which adjusts its parameters θ to minimize a loss function related to the observed data distribution. The formula can be represented as:

$$\hat{f}(x) = \text{NN}(x; \theta), \tag{1.4}$$

where $\text{NN}(x; \theta)$ is the neural network function with parameters θ, trained to approximate the probability density at each input point x. Training is performed by optimizing the parameters θ to minimize a suitable loss function, often based on maximum likelihood or other related objectives. Once trained, the network provides flexible and adaptive density estimates across the data space. The neural network-based density estimators are rapidly developing and gaining in popularity [Papamakarios et al., 2017] [Uria et al., 2016]. However, considerations such as training time, the complexity of the neural network, and avoiding overfitting are crucial for accurate estimation [Papamakarios et al., 2017].

- The **nearest-neighbor** estimator, proposed by [Fix and Hodges, 1989], estimates density by considering the distance to the k-th nearest neighbor in the dataset. The density at a point x is inversely proportional to the volume of the region containing the nearest k neighbors of x. The formula for the nearest-neighbor density estimate $\hat{f}(x)$ is given by:

$$\hat{f}(x) = \frac{k}{N \cdot V_k(x)}, \tag{1.5}$$

where k is the number of neighbors, N is the total number of data points, and $V_k(x)$ is the volume of the region containing the k nearest neighbors of x. This method adapts to the local structure of the data and provides flexibility, but it can be computationally intensive and sensitive to the chosen distance metric.

- **Kernel Density Estimation (KDE)** Introduced by [Rosenblatt, 1956] and developed by [Parzen, 1962], addresses some limitations of the histogram by providing a continuous estimate of the probability density function. KDE applies a kernel function to smooth the data, averaging contributions from nearby observations. This method yields a more refined and smooth density estimate compared to histograms [Silverman, 1986]. The formula for KDE is:

$$\hat{f}(x) = \frac{1}{Nh_N} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h_N}\right), \tag{1.6}$$

where $\hat{f}(x)$ is the estimated density at point x, N is the number of data points, h_N is the bandwidth (smoothing parameter), K is the kernel function, and x_i are the data points. The choice of the bandwidth h_N greatly affects the quality of the resulting density estimate. Generally, the kernels K used should have the following properties:

  - Symmetry, i.e., $K(u) = K(-u)$
  - $\int_{-\infty}^{\infty} K(u)\, du = 1$
  - $\int_{-\infty}^{\infty} u^j K(u)\, du = 0$ for $j = 1, \ldots, k-1$
  - $\int_{-\infty}^{\infty} u^k K(u)\, du \neq 0$

  These properties imply that K is an even function and that $\hat{f}_N(x)$ is a probability density function, i.e., $\int_{-\infty}^{\infty} \hat{f}_N(x)\, dx = 1$. Common kernel functions used in kernel Density Estimation include:

  - **Gaussian Kernel**: $K(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{u^2}{2}}$
  - **Epanechnikov Kernel**: $K(u) = \frac{3}{4}(1 - u^2)$ for $|u| \leq 1$
  - **Uniform Kernel**: $K(u) = \frac{1}{2}$ for $|u| \leq 1$
  - **Triangular Kernel**: $K(u) = (1 - |u|)$ for $|u| \leq 1$

As KDE offers a balance between smoothness, accurate estimation, and flexibility when the smoothing parameter is appropriately optimized we will further explore it in the subsequent section.

1.2.2 Kernel Density Estimation (KDE)

As noted by [DiNardo and Tobias, 2001], the choice of kernel function in kernel density estimation appears to have a relatively minor impact on the resulting estimate. The variety of kernel functions listed in the previous section can typically produce similar-looking density estimates. However, the choice of bandwidth is more critical. The efficiency of kernel density estimation (KDE) is highly dependent on the value of the smoothing parameter, which must be optimized. Several methods have been developed in the literature to optimize h_N. In the next section, we will review some of the most used bandwidth optimization approaches.

1.2.2.1 Kernel bandwidth optimization

The bandwidth parameter h_N controls the degree of smoothing applied to the data in kernel density estimation. A smaller h_N leads to an estimator with low bias but high variance, capturing finer details but potentially introducing noise. Conversely, a larger h_N results in high bias and low variance, smoothing out details and reducing noise but potentially obscuring important features.

The goal is to find the optimal h_N that balances bias and variance. This optimal value, denoted by h_N^*, represents the bandwidth that minimizes the overall error between the estimated density and the true underlying density.

One common metric for evaluating the quality of the estimate is the mean integrated squared error (MISE). The optimal h_N^* corresponds to the value that minimizes the MISE.

The MISE is calculated using the following formula:

$$MISE = E\left[\int_{-\infty}^{+\infty}\left(\hat{f}_N(x) - f(x)\right)^2 dx\right]$$

The Appendix 4.6 details the mathematical foundation for Bandwidth optimization. The optimal bandwidth h_N^* is expressed by the equation A.4

$$h_N^* = N^{-\frac{1}{5}} \left(J(f)\right)^{-\frac{1}{5}} \left(M(K)\right)^{\frac{1}{5}}$$

where $M(K)$ is the integral of the squared kernel function:

$$M(K) = \int_{-\infty}^{+\infty} K^2(u)du$$

and $J(f)$ is f" is the second derivative of f.

$$J(f) = \int_{-\infty}^{+\infty} (f"(x))^2 dx$$

$M(K)$ is straightforward to compute. The only term in h_N^* that is difficult to calculate is $J(f)$ since it depends on f, the unknown density function to be estimated.

Several researchers were interested in bandwidth optimization, we will develop in the next section the most popular ones.

a) Cross-Validation in Bandwidth Optimization: Several cross-validation approaches are used to optimize the bandwidth h in kernel density estimation [Hall and Marron, 1987] . One common approach is the Least Squares Cross-Validation (LSCV) method [Bowman, 1984], which seeks to minimize the Integrated Squared Error (ISE). The ISE is defined as:

$$ISE = \int_{-\infty}^{\infty} [\hat{f}_h(x) - f(x)]^2 \, dx,$$

where $\hat{f}_h(x)$ is the kernel density estimate and $f(x)$ is the true underlying density function. LSCV minimizes the ISE by approximating it through:

$$LSCV(h) = \int_{-\infty}^{\infty} \hat{f}_h^2(x) \, dx - \frac{2}{n} \sum_{i=1}^{n} \hat{f}_{-i}(x_i),$$

where $\hat{f}_{-i}(x)$ is the leave-one-out kernel estimate. The optimal bandwidth h^* is the one that minimizes this LSCV function, providing an unbiased estimator for the density. This

method, while effective, can be sensitive to sample variations, leading to multiple local minima, which complicates the selection of h. Other methods, such as Biased Cross-Validation (BCV) [Scott and Terrell, 1987] and Smoothed Cross-Validation (SCV)[Hall and Marron, 1991], offer alternatives that address some of these limitations.

b) Rule of Thumb (ROT) Methods in Bandwidth Selection: The Rule of Thumb (ROT) method, developed by Deheuvels in 1977 [Deheuvels, 1977], offers a simple approach for bandwidth selection by assuming a reference distribution to estimate the unknown density. Typically, the normal distribution is used, with its parameters estimated from the sample. The optimal bandwidth $\hat{h}_{rot}$ for a sample size N is given by:

$$\hat{h}_{rot} = 1.06\hat{\delta}_N N^{-\frac{1}{5}},$$

where $\hat{\delta}_N$ is the sample standard deviation. In other works, such as in [Härdle, 1991] , the standard deviation $\hat{\delta}_N$ is replaced by the interquartile range R, leading to an alternative form of the optimal bandwidth:

$$\hat{h}_{rot} = 1.06 \min(\hat{\delta}_N, \frac{\hat{R}}{1.34}) N^{-\frac{1}{5}}.$$

[Terrell, 1990] proposed an upper bound for the bandwidth, called the Maximal Smoothing Parameter (MSP), based on a lower bound for the functional $J(f)$, yielding:

$$\hat{h}_{MSP} = 3\left(\frac{35}{M(K)}\right)^{\frac{1}{5}} \hat{\delta}_N N^{-\frac{1}{5}}.$$

While Rule of Thumb methods are generally robust for unimodal distributions, they can provide suboptimal results for multimodal distributions.

c) Plug-in Method The Plug-in algorithm [Hall et al., 1992] [Delaigle and Gijbels, 2002] [Park and Marron, 1990] is an algorithm that estimates $J(f)$ over successive iterations until convergence is achieved. These algorithms are straightforward to implement and generate

interesting results for both unimodal and multimodal densities. The general procedure of the Plug-in algorithm is outlined in figure 1.1 and the following pseudo-code 1.

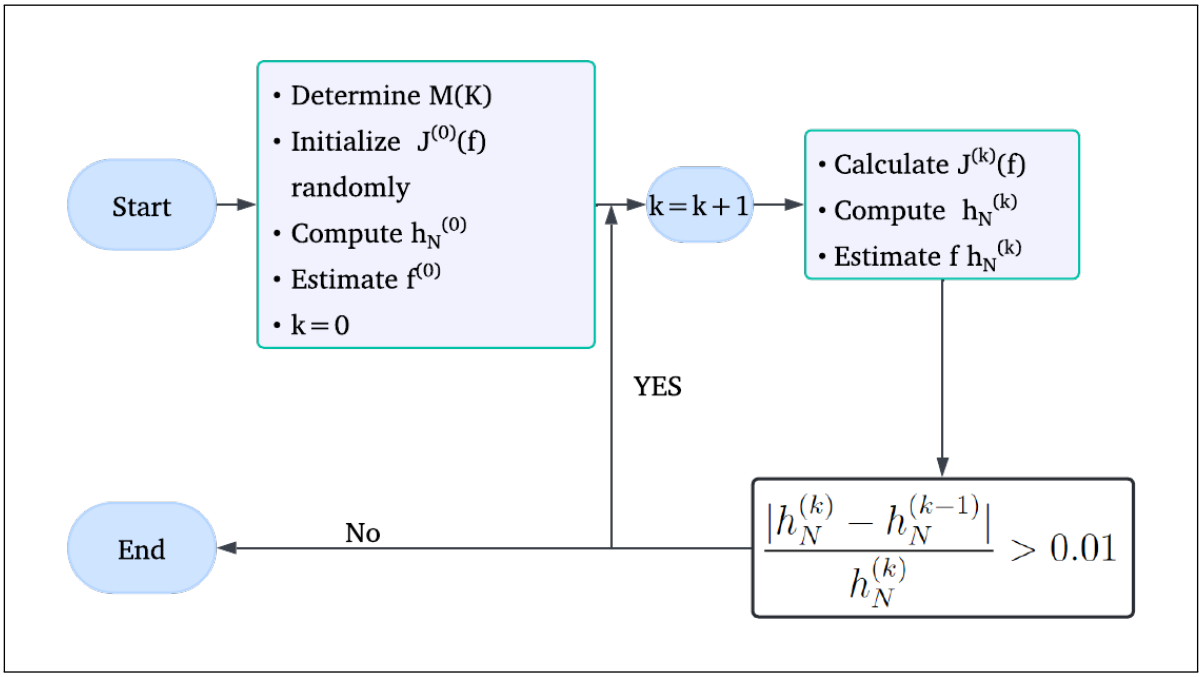


Figure 1.1: Schematic representation of the Plug-in algorithm

Algorithm 1 Standard Plug-in Algorithm

1: Determine $M(K)$
2: Initialize $J^{(0)}(f)$
3: Compute $h_N^{(0)}$
4: Estimate $f^{(0)}$ (the initial estimate of f)
5: **while** $\frac{|h_N^{(k)} - h_N^{(k-1)}|}{h_N^{(k)}} > 0.01$ **do**
6: Calculate $J^{(k)}(f)$
7: Compute $h_N^{(k)}$
8: Estimate f using $h_N^{(k)}$
9: **end while**
10: Return h_N

d) Comparison of Bandwidth Optimizers In this section, we compare three bandwidth selection methods for Kernel Density Estimation (KDE): Plug-in, Rule of Thumb (ROT), and Cross-Validation (CV). Using a bimodal distribution generated from three normal distributions with varying means and variances. The first distribution has a mean of 0.75 and variance 0.01, the second has a mean of 0.5 and variance 1, and the third has a mean of 1.75 and variance 0.01.

Table 1.1: MISE values for different bandwidth selection methods.

	Plug-in	**ROT**	**CV**
MISE	0.0175	0.0561	0.1226

The true probability density function (PDF) is computed as a combination of these distributions, and the Mean Integrated Squared Error (MISE) for each method is calculated to assess accuracy. The MISE values are summarized in Table 1.1.

For this mixture, the results indicate that the Plug-in method provides the most accurate density estimate, significantly outperforming the ROT and CV methods. The performance of each method is visualized in Figure 1.2, illustrating the estimated densities against the true density.

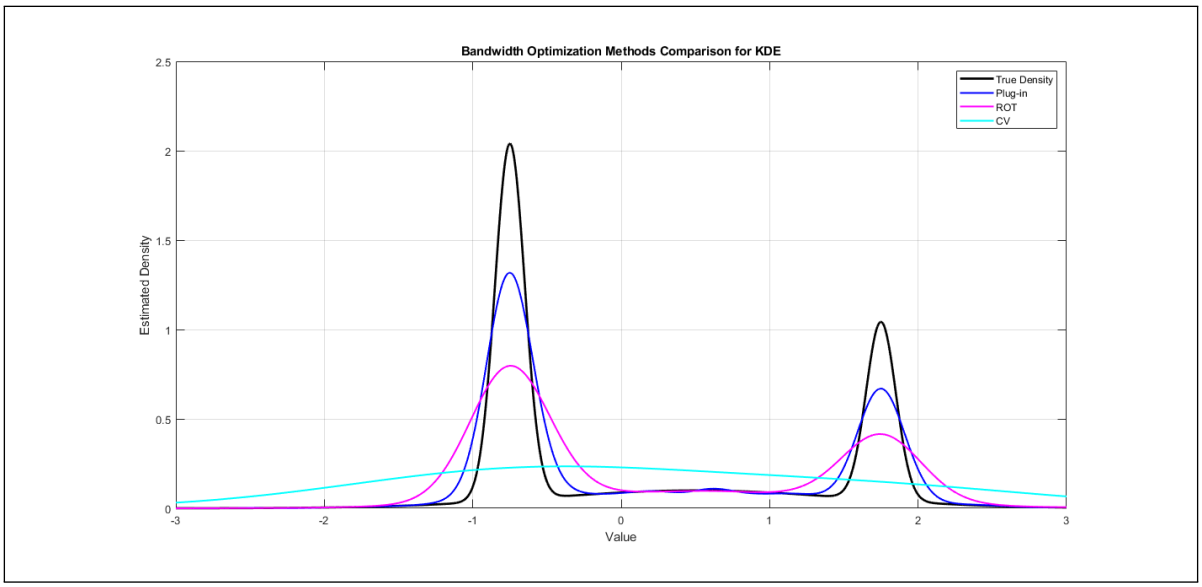


Figure 1.2: Comparison of Bandwidth Optimizers

Thus, in this research, we will be particularly interested in the plug-in algorithm.

1.2.2.2 KDE's variants

Beyond the classic KDE, several variants have been developed to enhance its performance in different scenarios.

- **The Adaptive Kernel Density Estimation** approach, as detailed by [Van Kerm, 2003], adjusts the bandwidth locally based on the density of the data, offering finer resolution in regions with high data concentration and smoother estimates where data are sparse.
- **The Robust Kernel Density Estimator**, introduced by [Kim and Scott, 2012], aims to mitigate the influence of outliers, ensuring that the density estimate remains stable even in the presence of aberrant data points.
- **the Diffeomorphic Kernel Density Estimation (DKDE)** introduced by [Saoudi et al., 1997] generalizes the kernel density estimation by applying diffeomorphic transformations, ensuring smooth and invertible mappings and improving boundary behavior.
- **the Fast Kernel Density Estimator (FKDE)**, proposed by [Troudi et al., 2008], is a computationally efficient variant that reduces the complexity of KDE by employing analytical approximations, making it particularly suitable for large datasets while maintaining accuracy in density estimation. It uses the Fast Plug-in algorithm, which iteratively estimates the optimal bandwidth while minimizing the Mean Integrated Squared Error (MISE). This approach significantly reduces the computational complexity, avoiding the need for multiple density estimations.

1.3 Expectation-Maximization: Algorithmic Foundations and Variants

The Expectation-Maximization (EM) algorithm is a widely used iterative method for finding maximum likelihood estimates (MLE) of parameters in statistical models, particularly when the data is incomplete or has missing values. Originally introduced by Dempster, Laird, and Rubin in 1977, the EM algorithm has become a foundational tool in many statistical fields [Dempster et al., 1977]. It operates by alternating between two key steps: the Expectation (E) step, which computes the expected value of the log-likelihood function with respect to the current estimate of the distribution of the missing data, and the Maximization (M) step, which maximizes this expected log-likelihood to update the parameters.

However, it is important to note that the EM algorithm only guarantees convergence to a local maximum of the likelihood function, which may not be the global maximum [Dempster et al., 1977]. Despite this limitation, the EM algorithm's flexibility and robustness make it a popular choice in various applications, including clustering, image processing, and machine learning.

The EM algorithm has several variants designed to address its limitations in different scenarios. In this section, we discuss some popular variants, including Monte Carlo EM (MCEM), Variational EM, Online EM, Bayesian EM, and Generalized EM (GEM).

1.3.1 Introduction to Maximum Likelihood Estimation (MLE)

The Expectation-Maximization (EM) algorithm is fundamentally based on the principle of Maximum Likelihood Estimation (MLE). In this framework, the observed data $y_1, \ldots, y_N$ are treated as realizations of a random variable Y within a range $\mathbb{R}$, while the values $x_1, \ldots, x_N$ are considered as realizations of a latent variable X with values in $\Omega = \{1, \ldots, K\}$.

In the context of a mixture model, the observations $y_1, \ldots, y_N$ are assumed to be independent realizations of a random variable Y, which follows a parameterized density function. The goal of MLE is to estimate the parameters that maximize the likelihood of observing the given data. The likelihood function for a mixture model can be expressed as:

$$P(y|\phi) = \sum_{k=1}^{K} \pi_k P(y|x_k, \phi_k), \tag{1.7}$$

where K denotes the number of mixture components, π_k are the mixing coefficients with $\sum_{k=1}^{K} \pi_k = 1$ and $0 \leq \pi_k \leq 1$, and $P(y|x_k, \phi_k)$ represents the density function of the k-th mixture component.

The objective is to estimate the parameter set $\phi = \{\pi_1, \ldots, \pi_K, \phi_1, \ldots, \phi_K\}$ that maximizes the likelihood function. The joint likelihood of the observed data is given by:

$$P(Y, \phi) = \prod_{i=1}^{N} P(y_i|\phi). \tag{1.8}$$

MLE aims to find the parameter set $\hat{\phi}$ that maximizes this joint likelihood, which is equivalent to maximizing the log-likelihood function:

$$L(Y, \phi) = \sum_{i=1}^{N} \log P(y_i|\phi). \tag{1.9}$$

The gradient of the log-likelihood function with respect to the parameters ϕ_k can be expressed as:

$$\nabla_{\phi_k} L = \sum_{i=1}^{N} \frac{1}{P(y_i|\phi_k)} \nabla_{\phi_k} \left(\sum_{k=1}^{K} \pi_k P(y_i|x_k, \phi_k) \right). \tag{1.10}$$

To simplify, the gradient can be rewritten using the posterior probability $P(x_k|y_i, \phi)$:

$$\nabla_{\phi_k} L = \sum_{i=1}^{N} P(x_k|y_i, \phi) \nabla_{\phi_k} \log P(y_i|x_k, \phi_k), \tag{1.11}$$

where:

$$P(x_k|y_i, \phi) = \frac{\pi_k f(y_i|x_k, \phi_k)}{P(y_i|\phi)}. \tag{1.12}$$

The parameter $\hat{\phi}_k$ that maximizes the log-likelihood function must satisfy:

$$\nabla_{\phi_k} L = \sum_{i=1}^{N} P(x_k|y_i, \phi) \nabla_{\phi_k} \log P(y_i|x_k, \phi_k) = 0, \quad k = 1, \ldots, K. \tag{1.13}$$

Given the complexity of the likelihood function, finding an analytical solution is often difficult. Therefore, iterative numerical methods, such as the EM algorithm, are employed to approximate the solution [Dempster et al., 1977].

1.3.2 The Univariate EM Algorithm

The univariate EM algorithm is a simplified version of the general EM framework, applied to models with a single observed variable. This version of the EM algorithm is useful for

introducing the core concepts before extending them to more complex multivariate cases. In the univariate case, we denote the probability $P(y|x_k, \phi_k)$ by the density function $f(y, \phi_k)$. The EM algorithm operates under the principle that maximizing the complete likelihood $L((y, x), \phi)$ is more straightforward than maximizing the marginal likelihood $L(y, \phi)$. The relationship between these likelihoods is:

$$L(y, \phi) = L((y, x), \phi) - \sum_{i=1}^{N} \log P(x_i|y_i, \phi). \tag{1.14}$$

However, because x is unknown, the complete likelihood is challenging to compute. To address this, Dempster, Laird, and Rubin proposed an iterative procedure that maximizes the expected complete log-likelihood given the current parameter estimates. The iterative process involves constructing a sequence $\{\phi^{(q)}\}$ that satisfies:

$$\phi^{(q+1)} = \arg\max_{\phi} Q(\phi|\phi^{(q)}), \tag{1.15}$$

where $Q(\phi|\phi^{(q)})$ is the expected log-likelihood:

$$Q(\phi|\phi^{(q)}) = \mathbb{E}[L((y, x), \phi)|y, \phi^{(q)}]. \tag{1.16}$$

Let $P_{jk}^{(q)} = P(x_k|Y = y_j, \phi^{(q)})$ denote the posterior probability of the k-th component given the j-th observation at iteration q. The expected log-likelihood can be written as:

$$Q(\phi|\phi^{(q)}) = \sum_{i=1}^{N} \sum_{k=1}^{K} P_{ik}^{(q)} \log(\pi_k f(y_i, \phi_k)). \tag{1.17}$$

Algorithm 2 EM Algorithm

1: **Initialize** parameters $\phi^{(0)}$ with initial guesses
2: **repeat**
3: **E-step:** Compute posterior probabilities $P_{ik}^{(q)}$ using current parameters $\phi^{(q)}$
4: **M-step:** Update parameters $\phi^{(q+1)}$ to maximize the expected log-likelihood function $Q(\phi|\phi^{(q)})$
5: Update iteration index $q \leftarrow q + 1$
6: **until** Convergence criterion is met (e.g., changes in log-likelihood or parameters are below a predefined threshold)

This iterative process continues until the algorithm converges to a set of parameter estimates that maximize the likelihood function given the observed data or until the stop condition is achieved (for example a limited number of iterations).

Having established the general framework of the EM algorithm, we now delve into the specifics of its components.

1.3.2.1 Initialization of the EM Algorithm

Before delving into the iterative process of the EM algorithm, it is crucial to address the initialization of the parameters. The initial values of the parameters, denoted by $\phi^(0)$, play a significant role in the convergence and performance of the EM algorithm. Poor initialization can lead to suboptimal solutions, slow convergence, or even convergence to incorrect parameter values.

Several initialization methods are commonly used in practice:

- **Random Initialization:** The parameters are randomly chosen from a specified range. Although simple, this approach may lead to variability in the results and requires multiple runs to ensure robustness [Kwedlo, 2015].
- **K-means Clustering:** The K-means algorithm is first applied to the data to assign initial cluster memberships. The resulting cluster centers and proportions are then used to initialize the parameters of the mixture components [Chelangat and Afullo, 2023]

- **Hierarchical Clustering:** Hierarchical clustering can be employed to group the data into clusters. The centroids and the distribution within each cluster are used to initialize the parameters [Scrucca and Raftery, 2015]
- **Quantile-Based Initialization:** In cases where the data is assumed to follow a specific distribution, initial parameters can be set based on quantiles or other statistical properties of the distribution [Mari and Baldassari, 2022]
- **Multiple Runs:** To mitigate the effects of poor initialization, multiple runs of the EM algorithm with different initial values are often performed, selecting the run that produces the highest likelihood [Biernacki et al., 2003]

Another critical parameter to initialize in the EM algorithm is the number of components. The random selection of this number can be unreliable and may result in poor convergence. To address this challenge, one effective technique is the Elbow method.

The Elbow method first introduced by [Thorndike, 1953], is a widely used approach to determine the optimal number of clusters within a dataset. This method evaluates the within cluster sum of squares (WCSS) as a function of the number of clusters. The WCSS is defined as:

$$\text{WCSS} = \sum_{k=1}^{K} \sum_{x \in C_k} \|x - \mu_k\|^2, \tag{1.18}$$

where K is the number of clusters, C_k represents the k-th cluster, x is a data point within cluster C_k, and μ_k is the centroid of cluster C_k.

In this approach, the WCSS is plotted against the number of clusters. The point at which the decrease in WCSS starts to slow down significantly is referred to as the "elbow" point, indicating the optimal number of clusters. This balance between cluster compactness and separation helps avoid overfitting or underfitting the data.

As Melnykov concluded in his research [Melnykov and Melnykov, 2012], no single initialization strategy consistently outperforms others in all scenarios. Therefore, the choice of

initialization strategy should be based on the specific characteristics of the dataset and the underlying distribution of the data.

Once the initialization strategy is chosen and the parameters initialized, the EM algorithm proceeds with the iterative process. In each iteration, the Expectation step (E-step) and the Maximization step (M-step) are performed to refine the parameter estimates, gradually improving the likelihood function. The initial parameter values serve as the starting point for this iterative refinement.

By establishing a solid foundation through appropriate initialization, the EM algorithm can effectively converge to an optimal solution. The following sections will explore the key components of the EM algorithm, starting with the E-step, where the expected value of the log-likelihood is calculated.

1.3.2.2 Expectation Step

Given the current estimate of the parameter $\theta^{(t)}$, the E-step involves calculating the expected value of the log-likelihood function, with respect to the conditional distribution of the missing data, given the observed data and the current parameter estimates. For a univariate Gaussian model, this expectation can be expressed as:

$$Q(\theta|\theta^{(t)}) = \mathbb{E}[\log L(\theta|X, Z)|X, \theta^{(t)}]$$

where $L(\theta|X, Z)$ represents the likelihood function, X is the observed data, and Z denotes the latent or missing data [Dempster et al., 1977].

In this context, for a mixture model, the posterior probability that the j-th observation belongs to the k-th component (i.e., $P_{jk}^{(q)} = P(x_k|y_j, \phi^{(q)})$) is computed as follows:

$$P_{jk}^{(q)} = \frac{\pi_k^{(q-1)} f(y_j|\phi_k^{(q-1)})}{\sum_{l=1}^{K} \pi_l^{(q-1)} f(y_j|\phi_l^{(q-1)})} \tag{1.19}$$

Here, $P_{jk}^{(q)}$ is the responsibility that the k-th component takes for explaining the j-th observation, $\pi_k^{(q-1)}$ is the prior probability of the k-th component, and $f(y_j|\phi_k^{(q-1)})$ is the probability density of y_j given the k-th component at the previous iteration $q-1$.

1.3.2.3 Maximization Step

In the M-step, the algorithm aims to maximize the expected log-likelihood function computed during the E-step to update the parameter estimates. Specifically, the new parameter estimate $\theta^{(t+1)}$ is obtained by solving:

$$\theta^{(t+1)} = \arg\max_{\theta} Q(\theta|\theta^{(t)})$$

where $Q(\theta|\theta^{(t)})$ is the expected log-likelihood function from the E-step. This maximization step is iteratively performed until convergence, which is typically determined by the change in the log-likelihood or the parameter estimates falling below a predefined threshold [Dempster et al., 1977].

The parameters π_k and ϕ_k are updated as follows: For the mixture weights π_k:

$$\pi_k^{(q)} = \frac{\sum_{i=1}^{N} P_{ik}^{(q)}}{N} \tag{1.20}$$

where $P_{ik}^{(q)}$ represents the posterior probability that the i-th observation belongs to the k-th component at the q-th iteration.

For the component-specific parameters ϕ_k:

$$\phi_k^{(q)} = \arg\max_{\phi_k} \sum_{i=1}^{N} P_{ik}^{(q)} \log(\pi_k f(y_i|\phi_k)) \tag{1.21}$$

Here, $f(y_i|\phi_k)$ denotes the probability density function of the i-th observation given the k-th component's parameters.

The convergence of the algorithm is typically assessed by monitoring the relative change in the log-likelihood:

$$\frac{\log L_{q+1} - \log L_q}{\log L_q} \leq \epsilon \tag{1.22}$$

where $\log L_q$ and $\log L_{q+1}$ are the log-likelihood values at iterations q and $q+1$, respectively, and ϵ is a small threshold value.

The application of the EM algorithm varies depending on the specifics of the mixture model being used.

1.3.2.4 Convergence and Limitations

The univariate EM algorithm is relatively simple to implement. It guarantees convergence to a local maximum but does not necessarily reach the global maximum [McLachlan and Krishnan, 2007]. This limitation can be mitigated in practice by using different initializations or applying more advanced variants of the algorithm.

1.3.3 EM Algorithm Variants

In addition to the well-known Expectation-Maximization (EM) algorithm, several variants have been developed to address specific challenges or enhance the algorithm's capabilities. Below, we discuss some notable variants.

1.3.3.1 Stochastic EM Algorithm

To address the potential issue of the EM algorithm stabilizing at a local maximum, Celeux et al. [Celeux and Diebolt, 1985] introduced a stochastic element before the maximization step. This variant, known as the Stochastic EM (SEM) algorithm, uses the estimated posterior distribution from the EM algorithm to simulate a set of realizations according to this distribution [Celeux et al., 1996]. By using this stochastic simulation, the SEM algorithm prevents the parameter estimates from stabilizing at a single point, except for the point that maximizes the likelihood.

A further variant, known as Simulated Annealing EM (SAEM), was proposed by Delyon, Lavielle, and Moulines [Delyon et al., 1999]. In SAEM, the parameters estimated during

the maximization phase are a weighted average between the EM and SEM estimates. Over the iterations, these weights evolve, gradually emphasizing the EM estimate as the algorithm progresses.

Algorithm 3 SAEM Algorithm

1: **Initialization:** Select an upper bound for the number of classes. Initialize weights and marginal distribution parameters similarly to the EM algorithm.
2: **repeat**
3: **E-step:** Compute the posterior probabilities based on the current parameters.
4: **S-step:** Generate a Bernoulli random variable using the posterior probabilities from the E-step.
5: **A-step:** Construct another random variable to adjust the posterior probabilities.
6: **M-step:** Replace the posterior probabilities in the EM algorithm's maximization equations with the constructed variable.
7: **until** Convergence criterion is met (e.g., parameter estimates or log-likelihood changes are below a predefined threshold)

SAEM combines the stochastic nature of SEM with the robust convergence properties of EM, offering a balance between exploration of the parameter space and convergence to the global maximum [Delyon et al., 1999].

1.3.3.2 Monte Carlo EM (MCEM)

The Monte Carlo EM (MCEM) algorithm was developed to address scenarios where the E-step of the standard EM algorithm becomes computationally intractable due to the complexity of the expectation calculation[Wei and Tanner, 1990]. In such cases, the MCEM algorithm approximates the expected value by drawing samples from the posterior distribution, utilizing Monte Carlo simulations.

The key idea behind MCEM is to replace the exact expectation in the E-step with an approximation derived from a set of samples. Specifically, given a current estimate of the parameters, the algorithm generates a sample from the posterior distribution of the latent variables and then uses this sample to compute an empirical average. This average serves as an approximation of the expectation required for the M-step.

The steps involved in the MCEM algorithm can be summarized as follows:

Algorithm 4 Algorithm with Monte Carlo Simulation

Initialization: Start with an initial estimate of the parameters, denoted as $\theta^{(0)}$.
repeat
    E-Step (Monte Carlo Simulation):
    At the q-th iteration, draw a sample $\{z^{(q,1)}, z^{(q,2)}, \ldots, z^{(q,m)}\}$ from the posterior distribution $P(z \mid x, \theta^{(q-1)})$.
    Approximate the expectation in the E-step by the empirical mean:

$$Q(\theta \mid \theta^{(q-1)}) \approx \frac{1}{m} \sum_{j=1}^{m} \log P(x, z^{(q,j)} \mid \theta).$$

    M-Step:
    Maximize the approximated expected log-likelihood with respect to the parameters:

$$\theta^{(q)} = \arg\max_{\theta} Q(\theta \mid \theta^{(q-1)}).$$

until Convergence criterion is met (e.g., change in parameter estimates between successive iterations falls below a predetermined threshold)

The advantage of MCEM is its flexibility in dealing with complex models, particularly when direct computation of the expectation is difficult or impossible. However, the accuracy of the MCEM algorithm is based upon the quality and quantity of the Monte Carlo samples. As such, increasing the number of samples can improve the approximation but at the cost of increased computational effort.

1.3.3.3 Variational EM

The Variational EM algorithm is another variant designed to handle cases where the E-step of the standard EM algorithm is difficult to compute. Instead of using Monte Carlo simulations like in MCEM, the Variational EM approach approximates the posterior distribution by a simpler distribution from a specified family [Beal, 2003]. This distribution is parameterized, and the parameters are optimized to make the approximation as close as possible to the true posterior.

In the Variational EM algorithm, the E-step involves optimizing a lower bound on the log-likelihood, known as the Evidence Lower BOund (ELBO), rather than the exact log likelihood. The M-step remains similar to that of the standard EM algorithm, where the parameters are updated to maximize the approximate lower bound.

The steps of the Variational EM Algorithm :

Algorithm 5 Variational EM Algorithm

1: **Initialization:** Start with an initial estimate of the parameters $\theta^{(0)}$ and an initial choice of the variational parameters.
2: **repeat**
3: **E-Step (Variational Inference):**
4: At each iteration, optimize the variational parameters to maximize the Evidence Lower Bound (ELBO), which serves as a lower bound to the log-likelihood.
5: **M-Step:**
6: Update the model parameters by maximizing the ELBO with respect to θ:
7:

$$\theta^{(q)} = \arg\max_{\theta} \text{ELBO}(\theta, \text{variational parameters}).$$

8: **until** Convergence criterion is met (e.g., changes in ELBO or parameters fall below a predetermined threshold)

The advantage of the Variational EM algorithm lies in its ability to handle high-dimensional latent variable models more efficiently than the standard EM algorithm, although it introduces approximation errors due to the use of a variational distribution.

1.3.3.4 Online EM

The Online EM algorithm is designed for scenarios where data arrive sequentially, and it is impractical to reprocess the entire dataset at each iteration [Cappé and Moulines, 2009]. Unlike the standard EM algorithm, which processes the entire dataset at each iteration, Online EM updates the model parameters incrementally as new data points become available.

The Online EM algorithm can be summarized as follows:

Algorithm 6 Online EM Algorithm

1: **Initialization:** Begin with an initial estimate of the parameters $\theta^{(0)}$.
2: **while** Data is available **do**
3: **E-Step:** For each incoming data point, compute a partial expectation using the current model parameters.
4: **M-Step:**
5: Update the parameters incrementally based on the partial expectation:
6:

$$\theta^{(q)} = \theta^{(q-1)} + \alpha^{(q)} \cdot \text{Gradient},$$

where $\alpha^{(q)}$ is a learning rate that decreases over time.
7: **end while**

Online EM is particularly useful for large-scale data applications where it is crucial to update the model in real-time without the need to store and reprocess large amounts of data.

1.3.3.5 Bayesian EM

The Bayesian EM algorithm is an extension of the standard EM algorithm within a Bayesian framework. Instead of finding point estimates of the parameters, Bayesian EM seeks to estimate the posterior distribution over the parameters given the data.

In the Bayesian EM algorithm, the M-step involves determining the mode of the posterior distribution, also known as the maximum a posteriori (MAP) estimate. The E-step remains similar, where the expected value is taken with respect to the posterior distribution of the latent variables.

The Bayesian EM algorithm can be summarized as follows:

Algorithm 7 Bayesian EM Algorithm

1: **Initialization:** Start with an initial estimate of the prior distribution over the parameters.
2: **while** Convergence criterion not met **do**
3: **E-Step:** Compute the expected value of the complete data log-likelihood with respect to the posterior distribution of the latent variables.
4: **M-Step:** Maximize the posterior distribution to obtain the MAP estimate:
5:

$$\theta^{(q)} = \arg\max_{\theta} P(\theta \mid \text{data}).$$

6: **end while**

Bayesian EM provides a more robust approach by incorporating prior information and uncertainty about the parameters, making it particularly valuable in cases where data is sparse or noisy.

1.3.3.6 Generalized EM (GEM)

The Generalized EM (GEM) algorithm is a flexible variant of the standard EM algorithm that allows for approximate maximization in the M-step. Instead of requiring the M-step to find the exact maximum of the expected complete data log-likelihood, GEM permits any update that increases the likelihood.

The GEM algorithm is useful in situations where the M-step is computationally expensive or difficult to solve exactly. By allowing approximate updates, GEM can converge more quickly and handle a broader class of models.

The steps in the GEM algorithm are as follows:

Algorithm 8 Generalized EM Algorithm

1: **Initialization:** Start with an initial parameter estimate, $\theta^{(0)}$.
2: **while** Convergence criterion not met **do**
3: **E-Step:** Compute the expected complete data log-likelihood given the current parameters to proceed to the next step.
4: **M-Step (Generalized Maximization):** Update the parameters to increase the likelihood, even if the exact maximum is not reached, for better accuracy.
5: $$Q(\theta^{(q)} \mid \theta^{(q-1)}) \geq Q(\theta^{(q-1)} \mid \theta^{(q-1)}).$$
6: **end while**

The flexibility of GEM makes it a powerful tool in applications where strict maximization is hard to achieve, providing a balance between computational efficiency and accuracy.

1.3.3.7 Conclusion

Having reviewed various EM algorithm variants, we will introduce a new EM variant in the following chapter 2. This new approach is designed to identify non-Gaussian mixture on a bounded support.

1.4 Regression Evaluation metric

Due to the absence of a "gold standard" evaluation metric, researchers have continuously developed new metrics in search of more effective solutions. As a result, the literature is filled with evaluation metrics, many of which remain unknown and understudied, while no universally optimal metric has yet been identified. The researchers have not been able to agree upon a single, universal evaluation metric because each has advantages and disadvantages that highlight different aspects of the model's error characteristics [Makridakis et al., 2022]. However, they

did agree that a single evaluation metric is not enough to properly assess the model's performance. In 2014, Chai and his colleague highlighted the limitations of relying on a single metric. They noted that individual metrics provide a limited perspective of the model's errors.[Chai and Draxler, 2014]

Therefore, the selection of the suitable set of metrics depends mainly on the nature of the algorithm and the desired properties to assess and highlight, as different metrics may produce different rankings of the model performance.

In this section, we will focus on evaluation metrics applied in regression analysis. The assessment of regression models usually depends on their capacity to forecast continuous results, and the selection of the evaluation criteria is essential as it directly impacts the evaluation of model effectiveness. We will only present the most popular used metrics

1.4.1 Error-Based Metrics

Error metrics are fundamental tools for evaluating regression models, offering clear assessments of the difference between predicted and actual values. These metrics facilitate straightforward interpretation and comparison across different models.

1.4.1.1 Traditional Metrics

Traditional error-based metrics are among the most frequently used methods for evaluating regression models. These metrics provide insights into a models predictive accuracy by quantifying the difference between actual and predicted values from a single iteration of the model. While they offer a quick and accessible measure of performance, they may not fully capture the variability of the model across different runs or data splits.

One of the earliest and most widely used metrics is the Mean Squared Error (MSE), which calculates the average squared difference between predicted and actual values. This metric places more weight on larger errors making it sensitive to outliers. As extreme values disproportionately influence the metric, they potentially distort the overall evaluation [Wang and Bovik, 2009]. The Root Mean Squared Error (RMSE) extends MSE by taking the square root of the

mean squared errors enhancing its interpretability by aligning the error metric with the original data units. RMSE is particularly favored in time series forecasting and for comparing model performance on identical datasets [Kumbure et al., 2022]. Despite its widespread use, RMSE retains the outlier sensitivity inherent in MSE, which can limit its effectiveness in datasets with high variability.

Alternatively, the Mean Absolute Error (MAE) calculates the average of the absolute differences between predicted and actual values. Unlike MSE and RMSE, MAE treats all errors equally without squaring them, making it less sensitive to outliers and a robust choice for scenarios where balanced error representation is required. MAE is simple to interpret, as it shares the same units as the original data, but its lack of sensitivity to data variance can be a limitation in some analyses. Consequently, as suggested by researchers such as Chai et al. (2014) [Chai and Draxler, 2014], a robust evaluation of a model's performance often necessitates the use of multiple metrics, including RMSE and MAE.

A further refinement is the Median Absolute Error (MdAE), a variation of MAE that uses the median of absolute errors instead of the mean. This modification minimizes the impact of outliers even more effectively, making MdAE a robust measure of central tendency, particularly valuable in datasets with unbalanced error distributions. MdAE is preferred in situations where the median offers a more accurate reflection of typical errors, as highlighted in studies like those by Yin et al. [Yin and Xie, 2021].

1.4.1.2 Normalized Error Metrics

Normalized error metrics offer scale-independent evaluations, allowing for equitable comparisons across different datasets and models. The Normalized Mean Squared Error (NMSE) evaluates the mean squared error relative to the variance of the target variable. This makes it particularly valuable in applications such as climate modeling, where performance comparisons across regions with varying climate variability are necessary [Tcheou et al., 2021]. While Mean Absolute Scaled Error (MASE) emphasizes absolute errors and is highly regarded for its consistent behavior across varying scales [Hyndman and Koehler, 2006], the Root Mean Squared

Scaled Error (RMSSE) scales the RMSE by a benchmark error, highlighting predictions that closely align with the mean [Ramos and Oliveira, 2023]. Both metrics are symmetrical, penalizing positive and negative forecast errors equally.

Weighted metrics like the Weighted Root Mean Squared Error (wRMSE) and Weighted Mean Absolute Error (wMAE) introduce weights to error terms, enabling the prioritization of certain errors based on their significance. This approach is particularly useful in scenarios where the importance of errors varies across data points [Cleger-Tamayo et al., 2012]. Similarly, the Weighted Root Mean Squared Scaled Error (WRMSSE), which normalizes each error term by the weighted error of a baseline model, has been employed in competitive settings, such as the M5 competition, to provide a nuanced evaluation of model performance [Makridakis et al., 2022].

The Mean Relative Absolute Error (MRAE) provides a relative error evaluation by comparing the mean of absolute errors to actual values, offering a normalized perspective on model performance. This metric is particularly useful for datasets with varying scales, as it accounts for the magnitude of the data, providing a balanced evaluation of prediction accuracy [Lin and Finlayson, 2021].

1.4.2 Percentage-Based Metrics

Percentage-based metrics are useful for assessing the relative accuracy of regression models, particularly when proportional errors are more critical than absolute differences. These metrics express errors as a percentage of the actual values, facilitating meaningful comparisons across datasets with varying scales. The Mean Absolute Percentage Error (MAPE) is a widely adopted metric, especially in fields like business and economics, where relative measures of gains and losses are essential [Gneiting, 2011]. It calculates the average absolute percentage difference between predicted and actual values, making it intuitive and accessible even to non-experts [De Myttenaere et al., 2016]. Despite its popularity, MAPE has limitations, particularly with data containing values close to zero, which can result in extremely large or undefined values [Kim and Kim, 2016]. To address these issues, the Symmetric Mean Absolute Percentage

Error (sMAPE) was developed. sMAPE improves upon MAPE by dividing the absolute error by the sum of the actual and predicted values, reducing the impact of extreme values and addressing some of MAPE's limitations. However, it can still exhibit asymmetry, often penalizing positive errors more than negative ones [Goodwin and Lawton, 1999]. Although not strictly percentage-based, the Mean Magnitude of Relative Error (MMRE) is used in software effort estimation but can be unreliable, often favoring models that systematically under-forecast [Jørgensen et al., 2022].

1.4.3 Variance Metrics

Variance metrics play a crucial role in evaluating how well a regression model captures the variability of the target variable, providing insights into the model's effectiveness in fitting the data and explaining underlying patterns. The coefficient of determination, commonly known as R^2, is a widely used metric that measures the proportion of variance in the target variable that is explained by the regression model. Mathematically, R^2 represents the ratio of explained variance to total variance, with values ranging from 0 to 1, where a higher R^2 indicates a better model fit. Chicco (2021) [Chicco et al., 2021] recommends using R^2 as a standard measure in regression analysis, acknowledging its widespread acceptance. Despite its popularity, R^2 has a key limitation: it tends to increase with the addition of more features, regardless of their actual predictive power, which can lead to overfitting and poor performance on unseen data. To address the overfitting issue associated with R^2, the Adjusted R^2 metric incorporates a penalty for the number of predictors in the model. Unlike R^2, which always remains the same or increases with the addition of new predictors, Adjusted R^2 can decrease if the new predictors do not sufficiently enhance the model. This adjustment provides a more balanced evaluation of model performance, accounting for both the goodness of fit and the model's complexity, and ensures that Adjusted R^2 is always less than or equal to R^2. This makes Adjusted R^2 a more reliable metric when comparing models with varying numbers of predictors, offering a clearer assessment of model performance.

1.4.4 Logarithmic Metrics

Logarithmic metrics are particularly useful for datasets composed of strictly positive values, as they apply a logarithmic transformation to the errors, which helps manage large-scale variations and provides a more balanced evaluation of model performance. The Mean Squared Logarithmic Error (MSLE) calculates the mean of the squared logarithmic differences between predicted and actual values. This metric is designed to penalize underestimates more than overestimates, making it especially valuable in contexts where underprediction is more critical than overprediction. Similarly, the Root Mean Squared Logarithmic Error (RMSLE) as presented in Liapis et al. (2023) [Liapis et al., 2023], extends the MSLE by taking the square root of the mean squared logarithmic errors, with an added constant of 1 to avoid undefined logarithms when dealing with zero or near-zero values. This modification makes RMSLE more robust against outliers and noise, providing a proportional penalty for large prediction errors, and is beneficial when accurate lower-bound predictions are crucial. Both MSLE and RMSLE are useful in scenarios where the target variable spans a wide range or where smaller errors are more significant than larger ones, as highlighted by Hodson in 2021 [Hodson et al., 2021].

The Mean Absolute Logarithmic Error (MALE), on the other hand, computes the mean of the absolute logarithmic differences between actual and predicted values. This metric, suited for multiplicative scale datasets, uses absolute differences rather than squared differences, making it less sensitive to large errors and outliers. As noted by Vamsi et al. (2021) [Vamsi Krishna et al., 2021], The combination of mean and logarithmic differences in MALE contributes to its robustness, offering a stable and interpretable measure for evaluating model performance, particularly in datasets where relative differences are more meaningful than absolute ones.

1.4.5 Bias Metrics

Bias metrics evaluate the systematic tendencies of a model to consistently overestimate or underestimate the target variable, helping to identify and correct biases that may skew predictions and compromise model reliability. The Fractional Gross Error (FGE) evaluates the absolute difference between predicted and actual values relative to their mean. It offers a more robust

measure of bias compared to The Mean Bias (MB) by being less affected by cancellation errors and providing a balanced assessment of both overestimation and underestimation. As highlighted by Benedetti et al. (2019) [Benedetti et al., 2019], FGE is symmetric and less sensitive to outliers, enhancing its effectiveness in bias assessment.

1.4.6 Relative Measures

Relative measures offer valuable insights into a model's performance by comparing it to a baseline, often the mean of the target variable, which helps in assessing its relative effectiveness. The Log of the Accuracy Ratio (logAR) is a well-known metric in this category, as it compares model performance to the mean of the target variable using a logarithmic transformation. As highlighted by [Tofallis, 2015], logAR is a robust alternative to traditional metrics like MAPE, effectively addressing issues with very small actual values that can distort MAPE and sMAPE. This normalization offers a relative measure of error and allows comparison of prediction errors relative to a baseline or reference model. Studies such as [Sudsawat et al., 2021] found The Relative Mean Squared Error (RMRSE) to be more informative about true performance than MSE. [Reich et al., 2016] focused on The Relative Mean Absolute Error (RMAE), noting its simplicity and intuitive interpretation, compared to MAPE, which systematically favors under-forecasting methods

1.4.7 Additional Evaluation Metrics

As mentioned in the introduction to this section, the literature on evaluation metrics is extensive, covering a vast array of approaches. We have presented only a selective overview of some of the most commonly used metrics. However, numerous other metrics exist, including distance-based metrics, correlation-based metrics, Median Absolute Percentage Error (MdAPE), Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), Fractional Bias (FB), Relative Absolute Error (RAE), Relative Root Mean Squared Error (RelRMSE), and Explained Variance, Geometric Mean Relative Absolute (GMRAE) among many others. Each of these metrics serves different purposes and offers unique insights depending on the context of the evaluation.

1.4.8 Averaging Single-Iteration Metrics

Relying on a single iteration for assessing model performance is often inadequate for obtaining a reliable estimate. To address this, researchers such as Bhandari et al. (2022) [Bhandari et al., 2022] have explored the approach of averaging metrics obtained from multiple training runs. In their study, Bhandari and colleagues replicated their experiments 30 times using three evaluation metrics: RMSE, MAPE, and R^2. They observed significant discrepancies between the maximum and minimum values across these replications, which underscores a key limitation of single-iteration metrics.

Running the model just once can produce results that significantly differ from the model's true performance, as model errors can be considered as a realization of a continuous random variable. Bhandari's experiments demonstrated this variability; for instance, in a single-layer experiment with 10 neurons, the RMSE ranged from a minimum of 34.7 to a maximum of 77.5. While averaging across multiple runs can account for some variability, it may lose critical details about model performance. The significant differences between extreme values observed by Bhandari et al. suggest that performance variability may not be fully captured by averaging alone. This underscores the need for more comprehensive evaluation methods.

1.5 Conclusion

This chapter explored the state of the art in various fields.

First, we explored the non-parametric probability density estimators, with a special focus on the Kernel Density Estimator. We explored varied Bandwidth optimization methods, and we were interested in the plugin method and the fast plugin method. We also presented some variants of, KDE particularly fast KDE .

In the second part, we described the Expectation Maximization Algorithm, detailed its steps, and discussed some of its variants.

In the final section, we reviewed the state of the art of the regression's evaluation metrics. We covered some of the most popular metrics used and highlighted some of their limitation and the need for a new method for evaluating algorithms.

The Diffeomorphism Expectation-Maximization Algorithm

Contents

Chapter 2 Abstract

This chapter introduces the Diffeomorphism Expectation-Maximization (EMD) algorithm, a novel variant of the EM algorithm tailored for handling bounded data. The chapter begins with the integration of a logarithmic transformation into the standard EM framework, laying the foundation for the EMD methodology. The EMD initialization and its mathematical underpinnings are discussed in detail, followed by experimentation on simulated datasets, including two-, three-, and five-component mixture models, which demonstrate the effectiveness of the algorithm.

The chapter also explores two key applications. The first application is ultrasound image segmentation, where the EMD algorithm is compared to reference segmentation techniques such as U-Net and edge detection. Various evaluation metrics, including Intersection over Union (IoU), Dice Coefficient, Precision, Recall, and the Symmetric Hausdorff Distance, are used to assess the algorithm's performance on benign, malignant, and normal ultrasound images. The second application focuses on Affine Multi-Scale Curve Registration (AMSCR) and the EM algorithm, which combines affine transformations with multi-scale smoothing to align curves. The AMSCR-EM uses the EM algorithm to identify the classes of relevant scales based on the L2 distance. This approach is tested on benchmark datasets, such as MPEG-7 and KIMIA, to validate its efficacy in shape retrieval and registration tasks.

2.1 Introduction

The Expectation-Maximization (EM) algorithm, first developed by Dempster and his colleagues in [Dempster et al., 1977], is a fundamental tool in statistical computing, extensively used for estimating parameters in probabilistic models due to its solid statistical basis [McLachlan and Krishnan, 2007]. The original paper by Dempster et al. [Dempster et al., 1977] has been cited more than 73,000 times according to Google Scholar, underscoring its widespread influence. Despite its popularity, traditional EM variants encounter challenges when dealing with data bounded by finite support. Typically, these algorithms do not account for boundary constraints, leading to estimates that may overflow beyond the data's natural limits. This issue becomes particularly problematic when there's clustering near the boundaries [Zhang et al., 2004] or when the data distribution deviates from the Gaussian assumption.

To overcome this limitation, we introduce a novel variant of the EM algorithmthe Diffeomorphism EM algorithm (EMD). This approach integrates the concept of diffeomorphism into the EM framework, allowing it to respect the boundary constraints of the data. The main concept of the EMD lies in the application of diffeomorphic mappings, which transform the original bounded data space into an infinite domain. This ensures that the algorithm's estimates remain within the finite support, improving accuracy, particularly for non-Gaussian mixture models.

In this chapter, we explore the theoretical basis of the EMD algorithm and explain how it differs from conventional EM methods. We also present a detailed performance evaluation of the EMD algorithm through a series of simulations and real-world applications, particularly focusing on its utility in ultrasound image segmentation. Ultrasound imaging, a common tool in medical diagnostics, is known for its low contrast and high speckle noise, with pixel values often clustering around zero [Saini et al., 2010]. This makes it an ideal candidate for evaluating the effectiveness of EMD. The results demonstrate the robustness and efficiency of EMD compared to existing EM variants, highlighting its effectiveness in parameter estimation and mixture identification tasks.

Finally, we applied the traditional EM algorithm to the Curve Registration problem, in conjunction with an Affine Multi-Scale Algorithm (AMSCR). We evaluate this combined approach across various datasets, showcasing its versatility and potential for broader applications.

2.2 Diffeomorphism Expectation Maximization Algorithm

The Diffeomorphism Expectation-Maximization (EMD) is composed of two main parts : the EM algorithm which was extensively presented in sec1.3 and a diffeomorphism to be chosen according to the nature of data and the final aim of the model. In this research, we chose the logarithm transformation as the diffeomorphism.

2.2.1 Integration of Logarithmic Transformation and EM Algorithm

The integration of the logarithmic transformation into the EM algorithm involves applying the transformation to the observed data. This approach transforms the data from a bounded or semi-bounded support to an unbounded (infinite) support, which allows us to avoid the overflowing issues that may arise during estimating near boundary values.

Consider a dataset $\{y_1, y_2, \ldots, y_n\}$ where each $y_i > 0$. The logarithmically transformed data $\{z_1, z_2, \ldots, z_n\}$ is defined as:

$$z_i = \log(y_i), \quad \text{for } i = 1, 2, \ldots, n$$

In the special context of image processing, the logarithmic transformation is defined as:

$$\text{log\_transformed\_image} = \log(1 + \text{image})$$

where *image* represents the original pixel values, and *log* denotes the natural logarithm. The addition of 1 ensures that the transformation is defined for all pixel values, including zero.

2.2.2 EMD Initialisatiion

The EMD algorithm is well-adapted for unsupervised classification, particularly for identifying mixture components. However, in unsupervised classification, the number of classes (or components) is typically unknown. Instead of randomly selecting this value, which can lead to inaccurate results, we employed the elbow method, to estimate the optimal number of classes.

By selecting the number of clusters at the elbow point, we provide an informed, reliable initialization for the EM algorithm, ensuring better convergence and accuracy.

Once the optimal number of clusters is determined using the elbow method, we proceed to estimate the remaining initial parameters for the EMD algorithm, namely the means and variances. Specifically, we apply the k-means clustering algorithm with the previously identified optimal number of clusters. The final result of the k-means algorithm provides an initial set of parameters: the centroids of the clusters are used as the initial mean estimates, and the variance within each cluster is computed and used as the initial variance estimates. These initial parameters are then fed into the EMD algorithm to begin the iterative process of parameter optimization for mixture model estimation.

2.2.3 the EMD Methodology

The EM algorithm is applied to the transformed data $\{z_1, z_2, \ldots, z_n\}$. The steps of the modified EM algorithm incorporating the logarithmic transformation are as follows:

- **Step 1: Apply the visualized Elbow method**
  - Choose from the plot the optimal number of class K
- **Step 2: Apply Logarithmic Transformation**
  - Transform the observed data $\{y_1, y_2, \ldots, y_n\}$ to $\{z_1, z_2, \ldots, z_n\}$.
- **Step 3: Initialize Parameters with K-means**

  - apply the k-means clustering algorithm to initialize the parameters $\phi^{(0)}$. The centroids from k-means serve as initial estimates for the component means, and the within-cluster variances are used to initialize the variance parameters in the EMD algorithm.

- **Step 4: Expectation Step (E-step)**

  - Compute the posterior probabilities $P_{ik}^{(q)}$ for the transformed data, based on the current parameter estimates $\phi^{(q)}$.

$$P_{ik}^{(q)} = \frac{\pi_k^{(q-1)} f(z_i|\phi_k^{(q-1)})}{\sum_{l=1}^{K} \pi_l^{(q-1)} f(z_i|\phi_l^{(q-1)})}$$

- **Step 5: Maximization Step (M-step)**

  - Update the parameters $\phi^{(q+1)}$ to maximize the expected log-likelihood function $Q(\phi|\phi^{(q)})$.

$$Q(\phi|\phi^{(q)}) = \sum_{i=1}^{N} \sum_{k=1}^{K} P_{ik}^{(q)} \log(\pi_k f(z_i, \phi_k))$$

- **Step 6: Iteration**

  - Repeat the E-step and M-step iteratively until convergence is achieved. Convergence can be defined as the point where the changes in the log-likelihood or the parameter estimates fall below a predefined threshold, or when the maximum number of iterations is reached.

- **Step 7: Apply the Inverse Diffeomorphic Transformation**

  - After the EM algorithm has converged and the parameters $\phi^{(q+1)}$ have been estimated for the transformed data $\{z_1, z_2, \ldots, z_n\}$, apply the inverse of the diffeomorphic transformation to revert the data back to the original space. This involves using the inverse function of the diffeomorphism applied in Step 2. The transformed parameters and results are then mapped back to the original data space to provide meaningful results in the context of the original data.

Algorithm 9 Pseudo-code for EMD

Input: Data $\{y_1, y_2, \ldots, y_n\}$
Utilize the Elbow method to determine the optimal number of clusters (K) visually.
Apply logarithmic transformation: $z_i = \log(y_i)$
Initialize $\phi^{(0)}$ using K-means
while Convergence criterion not met **do**
    Compute posterior probabilities:

$$P_{ik}^{(q)} = \frac{\pi_k^{(q-1)} f(z_i|\phi_k^{(q-1)})}{\sum_{l=1}^{K} \pi_l^{(q-1)} f(z_i|\phi_l^{(q-1)})}$$

    Update parameters to maximize expected log-likelihood:

$$\phi^{(q+1)} = \arg\max_{\phi} \sum_{i=1}^{N} \sum_{k=1}^{K} P_{ik}^{(q)} \log(\pi_k f(z_i, \phi_k))$$

end while
Apply the inverse diffeomorphic transformation to revert the data back to the original space.
Return: Final parameter estimates ϕ in the original space.

The pseudo-code 9 represents the principal steps of the EMD algorithm.

By incorporating the logarithmic transformation into the EM algorithm and initializing parameters using k-means and the elbow method, EMD benefits from a solid starting point that can lead to faster convergence and more stable results.

2.3 Experimentation on simulated data

Initially, we evaluated our algorithm using various combinations of simulated data. We experimented with varying: the number of components in the mixture, considering scenarios with two, three and five distributions; the types of distributions within the mixture, including log-normal, normal, and exponential distributions, among others. In the following section, we will present some interesting results obtained from our experiments.

In all the scenarios discussed in the next section, the distributions are equally weighted within the mixture. We generated a sample of 10,000 data points from each distribution. For each scenario, we will provide a visual comparison by plotting the probability density functions PDF of the theoretical distribution alongside the estimations obtained using the EM and EMD

algorithms. In addition to this visual comparison, we will present a quantitative analysis by calculating the Mean Integrated Squared Error MISE. The MISE was obtained based on 100 algorithm executions.

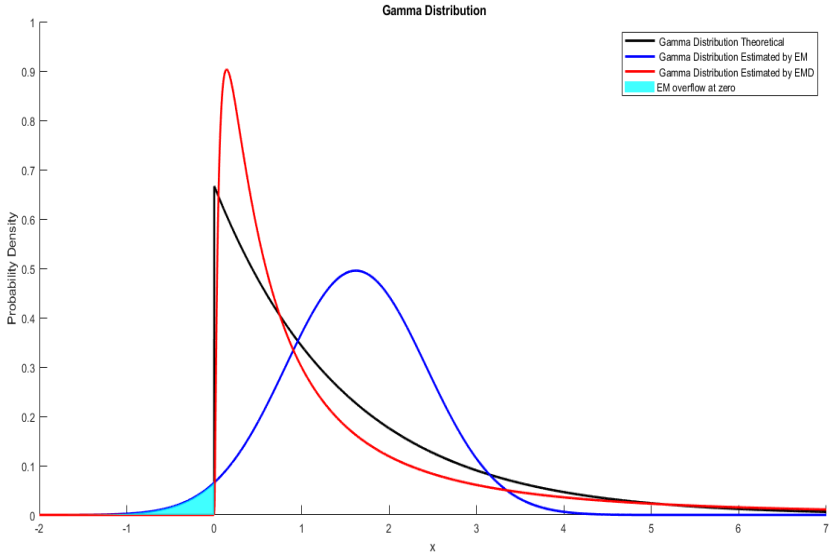
2.3.1 Two-Component Mixture Model

We began by evaluating our new algorithm on two distinct mixture models, each formed by combining two distributions. In the first scenario, we generated a bimodal mixture by combining two distinct distributions, resulting in a dataset with two modes. In the second scenario, we combined two distributions to create an unimodal mixture, producing a dataset with a single dominant mode.

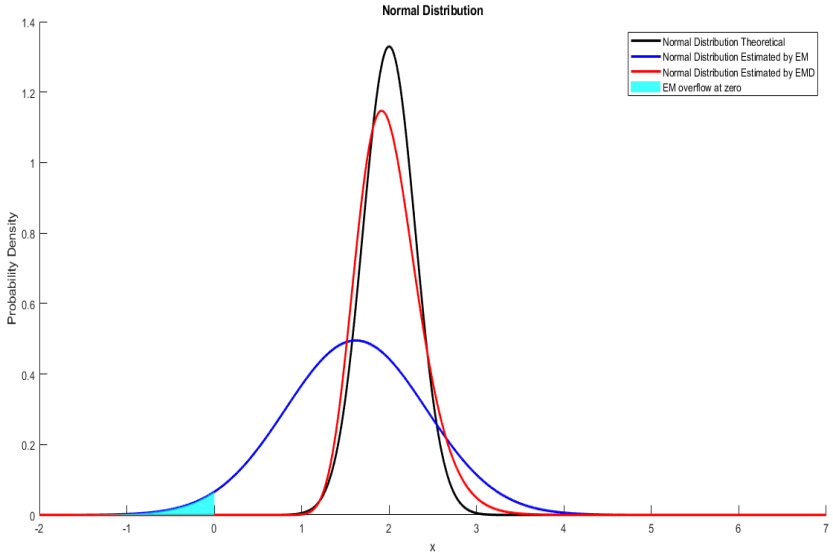
2.3.1.1 Bimodal Mixture Model

In this first scenario, we modeled a bimodal mixture consisting of a Gamma distribution with parameters (1, 1.5) and a normal distribution with parameters (2, 0.3).

Figure 2.1 illustrates the identification of the mixture components by both the EM and EMD algorithms. The EMD correctly identifies the mixture components and its curve closely matches each underlying distribution. Additionally, while EM produced negative estimates for these strictly positive distributions, the EMD algorithm correctly respected the positivity constraint of the data.



(a) Gamma distribution component



(b) Normal distribution component

Figure 2.1: Visualization of the individual components in the bimodal mixture

Using the parameters estimated by both EM and EMD, including the distribution weights, we reconstructed the mixture distributions and compared them to the theoretical mixture. This comparison is illustrated in Figure 2.2

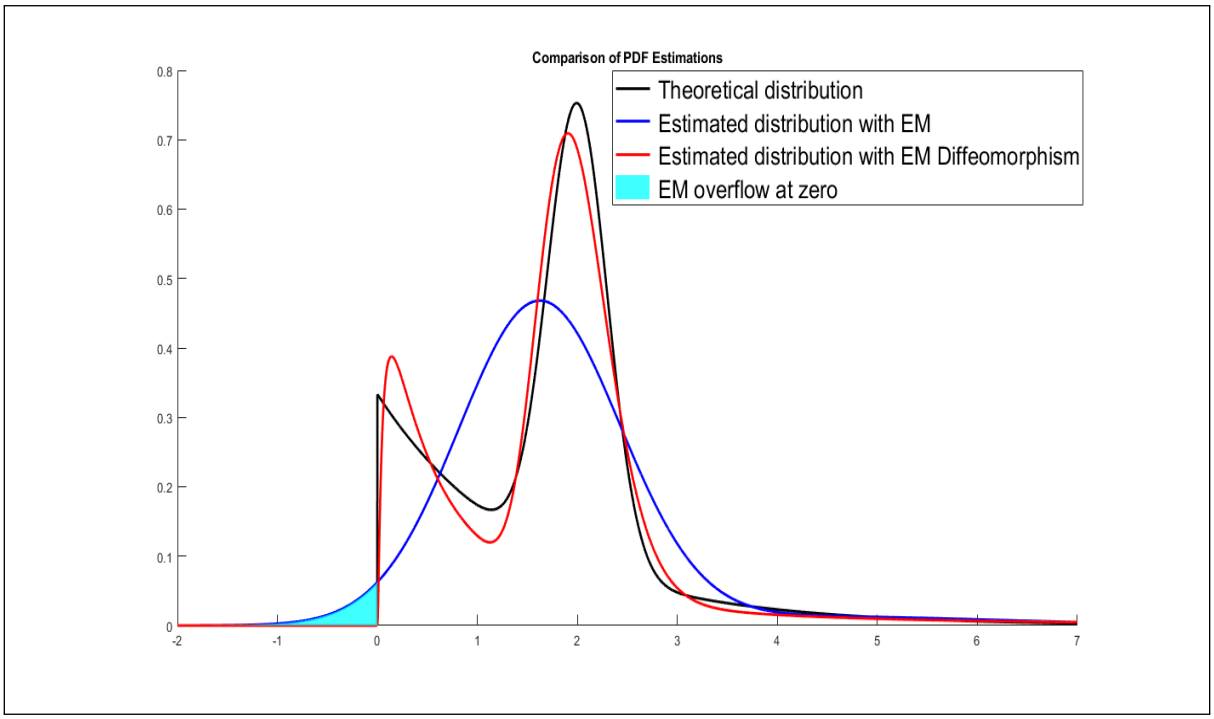


Figure 2.2: Comparison of the estimated bimodal mixture distribution by EM and EMD with the theoretical mixture.

Figure 2.2 displays the probability density functions (PDFs) of the theoretical data along with the estimates from both EM and EMD. The EMD algorithm effectively captured the two distinct modes of the mixture, aligning closely with the theoretical mixture distribution. In contrast, the EM algorithm failed to detect the bimodality, identifying only a single mode. These visual results are supported by the MISE values, as shown in Table 2.1, which highlight the superior performance of EMD over EM.

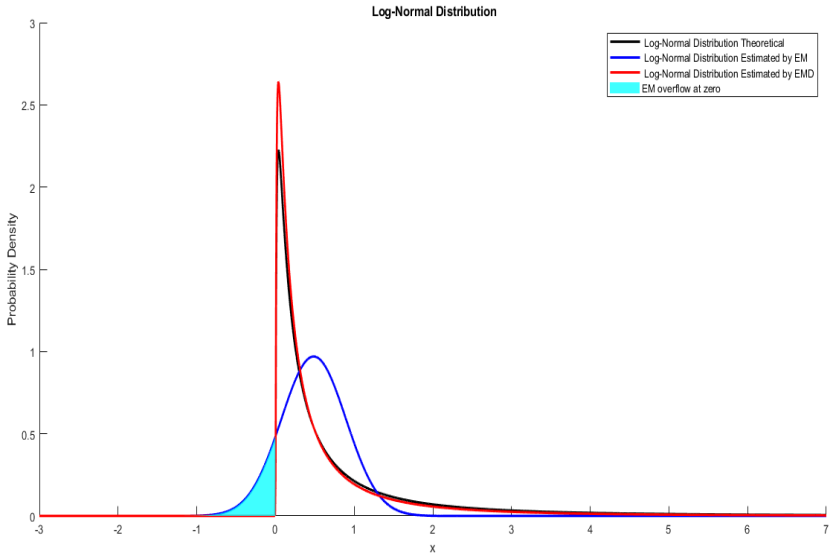
Table 2.1: Integrated Mean Squared Error (MISE) for the Bimodal Mixture Model

	EM	**EMD**
MISE	0.01031	9.8934e-04

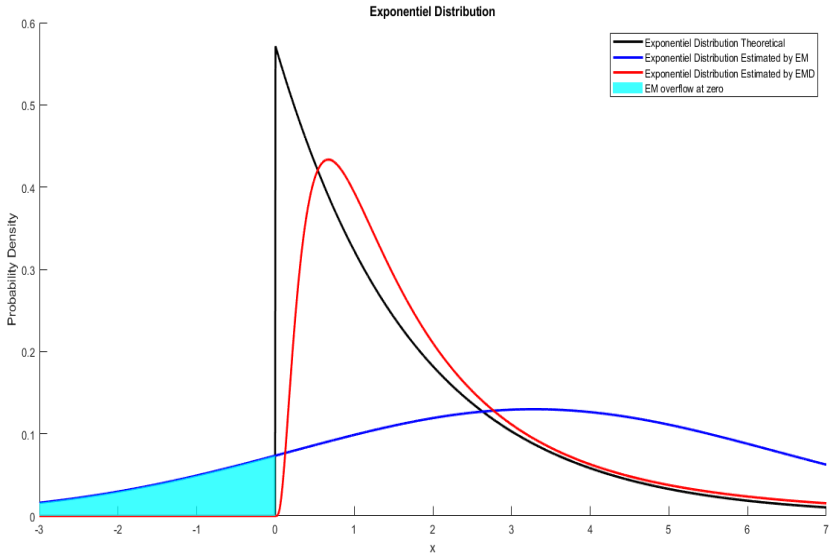
2.3.1.2 Unimodal Mixture Model

In this second scenario, we modeled a mixture comprising two distributions: a Log-Normal distribution with parameters (-1, 1.5) and an exponential distribution with parameter (1.75). Once again, the EMD algorithm accurately identified the mixture components, closely matching the true distributions, while the EM algorithm demonstrated poorer performance. In both cases,

the EM estimates did not preserve the positivity of the data, leading to incorrect estimates in the negative range, whereas EMD maintained the strictly positive nature of the data.



(a) Log-Normal distribution component



(b) Exponential distribution component

Figure 2.3: Visualizing the Unimodal mixture individual distribution components

Using the parameters estimated by EM and EMD, including the mixture weights, we reconstructed the unimodal mixture and compared it to the theoretical distribution. This comparison is shown in Figure 2.4

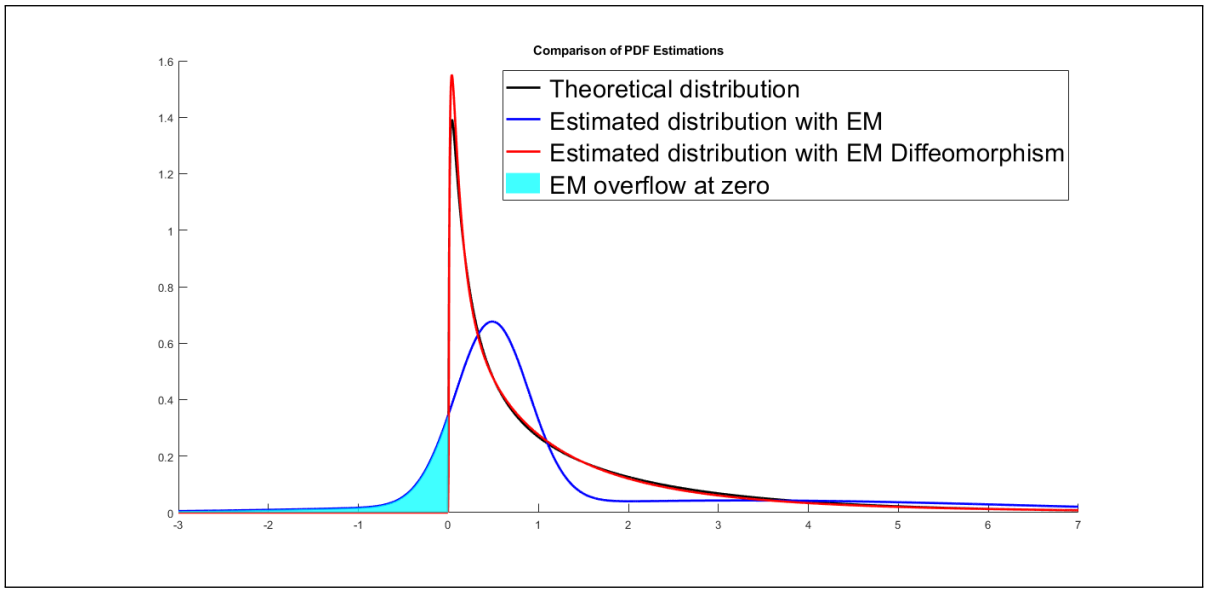


Figure 2.4: Comparison of the original unimodal mixture with the estimates from EM and EMD.

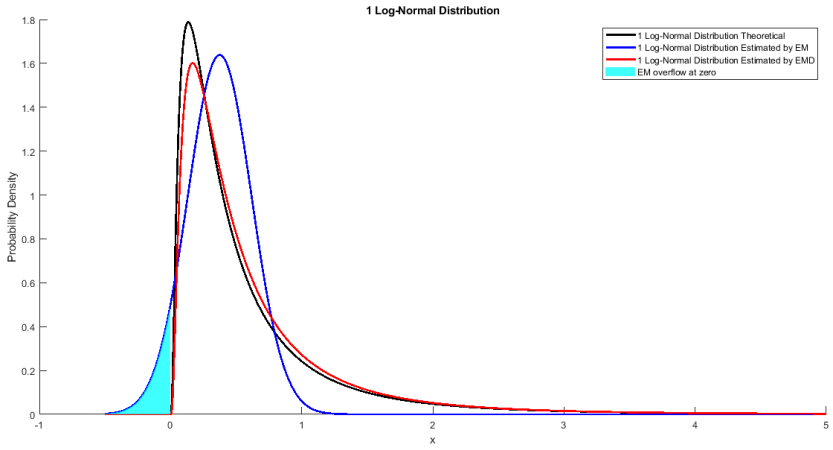
As illustrated in figure 2.4 the EMD algorithm provided a close approximation to the theoretical unimodal mixture, preserving data positivity. In contrast, the EM algorithm failed to respect this constraint, producing negative values. The superiority of the EMD algorithm is further confirmed by the MISE metric presented in Table 2.2, where EMD outperforms EM in accurately estimating the data.

Table 2.2: Integrated Mean Squared Error (MISE) for the Unimodal Mixture Model

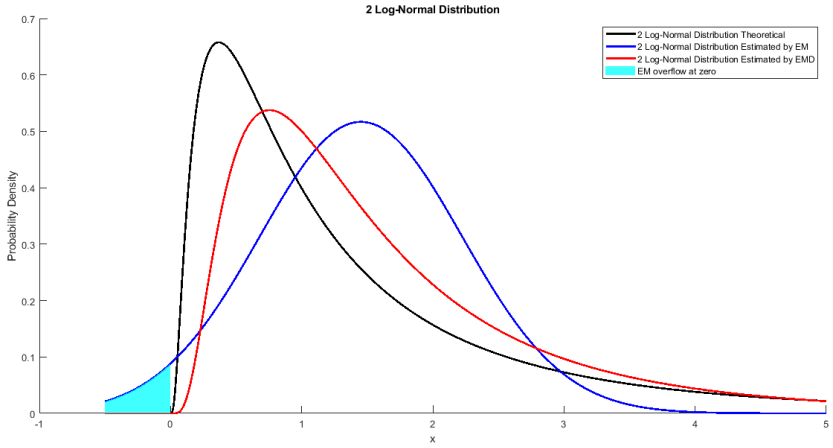
	EM	**EMD**
MISE	0.01605	1.9061 e-04

2.3.2 Three-Component Mixture Model

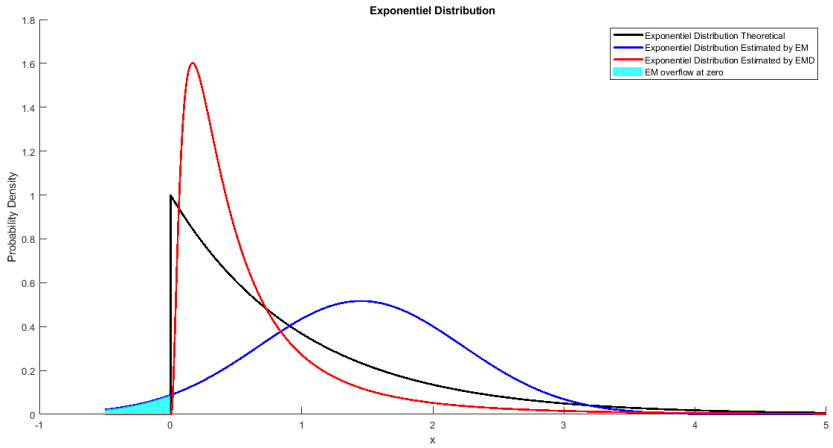
In this scenario, we applied our new algorithm to a mixture of three components: a log-normal distribution with parameters (0, 1), a second log-normal distribution with parameters (-1, 1), and an exponential distribution with parameter (1).



(a) The first Log-Normal distribution



(b) The second Log-Normal distribution



(c) The exponential distribution

Figure 2.5: Visualization of the individual components in the three-component mixture model.

Using the estimated parameters from both the EM and EMD algorithms, we compared the reconstructed mixture distributions to the original data. The results are shown in Figure 2.6.

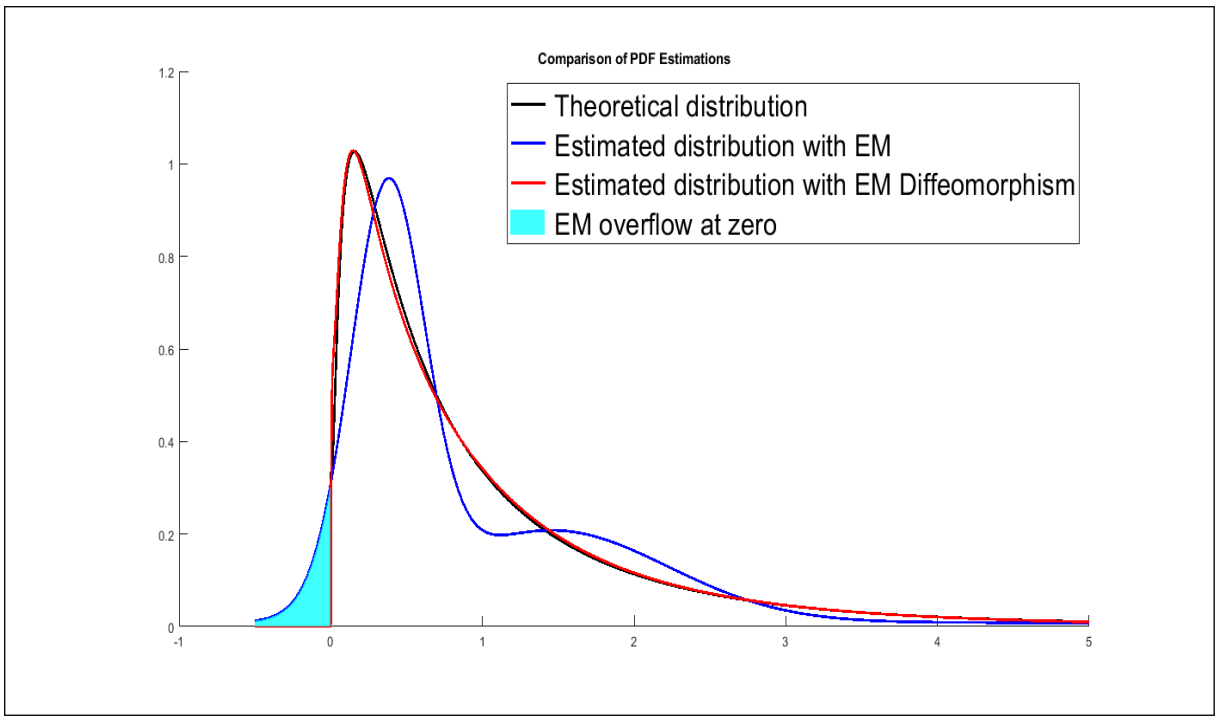


Figure 2.6: Comparison of the simulated data (original mixture of two log-normal and exponential distributions) with estimates from EM and EMD.

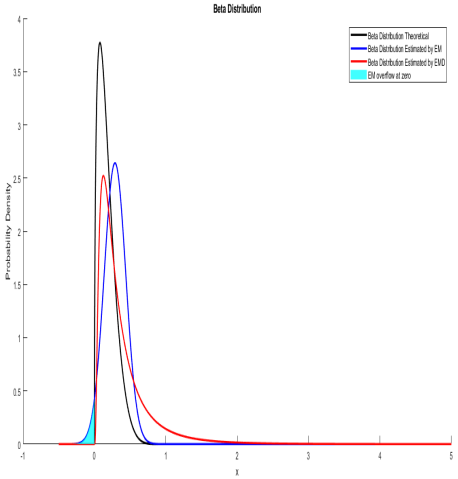
As seen in Figure 2.6, the EMD algorithm closely matches the theoretical mixture, preserving the strict positivity of the data. In contrast, the EM algorithm fails to maintain this constraint, showing overflow near zero. These visual results are further validated by the MISE metric, as presented in Table 2.3.

Table 2.3: Integrated Mean Squared Error (MISE) for the three-component mixture of log-normal and exponential distributions

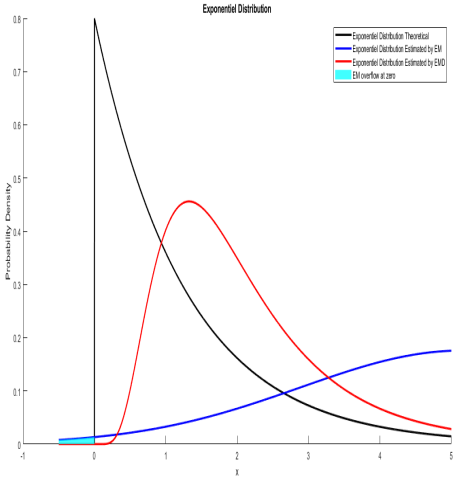
	EM	**EMD**
MISE	0.0093	2.5324 e-04

2.3.3 Five-Component Mixture Model

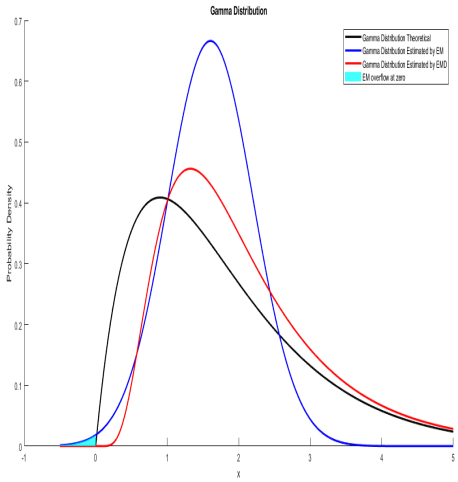
In this section, we tested our new algorithm on a more complex mixture model consisting of five distinct distributions: an Exponential distribution (1.25), a Gamma distribution (2, 0.9), a Beta distribution (1.5, 7), a Log-Normal distribution (-1, 0.1), and a Normal distribution (1, 0.1).



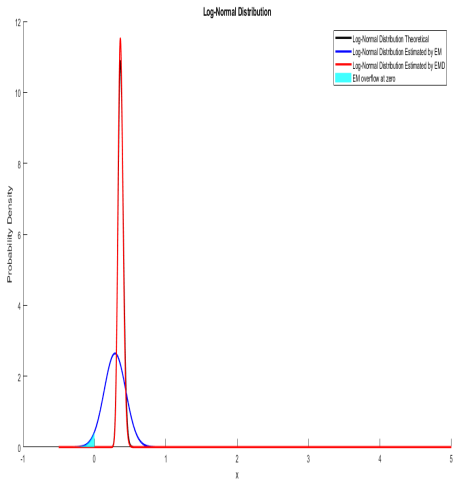
(a) The Beta distribution



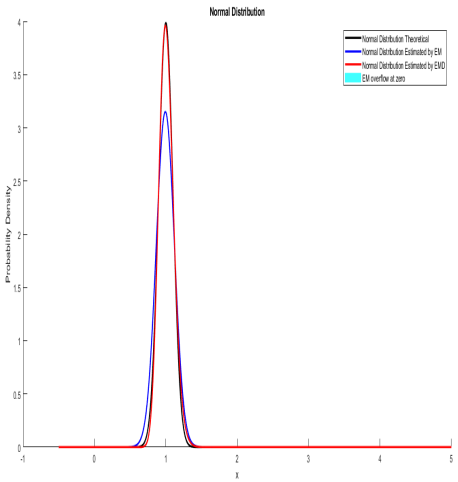
(b) The Exponential distribution



(c) The Gamma distribution



(d) The Log-Normal distribution



(e) The Normal distribution

Figure 2.7: Visualization of the individual components in the five-component mixture model.

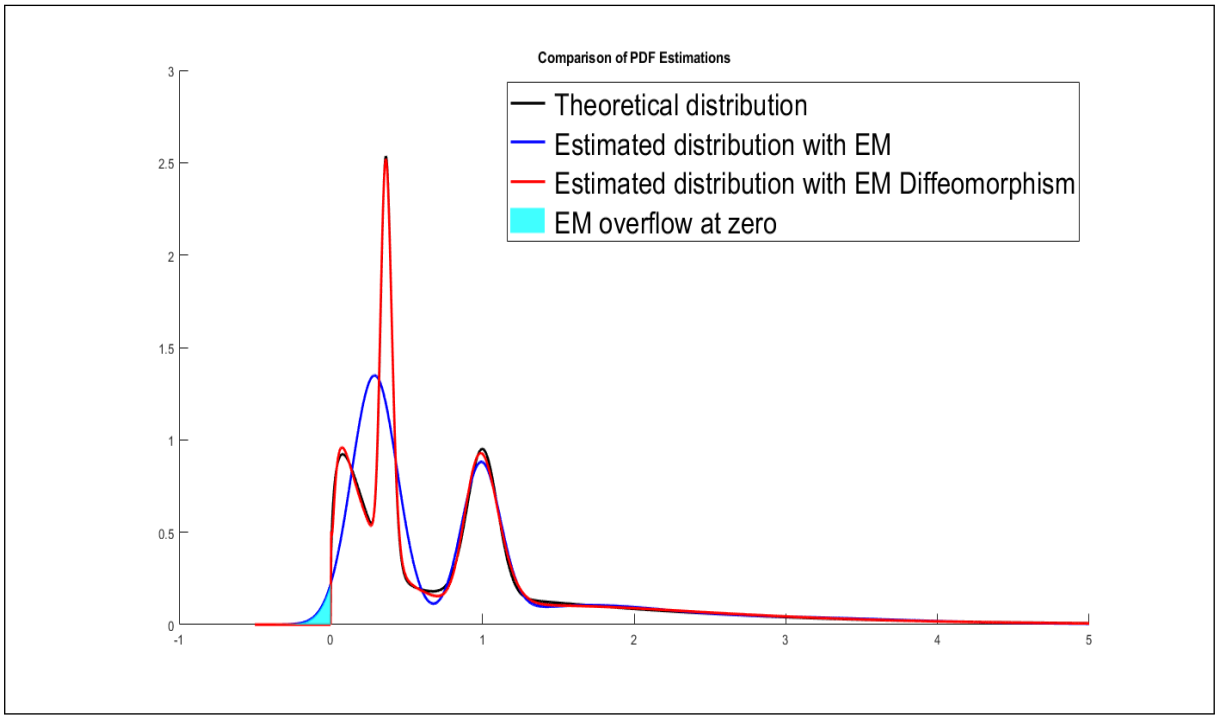


Figure 2.8: Simulated data with a mixture of five distributions, and estimates from EM and EMD.

In this more complex mixture model, the EMD algorithm provides an accurate approximation of the theoretical mixture, preserving the positivity constraint. In contrast, the EM algorithm produces estimates that include negative values, as shown in Figure 2.8. The MISE metric in Table 2.4 confirms the superior performance of EMD.

Table 2.4: Integrated Mean Squared Error (MISE) for the five-component mixture model

	EM	**EMD**
MISE	0.0309	0.1433 e-04

2.3.4 Discussion

The experiments conducted on simulated datasets produced encouraging outcomes, especially when the data points clustered near the boundary (in our simulations, this boundary was zero, as all the data values were positive). In these cases, the EMD algorithm successfully respected the inherent limits of the data, avoiding the overflow problems that were observed with the classic EM algorithm.

2.4 Application 1 : Ultrasound Image Segmentation

For our real data experiment, we applied our new algorithm to ultrasound images. This choice was driven by the unique characteristics of ultrasound images, which present specific challenges well-suited to the capabilities of our algorithm.

Ultrasound images are known for their particular data distribution, characterized by strictly positive pixel values ranging from 0 to 255, with a significant concentration of values near zero. This distribution is illustrated in Figure (2.9), which shows the histogram of a sample of an ultrasound image. The concentration of pixel values near zero, combined with the broad range of values, creates a scenario where traditional segmentation methods often fall short.

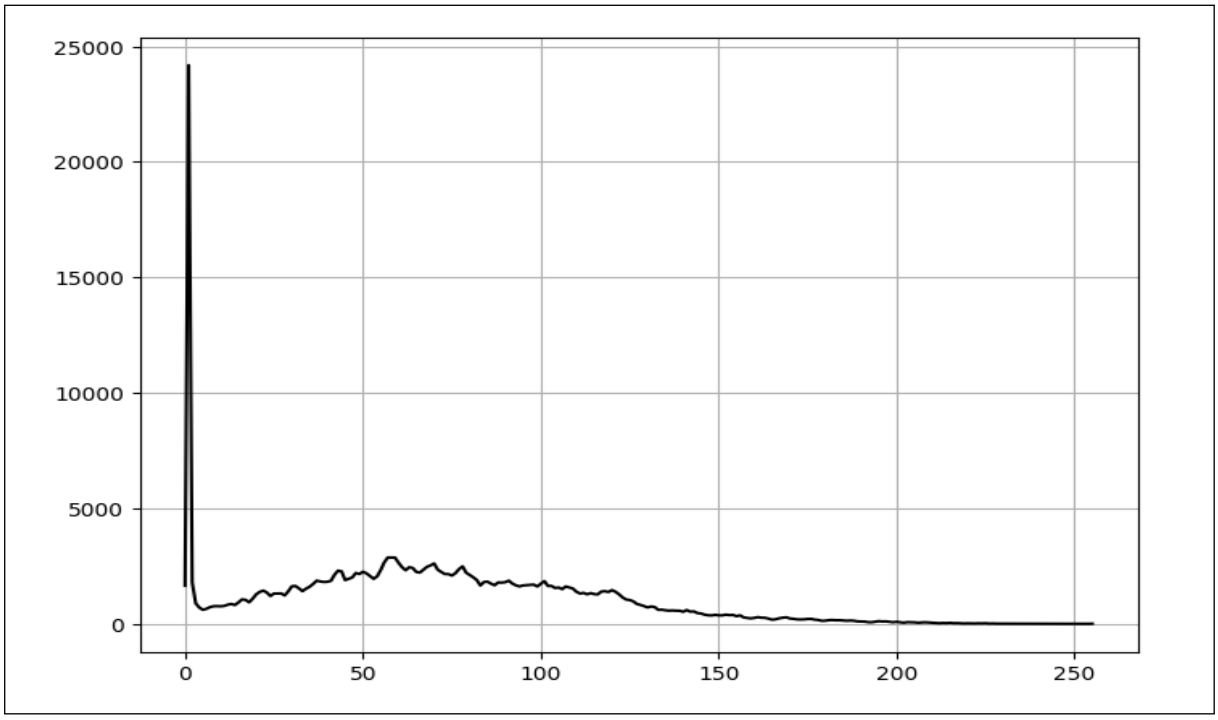


Figure 2.9: Histogram of an Ultrasound Image illustrating the pixel intensity distribution.

Segmentation of ultrasound images poses several challenges:

- **Low Contrast**: Ultrasound images typically have low contrast, which makes it difficult to distinguish between different tissues or structures. The lack of contrast reduces the ability of traditional segmentation methods to effectively differentiate between regions of interest [Tay et al., 2010].

- **Speckle Noise:** Ultrasound images are subjected to high levels of speckle noise, which can obscure the boundaries of structures and lead to inaccuracies in segmentation. Speckle noise results from the interference of ultrasound waves scattered by tissues and can significantly impact image quality [Mateo and Fernández-Caballero, 2009].

- **Complex Structures**: The structures within ultrasound images can be complex and varied, often requiring advanced algorithms capable of handling intricate patterns and textures.

The segmentation of ultrasound images has been extensively studied in the literature [Shamshad et al., 2023], with various algorithms proposed to address the challenges posed by this imaging technology. These approaches vary from traditional methods to more advanced methods involving machine learning and deep learning.

Among the traditional methods, we may cite threshold-based methods, which may be considered the simplest and earliest methods in ultrasound image segmentation, as they define a value (the threshold) and transform the grayscale image into a binary one according to that threshold ([Horsch et al., 2001] [Drukker et al., 2002] [Rodrigues and Giraldi, 2011]). The Clustering-based methods group pixels or regions with similar intensity or texture characteristics into clusters, enabling the identification of distinct tissue types or structures within the image without requiring prior labeling or supervision [Moon et al., 2014] . Another popular methods are the Graph-based methods. they represent the image as a graph, where pixels or groups of pixels serve as nodes, and edges between them encode (dis)similarity. Segmentation is achieved by partitioning the graph into meaningful regions based on criteria such as edge weights or cuts, often optimizing a cost function that balances boundary smoothness and region homogeneity ([Huang et al., 2012] [Huang et al., 2014] [Chang et al., 2015]).

The active contour models have been widely used for ultrasound image segmentation, employing energy minimization to capture object boundaries ([Fang et al., 2018] [Liu et al., 2010] [Fang et al., 2022]). However, these models often struggle with the low contrast and speckle noise present in ultrasound images. Another approach is the statistical shape models, which incorporate prior knowledge about the expected shape of structures within the image [Shen et al.,

2003]. While effective, these methods often require extensive preprocessing and fine-tuning to achieve optimal results.

The Canny edge detection algorithm [Canny, 1986], is also commonly used. Known for its simplicity and efficiency in detecting edges, Canny's algorithm is effective at identifying the boundaries of structures ([Hamou and El-Sakka, 2004] [Nikolic et al., 2016] [Zheng et al., 2015] but can struggle with the speckle noise and low contrast typical of ultrasound images, leading to potential inaccuracies in segmentation. The neuronal Networks are also widely used in ultrasound image segmentation, especially the watershed transformation which is a region-based approach that treats the image as a topographic surface. The regions of low intensity represent valleys. It segments the image by flooding these valleys from selected markers, identifying boundaries at the points where water from different valleys meets, effectively delineating structures and boundaries within the ultrasound image ([Huang and Chen, 2004] [Gómez et al., 2010] [Lo et al., 2014]).

In recent years, deep learning-based methods have gained significant attention due to their ability to learn complex patterns directly from data. Convolutional neural networks (CNNs), in particular, have shown great promise in improving segmentation accuracy ([Ma et al., 2017] [Alsinan et al., 2019] [Milletari et al., 2017]). However, challenges such as the reliance on large annotated datasets and the potential for overfitting remain significant in applying deep learning to ultrasound image segmentation.

One widely used deep learning approach is the U-Net architecture [Ronneberger et al., 2015] a type of CNN specifically designed for image segmentation tasks. U-Net has been adopted for medical image segmentation due to its ability to capture both spatial and contextual information. U-Net has demonstrated significant improvements in segmentation accuracy across various medical imaging modalities, including ultrasound [Luan et al., 2020] [Li et al., 2019] [Ansari et al., 2023][Tong et al., 2021]. However, its performance also relies on the existence of a large number of annotated data.

In the following section, we will introduce the two reference algorithmsU-Net and Canny edge detectionthat have been selected to benchmark our proposed method. These algorithms

are well-established in the literature and will provide a robust comparison for evaluating the effectiveness of our approach.

2.4.1 Reference Algorithms

To evaluate the effectiveness of EMD, we compared it with two algorithms used in ultrasound image segmentation literature. Initially, we assessed the performance of EMD against the traditional EM algorithm. Next, we compared EMD with a classic method, the active contour model, and specifically we used the Canny edge detection algorithm. Finally, we implemented a deep learning approach the U-Net architecture, a popular model in image segmentation to evaluate EMD.

2.4.1.1 U-Net

U-Net, a fully convolutional neural network for biomedical image segmentation, was introduced by Ronneberger et al. in 2015 [Ronneberger et al., 2015]. It is characterized by its U-shape, which consists of a symmetric encoder (contracting path) and a decoder (expanding path). This structure allows for precise localization and classification of pixels and effectively captures both local and global image features. While the encoder progressively extracts features through convolutional and pooling layers, the decoder reconstructs segmentation masks using upsampling and convolutional layers. Skip connections are essential to the success of U-Net as they integrate low-level details from the encoder into the decoder, leading to more accurate segmentation results. This design has made U-Net a basic model for many state-of-the-art ultrasound image segmentation algorithms ([Bal-Ghaoui et al., 2023], [Zhou et al., 2019],[Weng et al., 2019]...).

In our study, we trained a U-Net model on the CT2USforKidneySeg dataset [Song et al., 2022], which includes 4,568 synthesized ultrasound images generated from computed tomography (CT) scans, each paired with ground truth masks.

Below is a description of the U-Net architecture implemented in this study:

1. **Input Layer :** The model accepts input images of size 128x128 pixels with a single channel (grayscale).
2. **Encoder Path (Contracting Path) :** The encoding path consists of four levels of convolutional layer, each followed by max-pooling for down-sampling.

   Each convolutional block consists of two convolutional layers to maintain spatial dimensions. The number of filters doubles at each subsequent level, starting from 64 and going up to 1024.

   After each layer, max-pooling reduces the spatial dimensions by half, allowing the network to capture contextual information.

   - **Layer 1 :** Two convolutional layers with 64 filters each, using a 3x3 kernel, ReLU activation, and same padding.
     MaxPooling2D with a pool size of 2x2 is applied to reduce spatial dimensions and capture features.
   - **Layer 2 :** Two convolutional layers with 128 filters each, using a 3x3 kernel, ReLU activation, and same padding.
     MaxPooling2D with a pool size of 2x2 is applied.
   - **Layer 3 :** Two convolutional layers with 256 filters each, using a 3x3 kernel, ReLU activation, and same padding.
     MaxPooling2D with a pool size of 2x2 is applied.
   - **Layer 4 :** Two convolutional layers with 512 filters each, using a 3*3 kernel, ReLU activation, and same padding.
     MaxPooling2D with a pool size of 2x2 is applied.

3. **Bottleneck Layer :** The bottleneck is the deepest part of the U-Net, with two convolutional layers having 1024 filters, a 3x3 kernel size, and ReLU activation. This part captures the most abstract and context-rich features from the input image.
4. **Decoder Path (Expanding Path) :** The decoding path mirrors the encoding path, but with up-sampling to reconstruct the spatial resolution.

   At each level, the feature map from the corresponding encoding layer is concatenated with the up-sampled output. This helps in recovering fine-grained details lost during down-sampling. Each layer in the decoding path consists of a transposed convolutional layer (to up-sample the feature map) followed by two convolutional layers.

- **Layer 5 :** Conv2DTranspose with 512 filters and a kernel size of 2x2, followed by concatenation with the corresponding encoder layer (Layer 4).
  Two convolutional layers with 512 filters each, using a 3x3 kernel, ReLU activation, and same padding.
- **Layer 6 :** Conv2DTranspose with 256 filters and a kernel size of 2x2, followed by concatenation with the corresponding encoder layer (Layer 3).
  Two convolutional layers with 256 filters each, using a 3x3 kernel, ReLU activation, and same padding.
- **Layer 7 :** Conv2DTranspose with 128 filters and a kernel size of 2x2, followed by concatenation with the corresponding encoder layer (Layer 2).
  Two convolutional layers with 128 filters each, using a 3x3 kernel, ReLU activation, and same padding.
- **Layer 8 :** Conv2DTranspose with 64 filters and a kernel size of 2x2, followed by concatenation with the corresponding encoder layer (Layer 1).
  Two convolutional layers with 64 filters each, using a 3x3 kernel, ReLU activation, and same padding.

5. **Output Layer :** A final convolutional layer with 1 filter, using a 1x1 kernel, and a sigmoid activation function to produce the binary segmentation mask.
6. **Compilation :** The model is compiled with the Adam optimizer, binary cross-entropy loss function, and accuracy as the evaluation metric.

The architecture's components and their interrelationships are illustrated in the figure 2.10.

This architecture is known for its efficiency in segmenting images due to its use of skip connections, which combine high-level features from the encoder path with low-level features from the decoder path. This approach helps in retaining fine-grained details crucial for accurate segmentation.

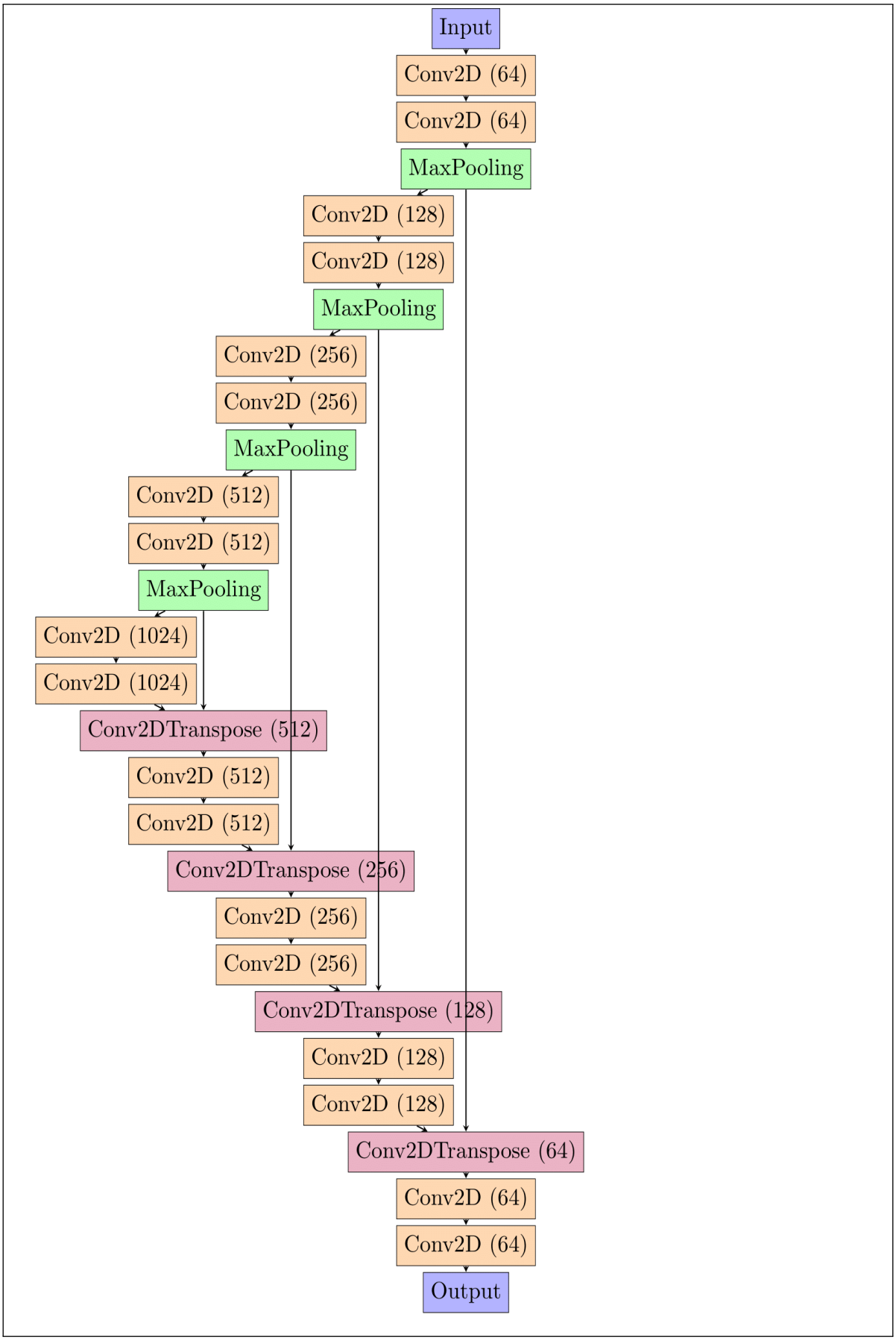


Figure 2.10: U-Net architecture diagram

2.4.1.2 Edge Detection

Edge detection is a traditional method for image segmentation that focuses on identifying the boundaries between different regions in an image [Mohana Priya and Mohamed Fathimal, 2023]. The Canny edge detection algorithm is among the most popular edge detection techniques [Thomas and Duela, 2024]. This algorithm is known for its robustness and precision in identifying edges.

The Canny edge detector operates through a sequence of stages designed to ensure accurate edge identification. First, it applies a Gaussian filter to smooth the image, which helps reduce noise and avoid false edge discovery. Next, it uses gradient operators to compute the intensity gradient of the image, highlighting areas where rapid changes in intensity occur, typically indicating edges.

After calculating the gradient, the algorithm performs non-maximum suppression to thin out the edges. This step ensures that only the most significant edge pixels are retained, providing a cleaner edge map. Finally, the Canny algorithm uses a double thresholding process to identify and classify edges as strong, weak, or non-edges. Strong edges are those that are clearly defined, while weak edges are considered only if they are connected to strong edges, helping to refine and complete the edge detection.

2.4.2 Evaluation Metrics

To evaluate the performance of the segmentation algorithms, we followed the guidelines and recommendations of [Müller et al., 2022], [Taha and Hanbury, 2015] and [Popovic et al., 2007]. So we picked confusion matrix-based metrics and one distance-based metric. We used the following metrics:

2.4.2.1 Intersection over Union (IoU)

Intersection over Union (IoU), also known as the Jaccard Index, is a metric used to measure the overlap between two sets, typically between a predicted segmentation and the ground truth in

image processing tasks. It is defined as the ratio of the intersection of the predicted and ground truth areas to their union, providing a clear indication of how well the predicted segmentation matches the actual object in the image, and is calculated as follows :

$$\text{IoU} = \frac{\text{Intersection Area}}{\text{Union Area}}$$

2.4.2.2 Dice Coefficient

The Dice Coefficient, also known as the Sørensen-Dice coefficient and Sørensen index, is a metric that measures the similarity between two sets. The Dice Coefficient is calculated as twice the area of overlap between the predicted and ground truth regions divided by the sum of the areas of the two regions. This metric is highly sensitive to the presence of common elements. It is calculated using the following formula :

$$\text{Dice Coefficient} = \frac{2 * \text{Intersection Area}}{\text{Area of Predicted Mask + Area of Ground Truth Mask}}$$

2.4.2.3 Precision

It is a performance metric that represents the proportion of true positive predictions among all positive predictions made by the model. This precision is calculated according to the following well-known formula:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives + False Positives}}$$

2.4.2.4 Recall

It is a performance metric that represents the proportion of true positive predictions among all actual positive instances. It measures the proportion of actual positive cases correctly identified by the model. It is given by the following formula :

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives + False Negatives}}$$

2.4.2.5 The Symmetric Hausdorff Distance (HD)

While confusion matrix-based metrics provide a general sense of segmentation performance, they may fail to capture boundary inaccuracies, which are particularly critical in medical applications. To address this limitation, we use a distance-based metrics. Specifically, we utilize the Symmetric Hausdorff Distance (HD), which offers a robust measure of boundary alignment between the segmented region and the ground truth.

HD is commonly used to assess the quality of segmentations in image processing. It is defined as the maximum of the directed Hausdorff distances between the two sets. Specifically, for two point sets A and B, the directed Hausdorff distance from A to B is the maximum of the minimum distances between any point in A and all points in B, and vice versa. The symmetric Hausdorff distance is calculated as follows:

$$d_H(A, B) = \max\left(\sup_{a\in A}\inf_{b\in B}\|a-b\|, \sup_{b\in B}\inf_{a\in A}\|a-b\|\right), \tag{2.1}$$

where $\|a-b\|$ represents the distance between points a and b. This measure provides a robust assessment of similarity between the sets by considering both directions, thus offering a balanced evaluation of discrepancies between them.

By incorporating the Symmetric Hausdorff Distance, we obtain a more clinically relevant assessment of segmentation quality, reflecting the spatial accuracy of the boundaries in ultrasound images.

2.4.3 dataset

We evaluated the segmentation algorithms using a dataset of ultrasound images and their corresponding ground truth segmentation masks. The dataset employed was the Breast Ultrasound Images Dataset (BUSI dataset) [Al-Dhabyani et al., 2020]), which includes breast ultrasound

images from women aged 25 to 75 years, collected in 2018. This dataset includes 600 female patients, yielding a total of 780 images with their ground truth masks divided into three categories : benign (437 images), malignant (210 images), and normal (133 images). It is one of the most widely used and reliable resources for ultrasound image segmentation [Abhisheka et al., 2023], [Yuan et al., 2024], [Chen et al., 2024], [Lawal and Yi, 2024]...

2.4.4 Results

This section presents the results of our segmentation experiments, evaluating the performance of four algorithms: U-Net, EM, EMD, and Canny edge detection. Each algorithm was tested on three categories of ultrasound images: benign, malignant, and normal. For each category, we randomly selected representative images to illustrate segmentation performance, displaying the original image, its corresponding ground truth mask, and the results from the four algorithms. Additionally, we provide a detailed evaluation using metrics such as accuracy, recall, precision, Intersection over Union (IoU), and Dice coefficient.

2.4.4.1 Benign Folder

The figure 2.11 represents the first step of our proposed algorithm, where we employed the elbow method to determine the optimal number of clusters (k=4). This step provided the necessary initialization for K-means and EMD.

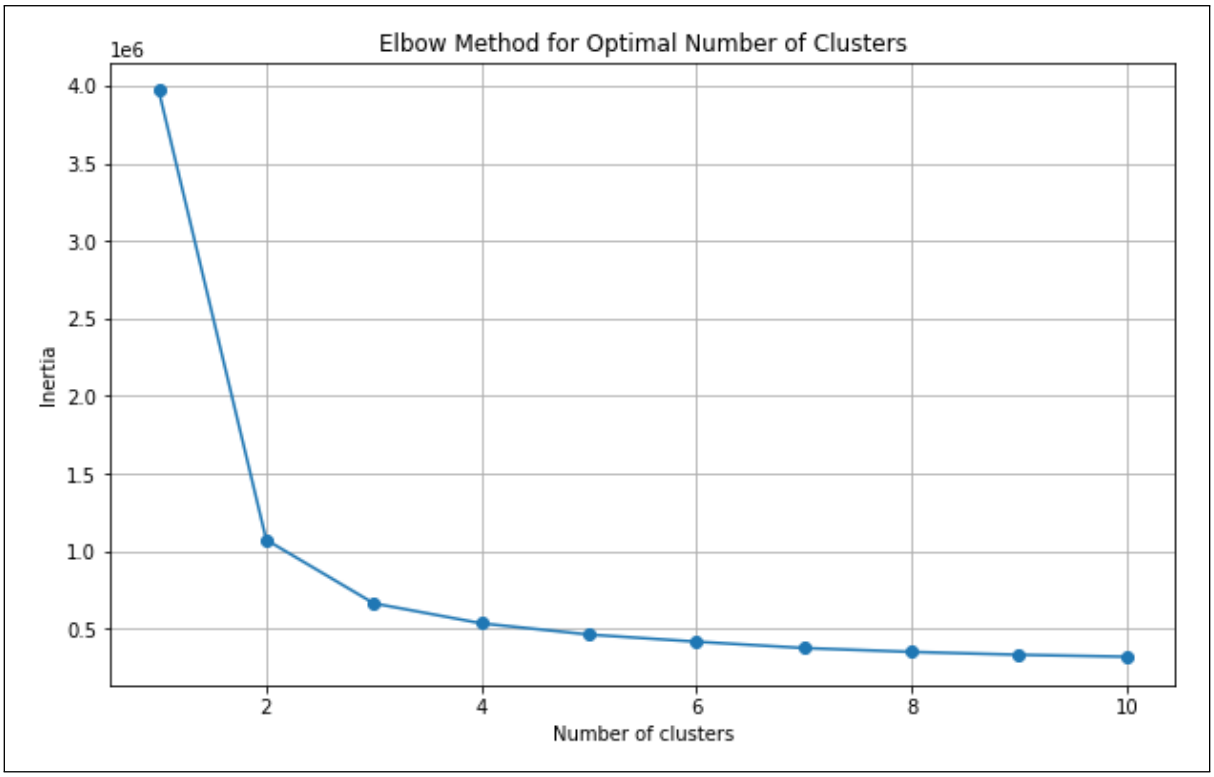


Figure 2.11: Elbow method applied to an ultrasound image.

Figure 2.12 illustrates the segmentation results for six randomly selected images from the benign folder, showcasing the original image, the ground truth mask, and the segmentation outcomes from the four algorithms.

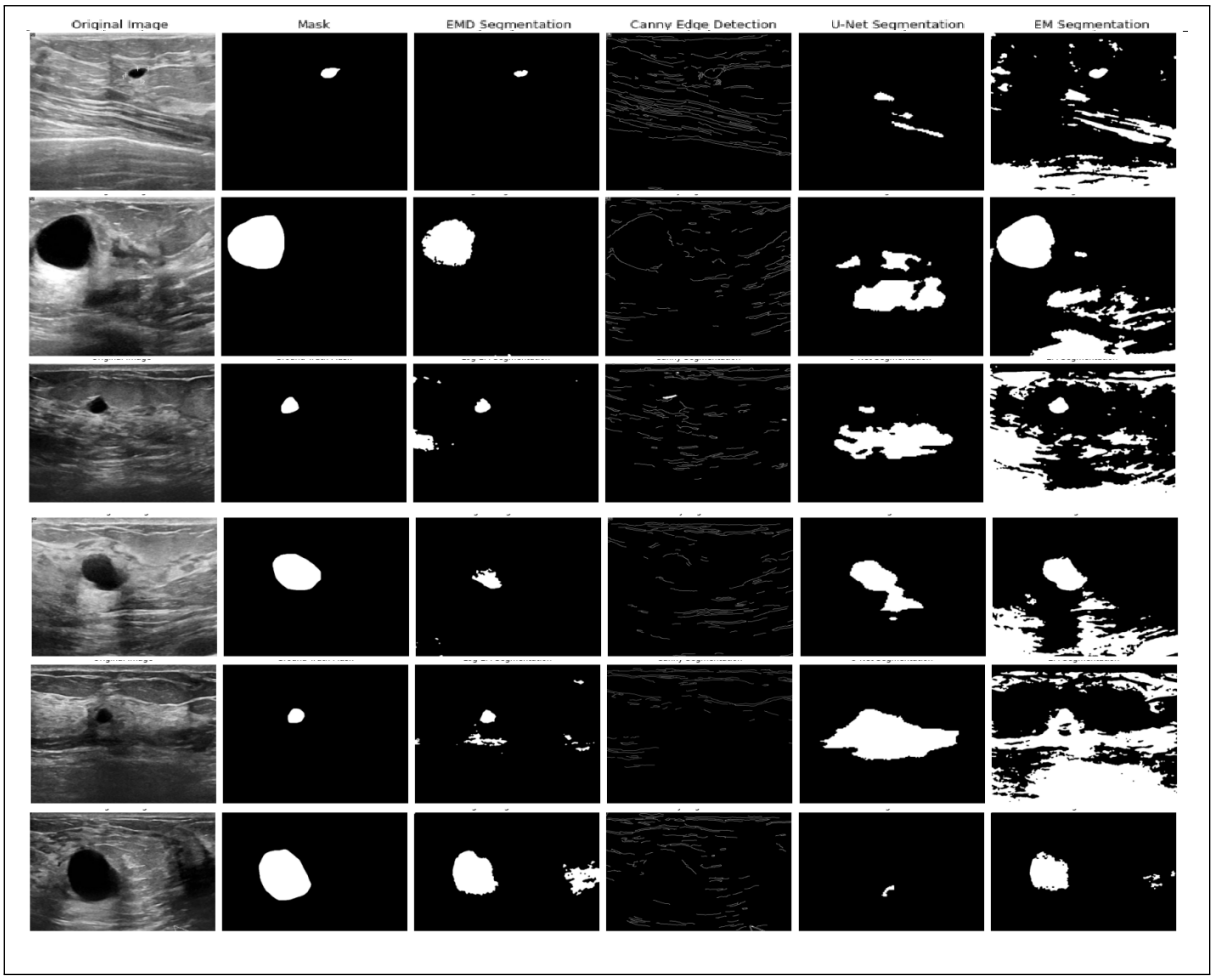


Figure 2.12: Segmentation results for six sample images from the benign folder: (a) Original image, (b) Ground truth mask, (c) EMD segmentation, (d) Canny segmentation, (e) U-Net segmentation, (f) EM segmentation.

The segmentation outcomes for the benign folder reveal that the EMD algorithm achieved the best overall performance, with its segmentation results closely matching the region of interest's shape. Both EM and Canny failed to delineate the boundaries accurately, particularly the EM, which detected significant noise in the images. U-Nets performance was inconsistent, sometimes failing to detect the entire region of interest, and at times misclassifying other regions as tumors. However, in certain images, U-Net performed excellently, detecting the tumor perfectly.

Table 2.5 summarizes the evaluation metrics for each algorithm in the benign folder.

Table 2.5: Evaluation metrics for benign Folder

Algorithm	Dice	IoU	Precision	Recall	Symmetric Hausdorff Distance
EMD	0.8514	0.7412	0.6401	0.7401	37.5065
Canny	0.4356	0.606	0.6848	0.8176	75.8481
U-Net	0.7903	0.7288	0.4631	0.64	38.3610
EM	0.830	0.709	0.3315	0.3330	76.6738

As indicated in Table 2.5, EMD consistently demonstrates superior performance, achieving the highest Dice coefficient (0.8514) and Intersection over Union (IoU) (0.7412), indicating better accuracy and overlap in segmenting the region of interest. Additionally, it shows the lowest Symmetric Hausdorff Distance (37.5065), suggesting better boundary alignment. On the other hand, while the Canny algorithm has the highest recall (0.8176), its poor Dice coefficient (0.4356) and IoU (0.6060) highlight its inability to precisely delineate the region, detecting excessive noise instead. U-Net achieves satisfactory results, performing comparably to EMD, with a Dice coefficient of 0.7903 and IoU of 0.7288. The EM algorithm showed performance similar to U-Net but detected more noise.

2.4.4.2 Malignant Folder

Figure 2.13 displays the segmentation results for six sample images from the malignant folder. Similar to the benign folder, the figure shows the original image, the ground truth mask, and the segmentation outcomes from the four algorithms.

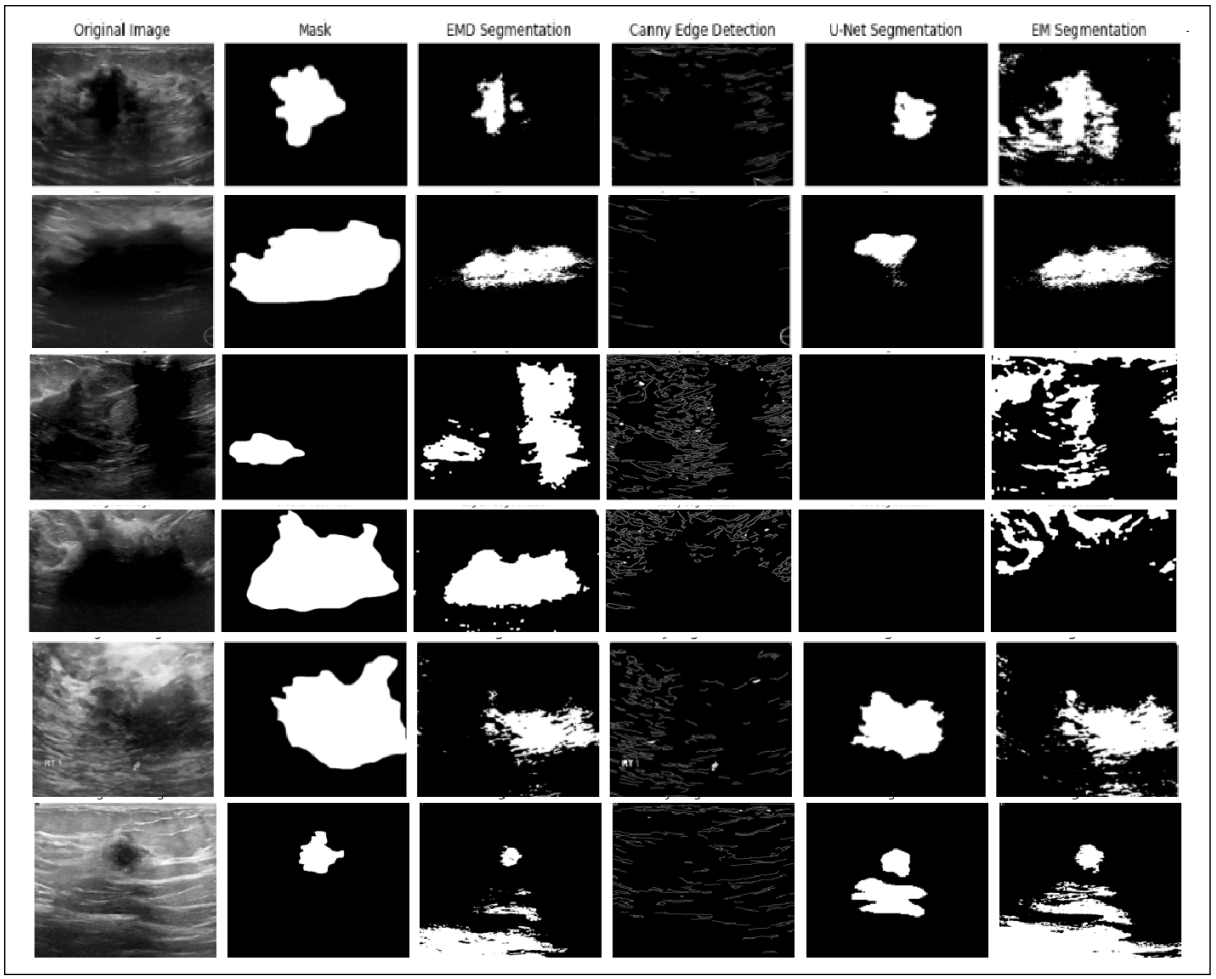


Figure 2.13: Segmentation results for six sample images from the malignant folder: (a) Original image,
(b) Ground truth mask, (c) EMD segmentation, (d) Canny segmentation, (e) U-Net segmentation, (f) EM segmentation.

In the malignant folder, the tumor regions exhibited less smooth boundaries and irregular shapes compared to those in the benign folder. This impacted the performance of the algorithms, making it more difficult to accurately capture fine details. The EMD algorithm continued to outperform the other methods, producing a segmentation mask that closely matched the ground truth with minimal noise. However, the irregular tumor boundaries posed challenges for all algorithms.

Table 2.6 provides the evaluation metrics for each algorithm applied to the malignant folder.

Table 2.6: Evaluation metrics for malignant Folder

Algorithm	Dice	IoU	Precision	Recall	Symmetric Hausdorff Distance
EMD	0.7378	0.5845	0.3988	0.6582	53.15072
Canny	0.4960	0.3381	0.5082	0.4583	84.4835
U-Net	0.6444	0.4227	0.2924	0.3133	59.8080
EM	0.5579	0.3869	0.3248	0.2180	81.7097

As indicated in Table 2.6, the performance metrics reveal notable differences among the algorithms. The EMD algorithm achieved the highest Dice score of 0.7378 and an IoU of 0.5845, indicating it provided the closest match to the ground truth while maintaining a reasonable balance between precision and recall. Additionally, the EMD algorithm had the lowest Symmetric Hausdorff Distance of 53.15072, suggesting it was the most effective at accurately delineating tumor boundaries compared to the other methods. The Canny algorithm, although demonstrating a higher precision of 0.5082, recorded the lowest Dice and IoU scores, reflecting its difficulties in accurately capturing the tumor region. U-Net achieved a moderate Dice score of 0.6444 and an IoU of 0.4227, but also exhibited a high Symmetric Hausdorff Distance of 59.8080, indicating challenges with precise boundary delineation. The EM algorithm's performance was somewhat comparable to that of U-Net in terms of precision and recall, but its metrics suggest less effective segmentation accuracy and boundary definition. Overall, the results demonstrate that while EMD is the most effective in capturing the tumor shape with the best boundary precision, all algorithms face significant challenges regarding tumor boundary accuracy in the malignant folder.

2.4.4.3 Normal Folder

Figure 2.14 demonstrates the segmentation results for six sample images from the normal folder. The original image, the ground truth mask, and the segmentation results from U-Net, EM, EMD, and Canny are displayed.

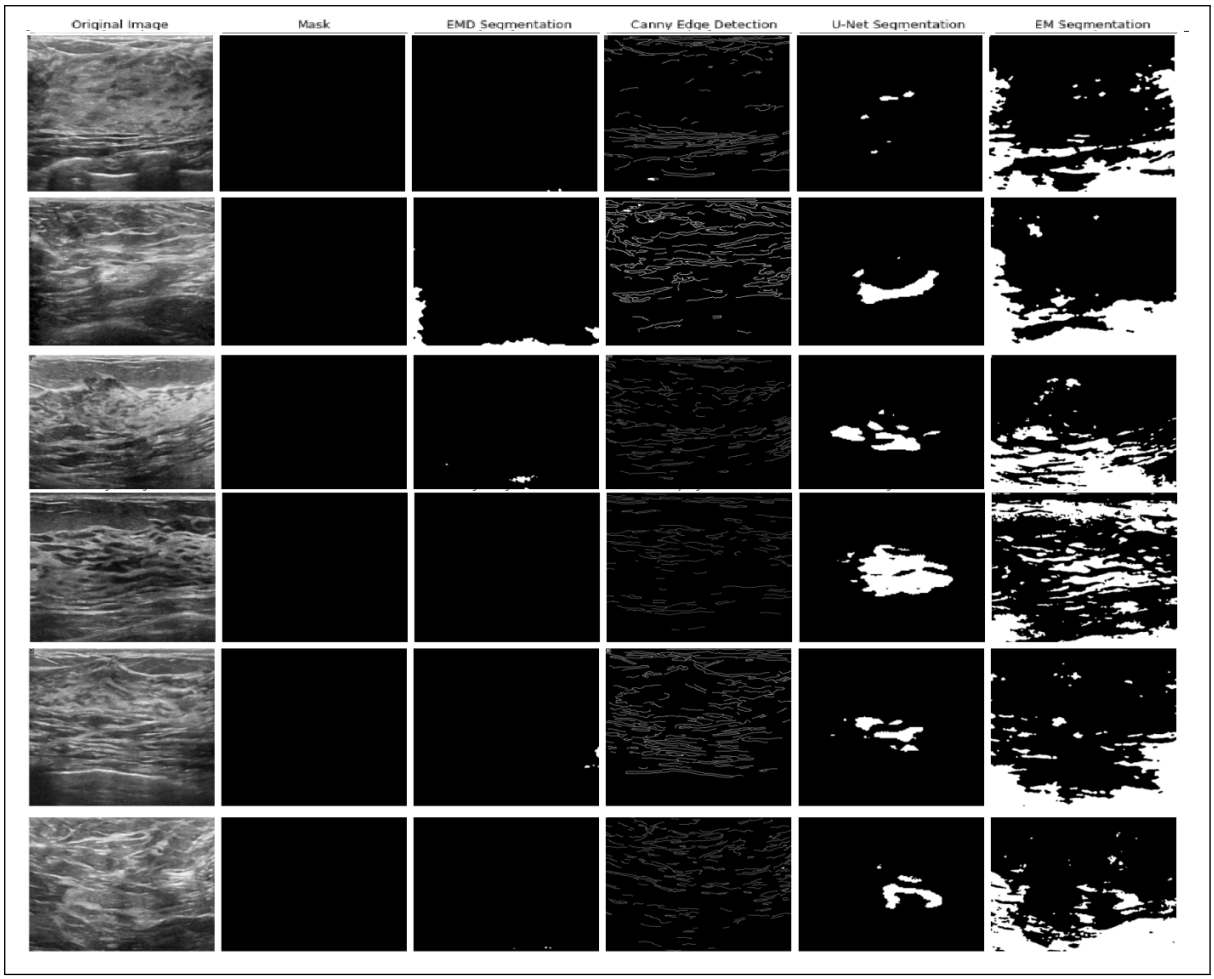


Figure 2.14: Segmentation results for six sample images from the normal folder: (a) Original image,
(b) Ground truth mask, (c) EMD segmentation, (d) Canny segmentation, (e) U-Net segmentation, (f) EM segmentation.

The normal folder presents a unique challenge due to the absence of any tumors, with the ground truth mask being entirely black. This situation complicates the evaluation of segmentation algorithms, as any detected features are likely to be noise rather than actual regions of interest. Both the EMD and U-Net algorithms identified some noise as potential regions of interest; however, EMD failed to delineate any meaningful structures. Similarly, the EM and Canny algorithms also detected significant amounts of noise, underscoring their limitations in distinguishing between actual tissue and artifacts in images where no clear regions of interest exist.

Given that the ground truth masks in the normal folder are completely black (i.e., 100% background), traditional evaluation metrics such as those based on confusion matrices cannot

be computed. Specifically, True Positives and False Negatives are both zero, preventing any meaningful calculation of metrics like Precision, Recall, or F1-Score. Furthermore, calculating the Symmetric Hausdorff Distance is not feasible due to the lack of a defined region to measure against. Therefore, in this scenario, we rely solely on visual comparisons to assess the performance of the algorithms.

2.4.5 Discussion

The visual results and evaluation metrics consistently demonstrate that the EMD algorithm outperforms the other methods, achieving the highest IoU and Dice coefficients. U-Net, though delivering competitive results, shows limitations due to its requirement for extensive labeled datasets for training. EMD offers a more efficient alternative, as it does not rely on large datasets or lengthy training processes. Canny, although computationally efficient, fails to capture the structures present in ultrasound images, leading to weaker segmentation performance. This is evident from both the low IoU and Dice values across all datasets.

The superior performance of EMD can be attributed to the logarithmic transformation applied to the data. This transformation enhances the contrast of ultrasound images, making the differentiation between regions clearer. By emphasizing the structural components of the images, the logarithmic transformation significantly improves the segmentation results.

While these findings underscore the effectiveness of the EMD algorithm on this dataset, further validation on larger and more diverse ultrasound datasets is necessary to confirm the generalizability of this approach across various medical imaging scenarios.

2.5 Application 2 : Affine Multi-Scale Curve Registration Using EM Algorithm

The work developed in this section was published in [Sakrani et al., 2021].

Affine Multi-Scale Curve Registration (AMSCR) [Sakrani et al., 2021] enhances the Affine Curve Matching Algorithm (ACMA) [Elghoul and Ghorbel, 2021b] by incorporating multi scale descriptors to improve curve alignment across various scales. This section details how the Expectation-Maximization (EM) algorithm is integrated into AMSCR to optimize the selection of scales and refine affine transformations.

Contour registration is a fundamental step in image processing and computer vision, particularly for aligning and matching shapes or objects across different images. It involves aligning the contours, or boundaries, of objects across different images, ensuring that corresponding points on the contours match as closely as possible. This process is essential for a wide range of applications, including medical imaging, object recognition, and shape analysis. However, accurately registering contours is challenging due to several factors.

Contour registration can be particularly complex because the same object may appear in different forms across images. These variations could be caused by different angles, times of capture, lighting conditions, or distortions in the images. Furthermore, contours are often subject to deformations, scale variations, noise, and occlusions [Mai et al., 2010]. These factors complicate the process of finding accurate correspondences between contours, especially when dealing with intricate shapes or highly variable data.

Several traditional methods have been proposed to improve contour registration. Many of these methods are based on Euclidean transformations, such as the work in [Huang et al., 2006], which employed Mutual Information (MI) to align shapes by retrieving optimal transformation parameters. Belongie et al. introduced the Shape Context method [Belongie et al., 2002], while the Inner-Distance Shape Context (IDSC) technique [Ling and Jacobs, 2007] was developed to capture the shortest path between feature points within shapes. Dynamic programming has also been employed to handle distortions and occlusions in contour alignment [Petrakis et al., 2002]. [Kaothanthong et al., 2016] proposed a shape signature called the Distance Interior Ratio (DIR) to align curves using a histogram-based method. These techniques have demonstrated effectiveness in various domains but often struggle with complex deformations and varying scales, particularly in noisy or occluded images.

However, traditional methods, especially those based on Euclidean or homography transformations, often face limitations when dealing with more complex deformations, as they require several numerical derivations. Affine transformations, which require fewer derivations, have been more successful in registering curves in challenging scenarios. Methods like the Affine Scale-Invariant Feature Transform (ASIFT) [Morel and Yu, 2009] and the Affine Curvature Scale Space (ACSS) [Mokhtarian and Abbasi, 2001] have shown promise in handling affine distortions effectively. Yet, challenges remain in the presence of noise and occlusions, particularly when using methods such as Curvature Scale Space (CSS) [Mokhtarian et al., 1996], which can be sensitive to local maxima.

To enhance the performance of contour registration, multi-scale approaches have been introduced. These methods incorporate information from multiple scales to capture both global shapes and finer local details. For example, triangular features have been used to create a multi-scale Fourier descriptor [Shu et al., 2015], while the Generalized Curvature Scale Space (GCSS) method [BenKhlifa and Ghorbel, 2019] has been employed to provide a more comprehensive shape descriptor. Such methods improve the robustness of contour matching by allowing for variations in scale, shape complexity, and local deformations.

In this section, we introduce the AMSCR-EM method, which combines affine arc-length parameterization and Gaussian smoothing for normalization. The EM algorithm is employed to optimize the registration by filtering the scales, enhancing the accuracy and performance of the registration process.

2.5.1 Affine Multi-Scale Curve Registration (AMSCR)

In this section, we provide an overview of the Affine Multi-Scale Curve Registration (AMSCR) method, outlining its key concepts and procedural steps.

2.5.1.1 Affine Arc-Length Reparametrization

When comparing curves extracted from different images, it is necessary to normalize the parameterization of each curve to ensure consistency. The method of affine arc-length reparametriza-

tion is applied to achieve this normalization. For a given curve $f(t)$, the normalized affine arc length parameterization $l(t)$ is expressed as follows:

$$l(t) = \frac{1}{L} \int_0^t \left| \det \left(\dot{f}(u), \ddot{f}(u) \right) \right|^{\frac{1}{3}} du \tag{2.2}$$

Here, L denotes the affine total length of the curve, $\dot{f}(u)$ and $\ddot{f}(u)$ represent the first and second derivatives of the curve $f(u)$, and $\det$ refers to the determinant operator. This parameterization allows us to compare curves that may initially have different parameterizations but represent the same geometric shape.

Following normalization, the affine transformation between two curves f and h is defined by the following relationship:

$$h(l) = Af(l) + B \tag{2.3}$$

In this equation, A is the affine transformation matrix, which encapsulates scaling, rotation, and shearing, while B is the translation vector. By using this transformation, the curves are aligned for comparison under affine transformations.

2.5.1.2 Affine Matrix Estimation

To find the affine transformation parameters, A and B, we minimize the distance between the transformed version of one curve and the other curve. The optimization problem can be written as:

$$\min_{A,B} \|Af(l) + B - h(l)\|^2 \tag{2.4}$$

This system of equations can be expressed in matrix form as follows:

$$H = DU \tag{2.5}$$

where H represents the target curve values, D is a matrix constructed from the parameterized points of the input curve, and U is the vector containing the unknown affine parameters A and B. To solve for U, we compute:

$$U = (D^T D)^{-1} D^T H \tag{2.6}$$

This pseudo-inverse solution allows us to estimate the affine transformation that minimizes the error between the two curves.

2.5.1.3 Multi-Scale Smoothing and Registration

Building upon the affine curve alignment, the AMSCR method introduces a multi-scale approach by filtering the curves at various scales. The smoothing of the curves f and h is performed through convolution with a Gaussian function at different scales σ_k, where $1 \leq k \leq p$ and p represents the number of scales:

$$f_x^\sigma(l, \sigma_k) = f_x(l) * g(l, \sigma_k), \quad f_y^\sigma(l, \sigma_k) = f_y(l) * g(l, \sigma_k) \tag{2.7}$$

The Gaussian function $g(l, \sigma_k)$ is defined as:

$$g(l, \sigma_k) = \frac{1}{2\pi\sigma_k^2} e^{-\frac{l^2}{2\sigma_k^2}} \tag{2.8}$$

This operation is applied to both curves, f and h, at each scale, resulting in a series of smoothed curves for both. The resulting system of equations at multiple scales can be written as:

$$\begin{cases} h_\sigma(l_1) = A f_\sigma(l_1) + B \\ h_\sigma(l_2) = A f_\sigma(l_2) + B \\ \vdots \\ h_\sigma(l_N) = A f_\sigma(l_N) + B \end{cases} \tag{2.9}$$

Each scale generates a system of equations. However, not all scales contribute equally to the registration process. Some scales capture fine local details, while others provide a more global alignment. Therefore, selecting the most relevant scales is crucial for accurate curve registration.

2.5.2 AMSCR-EM

Instead of empirically selecting the relevant scales, we propose to use the EM algorithm to identify the class of relevant scales. This section introduces the Affine Multi-Scale Curve Registration with Expectation Maximization (AMSCR-EM) algorithm, which enhances the registration process by unisg the EM algorithm with the affine reparametrization. The EM algorithm helps identify the most relevant scales from a mixture of possible scales.

In this context, the scales correspond to different levels of Gaussian smoothing applied to the curves. Since not all scales are equally relevant, the EM algorithm is used to identify the mixture components that correspond to the most important scales for registration. Once the relevant scales are identified, we refine the registration by focusing on these scales.

In cases where the squared Euclidean distance (L_2) between the curves is extremely small the EM may not perform well due to an overflow at zero in that case we propose using the (EMD) as an alternative.

To better manage this, we estimate the density of L_2 values using the Fast Kernel Density Estimator (FKDE), as shown in Figure 2.15. This visualization helps us understand the distribution of L_2 values.

As depicted in the figure 2.15 the L2 values are near zero, but they didn't overflow with FKDE. So we decided to apply the regular EM

The key steps of the AMSCR-EM algorithm are depicted in the figure 2.16 and described in the flowing paragraph :

1. **Re-sampling**: Curves f and h are re-sampled using affine arc-length reparametrization, ensuring that their parameterizations are comparable across different scales.

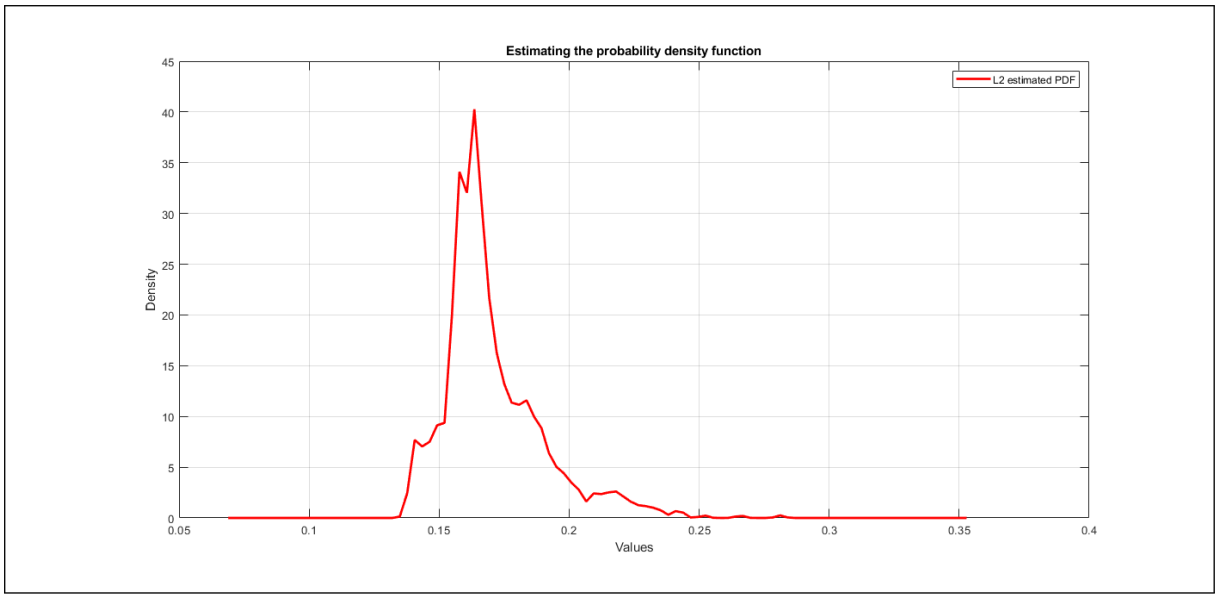


Figure 2.15: Estimated PDF of L_2 using FKDE

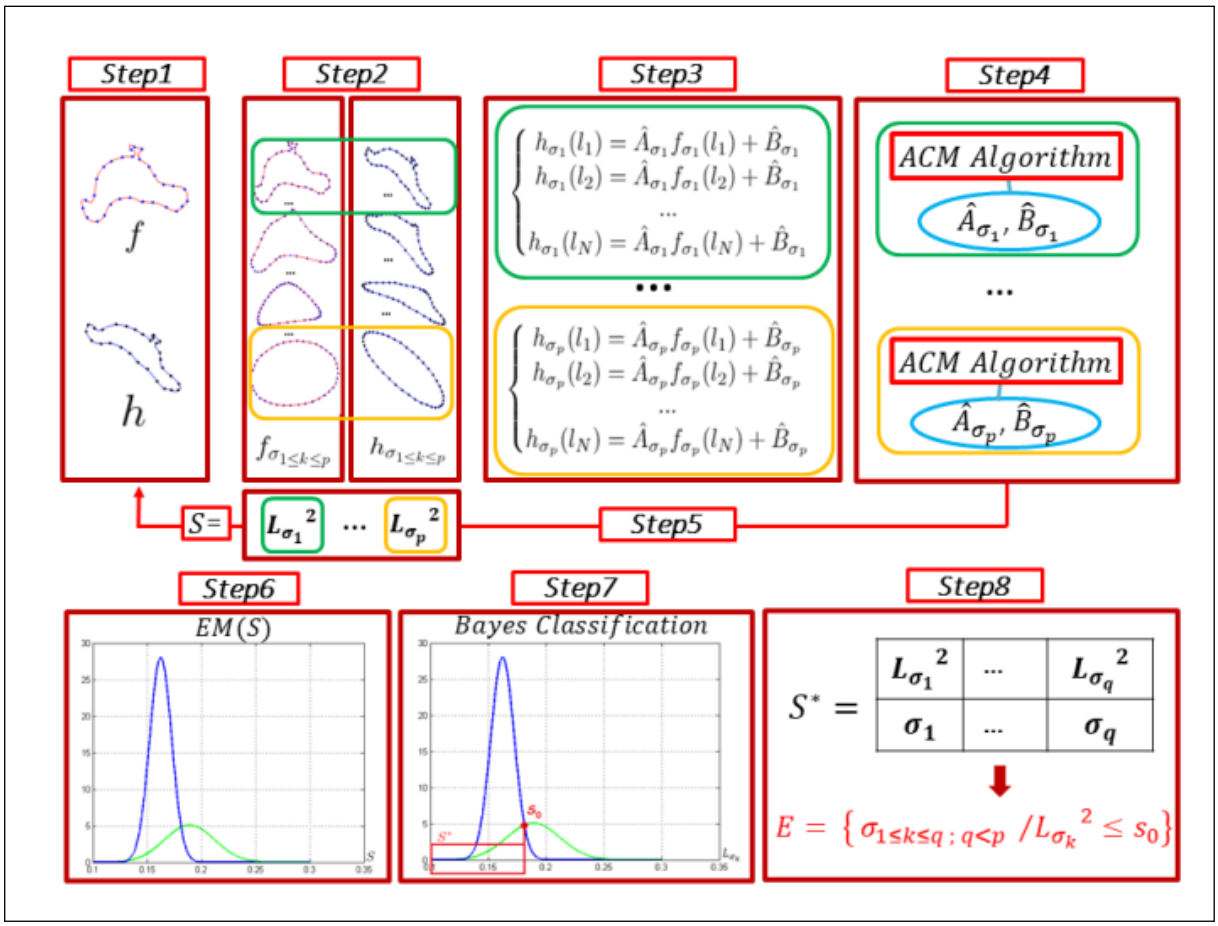


Figure 2.16: Diagram of the AMSCR-EM algorithm.

2. **Gaussian Smoothing**: The re-sampled curves are smoothed at multiple scales σ_k using Gaussian kernels.

3. **Affine Parameter Estimation**: For each scale σ_k, the affine parameters $(\hat{A}_{\sigma_k}, \hat{B}_{\sigma_k})$ are estimated by minimizing the squared Euclidean distance L_2 between the curves.
4. **Expectation Maximization (EM)**: The EM algorithm is applied to the mixture of scales to identify the mixture component of the most relevant components. Once the mixture components are identified, we plot the probability density functions (PDFs) of the mixture and select the threshold S^*. As shown in the figure 2.17, the threshold S^* is identified as the intersection between the two curves. We notice that EM didn't overflow at zero so no need to use EMD in this case.

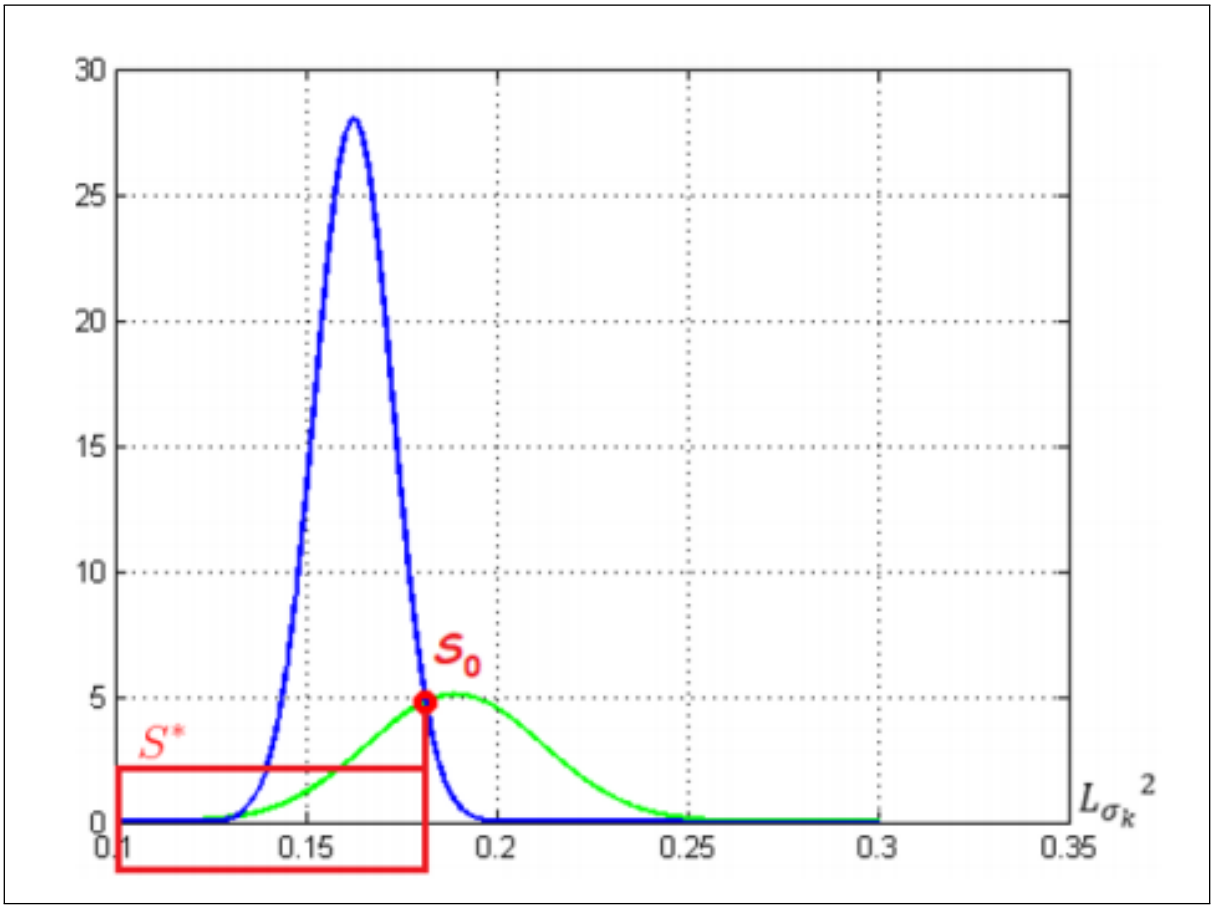


Figure 2.17: Visual representation of the PDF estimated by EM algorithm and the point of intersection between them

2.5.3 Experimentation and Results

We evaluate AMSCR-EM on two separate datasets the MPEG-7 and Kimia-99. The method's performance is assessed based on registration accuracy and robustness across different shapes

and scales. We compare our method against existing techniques, focusing on alignment precision and computational efficiency.

2.5.3.1 MPEG-7 Image Database Retrieval

The MPEG-7 shape database [Latecki et al., 2000] is a widely used database in computer vision. The MPEG-7 Set-B contains 70 shapes, each with 20 images. thus it contains 1400 images.

Figure 2.18 provides examples from each category. To evaluate the performance of the proposed approaches in shape retrieval, we use the Bulls Eye score, as described in [Yang et al., 2018]. This score measures how many correct contours from the same class are among the top 2Nc most similar shapes, with Nc being the number of samples per class. The Bull's Eye score [Fu et al., 2013] is calculated as following :

$$\text{Bull's Eye Score} = \frac{\text{Number of correct results retrieved}}{\text{Total possible correct results}} \times 100 \tag{2.10}$$

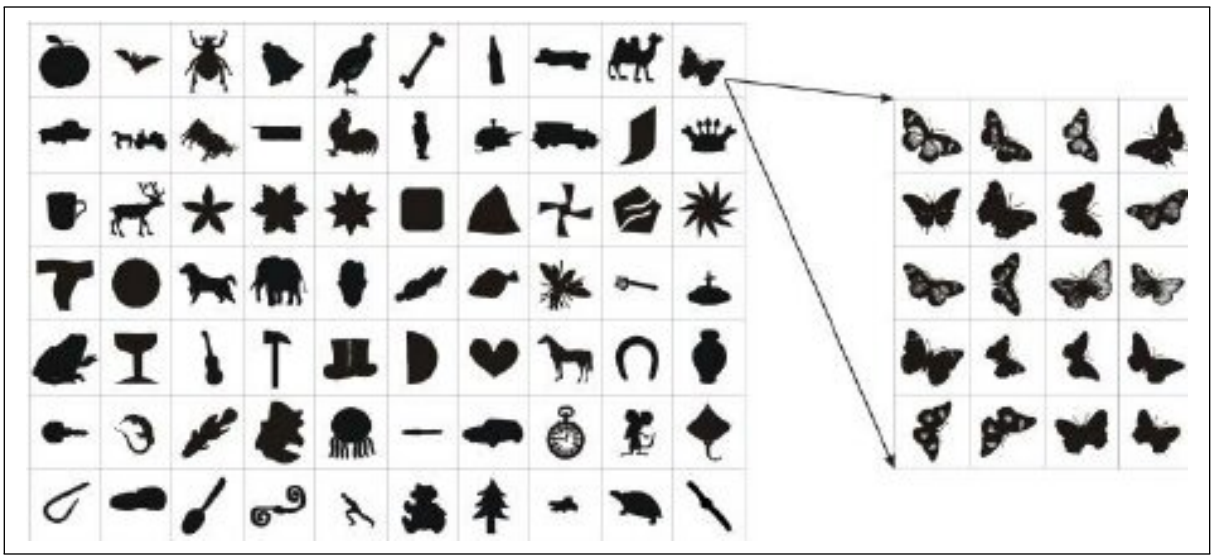


Figure 2.18: Examples from the MPEG-7 dataset. [[Wang and Gao, 2014]]

Table 2.7 presents the Bulls Eye scores for the proposed algorithms alongside existing methods. Our proposed method, Affine Multi-Scale Curve Registration using EM (AMSCR-EM), achieves a score of 94.36%, approaching the top-performing Invariant Multi-scale + LP method.

Table 2.7: Retrieval results on the entire MPEG-7 Set-B dataset.

Algorithm	**Bull's Eye Score (%)**
Multi-scale contour flexibility shape signature [Shu et al., 2015]	67.57 %
Shape Contexts [Belongie et al., 2002]	76.51 %
GCSS [BenKhlifa and Ghorbel, 2019]	78.84 %
SMR by data-driven EM [Tu et al., 2008]	80.03 %
Affine CSS [Mokhtarian and Abbasi, 2001]	81.12 %
CSS-SW [Mai et al., 2010]	81.33 %
Fast Non-Rigid Global Registration [Elghoul and Ghorbel, 2021a]	82.42 %
AICD [Fu et al., 2013]	84.26 %
Multiscale Representation [Adamek and O'Connor, 2004]	84.93 %
IDSC + AspectNorm + SR [Temlyakov et al., 2010]	88.39 %
Multiscale Fourier Descriptor [Yang and Yu, 2019]	83.94 %
MSFDGF-SH [Zheng et al., 2020]	87.76 %
Invariant Multi-scale [Xu et al., 2016]	91.25 %
IMTF [Yang and Yu, 2021]	91.26 %
ACMA [Elghoul and Ghorbel, 2022]	91.55 %
IDSC + LCDP [Yang et al., 2009]	93.32 %
AMSCR [Sakrani et al., 2021]	93.61 %
IDSC + Affine Normalization [Gopalan et al., 2010]	93.67 %
AMSCR using Binary-EM	94.36 %
Invariant Multi-scale + LP [Xu et al., 2016]	**94.51 %**

2.5.3.2 KIMIA Image Database Retrieval

The KIMIA databases [Sebastian et al., 2004] are widely used benchmarks in shape recognition and pattern matching. The KIMIA-99 dataset contains 99 curves, divided into 9 categories, with each category consisting of 11 shapes. Figure 2.19 shows samples from KIMIA-99.

2.8 presents the top 10 matching shapes for the KIMIA-99 dataset. The AMSCR algorithm combined with EM, achieves high matching performance, consistently ranking among the top methods. The AMSCR-EM variant shows excellent performance, matching the top shape correctly in all instances and slightly improving on the standard AMSCR method. This indicates that incorporating EM methods enhances the accuracy of shape matching.

Overall, the AMSCR-EM method shows promising results across both datasets, indicating its robustness and accuracy in shape matching tasks.

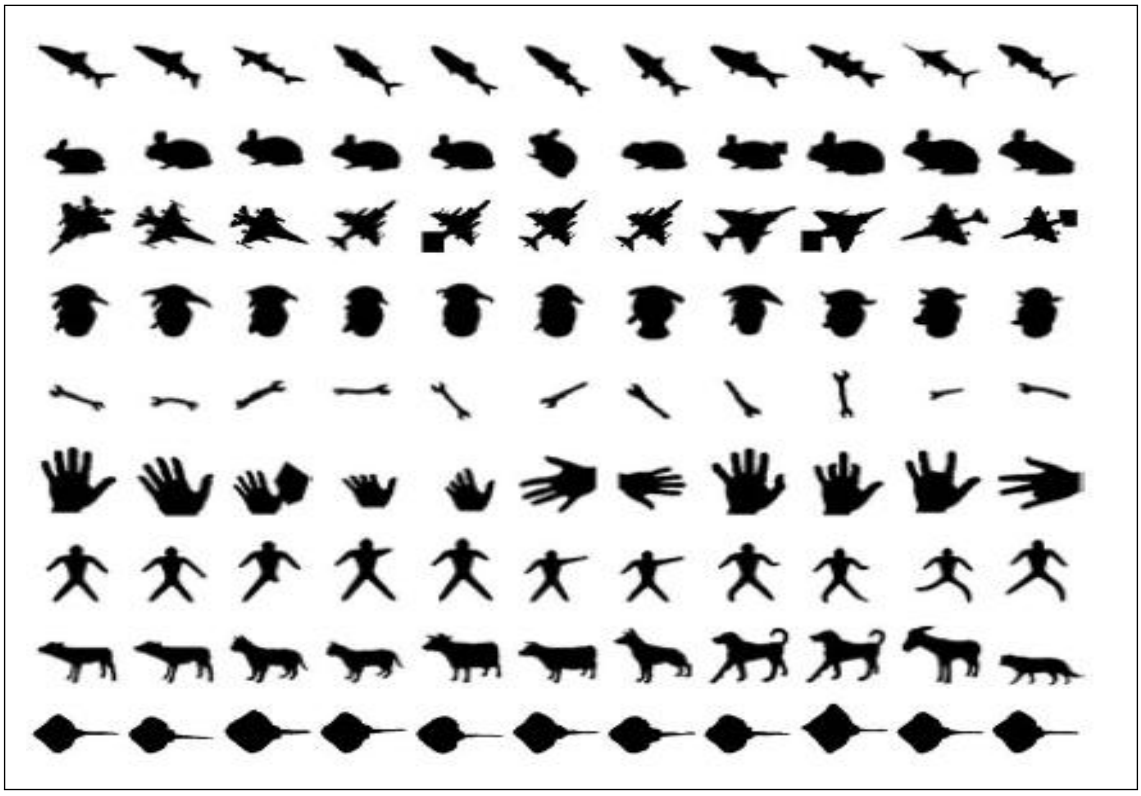


Figure 2.19: Example shapes from the KIMIA-99 dataset.

Table 2.8: Top 10 closest matching shapes for KIMIA-99 dataset.

Algorithm	**Top1**	**Top2**	**Top3**	**Top4**	**Top5**	**Top6**	**Top7**	**Top8**	**Top9**	**Top10**
Shape context	97	91	88	85	84	77	75	66	56	37
CPDH+EMD(eucl)	96	94	94	87	88	82	80	70	62	55
CPDH+EMD(shift)	98	94	95	92	90	88	85	84	71	52
Generative model	99	97	99	98	96	96	94	83	75	48
PS+LBP	99	97	97	88	88	86	86	90	80	77
Shock graphs	99	99	99	98	98	97	96	95	93	82
MDS+SC+DP	99	98	98	98	97	99	97	96	97	85
IDSC+DP	99	99	99	98	98	97	97	98	94	79
Shock Edit	99	99	99	98	98	98	96	95	94	86
Shape-tree	99	99	99	99	99	99	99	97	93	86
GM	99	99	99	99	99	99	99	97	93	86
Symbolic rep	99	99	99	99	96	96	99	95	93	88
IMC	99	99	99	99	98	97	95	94	90	83
HF	99	99	99	99	98	99	99	96	95	88
IDSC+LBP	99	99	99	99	98	97	97	98	98	96
ACMA	99	99	99	99	99	99	98	98	97	95
SMR by data-driven EM	99	97	99	98	96	96	94	83	75	48
AMSCR	99	99	99	99	99	99	98	98	97	96
AMSCR-EM	99	99	99	99	99	99	96	98	98	96

2.6 Conclusion

In this chapter, we have introduced a novel variant of the Expectation-Maximization algorithm: the Diffeomorphism EMD. By applying a diffeomorphism to the data prior to executing the EM algorithm, we enhance its ability to handle bounded and semi-bounded data. The integration of the elbow method for class estimation and the logarithmic transformation further improves the performance of EMD, especially in cases where traditional EM struggles with boundary overflow issues.

The effectiveness of EMD was validated through tests over simulated data and over ultrasound image segmentation tasks. It demonstrated superior accuracy and noise management compared to other established methods such as U-Net, EM, and Canny. These results underscore EMD's potential as a powerful segmentation tool for bounded or semi-bounded data clustered near the edge.

Also, we introduced a new algorithm for the Affine multi scale registration problem The AMSCR-EM. it uses the Em algorithm to optimize the identification of the most relevant scales among the mixture of scales. The tests conducted on two different datasets highlighted the efficiency of this new algorithm.

Assessment of deep learning algorithms for financial forecasting

Contents

Chapter 3 Abstract

This chapter presents a detailed assessment of deep learning algorithms developed for financial forecasting. It commences with an introduction to the field of financial forecasting.

The methodology section details the processes employed to evaluate the algorithms effectively, setting the groundwork for a thorough analysis. A probabilistic criterion for algorithm evaluation is used, emphasizing the importance of statistical rigor in assessing performance. This leads to a detailed exploration of the Cumulative Distribution Target Criterion (CDTC), a probabilistic evaluation criterion developed to enhance the assessment of algorithms.

The results and analysis section presents findings from both traditional and probabilistic evaluations of deep learning algorithms. Traditional evaluation techniques are discussed, emphasizing their limitation in detecting the instability in the performance across single iterations and multiple iterations (100 iterations), while the probabilistic evaluation section delves into the advantages of using probabilistic approaches to gauge the algorithms predictive capabilities more effectively.

In conclusion, this chapter synthesizes the insights gained from the evaluations, highlighting the advantages of the use of a probabilistic approach to evaluate deep learning algorithms in financial forecasting and offering directions for future research and improvement in algorithmic evaluation.

3.1 Introduction

In this chapter, we present a comparative study evaluating the performance of deep learning algorithms within the context of regression tasks. Our focus is on utilizing a probabilistic evaluation criterion to assess these algorithms more accurately. Traditional evaluation metrics, often based on a limited number of model executions, may lead to misleading assessments. This is due to the fact that errors can be considered realizations of continuous random variables, which introduces variability that might cover the true performance and stability of models, particularly in the context of complex financial data.

To address these challenges, we propose the use of "Cumulative Distribution Target Criterion" (CDTC), a probabilistic metric designed to provide a more nuanced assessment of model performance. Unlike conventional metrics, CDTC evaluates algorithms based on the distribution of their error rates over multiple executions, offering insights into both the accuracy and stability of models.

In this chapter, we apply CDTC to assess the performance of Long Short-Term Memory (LSTM) networks for financial forecasting, specifically focusing on stock close price prediction. Financial forecasting is by definition complex due to the multitude of factors influencing financial data, both internally and externally. Prices incorporate information quickly, making accurate predictions challenging, and financial data is often noisy and rapidly changing.

To address these challenges, financial analysts use technical analysis, which encompasses numerous features, such as those provided by Yahoo Finance (92 features). Thus, the financiers often uses Feature selection methods to be able to reduce the feature set to a more manageable number while retaining the most relevant features. A review by Nti in 2020 [Nti et al., 2020] found that 99% of the studies reviewed employed correlation-based feature selection methods.

In this work, we propose using Principal Component Analysis (PCA) as a dimensionality reduction technique for feature selection. PCA can significantly reduce the number of features while preserving as much information as possible.

We will utilize the CDTC criterion to evaluate the impact of dimensionality reduction via PCA on various LSTM architectures for stock price prediction. By applying CDTC, we aim to provide a robust comparison of different LSTM models and demonstrate how dimensionality reduction techniques can influence model performance.

3.2 Overview of Financial Forecasting

Financial forecasting, particularly stock price prediction, is a challenging task due to the uncertainty and multitude of influencing factors in the stock market [Zhao et al., 2023]. Researchers have explored various methodologies to enhance prediction accuracy, each with its strengths and limitations.

Early approaches to financial forecasting relied on time series analysis techniques. Models such as Autoregressive Moving Average (ARMA) [Rojas et al., 2008] and Autoregressive Integrated Moving Average (ARIMA) [Kumar et al., 2022] utilize historical data to predict future prices based on observed trends. These models are foundational in time series forecasting but often struggle with nonlinearity and complexity in financial data.

The evolution of machine learning introduced a variety of methods such as Artificial Neural Networks (ANNs) [Kurani et al., 2023] [Gurjar et al., 2018], Support Vector Machines (SVMs) [Sapankevych and Sankar, 2009], and fuzzy theory-based models [Boyacioglu and Avci, 2010] [Wang et al., 2023] have demonstrated considerable promise in capturing complex relationships in financial data. These methods offer greater flexibility and can model nonlinearities more effectively than traditional time series models.

Enhancements to forecasting models often involve advanced feature selection and extraction techniques. Principal Component Analysis (PCA) [Cao and Wang, 2020] reduces dimensionality by transforming data into principal components, which can improve model performance. Evolutionary algorithms such as Genetic Algorithms (GA) [Li et al., 2022], wavelet transforms [Liu et al., 2020], and Particle Swarm Optimization [Thakkar and Chaudhari, 2021b] are also used to optimize feature selection and model parameters, further refining prediction accuracy.

Unsupervised methods like clustering [Vilela et al., 2019] have also been applied to identify patterns in stock price data.

Recent advancements in deep learning have significantly impacted financial forecasting. Deep learning models, capable of uncovering complex, nonlinear relationships in noisy data, have shown great promise. Methods such as Deep Neural Networks (DNNs) [Yu and Yan, 2020] [Thakkar and Chaudhari, 2021a], Convolutional Neural Networks (CNNs) [Shah et al., 2022] [Chen and Huang, 2021] [Akehir and Kiliç, 2022], and Long Short-Term Memory networks (LSTMs) [Jin et al., 2020] [Li et al., 2021] [Ding and Qin, 2020] are increasingly used for stock price and return predictions.

Among these, LSTMs are particularly noteworthy for their robustness in capturing temporal dependencies [Hu et al., 2021]. Studies such as [Hua et al., 2019] and [Teng et al., 2022] highlight the effectiveness of LSTMs in time series forecasting. Despite their advantages, evaluating LSTM performance remains challenging due to their intrinsic instability [Ghazi et al., 2019].

In the context of these diverse methodologies, this thesis focuses specifically on the impact of dimensionality reduction through PCA on LSTM networks for financial stock prediction. By exploring this interaction, the thesis aims to assess the impact of PCA on the performance of LSTM models in forecasting stock prices, addressing both the strengths and limitations of integrating these techniques.

3.3 Methodology

This study investigates the influence of Principal Component Analysis PCA on a Long Short Term Memory LSTM network's ability to predict stock closing prices.

The experimental analysis is performed on different LSTM architectures, varying in their input features. These models were implemented using Python on Google Colab. We utilized the Optuna library to automatically optimize the hyperparameters for these models. The optimization process was executed multiple times (100 times) for each model to ensure the selection

of the most effective hyperparameters. For each model, the best hyperparameters found were tested on the other models, and we retained the configuration that yielded the best performance across all models. Once the configurations were established, we trained each model on a historical dataset of stock prices and evaluated their performance on a held-out test set.

The table 3.1 presents the list of the hyperparameters and their corresponding range of values used in Optuna and the best value found. The "Number of Layers" indicates the number of layers in the LSTM network. It was interesting to notice that for all four different architectures, the best number of LSTM layers was one. The single-layer LSTM model outperformed the multilayer LSTM model. This result corroborates the findings of [Bhandari et al., 2022]. However, the one-layer LSTM presented a consequent stability problem, so we had to re-run Optuna starting from two LSTM layers.

While the "Number of Units" corresponds to the number of units within the hidden layers of the LSTM network, the "Dropout Rate" is applied to mitigate overfitting. The "Activation" function can take on values such as sigmoid, relu, or tanh. "Batch Size" represents the number of samples per batch during training. The "Number of Epochs" denotes the total iterations over the entire training dataset. The "Optimizer" can be one of three options: stochastic gradient descent (SGD), Root Mean Square Propagation (RMSProp), or Adaptive Moment Estimation (ADAM). "Learning Rate" signifies the rate at which the back-propagation algorithm learns. Lastly, "Number of Steps" refers to the size of the time window used to segment the sequential stock price data denotes the time windows used to segment the sequential stock price data. Consequently, for each input data, the closing price is predicted based on the last 'The number of Steps' days. , and "Loss" can be either mean squared error or mean absolute error.

To thoroughly assess the predictive performance of each LSTM model, we conducted 100 runs for each model configuration. This allowed us to generate a sample of 100 performance indicators for each model. The coefficient of determination (R^2) was utilized as the representative error metric.

Table 3.1: List of parameters and their corresponding range of values used in Optuna

Hyperparameter	Considered values	Best value
number of layers	[2,10]	2
number of units	[16,256]	128
dropout rate	[0.0,0.9]	.034
activation	'relu', 'tanh', 'sigmoid'	'relu'
batch size	[8, 16, 32, 64]	16
epochs	[10, 100]	50
optimizer	'adam', 'sgd', 'rmsprop'	'rmsprop'
learning rate	[1e-5, 1e-2]	2.04 e-05
number of steps	[5, 60]	15
loss	'mean squared error', 'mean absolute error'	'mean absolute error'

The compared LSTM networks differ only in their input layers and more specifically in the number of neurons in the input layer. We detail below the features constituting the input layers of the 4 studied networks.

- **Architecture 1 :** The input layer is composed of 6 neurons relating to 6 features. It's about closing price close, the opening price open, the highest price of the session high, the lowest price of the session low, the trading volume volume and the price closing adjusted adj close.

- **Architecture 2 :** This architecture is more complex since the input layer is composed of 91 attributes derived from technical analysis, such as moving averages, relative strength index (RSI), stochastic oscillators, and others. All attributes included in our analysis displayed a missing data percentage under 25%

- **Architecture 3:** This architecture is a simplification of architecture 1 thanks to the use of PCA. Indeed, the inertia due to the first principal axis is already 97.67%. Therefore, we propose an input layer with only two features: the linear combination of the data projected on the main axis and the variable 'Close'.

- **Architecture 4** : The dimension reduction by the PCA for the data of architecture 2 is at the origin of this fourth architecture. Here the main axis has an inertia of only 40.12%. The second main axis has an inertia of 29.54%. We chose to keep as features the linear combination of the data projected on the first two axes and the 'Close' feature since the

variance explained on these first two axes is around 70%. The input layer of this network will therefore be composed of 3 features.

The dataset was obtained from Yahoo Finance. We chose to conduct our experiment on Google stock prices from 20-08-2004 to 01-01-2023 (4625 days). The data normalization was performed using MinMaxScaler, which scales each variable to the range [0,1].

Subsequently, we divided the dataset into two different sets. The initial set consists of 80% of the original dataset, covering 3685 days, utilized as the training dataset. The remaining 20% (910 days) constituted the testing dataset.

3.4 Probabilistic Criterion for Algorithms Evaluation

Traditional evaluation metrics for assessing model and algorithm efficiency often rely on a limited number of model executions. However, each execution can yield varying error rates, leading to inaccuracies in performance evaluation. This approach fails to capture the stability of models. For example, traditional metrics might not reveal this inconsistency if a model converges to two different means.

Therefore, a comprehensive evaluation should encompass a larger number of executions to better understand the variability and ensure a more robust assessment. It is crucial to develop new evaluation metrics to assess different machine learning techniques thoroughly and to measure the impact of dimensionality reduction on the accuracy and stability of stock price predictions.

In this work, we chose to use a probabilistic criterion called the Cumulative Distribution Target Criterion CDTC introduced in [Ben Slimen et al., 2022].

Since the error rate varies with each execution, it can be considered as a continuous random variable (CRV). This variability allows us to apply the Cumulative Distribution Target Criterion CDTC, which relies on estimating the probability density function PDF of performance indicators.

The CDTC provides a means to compare multiple algorithms by estimating the PDF of their error rates. To obtain this estimation, a significant number of performance indicators must be generated through repeated executions of the algorithm under evaluation. As a result, the CDTC criterion is particularly useful for assessing Machine Learning algorithms, including Long Short-Term Memory LSTM networks.

3.5 The CDTC criterion

This section presents the Cumulative Distribution Target Criterion CDTC introduced in [Ben Slimen et al., 2022]. As mentioned in the previous section, this criterion is based on the estimation of the PDF of the generated sets of performance indicators (prediction errors). Considering that a few numbers of executions of the model are not sufficient to accurately evaluate a model, the CDTC requires a significant number of executions, typically at least 100 executions. At each execution, we can observe a different value of the performance indicator, denoted by y_i. Therefore, we obtain a set of performance indicators. Each element of this set can be considered as a realization of a continuous random variable y. Thus, it will be possible to estimate the PDF of y by parametric or non-parametric methods.

Since we have no information about the distribution shape, employing non-parametric methods would be more appropriate As demonstrated in the section **??** combining FKDE with the plug-in algorithm leads to a robust convergence of the estimator within the Integrated Mean Squared Error **IMSE!**. Then, we will employ the FKDE combined with the plug-in algorithm to estimate the PDFs. Let $A_1, A_2, ..., A_k$ be k architecture models to be compared and $Y_1, Y_2, ..., Y_k$ their sets of prediction error value. The cardinal of each set is $|Y_i| = n_i$.
Estimating the probability density function provides a powerful visualization tool. By displaying the distribution of performance indicators for each architecture on a single plot, it allows a direct visual comparison of both performance and stability.

However, in certain cases, such visual comparison may not be straightforward. therefore, the CDTC [Ben Slimen et al., 2022] allows us to quantitatively measure the relative efficiencies

of the compared architectures. The CDTC, relies on the cumulative functions of the Y_i variables and is designed to distinguish the performance levels among the various architectural models.

To compute the $CDTC$ criterion, we consider that the mean value of Y_i, denoted as μ_i, is estimated as follows:

$$\mu_i = \frac{\sum_{j=1}^{n_i}(Y_{ij})}{n_i}$$

We denote μ as the average of the prediction error values across all architecture models $A_1, A_2, ..., A_k$. The estimation for μ is calculated as follows:

$$\mu = \frac{\sum_{i=1}^{k}(\mu_i)}{k}$$

We define $CDTC(A_i)$ as the value for the proposed criterion specific to architecture model A_i. This criterion represents the cumulative distribution function value of Y_i with respect to μ. The computation of the CDTC_{A_i} varies depending on whether it is a maximization or minimization problem.

In this work, we chose R^2 as the performance indicator of the prediction. As it is one of the most used metrics in the literature and according to a study conducted by Chicco and his colleague in 2021 R^2 is more informative and reliable than many other widely used metrics [Chicco et al., 2021].The R^2 score is calculated by the given formula 3.1

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{3.1}$$

In this formula y_i represents the real values, $\hat{y}_i$ the values predicted by the model, $\bar{y}$ the mean of the real values, and n is the number of observations.

Therefore, in this chapter, we are dealing with a maximization problem. A higher R^2 indicates a better fit between the predicted and actual values.

To calculate the CDTC, we consider the integral of f between μ and $+\infty$, (equation 3.2).

$$CDTC(A_i) = P\left[Y_i > \mu\right] = \int_{\mu}^{+\infty} f_i\left(y\right) dy \tag{3.2}$$

3.6 result and analysis

In this section, we present the evaluation of the LSTM architectures studied for predicting the price of financial assets. To show the importance and utility of the CDTC we tested our architectures as usual with one single iteration, then ran our models 100 times and computed the average results. Finally, we employed the CDTC to reveal latent information within the data.

3.6.1 Traditional evaluation

The conventional evaluation was conducted in two stages. In the first stage, we ran our models once and attempted to compare the results. In the second stage, we ran each of our models 100 times, calculated the average prediction, and then assessed the outcomes.

3.6.1.1 Single iteration

Initially, we aimed to visually assess the accuracy of our architectures, so we plotted the actual price alongside the predictions from models with and without PCA. Figures 3.1 and 3.2 show the actual price as the black curve, the architecture without PCA as the blue curve, and the architecture with PCA as the red curve.

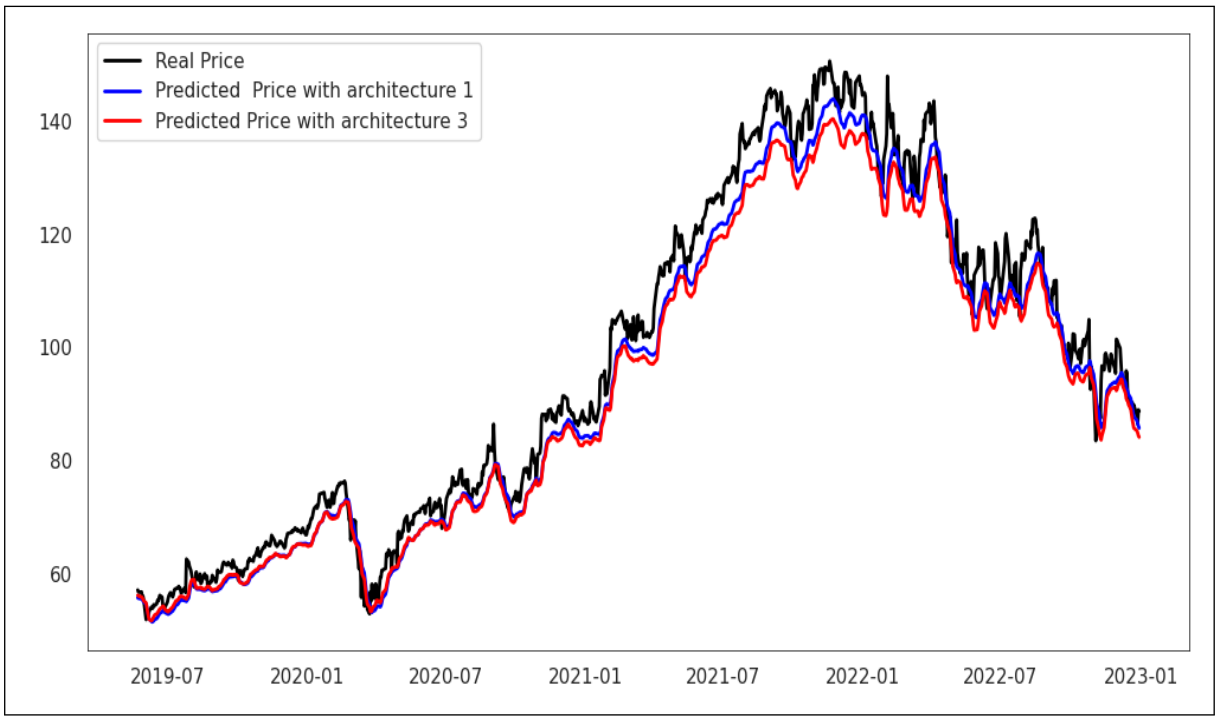


Figure 3.1: visual comparison of real and predicted prices -single iteration- using 6 input features, with and without PCA (architecture 1 and 3).

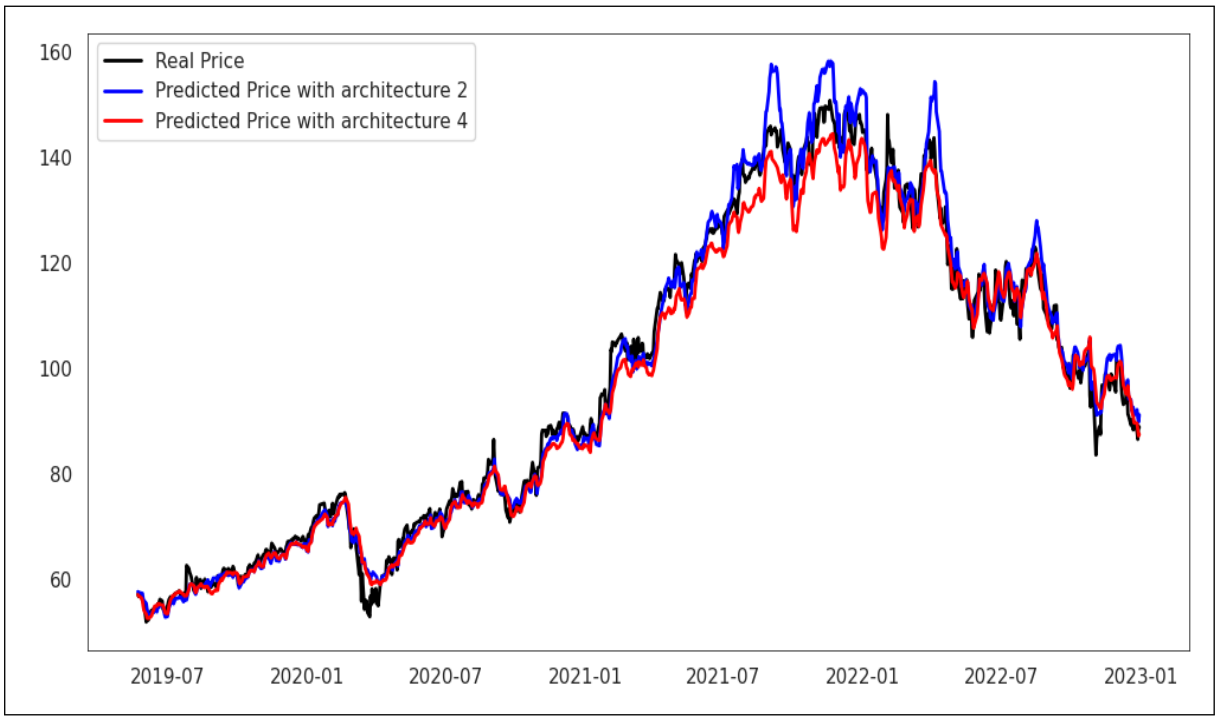


Figure 3.2: visual comparison of real and predicted prices -single iteration- using 91 input features, with and without PCA (architecture 2 and 4).

In both figures 3.1 and 3.2, the predictions generally align closely with the actual values, suggesting that PCA does not significantly affect the accuracy of the predictions in this run.

It is important to note that we found that in 77% of the cases, the PCA improves the performance for the LSTM model with 91 input features. With the 6 input features, it was in 72% of the runs the architecture with PCA outperformed the architecture without PCA. But, this single iteration run belongs to the minority of cases where the architecture without PCA outperformed the ones with PCA. As illustrated in the localized view in figure 3.3, where predictions with PCA are less accurate than those without PCA.

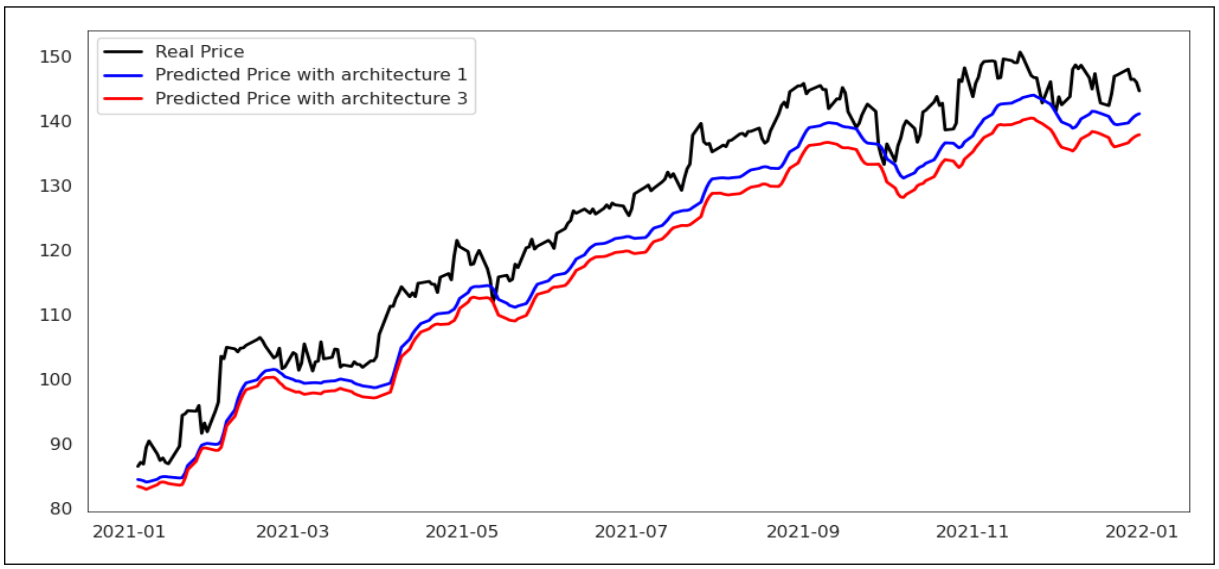


Figure 3.3: Localized View on one-year period : visual comparison of real and predicted prices -single iteration - using 6 input features, with and without PCA (architecture 1 and 3).

To validate these findings, we chose some of the widely used metrics in regression analysis defined in the first chapter 1.4 to compare our architectures. The results presented in Table 3.2 confirm our visual observation that PCA does not have a significant impact on prediction accuracy. In fact, in both cases, predictions without PCA outperformed those with PCA across all evaluation metrics. Additionally, architectures with 91 input features consistently yielded better results compared to those with 6 input features.

At this level, it is important to emphasize that these results only represent a single execution of the algorithm. Other runs might show different results. It is therefore not possible to draw conclusions from this single example.

Table 3.2: Comparing the stock price prediction results of the different architectures on one single iteration

Evaluation metric	Architecture 1	Architecture 2	Architecture 3	Architecture 4
R^2	0.973	**0.985**	0.958	0.982
RMSE	4.848	**3.602**	6.049	3.922
acsMAE	4.039	**2.588**	5.032	2.998
MSE	23.503	**12.975**	36.586	15.381
MAPE	4.167	**2.601**	4.927	2.935
Adjusted_R^2	0.973	**0.984**	0.958	0.982
Median_AE	3.541	**1.737**	4.225	2.361
sMAPE!	4.274	**2.580**	5.080	2.962
MSLE	0.002	**0.001**	0.003	**0.001**
MASE	2.935	**1.880**	3.656	2.178
sMSPE	0.001	**0.000**	0.001	**0.000**

As we can clearly see, in table 3.2 Architecture 2 outperforms all the other architectures in most metrics. According to this iteration PCA doesn't have a significant impact on prediction, it even fails to perform as well as the first two architecture

3.6.1.2 100 iterations

For the 100-iteration test, we ran each of our models 100 times. We then calculated the average prediction for each day and visualized the results in Figures 3.4 and 3.5. As in the previous figures, the black curve represents the actual price, the blue curve shows the average prediction without PCA, and the red curve represents the average prediction with PCA.

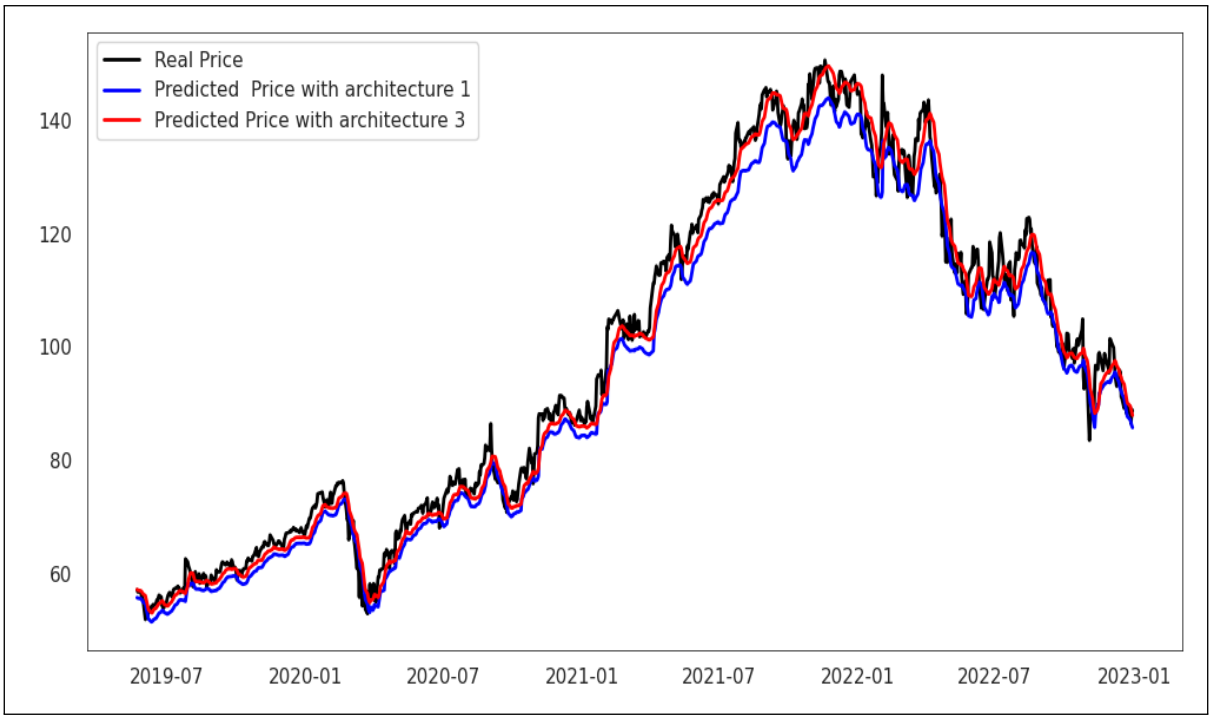


Figure 3.4: Comparison of real and predicted prices -100 iterations- using 6 input features, with and without PCA. (architecture 1 and 3)

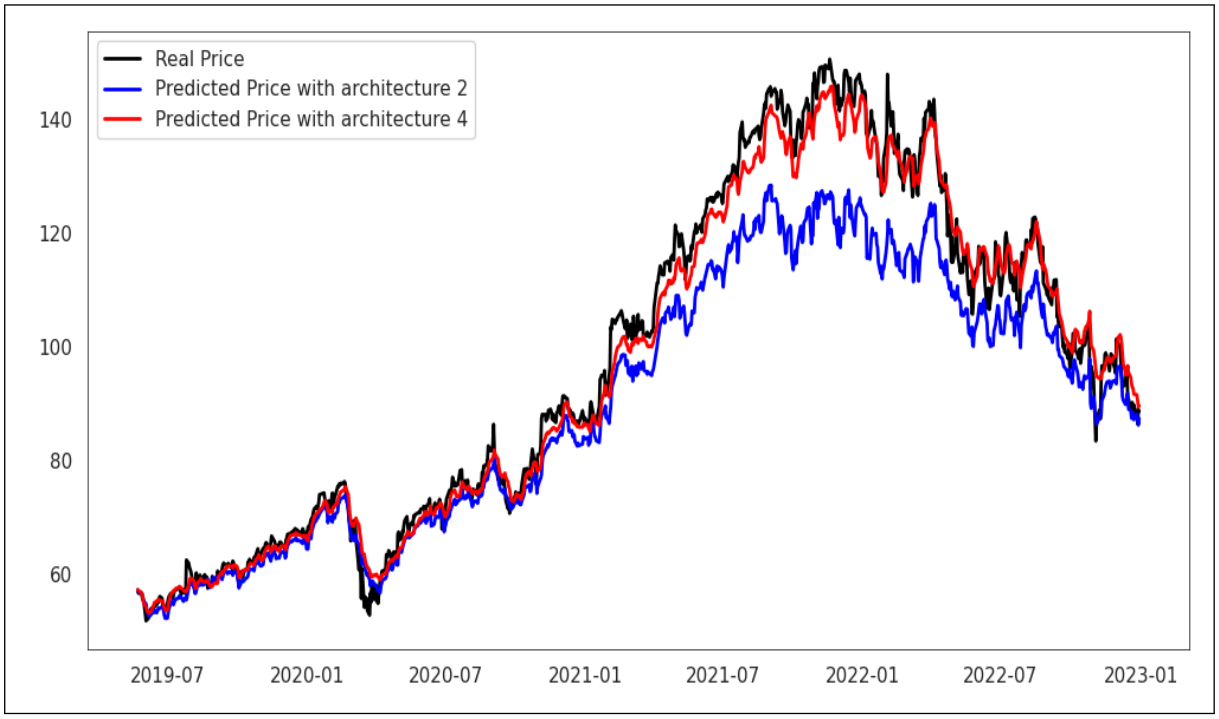


Figure 3.5: Comparison of real and predicted prices -100 iterations- using 91 Input features, with and without PCA.(architecture 2 and 4).

Unlike the single-iteration results, Figures 3.4 and 3.5 clearly show that PCA enhances prediction accuracy, with the predictions sometimes overlapping the actual prices. This is even more evident in Figure 3.6, where the predictions with PCA closely align with the real prices.

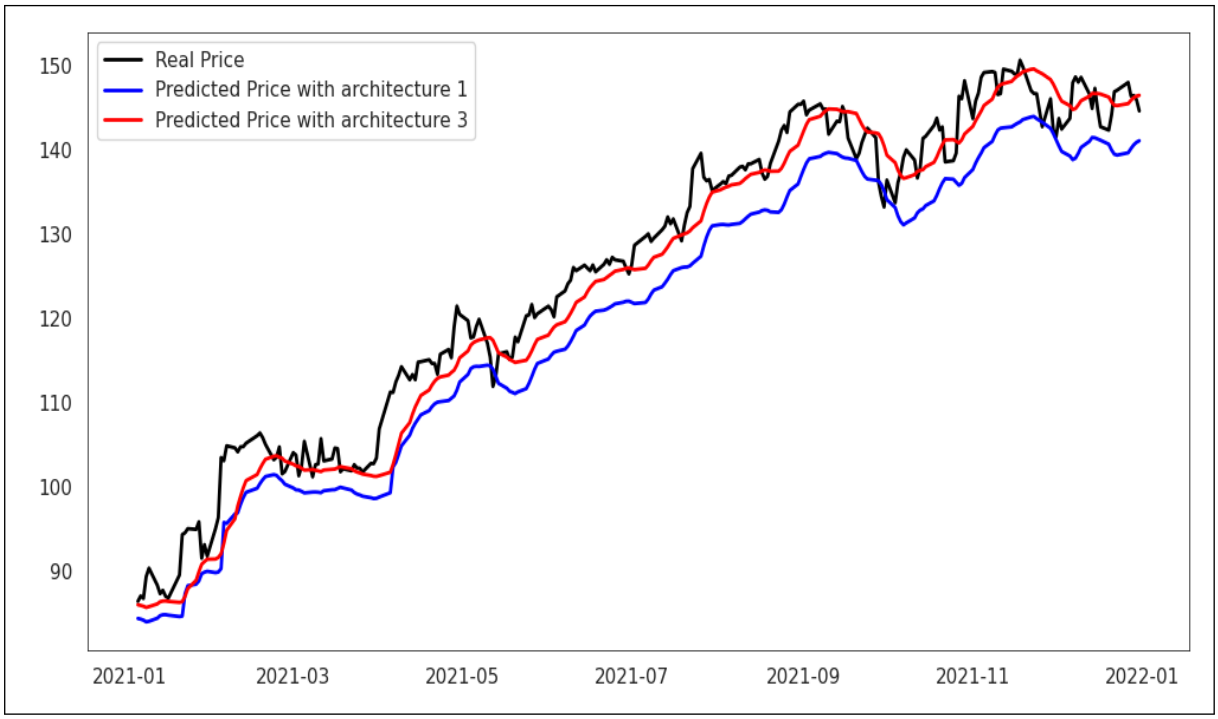


Figure 3.6: Localized View: Comparison of real and predicted prices -100 iterations- using 6 input features, with and without PCA on one-year period.(architecture 1 and 3)

To validate these findings, we used standard evaluation metrics. The results are summarized in Table 3.3.

Table 3.3: Comparing the stock price prediction results of the different architectures on 100 iteration

Evaluation metric	Architecture 1	Architecture 2	Architecture 3	Architecture 4
R^2	0.974	0.877	**0.986**	**0.986**
RMSE	4.742	10.373	3.491	**3.449**
MAE	3.971	7.679	2.744	**2.633**
MSE	22.489	107.598	12.189	**11.897**
MAPE	4.090	6.671	2.866	**2.638**
Adjusted_R^2	0.974	0.863	**0.986**	**0.986**
Median_AE	3.535	4.887	2.294	**2.051**
sMAPE!	4.191	7.005	2.887	**2.641**
MSLE	0.002	0.007	**0.001**	**0.001**
MASE	2.886	5.580	1.994	**1.913**
sMSPE	0.001	0.002	**0.000**	**0.000**

Based on the results in Table 3.3, it's evident that PCA has a positive impact, improving prediction accuracy in both cases.

Its worth noting that the architecture that produced the best results in the single iteration test turned out to be the worst-performing in this 100-iteration test, underscoring the importance of a large number of trials for reliable assessment.

3.6.2 Probabilistic evaluation

In this section, we will evaluate the architectures according to the probabilistic criterion CDTC.

Estimating the PDFs of the performance indicators will make it possible to better evaluate the relative prediction capacity of each algorithm. For this, the predictions are executed 100 times by each network generating a set composed of 100 performance indicators. We have chosen to use the most common performance indicator in the field of financial prediction, namely R^2. Each generated R^2 is considered as a realization of an absolutely continuous random variable. The PDFs of the distribution of R^2 from each network are estimated by the non-parametric kernel estimator with optimization of the bandwidth by the plug-in iterative algorithm. In Figure 3.7, the estimated distributions of R^2 generated from networks 1 and 3 are represented. We can observe that the density in red representing the distribution of R^2s resulting from architecture 3 is statistically closer to 1 compared to those of the distribution resulting from architecture 1. In addition, the distribution of R^2s resulting from network 1 is clearly bimodal showing an instability of convergence of this network. The figure 3.9 also shows the predominance of network 4 over network 2. Here too, the PDF of the random variable R^2 from architecture 4 converges towards a single mode with a low variance, unlike the PDF of R^2 from network 2 which presents a high number of small modes reflecting the instability of this network.

It is noteworthy to point out that in the figure 3.9 we can clearly see that the first architecture converges to two different values (bimodal cure) which can explain why in the one-iteration test it didn't perform well.

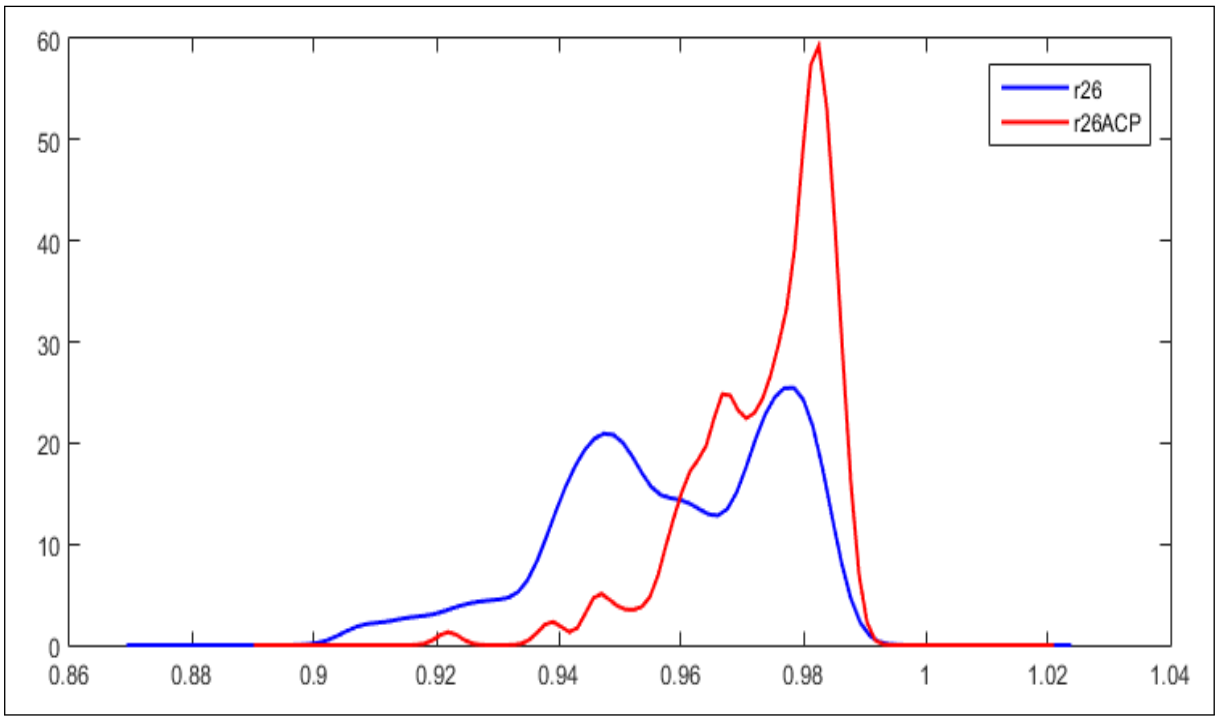


Figure 3.7: Probability density function of prediction errors of Architecture 1 (ARCH1) and Architecture 3 (ARCH3).

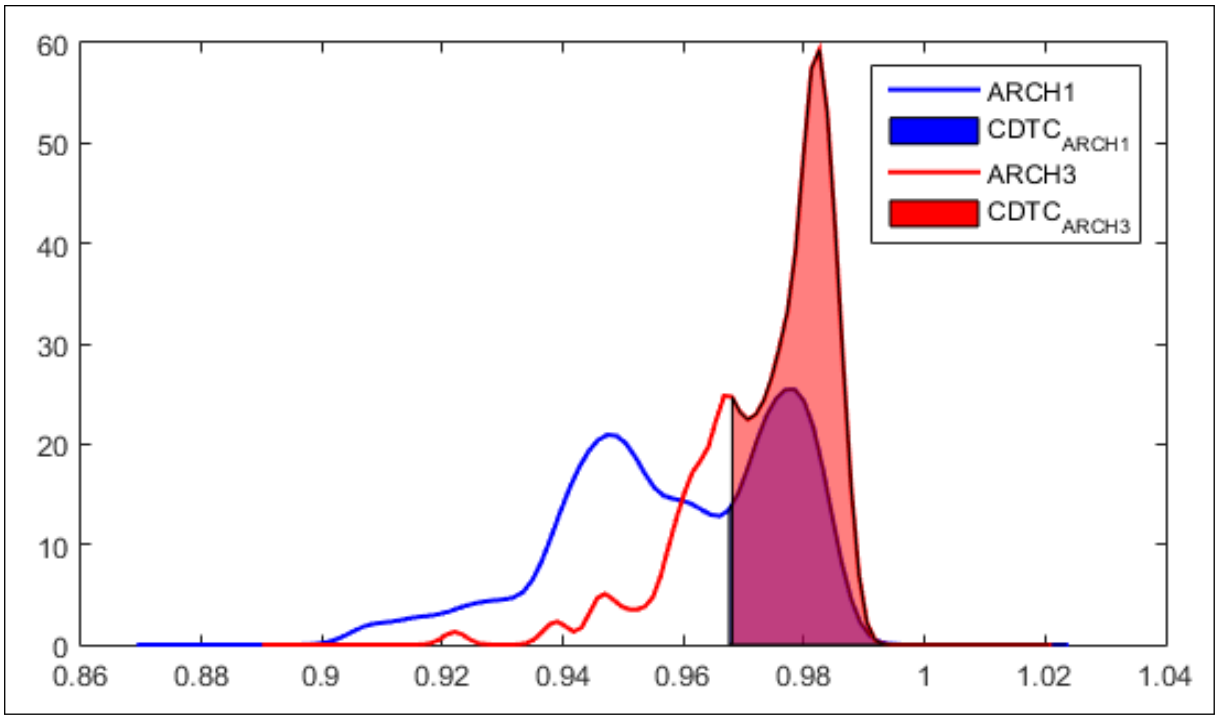


Figure 3.8: Visualization of CDTC_{ARCH1} and CDTC_{ARCH3}.

Figures 3.8 and 3.10 represent the CDTC criterion, which is evaluated by the integral value from the average of the concatenation of generated R^2 sets at $+\infty$ of the estimated PDFs. A high value of this integral indicates better prediction performance of the algorithm.

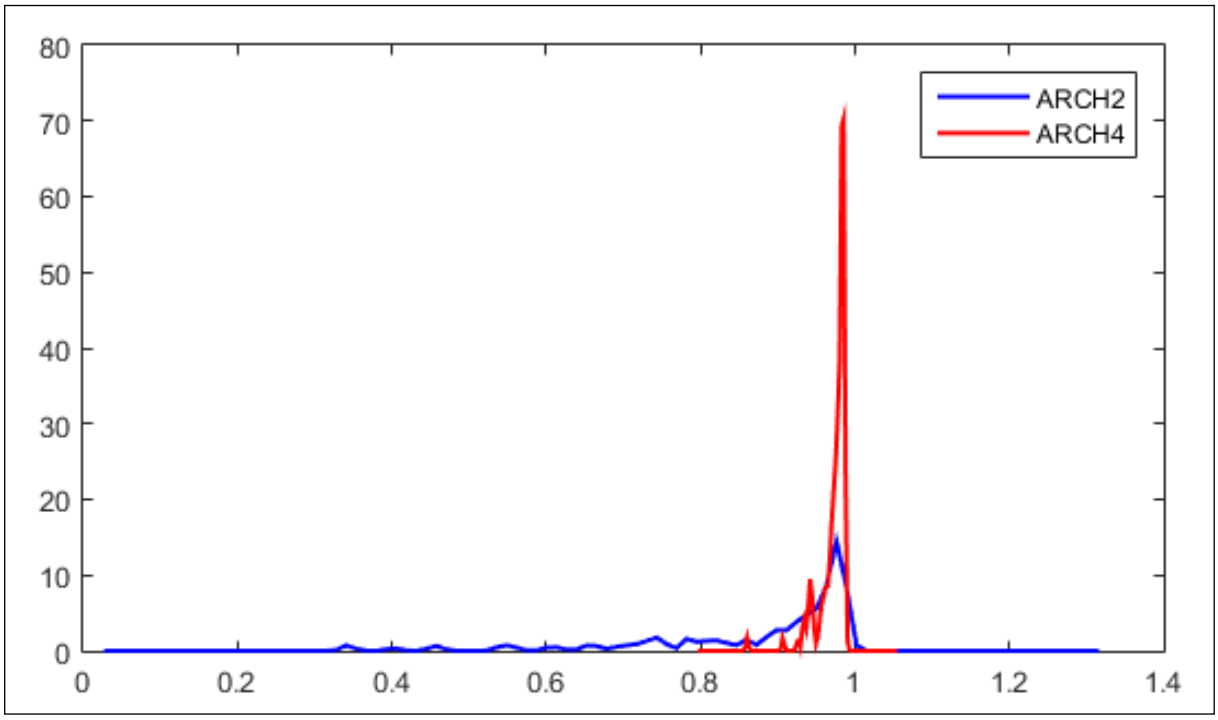


Figure 3.9: Probability density function of prediction errors of Architecture 2 (ARCH2) and Architecture 4 (ARCH4).

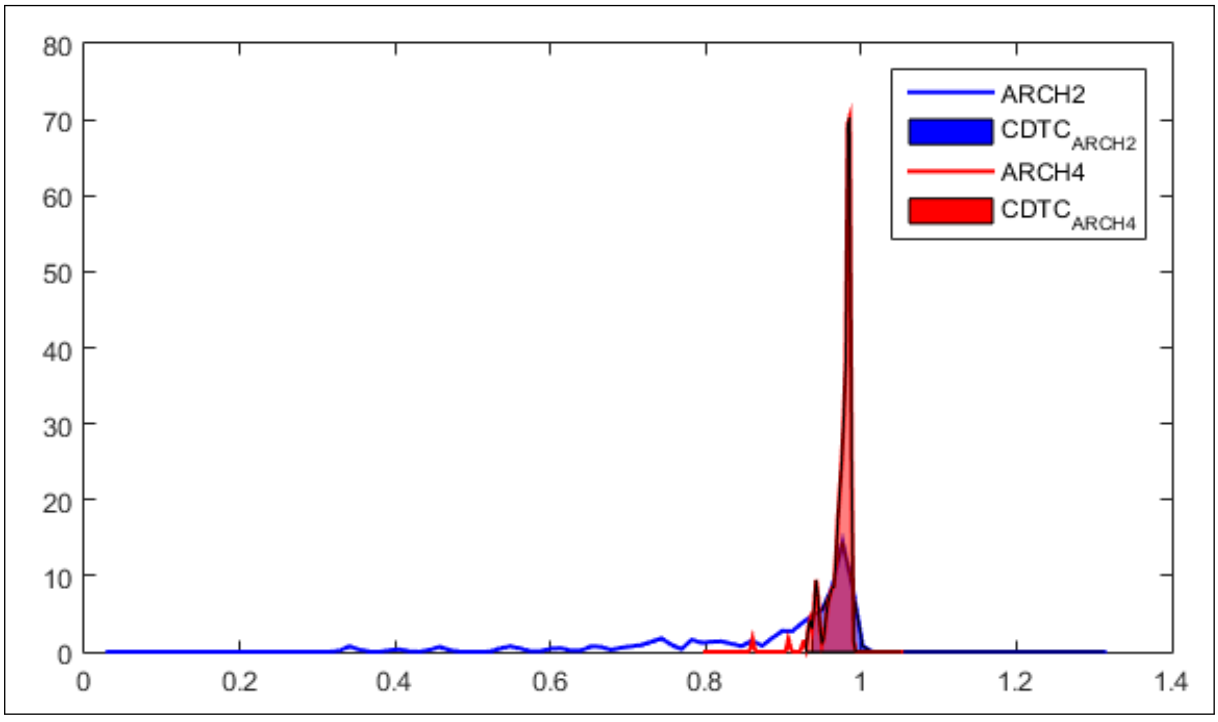


Figure 3.10: Visualization of CDTC_{ARCH2} and CDTC_{ARCH4}.

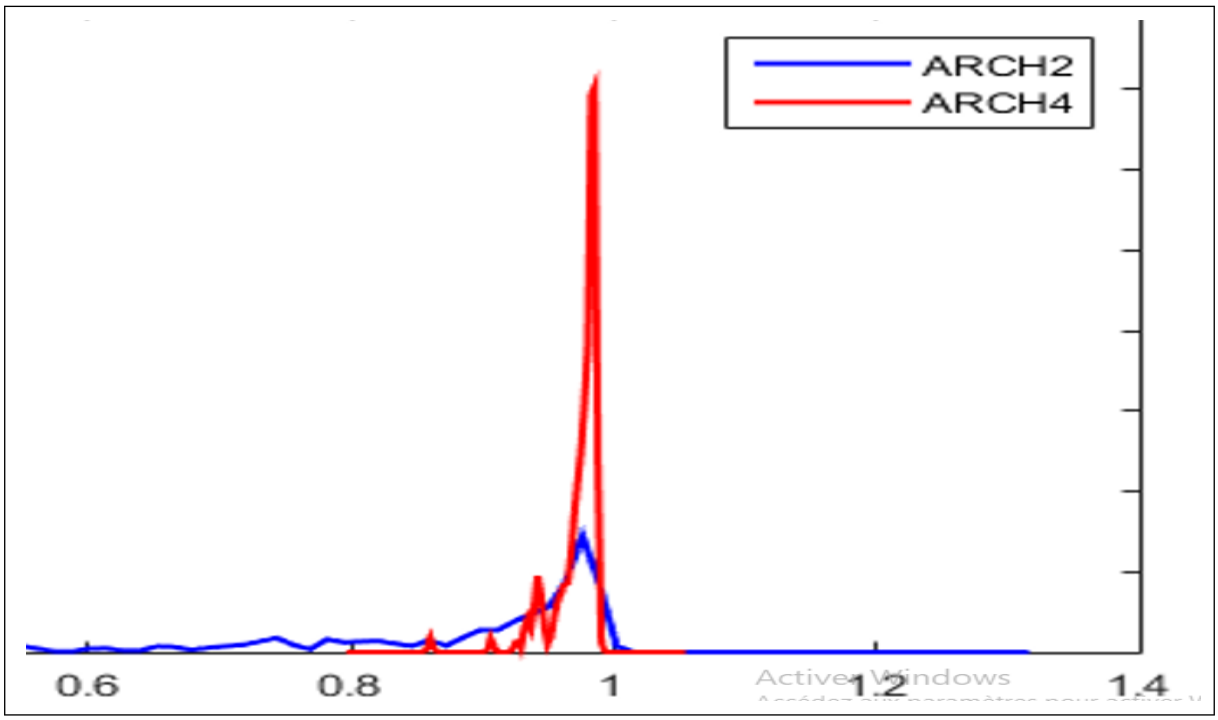


Figure 3.11: A zoomed view to the probability density function of prediction errors of Architecture 2 (ARCH2) and Architecture 4 (ARCH4)

It is interesting to notice, as illustrated in Figure 3.11, which provides a zoomed view of the PDF for architecture 2, that it does not respect the finite endpoint of the support. The chosen error metric, R^2, has a bounded support of $[0, 1]$, and since the errors are close to 1, we observed an overflow at the boundary. Normally, in such cases, the application of the Diffeomorphic KDE, rather than the fast KDE, would be more appropriate. The Diffeomorphic KDE is a generalization of KDE, specifically developed to handle cases with respect to the support. However, in this case, the overflow is insignificant, and it is clear that architecture 4 is outperforming architecture 2.

Table 3.4: CDTC values of studied architecture models

Architecture model	**CDTC**
Architecture 1	0.4004
Architecture 2	0.5514
Architecture 3	0.7375
Architecture 4	0.9804

The values CDTC are reported in table 3.4. The CDTC values highlight the significant impact of PCA on enhancing the model stability, increasing it from 0.4 with Architecture 1

(ARCH1) to 0.73 with Architecture 3 (ARCH3), and from 0.55 with Architecture 2 (ARCH2) to 0.98 with Architecture 4 (ARCH4).

The major limitation of our approach is that it requires the repeating of the system a statistically significant number of times (usually 100 times), which can be very time-consuming with some systems. But we strongly believe that as Machine learning are gaining more and more importance in all aspects of the life even in the critical ones and many people rely blindly on these technologies, it is worth the extra time and effort to have a more insightful evaluation of the model or the algorithm

3.7 Conclusion

In this chapter, we applied a probabilistic criterion based on the FKDE to assess the impact of dimensionality reduction on LSTM models. We considered the error variation as a realization of a continuous random variable, and we estimated their probability density function. This estimation uncovered that the models without PCA tend to converge to two different means. It is important to highlight that none of the other evaluation metrics have captured this tendency. And as You cant manage what you cant measure (a quote often attributed to Peter Drucker, a renowned management consultant, educator, and author) stands the importance of our method. We also emphasized the importance of using a significant number of iterations, as relying on too few iterations can lead to entirely inaccurate conclusions.

Hyperspectral Image Segmentation Using Geometric Deep Learning

Contents

Chapter 4 Abstract

This chapter addresses the segmentation of hyperspectral images using advanced geometric deep learning techniques. It begins with an introduction to hyperspectral imaging and highlights the challenges inherent in segmenting high-dimensional hyperspectral data.

The chapter then provides an overview of key concepts relevant to hyperspectral image segmentation. This includes a discussion on the application of Principal Component Analysis (PCA) in hyperspectral image processing, which aids in dimensionality reduction and feature extraction. The section also covers multi-scale graph construction, laying the groundwork for the subsequent application of geometric deep learning methods.

In the methodology section, we delve into specific deep learning architectures, including Graph Convolutional Networks (GCNs) and Graph Attention Networks (GATs). A hybrid GCN-GAT model is proposed, combining the strengths of both architectures to enhance segmentation performance. The proposed approach is described in detail, alongside the training and testing procedures employed in the experiments.

The experimental setup section outlines the datasets used for evaluation, as well as the metrics for assessing segmentation performance. This chapter presents results and discussions centered on the Pavia University dataset, showcasing the effectiveness of the proposed geometric deep learning approach.

In conclusion, this chapter summarizes the findings, highlighting the potential of geometric deep learning methods for improving hyperspectral image segmentation and suggesting future avenues for research and development in this field.

4.1 Introduction

Hyperspectral imaging (HSI) has emerged as a fundamental technology in remote sensing, enabling the acquisition of detailed spectral information across numerous spectral bands in a single image. This rich spectral data facilitates the identification and classification of materials and objects with high precision. However, factors like limited training data, mixed pixels, atmospheric effects, and geometric distortions raise significant challenges for accurate and efficient image segmentation [Gao et al., 2018]. Traditional machine learning approaches often struggle to capture the intricate spatial-spectral relationships present in HSI data, leading to suboptimal performance [Grewal et al., 2023].

In recent years, geometric deep learning (GDL) has gained popularity as a powerful framework for processing non-Euclidean data structures, such as graphs and manifolds [Bronstein et al., 2017]. Using the graph-based representation of data, GDL methods can effectively model the spatial dependencies and structural information in hyperspectral images, enhancing segmentation accuracy. Among the various GDL techniques, Graph Convolutional Networks (GCN) and Graph Attention Networks (GAT) have demonstrated remarkable success in capturing both local and global features, making them suitable for complex image segmentation tasks [Shi et al., 2023].

This chapter provides a brief overview of hyperspectral image segmentation, emphasizing the advancements brought by geometric deep learning. We compared GCN and GAT, with and without PCA and introduced a novel hybrid model designed to leverage the strengths of both architectures. The subsequent sections detail our proposed approach, including the algorithmic framework, experimental setup, and evaluation metrics. Finally, we present and discuss the experimental results, highlighting the efficacy of our model across various benchmark datasets.

4.2 An Overview on Hyperspectral Image Segmentation

Hyperspectral image segmentation involves partitioning an HSI into meaningful regions corresponding to different materials or objects. Traditional segmentation methods, such as k-means clustering [Ranjan et al., 2017], spectral angle mapper (SAM) [Park et al., 2007], and support vector machines (SVM) [Melgani and Bruzzone, 2004] [Pal and Foody, 2010], Random Forests (RF) [Ham et al., 2005], primarily rely on spectral information. While these methods can achieve reasonable accuracy, they often neglect the spatial context, leading to fragmented or noisy segmentation results.

To address these limitations, spatial-spectral approaches have been developed, which incorporate both spectral and spatial information to improve segmentation performance. Techniques such as Markov Random Fields (MRF) [Li et al., 2011] and Conditional Random Fields (CRF) [Zhong et al., 2019] have been employed to model spatial dependencies. Despite their effectiveness, these probabilistic models can be computationally intensive and may not scale well with high-dimensional data.

4.2.1 PCA in Hyperspectral Image Processing

The high dimensionality of hyperspectral data poses significant challenges, including the curse of dimensionality, increased computational complexity, and the risk of overfitting in machine learning models. Dimensionality reduction techniques are thus essential for simplifying the data while preserving its essential characteristics. Principal Component Analysis (PCA) is one of the most widely used dimensionality reduction methods in HSI processing.

In the context of HSI segmentation, PCA helps mitigate the curse of dimensionality, reducing computational complexity and enhancing model convergence. Recent studies have explored the integration of PCA with deep learning models to improve segmentation performance. For instance, combining PCA with Convolutional Neural Networks (CNN) has shown promising results in enhancing feature representation and reducing noise [Liu et al., 2017] [Rodarmel and Shan, 2002] [Kang et al., 2017].

4.2.2 Multi-Scale Graph Construction in HSI

Multi-scale graph construction involves creating multiple graph representations of the HSI data at different spatial or spectral resolutions. This approach allows models to capture both fine-grained and coarse-grained features, enhancing segmentation performance.

Recent studies have demonstrated that multi-scale graph constructions, when integrated with GDL models, significantly improve the robustness and accuracy of HSI segmentation [Ding et al., 2021] [Li et al., 2024].

4.2.3 Geometric Deep Learning for Image Segmentation

Geometric Deep Learning extends traditional deep learning paradigms to non-Euclidean domains, enabling the processing of graph-structured data [Bronstein et al., 2017]. In the context of HSI segmentation, GDL methods represent the image as a graph, where each pixel corresponds to a node, and edges capture the spatial or spectral relationships between pixels. This graph-based representation allows for the effective modeling of spatial dependencies and structural information, which are crucial for accurate segmentation.

4.2.3.1 Graph Convolutional Networks (GCN)

Graph Convolutional Networks generalize the concept of convolution to graph structures, allowing for the aggregation of information from neighboring nodes. GCNs have been successfully applied to various tasks, including node classification, link prediction, and image segmentation. In HSI segmentation, GCNs can effectively capture local spatial-spectral patterns, enhancing classification accuracy [Kipf and Welling, 2016] [Qin et al., 2018] [Qin et al., 2018].

The fundamental operation in GCNs involves the propagation and transformation of node features through multiple layers, where each layer aggregates features from a node's local neighborhood. This hierarchical feature extraction enables the model to learn complex patterns and dependencies inherent in hyperspectral data. However, standard GCNs may face limitations

in capturing long-range dependencies and varying node importance, which can be critical for accurately segmenting heterogeneous regions in HSI.

4.2.3.2 Graph Attention Networks (GAT)

Graph Attention Networks introduce an attention mechanism, enabling the model to weigh the importance of neighboring nodes dynamically. This adaptive weighting facilitates the capture of more nuanced spatial relationships, leading to improved segmentation performance. GATs have demonstrated superior performance in scenarios where the relevance of neighboring nodes varies across the graph [Veličković et al., 2017] [Zhao et al., 2021] [Shi et al., 2023].

In the context of HSI segmentation, GATs can selectively focus on the most relevant spectral and spatial features, enhancing the model's ability to distinguish between different materials and objects. Moreover, the attention mechanism can mitigate the effects of noise and irrelevant information, which are common in hyperspectral data.

4.3 Methodology

This section outlines the methodology employed for hyperspectral image segmentation using geometric deep learning. We focus on three distinct model architectures: Graph Convolutional Networks (GCN) only, Graph Attention Networks (GAT) only, and a Hybrid GCN-GAT model. Each architecture leverages the strengths of GDL to capture spatial-spectral relationships inherent in HSI data. Additionally, we explore the impact of Principal Component Analysis (PCA) on these models to assess dimensionality reduction's effect on segmentation performance.

4.3.1 Graph Convolutional Network (GCN)

The GCN model used in this research consists of the following layers:

1. **Input Layer**: Accepts the normalized spectral features of each pixel.

2. **Graph Convolutional Layers**: Two GCN layers are stacked, with hidden dimensions of 128 and 64, respectively. Each GCN layer performs convolution operations on the graph, aggregating features from neighboring nodes.
3. **Batch Normalization**: Applied after each GCN layer to stabilize and accelerate training.
4. **Activation Function**: ReLU is used to introduce non-linearity.
5. **Dropout Layers**: Applied after each GCN layer to prevent overfitting.
6. **Fully Connected Layer**: Transforms the aggregated features into class scores.
7. **Output Layer**: Produces log-softmax probabilities for each class.

4.3.2 Graph Attention Network (GAT)

The GAT model consists of the following layers:

1. **Input Layer**: Accepts the normalized spectral features of each pixel.
2. **Graph Attention Layers**: Two GAT layers are stacked, each with 4 attention heads. The first GAT layer transforms the input features into 128-dimensional representations, while the second GAT layer reduces them to 64 dimensions.
3. **Batch Normalization**: Applied after each GAT layer to stabilize and accelerate training.
4. **Activation Function**: ELU is used to introduce non-linearity.
5. **Dropout Layers**: Applied after each GAT layer to prevent overfitting.
6. **Fully Connected Layer**: Transforms the aggregated features into class scores.
7. **Output Layer**: Produces log-softmax probabilities for each class.

4.3.3 Hybrid GCN-GAT Model

The Hybrid GCN-GAT model combines the strengths of both GCN and GAT architectures. By integrating graph convolutional and attention-based layers, the model can effectively capture both local feature aggregation and dynamic weighting of neighbor contributions.

The HybridGCNGAT model operates by first constructing a multi-scale graph representation of the hyperspectral image data. This involves creating multiple graphs at different scales, capturing both fine-grained and coarse-grained spatial-spectral relationships among pixels. The model then processes these graphs through parallel GCN and GAT pathways:

1. **Input Layer**: Accepts the normalized spectral features of each pixel.
2. **GCN Branch**:
    - Two GCN layers with hidden dimensions of 128 and 64, respectively.
    - Batch normalization and ReLU activation after each GCN layer.
    - Dropout layers to prevent overfitting.
3. **GAT Branch**:
    - Two GAT layers with 4 attention heads each, transforming features into 128 and 64 dimensions, respectively.
    - Batch normalization and ELU activation after each GAT layer.
    - Dropout layers to prevent overfitting.
4. **Feature Fusion**:
    - Concatenation of the outputs from the GCN and GAT branches.
    - A fully connected layer (FC1) with 256 neurons, followed by batch normalization and ReLU activation.
    - Dropout layer to prevent overfitting.
    - Final fully connected layer (FC2) producing class scores.
5. **Output Layer**: Produces log-softmax probabilities for each class.

4.3.4 The proposed approach

The proposed methodology follows a systematic approach, encompassing data preprocessing, model training, and evaluation. Below is an overview of the algorithmic framework:

1. **Data Loading**: Import hyperspectral image data and corresponding ground truth labels.
2. **Dimensionality Reduction (PCA)**: Optionally apply PCA to reduce the number of spectral bands while retaining significant variance.
3. **Data Normalization**: Normalize the feature vectors to have zero mean and unit variance.

4. **Graph Construction**: Create a graph representation of the HSI using K-Nearest Neighbors (KNN) based on spectral similarity.
5. **Model Initialization**: Initialize the chosen model architecture (GCN, GAT, or Hybrid GCN-GAT).
6. **Training**: Train the model using the training dataset, employing techniques like batch normalization, dropout, and early stopping to enhance performance.
7. **Evaluation**: Assess the model's performance on the test dataset using metrics such as Overall Accuracy (OA), Average Accuracy (AA), Kappa Coefficient, and per-class accuracy. Additionally, compare processing times across different models.
8. **Visualization**: Generate visual representations of the segmentation results and confusion matrices to interpret model performance.

4.3.5 Model Training and Testing

To comprehensively evaluate the proposed models, we conduct experiments using three different architectures: GCN-only, GAT-only, and Hybrid GCN-GAT. Each model is trained and tested under identical conditions to ensure a fair comparison.

a) Training Procedure:

- **Optimizer**: Utilize the AdamP optimizer, which offers adaptive weight decay and efficient convergence properties.
- **Learning Rate Scheduler**: Implement a ReduceLROnPlateau scheduler to adjust the learning rate based on validation loss, promoting stable training.
- **Early Stopping**: Incorporate early stopping with a patience parameter to prevent overfitting by halting training when performance ceases to improve.
- **Batch Normalization and Dropout**: Apply batch normalization and dropout layers throughout the network to enhance generalization and reduce overfitting.

b) Testing Procedure:

- **Inference**: Perform forward passes on the test dataset to obtain class predictions.
- **Performance Metrics**: Calculate OA, AA, Kappa Coefficient, and per-class accuracy to quantify model performance.
- **Processing Time**: Measure the time taken for training and inference to compare the computational efficiency of each model.
- **Confusion Matrix**: Generate confusion matrices to visualize class-wise performance and identify potential misclassifications.

4.4 Experimental Setup

4.4.1 Datasets

We evaluate our proposed model on a widely used hyperspectral datasets the Pavia University (an Urban Area).

The Pavia University dataset consists of 610 $\times$ 340 pixels with 103 spectral bands. It includes 9 classes such as Asphalt, Grass, Gravel, Trees, Metal, Bare Soil, Bitumen, Shadows, and Buildings.

4.4.2 Evaluation Metrics

To assess the performance of the segmentation models, we employ the following metrics:

- **Overall Accuracy (OA)**: Measures the proportion of correctly classified pixels, calculated as:

$$OA = \frac{TP + TN}{TP + TN + FP + FN}$$

where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives.

- **Average Accuracy (AA)**: Computes the average of the per-class accuracies, given by:

$$AA = \frac{1}{C} \sum_{i=1}^{C} \frac{TP_i}{TP_i + FP_i}$$

where C is the number of classes, TP_i is the number of true positives for class i, and FP_i is the number of false positives for class i.

- **Kappa Coefficient** (κ): Evaluates the agreement between predicted and true labels, accounting for chance agreement, calculated as:

$$\kappa = \frac{P_o - P_e}{1 - P_e}$$

where P_o is the observed agreement and P_e is the expected agreement by chance, given by:

$$P_e = \sum_{i=1}^{C} \frac{(TP_i + FP_i)(TP_i + FN_i)}{N^2}$$

and N is the total number of instances.

- **Accuracy per Class**: Provides detailed insight into the model's performance across each class, defined as:

$$\text{Accuracy}_i = \frac{TP_i}{TP_i + FP_i + FN_i}$$

for each class i.

4.5 Results and Discussion

4.5.1 Pavia University Dataset

For the dataset Pavia University, The Cumulative Variance is depicted in the figure 4.1. So we decided to retain 3 dimensions, thus 99.16% of the information.

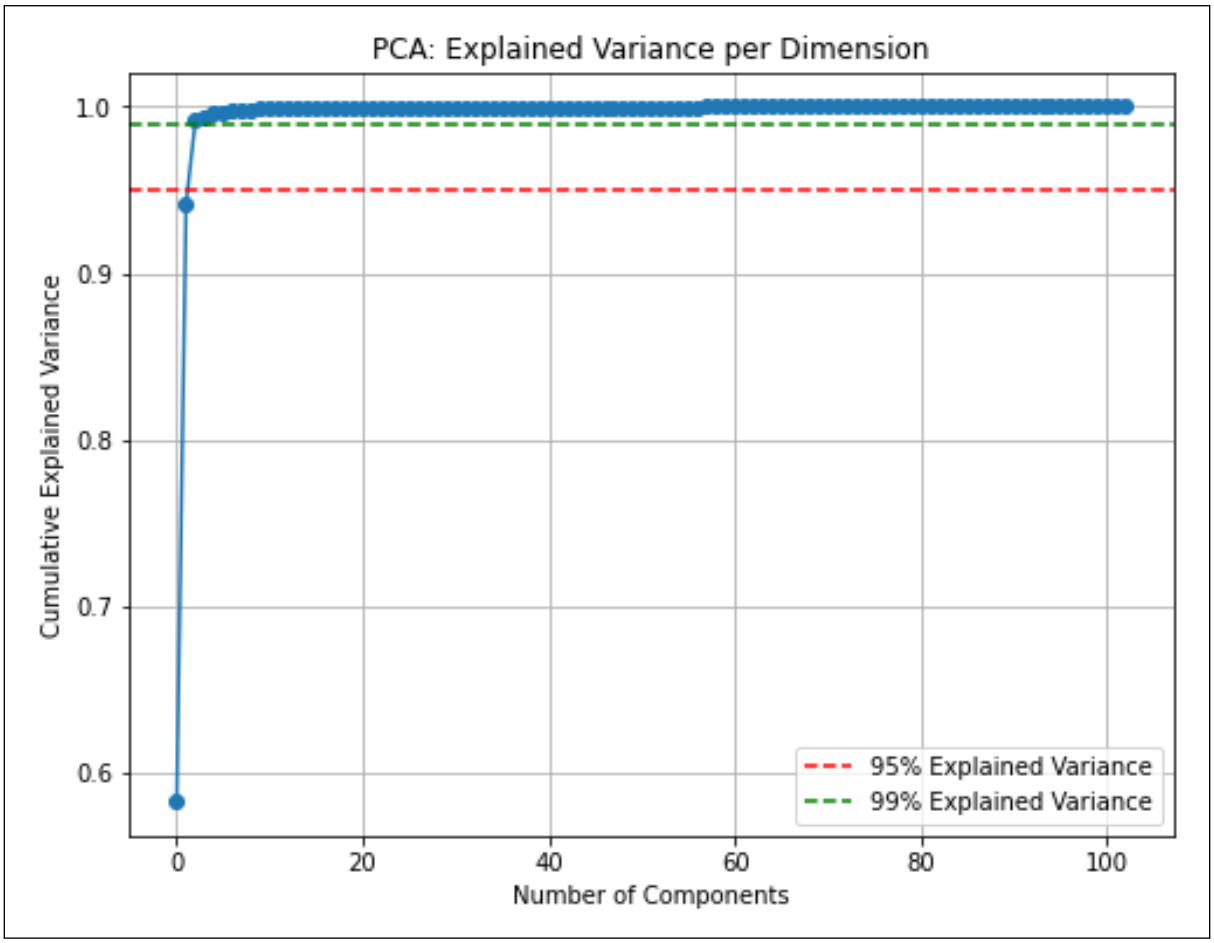


Figure 4.1: Cumulative Variance for the PaviaU Dataset Explained by PCA Components.

Figure 4.2 visually compares the segmentation results of the six algorithms, while Table 4.1 provides a quantitative analysis of their performance.

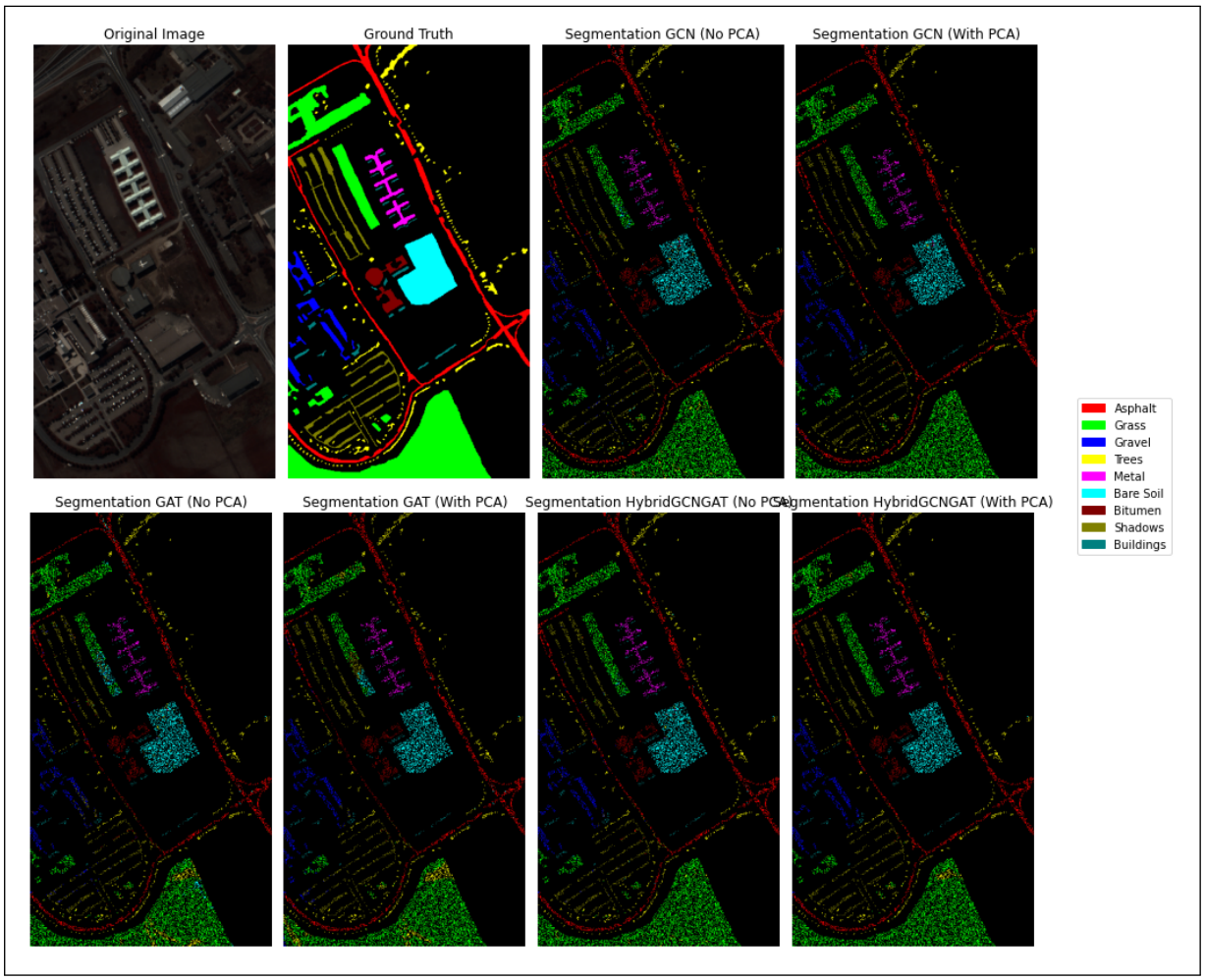


Figure 4.2: Segmentation Results on Pavia University Dataset.

Table 4.1: Performance Metrics for Pavia University Dataset

Algorithm	**Training Time**	**O A %**	**A A %**	**Kappa Score**
GCN (No PCA)	250.98 s	96.24	96.55	0.9505
GCN (With PCA)	220.73 s	98.21	98.84	0.9764
GAT (No PCA)	194.29 s	90.51	93.35	0.8774
GAT (With PCA)	**168.73** s	91.07	95.80	0.8853
Hybrid GCN-GAT (No PCA)	223.38 s	97.50	97.70	0.9670
Hybrid GCN-GAT (With PCA)	199.62 s	**98.27**	**99.15**	**0.9772**

The results presented in Table 4.1 highlight the comparative performance of the six algorithms. Overall, the Hybrid GCN-GAT model, both with and without Principal Component Analysis (PCA), demonstrated superior performance, achieving the highest overall accuracy of 98.27% with PCA and 97.50% without PCA. This indicates that the hybrid approach effectively leverages the strengths of both Graph Convolutional Networks (GCN) and Graph Attention Networks (GAT) to enhance classification accuracy. Notably, the GCN with PCA also performed

exceptionally well, achieving an overall accuracy of 98.21%, which underscores the importance of dimensionality reduction in improving model performance. In contrast, the GAT algorithm, while showing some promise, exhibited lower accuracy, particularly in the absence of PCA, with an overall accuracy of 90.51%.

Table 4.2: Per Class Precision for Pavia University Dataset

Class	GCN	GCN + PCA	GAT	GAT + PCA	GCN-GAT	GCN-GAT + PCA
Asphalt	94.17	95.53	87.68	86.68	94.67	**97.34**
Grass	96.60	**98.28**	87.81	85.86	98.07	97.41
Gravel	94.13	99.37	84.92	96.35	96.51	**99.68**
Trees	98.26	**99.13**	97.82	98.59	98.48	99.02
Metal	99.01	99.75	98.26	99.26	99.26	**100.00**
Bare Soil	98.14	99.01	97.28	**99.87**	98.81	**99.87**
Bitumen	95.24	**100.00**	96.74	**100.00**	96.99	**100.00**
Shadows	93.39	98.46	89.59	95.57	96.56	**99.00**
Buildings	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**

Furthermore, the table 4.2 present the class-level accuracy results. For instance, the GAT (No PCA) model did not perform well with the class "Gravel," achieving only 84.92% accuracy, while the Hybrid GCN-GAT (With PCA) model excelled across all classes, attaining a perfect score of 100% for "Bitumen" and "Buildings." This consistent high performance across multiple classes indicates the robustness of the hybrid approach, making it well-suited for complex classification tasks in hyperspectral imaging. Overall, the findings suggest that integrating multiple deep learning techniques and applying PCA can significantly enhance classification accuracy, providing valuable insights for future research and practical applications in hyperspectral image analysis.

4.6 Conclusion

This chapter presented a start point exploration of hyperspectral image segmentation using geometric deep learning, with a focus on Graph Convolutional Networks and Graph Attention Networks. We introduced a novel hybrid model, HybridGCNGAT, which combines the strengths of both GCN and GAT architectures to achieve superior segmentation performance. Through comprehensive experiments on the Pavia University we demonstrated the significant impact of

PCA-based dimensionality reduction on model accuracy and efficiency. The HybridGCNGAT model consistently outperformed traditional GCN and GAT models, highlighting its potential for advanced hyperspectral image analysis. Future work may explore the integration of additional deep learning techniques and further optimization of graph construction methods to enhance segmentation performance.

GENERAL CONCLUSION

This chapter concludes the Ph.D. thesis, summarizing its key findings and highlighting future research directions. The thesis encompassed two primary domains: unsupervised classification with the Expectation Maximization (EM) algorithm and regression evaluation using Long Short Term Memory (LSTM) networks.

Summary

The first part of the thesis introduced a novel variant of the EM algorithm, the Diffeomorphism EM (EMD), designed to address the boundary issues inherent in applying EM to Gaussian Mixture Models (GMMs) with bounded support. By transforming the data from bounded to unbounded support using a diffeomorphic mapping, EMD applies unsupervised classification on infinite support before reverting the data to its original space. This approach was validated on simulated datasets and successfully applied to ultrasound image segmentation, showcasing its effectiveness. The promising results encourage future research into applying EMD to other medical imaging modalities, such as X-ray, CT scans, and mammography.

In the second part, we evaluated the performance of regression algorithms, focusing on LSTM networks for financial forecasting. Through a comprehensive review of existing regression evaluation metrics, we identified a significant gap: the lack of an evaluation metric that could effectively assess the stability of deep learning models. Most metrics were found to rely on single iterations, which could lead to erroneous conclusions. To address this, we proposed to estimate their error distribution and to calculate the Cumulative Distribution Target Criterion (CDTC), a probabilistic evaluation metric based on non-parametric density estimation. The CDTC, utilizing the Fast Kernel Density Estimator (FKDE) with plug-in bandwidth optimization was applied to evaluate the stability and accuracy of LSTM models, and how they are influenced by dimensionality reduction via PCA. The results confirmed that

the error's distribution and the CDTC provides a deeper and more reliable assessment of model performance compared to traditional metrics.

The last part of this thesis focused on hyperspectral image segmentation, employing geometric deep learning methods. We compared Graph Convolutional Networks (GCNs), Graph Attention Networks (GATs), and a hybrid GCN-GAT model. These models were tested with and without Principal Component Analysis (PCA). The results demonstrated that merging geometric deep learning techniques offers substantial improvements in segmentation accuracy, highlighting the potential of these methods for complex, high-dimensional data like hyperspectral images.

Perspectives and Future Research

The research conducted in this thesis opens several avenues for future exploration.

As a future perspective, we aim to extend the EMD (Diffeomorphism Expectation Maximization) approach to handle multivariate data. Currently, the method operates on univariate data, limiting its application to problems where relationships between multiple variables are not considered. By transforming the algorithm to work with multivariate data, we can significantly broaden its applicability, allowing it to model more complex datasets, capture inter-variable dependencies, and improve its performance in multidimensional classification problems.

The CDTC (Cumulative Distribution Target Criterion) holds significant potential for broader applications beyond its initial use in financial forecasting. One promising direction is its application in evaluating machine learning algorithms across various domains, such as healthcare, image processing, and environmental modeling. The criterion's focus on error distributions rather than single-point metrics makes it especially valuable for assessing models in fields where accurate prediction intervals and uncertainty estimation are crucial.

The promising results obtained from the hybrid GCN-GAT model in hyperspectral image classification suggest several avenues for future research and development. One significant perspective is the exploration of more advanced ensemble learning techniques that could further

enhance the robustness and accuracy of classification models. Investigating other combinations of deep learning architectures and their potential synergies could yield models that better capture the intricacies of hyperspectral data.

Another area of interest lies in the optimization of hyperparameters and network architectures to improve model performance. Implementing automated hyperparameter tuning methods, such as Bayesian optimization or genetic algorithms, could provide insights into the most effective configurations for achieving higher accuracy and efficiency.

BIBLIOGRAPHY

B. Abhisheka, S. K. Biswas, and B. Purkayastha. A comprehensive review on breast cancer detection, classification and segmentation using deep learning. *Archives of Computational Methods in Engineering*, 30(8):5023–5052, 2023.

T. Adamek and N. E. O'Connor. A multiscale representation method for nonrigid shapes with a single closed contour. *IEEE Transactions on Circuits and Systems for Video Technology*, 14 (5):742–753, 2004.

Z. D. Akehir and E. Kiliç. How to handle data imbalance and feature selection problems in cnn-based stock price forecasting. *IEEE Access*, 10:31297–31305, 2022. doi: 10.1109/ACCESS.2022.3160797.

W. Al-Dhabyani, M. Gomaa, H. Khaled, and A. Fahmy. Dataset of breast ultrasound images. *Data in Brief*, 28:104863, 2020. ISSN 2352-3409. doi: https://doi.org/10.1016/j.dib.2019.104863. URL https://www.sciencedirect.com/science/article/pii/S2352340919312181.

R. Allison and J. Dunkley. Comparison of sampling techniques for bayesian parameter estimation. *Monthly Notices of the Royal Astronomical Society*, 437(4):3918–3928, 2014.

N. M. Alpert and F. Yuan. A general method of bayesian estimation for parametric imaging of the brain. *NeuroImage*, 41(2):1183–1189, 2008.

A. Z. Alsinan, V. M. Patel, and I. Hacihaliloglu. Automatic segmentation of bone surfaces from ultrasound using a filter-layer-guided cnn. *International journal of computer assisted radiology and surgery*, 14:775–783, 2019.

M. Y. Ansari, Y. Yang, P. K. Meher, and S. P. Dakua. Dense-psp-unet: A neural network for fast inference liver ultrasound segmentation. *Computers in Biology and Medicine*, 153:106478, 2023.

M. Bal-Ghaoui, M. H. El Yousfi Alaoui, A. Jilbab, and A. Bourouhou. U-net transfer learning backbones for lesions segmentation in breast ultrasound images. *International*

Journal of Electrical and Computer Engineering, 13(5), 10 2023. doi: 10.11591/ijece.v13i5.pp5747-5754.

L. A. Barboza and F. G. Viens. Parameter estimation of gaussian stationary processes using the generalized method of moments. *Electronic Journal of Statistics*, 11(1):401 – 439, 2017. doi: 10.1214/17-EJS1230. URL https://doi.org/10.1214/17-EJS1230.

M. J. Beal. *Variational algorithms for approximate Bayesian inference*. University of London, University College London (United Kingdom), 2003.

S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE transactions on pattern analysis and machine intelligence*, 24(4):509–522, 2002.

N. E. Ben Slimen, M. Troudi, and J. Chaouachi. Critère probabiliste de comparaison des métaheuristiques pour les problèmes à grande echelle. In *TAIMA Actes*, 06 2022.

A. Benedetti, F. Di Giuseppe, L. Jones, V.-H. Peuch, S. Rémy, and X. Zhang. The value of satellite observations in the analysis and short-range prediction of asian dust. *Atmospheric Chemistry and Physics*, 19(2):987–998, 2019.

A. BenKhlifa and F. Ghorbel. An almost complete curvature scale space representation: Euclidean case. *Signal Processing: Image Communication*, 75:32–43, 2019.

A. Bernacchia and S. Pigolotti. Self-consistent method for density estimation. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 73(3):407–422, 2011.

H. N. Bhandari, B. Rimal, N. R. Pokhrel, R. Rimal, K. R. Dahal, and R. K. Khatri. Predicting stock market index using lstm. *Machine Learning with Applications*, 9:100320, 2022. ISSN 2666-8270. doi: https://doi.org/10.1016/j.mlwa.2022.100320. URL https://www.sciencedirect.com/science/article/pii/S2666827022000378.

C. Biernacki, G. Celeux, and G. Govaert. Choosing starting values for the em algorithm for getting the highest likelihood in multivariate gaussian mixture models. *Computational Statistics & Data Analysis*, 41(3-4):561–575, 2003.

A. W. Bowman. An alternative method of cross-validation for the smoothing of density estimates. *Biometrika*, 71(2):353–360, 1984.

M. A. Boyacioglu and D. Avci. An adaptive network-based fuzzy inference system (anfis) for the prediction of stock market return: the case of the istanbul stock exchange. *Expert Systems with Applications*, 37(12):7908–7912, 2010.

M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017. doi: 10.1109/MSP.2017.2693418.

J. Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, 8(6):679–698, 1986.

J. Cao and J. Wang. Exploration of stock index change prediction model based on the combination of principal component analysis and artificial neural network. *Soft Computing*, 24:7851–7860, 2020.

O. Cappé and E. Moulines. On-line expectation-maximization algorithm for latent data models. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 71(3):593–613, 2009.

G. Celeux and J. Diebolt. The sem algorithm: a probabilistic teacher algorithm derived from the em algorithm for the mixture problem. *Computational statistics quarterly*, 2:73–82, 1985.

G. Celeux, D. Chauveau, and J. Diebolt. Stochastic versions of the em algorithm: An experimental study in the mixture case. *Journal of Statistical Computation and Simulation*, 55(4):287–314, 1996.

T. Chai and R. Draxler. Root mean square error (rmse) or mean absolute error (mae)? arguments against avoiding rmse in the literature. *Geoscientific Model Development*, 7:1247–1250, 06 2014. doi: 10.5194/gmd-7-1247-2014.

H. Chang, Z. Chen, Q. Huang, J. Shi, and X. Li. Graph-based learning for segmentation of 3d ultrasound images. *Neurocomputing*, 151:632–644, 2015.

F. Chelangat and T. Afullo. K-means initialisation for the modelling of plc noise in indoor environment. In *2023 IEEE AFRICON*, pages 1–6, 2023. doi: 10.1109/AFRICON55910.2023.10293552.

Y.-C. Chen and W.-C. Huang. Constructing a stock-price forecast cnn model with gold and crude oil indicators. *Applied Soft Computing*, 112:107760, 2021.

Y.-X. Chen, Y.-J. Xiong, X.-H. Qiu, and C.-M. Xia. Harmonious parameters and performance: Lightweight convolutional stage and local feature weighted fusion mlp for medical image segmentation. *Biomedical Signal Processing and Control*, 98:106726, 2024.

D. Chicco, M. J. Warrens, and G. Jurman. The coefficient of determination r-squared is more informative than smape, mae, mape, mse and rmse in regression analysis evaluation. *Peerj computer science*, 7:e623, 2021.

J. Choi, H. Eom, and S.-M. Baek. A wind power probabilistic model using the reflection method and multi-kernel function kernel density estimation. *Energies*, 15(24):9436, 2022.

S. Cleger-Tamayo, J. M. Fernández-Luna, and J. F. Huete. On the use of weighted mean absolute error in recommender systems. In *RecSys'12 - Proceedings of the 6th ACM Conference on Recommender Systems*, 2012. URL https://api.semanticscholar.org/CorpusID:16692759.

G. Coq, X. Li, O. Alata, Y. Pousset, and C. Olivier. Law recognition via histogram-based estimation. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3425–3428. IEEE, 2009.

A. De Myttenaere, B. Golden, B. Le Grand, and F. Rossi. Mean absolute percentage error for regression models. *Neurocomputing*, 192:38–48, 2016.

P. Deheuvels. Estimation non paramétrique de la densité par histogrammes généralisés. *Revue de statistique appliquée*, 25(3):5–42, 1977.

A. Delaigle and I. Gijbels. Estimation of integrated squared density derivatives from a contaminated sample. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 64(4):869–886, 2002.

B. Delyon, M. Lavielle, and E. Moulines. Convergence of a stochastic approximation version of the em algorithm. *Annals of statistics*, pages 94–128, 1999.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society: series B (methodological)*, 39(1): 1–38, 1977.

J. DiNardo and J. L. Tobias. Nonparametric density and regression estimation. *Journal of Economic Perspectives*, 15(4):11–28, 2001.

F. Ding. Least squares parameter estimation and multi-innovation least squares methods for linear fitting problems from noisy data. *Journal of Computational and Applied Mathematics*, 426:115107, 2023.

G. Ding and L. Qin. Study on the prediction of stock price based on the associated network model of lstm. *International Journal of Machine Learning and Cybernetics*, 11:1307–1317, 2020.

Y. Ding, X. Zhao, Z. Zhang, W. Cai, and N. Yang. Multiscale graph sample and aggregate network with context-aware learning for hyperspectral image classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 14:4561–4572, 2021.

K. Drukker, M. L. Giger, K. Horsch, M. A. Kupinski, C. J. Vyborny, and E. B. Mendelson. Computerized lesion detection on breast ultrasound. *Medical physics*, 29(7):1438–1446, 2002.

S. Elghoul and F. Ghorbel. Fast global sa (2, r) shape registration based on invertible invariant descriptor. *Signal Processing: Image Communication*, 90:116058, 2021a.

S. Elghoul and F. Ghorbel. Partial contour matching based on affine curvature scale space descriptors. In R. Kountchev, R. Mironov, and S. Li, editors, *New Approaches for Multidimensional Signal Processing*, pages 73–81, Singapore, 2021b. Springer Singapore. ISBN 978-981-33-4676-5.

S. Elghoul and F. Ghorbel. A fast and robust affine-invariant method for shape registration under partial occlusion. *International Journal of Multimedia Information Retrieval*, 11(1): 39–59, 2022.

L. Fang, T. Qiu, Y. Liu, and C. Chen. Active contour model driven by global and local intensity information for ultrasound image segmentation. *Computers & Mathematics with Applications*, 75(12):4286–4299, 2018.

L. Fang, L. Zhang, Y. Yao, and L. Chen. Ultrasound image segmentation using an active contour model and learning-structured inference. *Multimedia Tools and Applications*, 81 (10):13389–13407, 2022.

E. Fix and J. Hodges. Discriminatory analysis. nonparametric discrimination: Consistency properties. *International Statistical Review/Revue Internationale de Statistique*, 57(3): 238–247, 1989.

J. M. Francos, A. Narasimhan, and J. W. Woods. Maximum likelihood parameter estimation of textures using a wold-decomposition based model. *IEEE Transactions on Image Processing*, 4(12):1655–1666, 1995. doi: 10.1109/TIP.1995.8875999.

H. Fu, Z. Tian, M. Ran, and M. Fan. Novel affine-invariant curve descriptor for curve matching and occluded object recognition. *IET Computer Vision*, 7(4):279–292, 2013.

M. Galrinho, C. Rojas, and H. Hjalmarsson. A weighted least-squares method for parameter estimation in structured models. In *53rd IEEE conference on decision and control*, pages 3322–3327. IEEE, 2014.

Q. Gao, S. Lim, and X. Jia. Hyperspectral image classification using convolutional neural networks and multiple feature learning. *Remote Sensing*, 10(2):299, 2018.

M. M. Ghazi, M. Nielsen, A. Pai, M. Modat, M. J. Cardoso, S. bastien Ourselin, and L. Sørensen. On the initialization of long short-term memory networks. In *Neural Information Processing*, pages 275–286. Springer International Publishing, 2019. doi: 10.1007/978-3-030-36708-4_23. URL https://doi.org/10.1007%2F978-3-030-36708-4_23.

T. Gneiting. Making and evaluating point forecasts. *Journal of the American Statistical Association*, 106(494):746–762, 2011.

W. Gómez, L. Leija, A. V. Alvarenga, A. F. C. Infantosi, and W. C. A. Pereira. Computerized lesion segmentation of breast ultrasound based on marker-controlled watershed transformation. *Medical physics*, 37(1):82–95, 2010.

P. Goodwin and R. Lawton. On the asymmetry of the symmetric mape. *International journal of forecasting*, 15(4):405–408, 1999.

R. Gopalan, P. Turaga, and R. Chellappa. Articulation-invariant representation of non-planar shapes. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *Computer Vision – ECCV 2010*, pages 286–299, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. ISBN 978-3-642-15558-1.

R. Grewal, S. S. Kasana, and G. Kasana. Hyperspectral image segmentation: a comprehensive survey. *Multimedia Tools and Applications*, 82(14):20819–20872, 2023.

M. Gurjar, P. Naik, G. Mujumdar, and T. Vaidya. Stock market prediction using ann. *International Research Journal of Engineering and Technology*, 5(3):2758–61, 2018.

P. Hall. Comparison of two orthogonal series methods of estimating a density and its derivatives on an interval. *Journal of multivariate analysis*, 12(3):432–449, 1982.

P. Hall and J. S. Marron. Extent to which least-squares cross-validation minimises integrated square error in nonparametric density estimation. *Probability Theory and Related Fields*, 74 (4):567–581, 1987.

P. Hall and J. S. Marron. Local minima in cross-validation functions. *Journal of the Royal Statistical Society: Series B (Methodological)*, 53(1):245–252, 1991.

P. Hall, J. Marron, and B. U. Park. Smoothed cross-validation. *Probability theory and related fields*, 92(1):1–20, 1992.

J. Ham, Y. Chen, M. M. Crawford, and J. Ghosh. Investigation of the random forest framework for classification of hyperspectral data. *IEEE Transactions on Geoscience and Remote Sensing*, 43(3):492–501, 2005.

A. K. Hamou and M. R. El-Sakka. A novel segmentation technique for carotid ultrasound images. In *2004 IEEE international conference on acoustics, speech, and signal processing*, volume 3, pages iii–521. IEEE, 2004.

W. Härdle. *Smoothing techniques: with implementation in S*. Springer Science & Business Media, 1991.

T. O. Hodson, T. M. Over, and S. S. Foks. Mean squared error, deconstructed. *Journal of Advances in Modeling Earth Systems*, 13(12):e2021MS002681, 2021.

K. Horsch, M. L. Giger, L. A. Venta, and C. J. Vyborny. Automatic segmentation of breast lesions on ultrasound. *Medical physics*, 28(8):1652–1659, 2001.

Z. Hu, Y. Zhao, and M. Khushi. A survey of forex and stock price prediction using deep learning. *Applied System Innovation*, 4(1), 2021. ISSN 2571-5577. doi: 10.3390/asi4010009. URL https://www.mdpi.com/2571-5577/4/1/9.

Y. Hua, Z. Zhao, R. Li, X. Chen, Z. Liu, and H. Zhang. Deep learning with long short-term memory for time series prediction. *IEEE Communications Magazine*, 57(6):114–119, 2019. doi: 10.1109/MCOM.2019.1800155.

Q. Huang, X. Bai, Y. Li, L. Jin, and X. Li. Optimized graph-based segmentation for ultrasound images. *Neurocomputing*, 129:216–224, 2014.

Q.-H. Huang, S.-Y. Lee, L.-Z. Liu, M.-H. Lu, L.-W. Jin, and A.-H. Li. A robust graph-based segmentation method for breast tumors in ultrasound images. *Ultrasonics*, 52(2):266–275, 2012.

X. Huang, N. Paragios, and D. N. Metaxas. Shape registration in implicit spaces using information theory and free form deformations. *IEEE transactions on pattern analysis and machine intelligence*, 28(8):1303–1318, 2006.

Y.-L. Huang and D.-R. Chen. Watershed segmentation for breast tumor in 2-d sonography. *Ultrasound in medicine & biology*, 30(5):625–632, 2004.

R. J. Hyndman and A. B. Koehler. Another look at measures of forecast accuracy. *International journal of forecasting*, 22(4):679–688, 2006.

Z. Jin, Y. Yang, and Y. Liu. Stock closing price prediction based on sentiment analysis and lstm. *Neural Computing and Applications*, 32:9713–9729, 2020.

M. Jørgensen, T. Halkjelsvik, and K. Liestøl. When should we (not) use the mean magnitude of relative error (mmre) as an error measure in software development effort estimation? *Information and Software Technology*, 143:106784, 2022.

X. Kang, X. Xiang, S. Li, and J. A. Benediktsson. Pca-based edge-preserving features for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(12):7140–7151, 2017.

N. Kaothanthong, J. Chun, and T. Tokuyama. Distance interior ratio: A new shape signature for 2d shape retrieval. *Pattern recognition letters*, 78:14–21, 2016.

J. Kim and C. D. Scott. Robust kernel density estimation. *The Journal of Machine Learning Research*, 13(1):2529–2565, 2012.

S. Kim and H. Kim. A new metric of absolute percentage error for intermittent demand forecasts. *International Journal of Forecasting*, 32(3):669–679, 2016. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2015.12.003. URL https://www.sciencedirect.com/science/article/pii/S0169207016000121.

T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

J. L. Kirkby, Á. Leitao, and D. Nguyen. Spline local basis methods for nonparametric density estimation. *Statistic Surveys*, 17:75–118, 2023.

R. Kumar, P. Kumar, and Y. Kumar. Multi-step time series analysis and forecasting strategy using arima and evolutionary algorithms. *International Journal of Information Technology*, 14(1):359–373, 2022.

M. M. Kumbure, C. Lohrmann, P. Luukka, and J. Porras. Machine learning techniques and data for stock market forecasting: A literature review. *Expert Systems with Applications*, 197: 116659, 2022.

A. Kurani, P. Doshi, A. Vakharia, and M. Shah. A comprehensive comparative study of artificial neural network (ann) and support vector machines (svm) on stock forecasting. *Annals of Data Science*, 10(1):183–208, 2023.

W. Kwedlo. A new random approach for initialization of the multiple restart em algorithm for gaussian model-based clustering. *Pattern Analysis and Applications*, 18:757–770, 2015.

L. J. Latecki, R. Lakamper, and T. Eckhardt. Shape descriptors for non-rigid shapes with a single closed contour. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, volume 1, pages 424–429. IEEE, 2000.

M. Lawal and D. Yi. Polar contrast attention and skip cross-channel aggregation for efficient learning in u-net. *Computers in Biology and Medicine*, 181:109047, 2024.

A. Li, Y. Sun, C. Feng, Y. Cheng, and L. Xi. Multiscale graph convolution residual network for hyperspectral image classification. *Journal of Applied Remote Sensing*, 18(1): 014504–014504, 2024.

H. Li, J. Fang, S. Liu, X. Liang, X. Yang, Z. Mai, M. T. Van, T. Wang, Z. Chen, and D. Ni. Cr-unet: A composite network for ovary and follicle segmentation in ultrasound images. *IEEE journal of biomedical and health informatics*, 24(4):974–983, 2019.

J. Li, J. M. Bioucas-Dias, and A. Plaza. Spectral–spatial hyperspectral image segmentation using subspace multinomial logistic regression and markov random fields. *IEEE Transactions on Geoscience and Remote Sensing*, 50(3):809–823, 2011.

Q. Li, J. Tan, J. Wang, and H. Chen. A multimodal event-driven lstm model for stock prediction using online news. *IEEE Transactions on Knowledge and Data Engineering*, 33 (10):3323–3337, 2021. doi: 10.1109/TKDE.2020.2968894.

X. Li, Q. Yu, C. Tang, Z. Lu, and Y. Yang. Application of feature selection based on multilayer ga in stock prediction. *Symmetry*, 14(7), 2022. ISSN 2073-8994. doi: 10.3390/sym14071415. URL https://www.mdpi.com/2073-8994/14/7/1415.

C. M. Liapis, A. Karanikola, and S. Kotsiantis. Investigating deep stock market forecasting with sentiment analysis. *Entropy*, 25(2):219, 2023.

Y.-T. Lin and G. D. Finlayson. On the optimization of regression-based spectral reconstruction. *Sensors*, 21(16):5586, 2021.

H. Ling and D. W. Jacobs. Shape classification using the inner-distance. *IEEE transactions on pattern analysis and machine intelligence*, 29(2):286–299, 2007.

B. Liu, H.-D. Cheng, J. Huang, J. Tian, X. Tang, and J. Liu. Probability density difference-based active contour for ultrasound image segmentation. *Pattern Recognition*, 43(6):2028–2042, 2010.

X. Liu, Q. Sun, B. Liu, B. Huang, and M. Fu. Hyperspectral image classification based on convolutional neural network and dimension reduction. In *2017 Chinese automation congress (CAC)*, pages 1686–1690. IEEE, 2017.

X. Liu, H. Liu, Q. Guo, and C. Zhang. Adaptive wavelet transform model for time series data prediction. *Soft Computing*, 24:5877–5884, 2020.

C.-M. Lo, R.-T. Chen, Y.-C. Chang, Y.-W. Yang, M.-J. Hung, C.-S. Huang, and R.-F. Chang. Multi-dimensional tumor detection in automated whole breast ultrasound using topographic watershed. *IEEE transactions on medical imaging*, 33(7):1503–1511, 2014.

K. Luan, Z. Li, and J. Li. An efficient end-to-end cnn for segmentation of bone surfaces from ultrasound. *Computerized Medical Imaging and Graphics*, 84:101766, 2020.

J. Ma, F. Wu, T. Jiang, Q. Zhao, and D. Kong. Ultrasound image-based thyroid nodule automatic segmentation using convolutional neural networks. *International journal of computer assisted radiology and surgery*, 12:1895–1910, 2017.

M. Magdon-Ismail and A. Atiya. Neural networks for density estimation. *Advances in Neural Information Processing Systems*, 11, 1998.

F. Mai, C. Chang, and Y. Hung. Affine-invariant shape matching and recognition under partial occlusion. In *2010 IEEE international conference on image processing*, pages 4605–4608. IEEE, 2010.

S. Makridakis, E. Spiliotis, and V. Assimakopoulos. M5 accuracy competition: Results, findings, and conclusions. *International Journal of Forecasting*, 38(4):1346–1364, 2022.

ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2021.11.013. URL https://www.sciencedirect.com/science/article/pii/S0169207021001874. Special Issue: M5 competition.

P. Malec and M. Schienle. Nonparametric kernel density estimation near the boundary. *Computational Statistics & Data Analysis*, 72:57–76, 2014.

C. Mari and C. Baldassari. Em algorithm initialization for mixture models: A complex network driven approach for financial time series. *Information Sciences*, 617:1–16, 2022.

J. S. Marron and D. Ruppert. Transformations to reduce boundary bias in kernel density estimation. *Journal of the Royal Statistical Society: Series B (Methodological)*, 56(4): 653–671, 1994.

J. C. Marshall and M. L. Hazelton. Boundary kernels for adaptive density estimators on regions with irregular boundaries. *Journal of Multivariate Analysis*, 101(4):949–963, 2010.

J. L. Mateo and A. Fernández-Caballero. Finding out general tendencies in speckle noise reduction in ultrasound images. *Expert systems with applications*, 36(4):7786–7797, 2009.

G. J. McLachlan and T. Krishnan. *The EM algorithm and extensions*. John Wiley & Sons, 2007.

F. Melgani and L. Bruzzone. Classification of hyperspectral remote sensing images with support vector machines. *IEEE Transactions on geoscience and remote sensing*, 42(8):1778–1790, 2004.

V. Melnykov and I. Melnykov. Initializing the em algorithm in gaussian mixture models with an unknown number of components. *Computational Statistics & Data Analysis*, 56 (6):1381–1395, 2012.

F. Milletari, S.-A. Ahmadi, C. Kroll, A. Plate, V. Rozanski, J. Maiostre, J. Levin, O. Dietrich, B. Ertl-Wagner, K. Bötzel, and N. Navab. Hough-cnn: Deep learning for segmentation of deep brain regions in mri and ultrasound. *Computer Vision and Image Understanding*, 164: 92–102, 2017.

G. Mohana Priya and P. Mohamed Fathimal. Fetal head ultrasound image segmentation using region-based, edge-based and clustering strategies. In *Proceedings of International Conference on Recent Trends in Computing: ICRTC 2022*, pages 581–592. Springer, 2023.

F. Mokhtarian and S. Abbasi. Affine curvature scale space with affine length parametrisation. *Pattern Analysis & Applications*, 4:1–8, 2001.

F. Mokhtarian, S. Abbasi, and J. Kittler. Robust and efficient shape indexing through curvature scale space. In *British Machine Vision Conference*, 1996. URL https://api.semanticscholar.org/CorpusID:2121487.

W. K. Moon, C.-M. Lo, R.-T. Chen, Y.-W. Shen, J. M. Chang, C.-S. Huang, J.-H. Chen, W.-W. Hsu, and R.-F. Chang. Tumor detection in automated breast ultrasound images using quantitative tissue clustering. *Medical physics*, 41(4):042901, 2014.

J.-M. Morel and G. Yu. Asift: A new framework for fully affine invariant image comparison. *SIAM journal on imaging sciences*, 2(2):438–469, 2009.

D. Müller, I. Soto-Rey, and F. Kramer. Towards a guideline for evaluation metrics in medical image segmentation. *BMC Research Notes*, 15(1):210, 2022.

I. J. Myung. Tutorial on maximum likelihood estimation. *Journal of Mathematical Psychology*, 47(1):90–100, 2003.

M. Nikolic, E. Tuba, and M. Tuba. Edge detection in medical ultrasound images using adjusted canny edge detection algorithm. In *2016 24th telecommunications forum (TELFOR)*, pages 1–4. IEEE, 2016.

I. K. Nti, A. F. Adekoya, and B. A. Weyori. A systematic review of fundamental and technical analysis of stock market predictions. *Artificial Intelligence Review*, 53(4):3007–3057, 2020.

M. Pal and G. M. Foody. Feature selection for classification of hyperspectral data by svm. *IEEE Transactions on Geoscience and Remote Sensing*, 48(5):2297–2307, 2010.

G. Papamakarios, T. Pavlakou, and I. Murray. Masked autoregressive flow for density estimation. *Advances in neural information processing systems*, 30, 2017.

B. Park, W. Windham, K. Lawrence, and D. Smith. Contaminant classification of poultry hyperspectral imagery using a spectral angle mapper algorithm. *Biosystems Engineering*, 96(3):323–333, 2007.

B. U. Park and J. S. Marron. Comparison of data-driven bandwidth selectors. *Journal of the American Statistical Association*, 85(409):66–72, 1990.

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

E. G. M. Petrakis, A. Diplaros, and E. Milios. Matching and retrieval of distorted and occluded shapes using dynamic programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(11):1501–1516, 2002.

A. Popovic, M. De la Fuente, M. Engelhardt, and K. Radermacher. Statistical validation metric for accuracy assessment in medical image segmentation. *International Journal of Computer Assisted Radiology and Surgery*, 2:169–181, 2007.

A. Qin, Z. Shang, J. Tian, Y. Wang, T. Zhang, and Y. Y. Tang. Spectral–spatial graph convolutional networks for semisupervised hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters*, 16(2):241–245, 2018.

P. Ramos and J. M. Oliveira. Robust sales forecasting using deep learning with static and dynamic covariates. *Applied System Innovation*, 6(5):85, 2023.

S. Ranjan, D. R. Nayak, K. S. Kumar, R. Dash, and B. Majhi. Hyperspectral image classification: A k-means clustering based approach. In *2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pages 1–7. IEEE, 2017.

N. G. Reich, J. Lessler, K. Sakrejda, S. A. Lauer, S. Iamsirithaworn, and D. A. Cummings. Case study in evaluating time series prediction models using the relative mean absolute error. *The American Statistician*, 70(3):285–292, 2016.

C. Rodarmel and J. Shan. Principal component analysis for hyperspectral image classification. *Surveying and Land Information Science*, 62(2):115–122, 2002.

P. S. S. Rodrigues and G. A. Giraldi. Improving the non-extensive medical image segmentation based on tsallis entropy. *Pattern Analysis and Applications*, 14:369–379, 2011.

I. Rojas, O. Valenzuela, F. Rojas, A. Guillen, L. Herrera, H. Pomares, L. Marquez, and M. Pasadas. Soft-computing techniques and arma model for time series prediction.

Neurocomputing, 71(4):519–537, 2008. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2007.07.018. URL https://www.sciencedirect.com/science/article/pii/S0925231207002858. Neural Networks: Algorithms and Applications 50 Years of Artificial Intelligence: a Neuronal Approach.

O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

M. Rosenblatt. Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, 27(3):832 – 837, 1956. doi: 10.1214/aoms/1177728190. URL https://doi.org/10.1214/aoms/1177728190.

K. Saini, M. Dewal, and M. Rohit. Ultrasound imaging and image segmentation in the area of ultrasound: a review. *International Journal of advanced science and technology*, 24, 2010.

K. Sakrani, S. Elghoul, S. Falleh, and F. Ghorbel. Sa (2, r) multi-scale contour registration based on em algorithm. In *2021 International Conference on Visual Communications and Image Processing (VCIP)*, pages 1–5. IEEE, 2021.

N. Sammaknejad, Y. Zhao, and B. Huang. A review of the expectation maximization algorithm in data-driven process identification. *Journal of Process Control*, 73:123–136, 2019.

S. Saoudi, F. Ghorbel, and A. Hillion. Some statistical properties of the kernel-diffeomorphism estimator. *Applied stochastic models and data analysis*, 13(1):39–58, 1997.

N. I. Sapankevych and R. Sankar. Time series prediction using support vector machines: A survey. *IEEE Computational Intelligence Magazine*, 4(2):24–38, 2009. doi: 10.1109/MCI.2009.932254.

S. C. Schwartz. Estimation of probability density by an orthogonal series. *The Annals of Mathematical Statistics*, pages 1261–1265, 1967.

D. W. Scott and G. R. Terrell. Biased and unbiased cross-validation in density estimation. *Journal of the american Statistical association*, 82(400):1131–1146, 1987.

L. Scrucca and A. E. Raftery. Improved initialisation of model-based clustering using gaussian hierarchical partitions. *Advances in data analysis and classification*, 9:447–460, 2015.

T. B. Sebastian, P. N. Klein, and B. B. Kimia. Recognition of shapes by editing their shock graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):550–571, 2004. doi: 10.1109/TPAMI.2004.1273924.

J. Shah, D. Vaidya, and M. Shah. A comprehensive review on multiple hybrid deep learning approaches for stock prediction. *Intelligent Systems with Applications*, 16:200111, 2022.

F. Shamshad, S. Khan, S. W. Zamir, M. H. Khan, M. Hayat, F. S. Khan, and H. Fu. Transformers in medical imaging: A survey. *Medical Image Analysis*, 88:102802, 2023. ISSN 1361-8415. doi: https://doi.org/10.1016/j.media.2023.102802. URL https://www.sciencedirect.com/science/article/pii/S1361841523000634.

D. Shen, Y. Zhan, and C. Davatzikos. Segmentation of prostate boundaries from ultrasound images using statistical shape model. *IEEE transactions on medical imaging*, 22(4):539–551, 2003.

C. Shi, H. Wu, and L. Wang. Cegat: A cnn and enhanced-gat based on key sample selection strategy for hyperspectral image classification. *Neural Networks*, 168:105–122, 2023.

X. Shu, L. Pan, and X.-J. Wu. Multi-scale contour flexibility shape signature for fourier descriptor. *Journal of Visual Communication and Image Representation*, 26:161–167, 2015.

B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall, London, 1986.

Y. Song, J. Zheng, L. Lei, Z. Ni, B. Zhao, and Y. Hu. Ct2us: Cross-modal transfer learning for kidney segmentation in ultrasound images with synthesized data. *Ultrasonics*, 122:106706, 2022. ISSN 0041-624X. doi: https://doi.org/10.1016/j.ultras.2022.106706. URL https://www.sciencedirect.com/science/article/pii/S0041624X22000191.

M. Sudsawat, S. Unhapipat, and N. Pal. A comprehensive simulation study to compare various estimators of the model parameters, model mean, as well as model percentiles of

a two-parameter generalized half-normal distribution (2p-ghnd) with applications. *Thailand Statistician*, 19(1):13–42, 2021.

M. Sumair, T. Aized, M. M. A. Bhutta, F. A. Siddiqui, L. Tehreem, and A. Chaudhry. Method of four moments mixture-a new approach for parametric estimation of weibull probability distribution for wind potential estimation applications. *Renewable Energy*, 191:291–304, 2022.

A. A. Taha and A. Hanbury. Metrics for evaluating 3d medical image segmentation: analysis, selection, and tool. *BMC medical imaging*, 15:1–28, 2015.

P. C. Tay, C. D. Garson, S. T. Acton, and J. A. Hossack. Ultrasound despeckling for contrast enhancement. *IEEE Transactions on Image Processing*, 19(7):1847–1860, 2010.

M. P. Tcheou, L. Lovisolo, A. R. Freitas, and S. C. Chou. Reducing forecast errors of a regional climate model using adaptive filters. *Applied Sciences*, 11(17):8001, 2021.

A. Temlyakov, B. C. Munsell, J. W. Waggoner, and S. Wang. Two perceptually motivated strategies for shape classification. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 2289–2296. IEEE, 2010.

X. Teng, X. Zhang, and Z. Luo. Multi-scale local cues and hierarchical attention-based lstm for stock price trend prediction. *Neurocomputing*, 505:92–100, 2022.

G. R. Terrell. The maximal smoothing principle in density estimation. *Journal of the American statistical association*, 85(410):470–477, 1990.

A. Thakkar and K. Chaudhari. A comprehensive survey on deep neural networks for stock market: The need, challenges, and future directions. *Expert Systems with Applications*, 177: 114800, 2021a.

A. Thakkar and K. Chaudhari. A comprehensive survey on portfolio optimization, stock price and trend prediction using particle swarm optimization. *Archives of Computational Methods in Engineering*, 28:2133–2164, 2021b.

A. G. E. Thomas and J. S. Duela. A neoteric segmentation approach for lung ultrasound images. *New Generation Computing*, pages 1–14, 2024.

R. L. Thorndike. Who belongs in the family? *Psychometrika*, 18(4):267–276, 1953.

C. Tofallis. A better measure of relative prediction accuracy for model selection and model estimation. *Journal of the Operational Research Society*, 66(8):1352–1362, 2015.

Y. Tong, Y. Liu, M. Zhao, L. Meng, and J. Zhang. Improved u-net malf model for lesion segmentation in breast ultrasound images. *Biomedical Signal Processing and Control*, 68: 102721, 2021.

M. Troudi, A. M. Alimi, and S. Saoudi. Analytical plug-in method for kernel density estimator applied to genetic neutrality study. *EURASIP Journal on Advances in Signal Processing*, 2008(1):739082, 2008.

Z. Tu, S. Zheng, and A. Yuille. Shape matching and registration by data-driven em. *Computer Vision and Image Understanding*, 109(3):290–304, 2008.

B. Uria, M.-A. Côté, K. Gregor, I. Murray, and H. Larochelle. Neural autoregressive distribution estimation. *Journal of Machine Learning Research*, 17(205):1–37, 2016.

U. Vamsi Krishna, R. Priyamvada, G. Jyothish Lal, V. Sowmya, and K. Soman. A comparative evaluation of decomposition methods based on pitch estimation of piano notes. In *Smart Computing Techniques and Applications: Proceedings of the Fourth International Conference on Smart Computing and Informatics, Volume 1*, pages 833–843. Springer, 2021.

W. J. Van der Linden. *Handbook of item response theory: Three volume set*. Chapman and Hall/CRC, 2016.

P. Van Kerm. Adaptive kernel density estimation. *The Stata Journal*, 3(2):148–156, 2003.

P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

L. F. Vilela, R. C. Leme, C. A. Pinheiro, and O. A. Carpinteiro. Forecasting financial series using clustering methods and support vector regression. *Artificial Intelligence Review*, 52: 743–773, 2019.

G. Wahba. Data-based optimal smoothing of orthogonal series density estimates. *The annals of statistics*, 9(1):146–156, 1981.

B. Wang and Y. Gao. Hierarchical string cuts: a translation, rotation, scale, and mirror invariant descriptor for fast shape retrieval. *IEEE Transactions on Image Processing*, 23(9):4101–4111, 2014.

W. Wang, W. Lin, Y. Wen, X. Lai, P. Peng, Y. Zhang, and K. Li. An interpretable intuitionistic fuzzy inference model for stock prediction. *Expert Systems with Applications*, 213:118908, 2023.

Z. Wang and A. C. Bovik. Mean squared error: Love it or leave it? a new look at signal fidelity measures. *IEEE Signal Processing Magazine*, 26(1):98–117, 2009. doi: 10.1109/MSP.2008.930649.

G. C. Wei and M. A. Tanner. A monte carlo implementation of the em algorithm and the poor man's data augmentation algorithms. *Journal of the American statistical Association*, 85 (411):699–704, 1990.

Y. Weng, T. Zhou, Y. Li, and X. Qiu. Nas-unet: Neural architecture search for medical image segmentation. *IEEE Access*, 7:44247–44257, 2019. doi: 10.1109/ACCESS.2019.2908991.

H. Xu, J. Yang, and J. Yuan. Invariant multi-scale shape descriptor for object matching and recognition. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 644–648, 2016. doi: 10.1109/ICIP.2016.7532436.

C. Yang and Q. Yu. Multiscale fourier descriptor based on triangular features for shape retrieval. *Signal Processing: Image Communication*, 71:110–119, 2019. ISSN 0923-5965. doi: https://doi.org/10.1016/j.image.2018.11.004. URL https://www.sciencedirect.com/science/article/pii/S0923596518304521.

C. Yang and Q. Yu. Invariant multiscale triangle feature for shape recognition. *Applied Mathematics and Computation*, 403:126096, 2021.

C. Yang, H. Wei, and Q. Yu. A novel method for 2d nonrigid partial shape matching. *Neurocomputing*, 275:1160–1176, 2018.

X. Yang, S. Koknar-Tezel, and L. J. Latecki. Locally constrained diffusion process on locally densified distance spaces with applications to shape retrieval. In *2009 IEEE Conference on*

Computer Vision and Pattern Recognition, pages 357–364, 2009. doi: 10.1109/CVPR.2009.5206844.

L. Yin and J. Xie. Multi-temporal-spatial-scale temporal convolution network for short-term load forecasting of power systems. *Applied Energy*, 283:116328, 2021.

P. Yu and X. Yan. Stock price prediction based on deep neural networks. *Neural Computing and Applications*, 32:1609–1628, 2020.

S. Yuan, J. Yin, R. Li, Y. Chen, and Y. Zhang. Unified semantic model for medical image segmentation. *Biomedical Signal Processing and Control*, 98:106711, 2024.

B. Zhang, C. Zhang, and X. Yi. Competitive em algorithm for finite mixture models. *Pattern recognition*, 37(1):131–144, 2004.

C. Zhao, M. Wu, J. Liu, Z. Duan, J. li, L. Shen, X. Shangguan, D. Liu, and Y. Wang. Progress and prospects of data-driven stock price forecasting research. *International Journal of Cognitive Computing in Engineering*, 4:100–108, 2023. ISSN 2666-3074. doi: https://doi.org/10.1016/j.ijcce.2023.03.001. URL https://www.sciencedirect.com/science/article/pii/S2666307423000116.

Z. Zhao, H. Wang, and X. Yu. Spectral–spatial graph attention network for semisupervised hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters*, 19:1–5, 2021.

Y. Zheng, Y. Zhou, H. Zhou, and X. Gong. Ultrasound image edge detection based on a novel multiplicative gradient and canny operator. *Ultrasonic Imaging*, 37(3):238–250, 2015.

Y. Zheng, F. Meng, J. Liu, B. Guo, Y. Song, X. Zhang, and L. Wang. Fourier transform to group feature on generated coarser contours for fast 2d shape matching. *IEEE Access*, 8: 90141–90152, 2020.

Z. Zhong, J. Li, D. A. Clausi, and A. Wong. Generative adversarial networks and conditional random fields for hyperspectral image classification. *IEEE transactions on cybernetics*, 50 (7):3318–3329, 2019.

Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang. Unet++: Redesigning skip connections to exploit multiscale features in image segmentation. *IEEE transactions on medical imaging*, 39(6):1856–1867, 2019.

APPENDIX : The Mathematical Foundation for Bandwidth Optimization

To find the optimal bandwidth h_N^* for a Kernel Density Estimator (KDE), we need to minimize the Mean Integrated Squared Error (MISE). We start by considering the Mean Square Error (MSE) of the estimator:

$$MSE = E\left[\left|\hat{f}_N - f\right|^2\right] = E\left[\hat{f}_N^2\right] + f^2 - 2fE\left[\hat{f}_N\right]$$

$$= E\left[\hat{f}_N^2\right] - \left(E\left[\hat{f}_N\right]\right)^2 + \left(E\left[\hat{f}_N\right]\right)^2 + f^2 - 2fE\left[\hat{f}_N\right]$$

$$= \text{var}\left(\hat{f}_N\right) + \left(f - E\left[\hat{f}_N\right]\right)^2$$

The variance of the density estimator $\hat{f}$ can be expressed as follows:

$$\text{var}\left(\hat{f}_N\right) = \frac{1}{Nh_N}\int K^2(u) f\left(x - h_N u\right) du - \frac{1}{N}\left(\int K(u) f\left(x - h_N u\right) du\right)^2 \qquad \text{(A.1)}$$

And the expected value of $\hat{f}$ is given by:

$$E\left[\hat{f}(x)\right] = \int K(u) f(x - uh) du.$$

Now we the expression $(f - E[\hat{f}_N])^2$ can be writing as

$$\left(f - E[\hat{f}_N]\right)^2 = \left(E[\hat{f}_N] - f\right)^2 == \left[\int K(u)\left(f(x - uh_N) - f(x)\right) du\right]^2 \qquad \text{(A.2)}$$

Using equations (A.1) and (A.1), the MSE can be writing as :

$$E\left[\left|\hat{f}_N - f\right|^2\right] = \frac{1}{Nh_N}\int K^2(u)f(x-h_Nu) + \left[\int K(u)\left(f(x-uh_N)-f(x)\right)du\right]^2 - \frac{1}{N}\left(\int K(u)f(x-h_Nu)du\right)^2$$

By introducing the following Taylor expansion:

$$f(x-h_Nu) = f(x) - h_Nuf'(x) + \frac{u^2}{2}h_N^2f''(x) - \frac{u^3h_N^3}{6}f^{(3)}(x-\theta h_Nu)$$

where $0 < \theta < 1$, MISE can be expressed in terms of h_N by Δh_N

$$MISE \approx \Delta(h_N) = \frac{M(K)}{Nh_N} + \frac{J(f)h_N^4}{4} \tag{A.3}$$

with $M(k) = \int_{-\infty}^{+\infty} K^2(u)du$ and $J(f) = \int_{-\infty}^{+\infty}(f''(x))^2dx$

where f" is the second derivative of f.

The minimum value of the function $\Delta(h_N)$ is obtained by annulling its derivative $\Delta'(h_N) = 0$.

$$\Delta'(h_N) = -\frac{M(K)}{Nh_N^2} + h_N^3J(f) = 0$$

Therefore, the optimal value of h_N noted by h_N^* becomes:

$$h_N^* = N^{-\frac{1}{5}}.\left(J(f)\right)^{-\frac{1}{5}}.\left(M(K)\right)^{\frac{1}{5}} \tag{A.4}$$

with

$$M(k) = \int_{-\infty}^{+\infty} K^2(u)du$$

and

$$J(f) = \int_{-\infty}^{+\infty}(f''(x))^2dx$$

Analysis of hyperspectral images by content using a Geometric Deep Learning approach

Sarra FALLEH

Abstract

This thesis explores two central tasks in machine learning: classification, and regression, with a particular focus on unsupervised classification using the Expectation-Maximization (EM) algorithm, and regression using deep learning models, specifically Long Short-Term Memory (LSTM) networks.

The first part of this thesis focuses on unsupervised classification using the EM algorithm. A key limitation of the traditional EM algorithm when applied to Gaussian Mixture Models (GMMs) with bounded or semi-bounded support is its tendency to produce overflow problems near the boundary of the support. To address this, we propose a novel variant of the EM algorithm based on a diffeomorphic transformation that maps data from bounded support to unbounded support, performs unsupervised estimation in the transformed space, and then maps the data back to its original space.

In the second part, we evaluate regression algorithms, particularly the LSTM model, under various configurations. We assess LSTM performance using existing parameters, post-technical analysis, and after applying PCA to reduce the dimensionality of the input features. Additionally, we employ Kernel Density Estimation (KDE) with bandwidth optimization to estimate the probability distribution of model errors. The Cumulative Distribution Target Criterion (CDTC), a more robust evaluation metric, is utilized to provide deeper insights into model performance compared to traditional metrics.

In the last part, we investigate the application of Geometric Deep Learning (GDL) techniques to hyperspectral image (HSI) segmentation. Hyperspectral images, due to their high dimensional nature, pose significant challenges for traditional machine learning approaches. To address these challenges, we compare three models: Graph Convolutional Networks (GCNs),

Graph Attention Networks (GATs), and a hybrid GCN-GAT model. These models are evaluated with and without Principal Component Analysis (PCA) to assess their performance in segmenting hyperspectral data.

Key words: Machine Learning, Classification, Regression, Expectation-Maximization Algorithm, Gaussian Mixture Models, LSTM, Bounded Support, Unbounded Support, Kernel Density Estimation, Fast Plug-in Algorithm, Cumulative Distribution Target Criterion (CDTC), Principal Component Analysis (PCA), Hyperspectral image segmentation, Geometric Deep Learning.