AGGREGATED WASSERSTEIN DISTANCE FOR HIDDEN MARKOV MODELS AND AUTOMATED MORPHOLOGICAL CHARACTERIZATION OF PLACENTA FROM PHOTOS

A Dissertation in
Information Sciences and Technology
by
Yukun Chen

© 2020 Yukun Chen

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

May 2020
The dissertation of Yukun Chen was reviewed and approved by the following:

James Ze Wang  
Professor of Information of Sciences and Technology  
Dissertation Adviser, Co-Chair of Committee

Jia Li  
Professor of Statistics  
Co-Chair of Committee

Xinyu Xing  
Assistant Professor of Information of Sciences and Technology

Xiang Zhang  
Associate Professor of Information of Sciences and Technology

David Hunter  
Professor of Statistics

Dashun Wang  
Associate Professor of Kellogg School of Management  
Northwestern University  
Special Member

Mary Beth Rosson  
Professor of Information Sciences and Technology  
Graduate Program Director
Abstract

In the past decade, fueled by the rapid advances of big data technology and machine learning algorithms, data science has become a new paradigm of science and has more and more emerged into its own field. At the intersection of computational methods, data modeling and domain knowledge, data science aims to solve a wide spectrum of problems with data-driven approaches. To really advance the field, a number of unique challenges are ahead of us. For example, from an algorithmic perspective, new fundamental modeling tools are needed to support innovative computing paradigms. And from a solving real-world problem perspective, novel data-driven solutions synergizing carefully-tailored cutting-edge models are yet to be developed for addressing meaningful problems not technology ready before, which include but not are limited to those in the fields of medical image analysis and diagnosis, climate change and weather forecasting.

Within that big picture, the first part of this thesis proposed a new fundamental tool – aggregated Wasserstein distances for hidden Markov models (HMMs) with state conditional distributions being Gaussian. By exploiting the fact that the marginal distribution for the type of HMMs in consideration at any time position follows a Gaussian mixture distribution (GMM), our proposed distances first softly register the states in two HMMs by solving an optimal transport optimization problem where the cost between components is the Wasserstein metric for Gaussian distributions. The solution of such optimization is a fast approximation to the
Wasserstein metric between two GMMs. After state registration, our framework quantifies the dissimilarity of HMMs by measuring both the difference between the two marginal GMMs and that between the two transition matrices. Distances defined using our framework offer three key advantages. First, the defined distances are invariant to relabeling or permutation of states. Second, their definition is meaningful even for two HMMs that are estimated from data of different dimensionality, a situation that can arise due to missing variables. Third, they outperform Kullback-Leibler divergence (KLD)-based distances in terms of accuracy as well as efficiency in a variety of tasks, which is demonstrated through extensive experiments consisting of retrieval, classification and t-Distributed Stochastic Neighbor Embedding (t-SNE) visualization of time series, on both synthetic and real data.

The second part of the thesis addressed an important medical image analysis problem: automated placental assessment and examination from photos. Specifically, we focus on morphological characterization, which includes the tasks of placental image segmentation, umbilical cord insertion point localization, and maternal/fetal side classification. To that end, we first curated a dataset consisting of approximately 1,300 placenta images taken at Northwestern Memorial Hospital, with hand-labeled pixel-level segmentation map, cord insertion point and other information extracted from the associated pathology report. Then we developed three encoder-decoder convolutional neural networks with a shared encoder to address those morphological characterization tasks by employing a transfer-learning training strategy. Through extensive experiments, we demonstrate that our method is able to produce accurate morphological characterization. We also show how our proposed method fits into a comprehensive two-stage placental assessment pipeline, which includes more placental feature analysis tasks, such as retained placenta (i.e., incomplete placenta), umbilical cord knot, meconium, abruption, chorioamnionitis, and hypercoiled cord, and categorization of umbilical cord insertion type. Our method and results for automated placental assessment may possess clinical impact and contribute to future pregnancy research. Moreover, this part of research is the first for comprehensive, automated, computer-based placental analysis and could serve as a launchpad for potentially multiple future innovations.
Table of Contents

List of Figures viii
List of Tables xiii
Acknowledgments xiv

Chapter 1
Introduction 1
1.1 Background ............................................. 4
1.1.1 Edge Computing ...................................... 4
1.1.2 Hidden Markov Model ................................. 5
1.1.3 Optimal Transport .................................... 11
1.1.4 Geometric Model – Why defining a distance is essential . . . . 15
1.1.5 Transfer Learning .................................... 15
1.1.6 Placenta Pathology and Imaging ..................... 16
1.2 Organization of the Thesis ............................... 17

Chapter 2
Aggregated Wasserstein Distance for Hidden Markov Models 19
2.1 Introduction ............................................. 19
2.2 Preliminaries ............................................ 23
2.2.1 Notations and Definitions ............................ 23
2.2.2 The Wasserstein Distance and the Gaussian Case ........ 24
2.3 The Framework of Aggregated Wasserstein ................. 29
2.3.1 The Registration of States ........................... 29
2.3.2 The Distance between Two Marginal GMMs ........... 31
2.3.3 The Distance between Two Transition Matrices ........ 32
Chapter 3
Automated Morphological Characterization of Placenta from Photos

3.1 Introduction ................................................................. 65
  3.1.1 Related Work ......................................................... 66
  3.1.2 Rationale for a Two-stage Pipeline ............................ 68
  3.1.3 Contributions ....................................................... 68
3.2 Dataset ................................................................. 69
3.3 Stage I: Morphological Characterization .......................... 71
  3.3.1 Encoder as Feature Pyramid Extractor ........................ 72
  3.3.2 SegDecoder for Segmentation .................................... 73
  3.3.3 Classification Subnet for Fetal/Maternal Side Classification 73
  3.3.4 IPDecoder for Insertion Point Localization .................. 74
  3.3.5 Training and Testing .............................................. 76
  3.3.6 Implementation .................................................... 76
3.4 Stage II: Placenta Feature Analysis ............................... 77
  3.4.1 Umbilical Cord Insertion Type Categorization .............. 77
  3.4.2 Other Placental Feature Analysis Tasks ........................ 78
3.5 Experiments and Results ............................................. 78
  3.5.1 Morphological Characterization ................................ 78
    3.5.1.1 Segmentation .............................................. 78
    3.5.1.2 Fetal/Maternal Side Classification .................... 81
    3.5.1.3 Insertion Point Localization ............................ 82
  3.5.2 Placenta Feature Analysis ...................................... 84
List of Figures

1.1 Search term popularity of “data science” compared with “deep learning” in the United States. Data source: Google Trend (https://www.google.com/trends). .................................................. 2

1.2 Illustration of the model of models. left: Traditional central computing approach which assume all data communicated to single computing node. right: Edge computing approach where learning is conducted at two levels: 1) At the data-level, local models are computed at decentralized computing sites where the original data is available. 2) At the model-level, we perform learning at a central site where all models computed at decentralized sites are communicated to. Here the objects of learning is model parameters. We call such approach “model of models” ................................................................. 5

1.3 Motivating examples for optimal transport. The figure (a) is reproduced from the Chapter 1 of [90]. ................................................................. 12

1.4 The Placenta. (The image source for (a) is National Institute of Diabetes and Digestive and Kidney Diseases, National Institutes of Health [3]. The photos in (b) and (c) are from the dataset we curated (see in Chapter 3.)) .................................................. 16

2.1 (a) Experiment scheme for varying $\mu$ and varying $\Sigma$. A re-estimated $\hat{\phi}_0$ is denoted as the dashed blue line. (b) (c) Mean estimates of $W_2(\hat{\phi}_0, \phi_i)$ (blue) and $KL(\hat{\phi}_0, \phi_i)$ (orange) and their $3\sigma$ confidence intervals w.r.t different Gaussian $\phi_i$. (b) is for varying $\mu$, and (c) is for varying $\Sigma$. ................................................................. 25

2.2 A simple registration example about how $\mathbf{T}_2$ in $\Lambda_2$ is registered towards $\Lambda_1$ such that it can be compared with $\mathbf{T}_1$ in $\Lambda_1$. For this example, $\mathbf{W}$ encodes a “hard matching” between states in $\Lambda_1$ and $\Lambda_2$ 30
2.3 Top: intuition for $w_{i,j}$ solved in MAW. $w_{i,j}$ represents a proportion of $\phi_{1,i}$ (more precisely, $w_{i,j}\phi_{1,i}$ of $\pi_{1,i}\phi_{1,i}$) that is transported to $\phi_{2,j}$ ($w_{i,j}\phi_{2,j}$ of $\pi_{2,i}\phi_{2,j}$). Bottom: intuition for $w_{i,j}^*$ solved in IAW. Imagine the blue dots under the slash lined area of $\phi_{1,i}$ are matched to red dots under the slash lined area of $\phi_{2,j}$ based on the point-wise transport plan $\Pi_n$ (see Eq.(2.42)). Suppose these dots are sampled from the two mixture distributions respectively. Let $w_{i,j}^*$ quantify the proportion of $\phi_{1,i}$ that is transported to $\phi_{2,j}$. Eq.(2.38) determines $w_{i,j}^*$ based on the true Wasserstein transport plan for the two continuous mixture distributions, which is approximated here by the optimal transport plan $\Pi_n$ between two sets of sampled points.

2.4 Precision-recall plot for the study to compare KL, KL_var_gmm, KL_var_hmm, MAW and IAW’s sensitivity to the perturbation of HMM’s parameters. (best viewed in color) .................................................. 39

2.5 Nearest neighbor retrieval experiment Scale v.s. AUCs. under (a) the perturbation of $\mu$, (b) the perturbation of $\Sigma$ and (c) the perturbation of transition matrix. (best viewed in color) ................................................................. 47

2.6 Analysis of how the accuracy of the estimated HMMs affects results of the NN Retrieval experiment (perturbation of $\mu$). $T = M \times d \times \tau$, where $M = 3$, $d = 3$. The case $\tau = \infty$ means the true parameters of the seed HMMs are used (instead of estimated parameters) .................. 51

2.7 Visualization of CMU motion capture data. Top: Jump. Bottom: Walk ................................................................. 53

2.8 Precision Recall Plot for Motion Retrieval. The plot for 6 joint-groups, i.e. root$_{12}$, head$_{neck}$, thorax$_{12}$, rbody$_{12}$, lbody$_{12}$, rleg$_6$, lleg$_6$, are displayed separately. ................................................................. 54

2.9 Precision Recall Plot for MAW based Motion Retrieval under data missing for certain dimension(s) setting. The plot for 6 joint-groups, i.e. root$_{12}$, head$_{neck}$, thorax$_{12}$, rbody$_{12}$, lbody$_{12}$, rleg$_6$, lleg$_6$, are displayed separately. ................................................................. 57

2.10 Precision Recall Plot for IAW-based Motion Retrieval. In this experiment, a certain number of randomly chosen variables are missing in each query sequence. The plots for the 6 joint-groups, i.e. root$_{12}$, head$_{neck}$, thorax$_{12}$, rbody$_{12}$, lbody$_{12}$, rleg$_6$, lleg$_6$, are displayed separately (best viewed in color). ................................................................. 58

2.11 Testing accuracies with respect to the iteration number in Adaboost (number of weak classifiers selected). (a) Motion Classification by Adaboost on 6 joints. (b) Motion Classification by Adaboost on 27 joints. The iteration number means the number of features incrementally acquired in Adaboost. ................................................................. 59
2.12 Histogram of 20-concat phoneme segments ................................................. 60
2.13 KL-based, MAW-based and IAW-based t-SNE visualization compar-
ison. ......................................................................................................................... 62

3.1 Data curation process. (a) Placenta photography equipment at Northwestern Memorial University. This equipment is used to collect high and consistent quality placenta photos for the curation of our dataset. (b-c) Extracting diagnoses and measurements from unidentified pathological report in p.d.f format. (d-f) Collecting pixel level segmentation map for cord, disc and ruler, insertion point location and whether a image captures fetal or maternal side placenta through our web-based labeling tool. ......................................................... 70

3.2 Schematic diagram of our proposed two-stage framework for auto-
mated placental assessment and examination using photos. Details of Stage I models and Stage II models will be described in Sec. 3.3 and Sec. 3.4 respectively. .......................................................... 70

3.3 The architecture of our model for morphological characterization: a multi-head convolutional neural network for placenta image segmen-
tation, cord insertion point localization, and placenta disc side classi-
fication. “Up sample & Conv” is implemented by a transposed convolution layer. “Res conv blocks” are residual blocks with two convolutional layers with stride 2 and 1, respectively, and the same kernel size $3 \times 3$. “Score blocks” are convolutional layers with kernel size $1 \times 1$ and the number of output channel 1. The soft-max layers are omitted. We use dice loss, binary cross entropy (BCE) loss and mean square error (MSE) loss for the segmentation, classification, and insertion point localization, respectively. .............................................. 72

3.4 The insertion type categorization and related automated measure-
ments procedures consists of steps 1) to 5). (a), (b), and (c) illustrate the detailed procedure for steps 1), 2), and 3), respectively. (For (b) ruler scale extraction, Ms. Zhuomin Zhang helped with the coding and the illustration.) .................................................... 75

3.5 (a), (b), and (c) are confusion matrices of our approach, U-Net, and SegNet, respectively. .......................................................... 79

3.6 Examples of segmentation results. We show both fetal-side results (top two rows) and maternal-side results (bottom two rows). .......... 80

3.7 Fetal/maternal side classification confusion matrices comparison. (a) Without shared encoder weights. (b) Ours. ............................. 80

3.8 Quantitative evaluation of insertion point localization with percent-
age of correct keypoints (PCK) curves. ......................................................... 81
3.9 Qualitative examples of insertion point heat map predictions.

3.10 Quantitative and qualitative evaluation for insertion point type categorization. (a) The confusion matrix for insertion type categorization. (b) Quantitative evaluation of our estimation on the distance from the insertion point to the nearest disc margin.

3.11 Qualitative examples of insertion point type categorization.

A1 overall accuracy for $r_{leg}$ w.r.t MAW’s $\alpha$.

A2 Precision Recall Plot for MAW based Motion Retrieval at $\alpha = 0$ and $\alpha = 1$. The plot for 6 joint-groups, i.e. $\text{root}_{12}, \text{head}_\text{neck}_\text{thorax}_{12}, \text{rbody}_{12}, \text{lbody}_{12}, \text{rleg}_{6}, \text{lleg}_{6}$, are displayed separately. (Best view in color.)

A3 Precision Recall Plot for IA W based Motion Retrieval at $\alpha = 0$ and $\alpha = 1$. The plot for 6 joint-groups, i.e. $\text{root}_{12}, \text{head}_\text{neck}_\text{thorax}_{12}, \text{rbody}_{12}, \text{lbody}_{12}, \text{rleg}_{6}, \text{lleg}_{6}$, are displayed separately. (Best view in color.)

A4 (a) 6 joints, KL, corresponding to the blue dot in the left sub-figure of Fig. 8, (b) 6 joints, MAW, corresponding to the red dot in the right sub-figure of Fig. 8, (c) 6 joints, IA W, corresponding to the yellow dot in the left sub-figure of Fig. 8, (d) 27 joints, KL, corresponding to the blue dot in the right sub-figure of Fig. 8, (e) 27 joints, MA W, corresponding to the red dot in the left sub-figure of Fig. 8, (f) 27 joints, IA W, corresponding to the yellow dot in the right sub-figure of Fig. 8.

A5 Testing accuracies with respect to the iteration number in Adaboost (number of weak classifiers selected). (a) Motion Classification by Adaboost on 6 joints. (b) Motion Classification by Adaboost on 27 joints. The iteration number means the number of features incrementally acquired in Adaboost. MA W for GMM and IA W for GMM here refers to the experiment settings where we solely consider $R_p(M_1, M_2; W)$ when we compute distance matrix by MA W or IA W.

A6 Analyzing the effect of the accuracy of the re-estimated HMMs on the NN retrieval results. In the experiment, $\mu$ is perturbed. Each column corresponds to a different length factor $\tau$. Note that $T = M \times d \times \tau$. The rightmost column with $\tau = \infty$ means that we used the true parameters of the seed HMMs.
Analyzing the effect of the accuracy of the re-estimated HMMs on the
NN retrieval results. In the experiment, $\Sigma$ is perturbed. Each column
corresponds to a different length factor $\tau$. Note that $T = M \times d \times \tau$.
The rightmost column with $\tau = \infty$ means that we used the true
parameters of the seed HMMs.

Analyzing the effect of the accuracy of the re-estimated HMMs on
the NN retrieval results. In the experiment, the transition matrix is
perturbed. Each column corresponds to a different length factor $\tau$.
Note that $T = M \times d \times \tau$. The rightmost column with $\tau = \infty$ means
that we used the true parameters of the seed HMMs.

The analysis of cutoff sequence length $N$’s effect on the recision-recall
performance of KL_var_hmm.
## List of Tables

2.1 Summary of the parameters setup for parameter perturbation experiments. *rand*(2) here means random matrix of dimension 2 by 2. *Dirichlet*(\(\vec{x}\)) here means generating samples from Dirichlet distribution with parameter \(\vec{x}\). ................................. 46

2.2 Synthetic data per distance computation time comparison. (KL sample size: 2000, IAW sample size: 500) ................................. 48

2.3 Phoneme k-NN classification accuracy comparison. ................................. 60

2.4 Quantitative comparison of t-SNE visualization based on KL, MAW and IAW. The best of each row (in terms of one measure) is underscored. 63

3.1 Segmentation evaluation ................................. 79

A1 Choice of \(\alpha\) for MAW and IAW in Motion Retrieval Experiments. . 105

B2 Information Extracted from Pathology Reports ................................. 117
Acknowledgments

Pursuing a Ph.D is challenging, both physically and mentally. For me, It’s hard to imagine surviving this undertaking alone without the help from many people. Here, I want to express my sincere gratitude for them.

First and foremost, I would like to thank my advisors, Prof. James Z. Wang and Prof. Jia Li. Without their support and guidance, this dissertation would not have been possible. They taught me innumerable lessons on how to plan, conduct and present meaningful research. During inevitable difficult times, their encouragement always gave me more courage to face and overcome those challenges. I remember times when they revised my papers at late night in order to meet submission deadlines. I am deeply respectful for their devotion to and rigorous attitude toward research. I am also grateful for the freedom they have allowed me in my research and study.

I would like to thank my other committee members, Prof. David Hunter and Prof. Xinyu Xing, Prof. Xiang Zhang and Prof. Dashun Wang for their help, encouragement and suggestions on my dissertation. Dr. Alison D. Gernand and Dr. Jeffery A. Goldstein offered very insightful and helpful discussion and suggestion on the placenta project from the pathology perspective, and I am grateful for them.

I’m fortunate to have three summer internship experiences during my Ph.D. study. I would like to thank my intern managers, Dr. Hongbo Deng, Dr. Xuanhui Wang, Dr. Satish Chandra, Dr. Swarat Chaudhuri and Mr. Minghai Chen for the generous help and advice they offered me during the internships. From them, I learnt a lot on how to manage and execute projects that are meaningful to real products.
I would like to thank all my fellow lab-mates, Dr. Yu Zhang, Dr. Xin Lu, Dr. Jianbo Ye, Dr. Baris Kandemir, Farshid Farhat, Xinye Zheng, Yu Luo, Mohammad M. Kamani, Zhuomin Zhang, Chenyan Wu and Dolzodmaa Davaasuren, Tongan Cai. It has been a great fortune and pleasure for me to work and study with them and I appreciate their inspiring discussion and company.

Last but not the least, I would like to thank my parents, Huiwen Wu and Hong Chen and my wife, Jiawei Chen, for their perpetual love and support.

Part of this dissertation is based upon work supported by the NSF Award ECCS-1462230, DMS-1521092, IIS-1027854 and Bill & Melinda Gates Foundation grant No. OPP1195074. Any opinions, findings, and conclusions or recommendations expressed in this dissertation do not necessarily reflect the views of the NSF or Bill & Melinda Gates Foundation.
To my parents and my wife, Jiawei.
Chapter 1

Introduction

Data science is a young but thriving interdisciplinary field. According to Google Trends [1], the popularity of the search term “data science” has quadrupled from 2014 to 2020. This growth rate of popularity is even greater than that of the search term “deep learning” (Fig. 1.1). Many universities in the U.S. have begun offering data science programs recently [2], which reflect industry’s increasing demand for the methodology and skillset the data science field offers. At its core, data science stresses innovation in data-driven solutions that can turn abundantly available data into insights and value by unifying knowledge from classical disciplines, such as statistics, machine learning, and databases. It has three pillars: data modeling, computational methods, and domain knowledge. The fast growth of data science is fueled by not only the rapid advances of big data technology and machine learning algorithms, but also by the eagerness for transforming the new technologies to productivity in much broader fields that are most meaningful to individuals, organizations, and society. Thus, both the interdisciplinary nature of data science and the fundamental momentum for the growth of the field above give us the opportunity to advance the fields from different perspectives. This thesis takes two different perspectives for advancing data science – one is the algorithmic perspective and the other one is real-world problem solving perspective. From the algorithmic perspective, we are motivated by developing new fundamental modeling tools to support innovative computing paradigms (e.g., edge computing). From the perspective of solving a real-
world problem, we are motivated to address automating placenta image analysis by novel data-driven solutions synergizing carefully-tailored cutting-edge models.

![Image of search term popularity comparison between data science and deep learning](https://www.google.com/trends)

**Figure 1.1.** Search term popularity of “data science” compared with “deep learning” in the United States. Data source: Google Trend (https://www.google.com/trends).

In Chapter 2, we propose a new optimal transport based framework, which we named aggregated Wasserstein, for defining a dissimilarity measure or distance between two hidden Markov models (HMMs) with state conditional distributions being Gaussian. Hidden Markov models have achieved great success in modeling sequential data since its invention in the 1960s and Gaussian emission function is often used when the observation space for the HMM in consideration is continuous. Traditionally, Kullback-Leibler divergence (KLD) and its variants have been the dominating methods for comparing two HMMs. Different from the probabilistic perspective where KLD based distances originate, our proposed framework is built up from a geometric perspective. In essence, we exploit the fact that the marginal distribution for the type of HMMs in consideration at any time position follows a Gaussian mixture distribution (GMM), to softly match, aka register, the states in two HMMs. The registration of states is inspired by the intrinsic relationship of optimal transport and the Wasserstein metric between distributions. Specifically, the components of the marginal GMMs are matched by solving an optimal transport problem where
the cost between components is the Wasserstein metric for Gaussian distributions. The solution of the optimization problem is a fast approximation to the Wasserstein metric between two GMMs. After state registration, our framework quantifies the dissimilarity of HMMs by measuring both the difference between the two marginal GMMs and that between the two transition matrices. Proofs are provided to establish key properties of our distances. We also demonstrate the characteristics and advantages of the distances defined using our framework through extensive experiments including retrieval, classification and t-SNE visualization of time series, on both synthetic and real data.

In Chapter 3, we address an important medical image analysis problem, automated placental assessment and examination from photos. Analysis of the placenta is useful for evaluating health risks of both mother and baby after delivery. However, only about 20% of placentas in the U.S. are assessed by pathology exams and placental data is often missed in pregnancy research because of the additional time, cost, and expertise needed. A computer-based tool that can be used in any delivery setting at the time of birth to provide an immediate and comprehensive placental assessment would have the potential to both improve health care and radically improve medical knowledge. We mainly focus on morphological characterization in this thesis, which includes the tasks of placental image segmentation, umbilical cord insertion point localization, and maternal/fetal side classification. To that end, we first curated a dataset consisting of approximately 1,300 placenta images taken at Northwestern Memorial Hospital. Apart from images of placenta, the dataset also provides hand-labeled pixel-level segmentation map, cord insertion point and other information extracted from the associated pathology report. Then we developed three encoder-decoder convolutional neural networks with a shared encoder to address those morphological characterization tasks by employing a transfer-learning training strategy. Through extensive experiments, we demonstrate our method is able to produce accurate morphological characterization. We also show how our proposed method fits into a comprehensive two-stage placental assessment pipeline, which we named AI-based Placental Assessment and Examination system (AI-PLAX). It includes more
placental feature analysis tasks, such as retained placenta (i.e., incomplete placenta), umbilical cord knot, meconium, abruption, chorioamnionitis, and hypercoiled cord, as well as categorization of umbilical cord insertion type. Since parts of this two-stage pipeline involve works done by a larger research team, the focus of this thesis chapter is on the categorization of umbilical cord insertion type and related measurements. We only briefly discuss other placental analysis tasks as a demonstration of the usefulness of our placental morphological characterization models.

At the first glance, Chapters 2 and 3 address quite different problems and they are hard to connect with each other; however, from a modeler’s perspective, the two different problems allow this thesis to consider both generative models and discriminative models. Even though discriminative models, exemplified by neural networks in the era of deep learning, has gathered more attention in recent years because of stellar performance in all kinds of benchmarks, this thesis asserts that both generative and discriminative models will play an important role in addressing data science problems in the future.

In the rest of this chapter, we will review some key concepts as the background for the thesis research in Sec. 1.1. The goal is to present the broader context and relevant concepts in a more detailed way before delving into the main research. Then we outline the structure of the thesis in Sec. 1.2.

1.1 Background

1.1.1 Edge Computing

One prominent challenge for big data is the limitation brought by the distributed data collection, communication networks and decentralized computing platforms. Due to the communication constraints as well as the sheer size of data, it can be very costly or even impossible to communicate all training data to a single computing site. This difficulty motivates us to revisits those established models that were originally developed under the assumption that all training data could be available at a single site. The new setting we targeted is the so called edge computing paradigm
In this new paradigm, learning is conducted on a distributed computing platform. First, local models are estimated at multiple sites based on local available data. Then, only the model parameters (much compressed information from the original data) are transmitted to a central site. Finally, learning tasks such as classification or clustering are conducted on the model parameters. Note that at this stage, the learning instance is no longer the original data but rather the models.

Figure 1.2. Illustration of the model of models. *left:* Traditional central computing approach which assume all data communicated to single computing node. *right:* Edge computing approach where learning is conducted at two levels: 1) At the data-level, local models are computed at decentralized computing sites where the original data is available. 2) At the model-level, we perform learning at a central site where all models computed at decentralized sites are communicated to. Here the objects of learning is model parameters. We call such approach “model of models”.

1.1.2 Hidden Markov Model

A hidden Markov model (HMM) is a probabilistic model for sequential data, which could be a discrete sequence consisting of symbols that belong to a finite set, or a continuous sequence consisting of float vectors. It is a generative model assuming the observed sequence is generating from a doubly embedded stochastic process: First, a discrete Markov process governs the generation of a sequence of hidden states (by
“hidden” we mean those states are not directly observable and they are just hypothetical variables explaining the dynamics of the observed sequential data.); Second, a set of probabilistic functions governs the generation of an observable symbol or float vector given a hidden state. We can see that the name “hidden Markov” actually comes from the first stochastic process. More formally, a hidden Markov model with $M$ states models an sequence $O = o_1 o_2 ... o_T$ with length $T$ by assuming that there is a corresponding sequence of states $S = s_1 s_2 ... s_T \ (s_t \in \{1, ..., M\} \ \forall \ t \in \{1, ..., T\})$ governing the underlying dynamics for the sequence. The state sequence $S$ is assumed to be generated from a Markov chain specified by a $M \times M$ transition matrix $T$ (where $T_{i,j} = P(s_{t+1} = j|s_t = i)$), and the initial state probability $\pi = [\pi_1, \pi_2, ..., \pi_M]$. At time step $t$, given the state $s_t$, the observation $o_t$ is generated from a probabilistic distribution $\phi_i(o_t) = P(o_t|s_t = i)$ often called the emission function. For a discrete sequence, $\phi_i(o_t)$ is a probability table that sums to one. For a continuous sequence, $\phi_i(o_t)$ is a continuous probability distribution, of which a Gaussian or a Gaussian mixture is the common choice, especially for speech recognition applications. In this thesis, $\phi_i(o_t)$ is further assumed to be a Gaussian for the continuous sequence since a state with a number of mixture components can be split into sub-states, each with a single Gaussian. Thus, the emission function can be expressed as:

$$\phi_i(o_t) = \frac{1}{\sqrt{(2\pi)^k \det(\Sigma_i)}} e^{-\frac{1}{2}(t-\mu_i)^T \Sigma_i^{-1} (t-\mu_i)},$$

where $k$ is the dimension of $o_t$. $\mu_i$ and $\Sigma_i$ are the mean vector and covariance matrix respectively. Thus we can fully specify a HMM by $\Lambda (T, \pi, \{\phi_i\}_{i=1}^M)$.

Given the above full specification for a hidden Markov model, we are ready to review the three most important algorithms about its estimation and usage:

1. **Forward-backward procedure**: Given a sequence $O = o_1 o_2 ... o_T$ and a HMM $\Lambda$, this algorithm efficiently computes $P(O|\Lambda)$, the probability of the observed sequence given the model. By definition,

$$P(O|\Lambda) = \sum_S P(O, S|\Lambda)$$
\[= \sum_S P(O|S, \Lambda)P(\Lambda)\]
\[= \sum_{s_1 s_2 \ldots s_T} \pi_{s_1} \phi(o_1|s_1)T_{s_1, s_2} \ldots T_{s_{T-1}, s_T} \phi(o_T|s_T).\]

Direct calculation based on the definition above, however, has a time complexity of \(T N T\), which is too expensive to be practical. Forward-backward procedure uses dynamic programming to save the computation. First, for ease of describing the procedure, a forward variable \(\alpha_t(i)\) is defined as:

\[\alpha_t(i) = P(o_1 o_2 \ldots o_t, s_t = i|\Lambda) .\]

Then \(P(O|\Lambda)\) can be computed by three steps:

(a) Initialization:
\[\alpha_1(i) = \pi_i \phi(o_1|i), \ \forall 1 \leq i \leq M .\]

(b) Induction:
\[\alpha_{t+1}(j) = \left( \sum_{i=1}^{M} \alpha_t(i)T_{i,j} \right) \phi_j(o_{t+1}), \ t \in \{2, 3, \ldots, T\}, \ 1 \leq i \leq M .\]

(c) Termination:
\[P(O|\Lambda) = \sum_{i=1}^{M} \alpha_T(i) .\]

The time complexity has been reduced to \(M^2 T\) by the above procedure. Similarly a backward variable \(\beta_t(i)\) can be defined as:

\[\beta_t(i) = P(o_{t+1} o_{t+2} \ldots o_T|s_t = i, \Lambda) .\]

Then \(P(O|\Lambda)\) can be computed by three steps:

(a) Initialization:
\[\beta_T(i) = 1 \ \forall 1 \leq i \leq M .\]
(b) Induction:

\[ \beta_t(j) = \sum_{j=1}^{M} \beta_{t+1}(j) T_{ij} \phi_j(o_{t+1}), \quad t \in \{T-1, T-2, \ldots, 1\}, \quad 1 \leq i \leq M. \]

c (c) Termination:

\[ P(O|\Lambda) = \sum_{i=1}^{M} \beta_1(i). \]

to compute \( P(O|\Lambda) \) in backward. Though this backward procedure accomplishes the same thing as the forward procedure and thus is not necessary for the computation of \( P(O|\Lambda) \), it will be useful for the Baum-Welch algorithm which we will review next.

2. **Viterbi algorithm**: Given a sequence \( O = o_1o_2...o_T \) and a HMM \( \Lambda \), this algorithm estimates the optimal sequence of states \( S = s_1s_2...s_T \) that best explains the observation \( O \). The optimality is in the sense of the maximum likelihood of the whole state sequence given the model and the observation. An intermediate variable, that stands for the highest probability of the sub state sequence, that accounts for the first \( t \) observations \( o_1\ldots o_t \) that ends in state \( i \) at time \( t \), is defined:

\[ \delta_t(i) = \max_{s_1s_2...s_{t-1}} P(s_t = i, o_1o_2...o_t|\Lambda). \]

The recursion relation between \( \delta_t(i) \) and \( \delta_{t+1}(j) \) is:

\[ \delta_{t+1}(j) = \max_i (\delta_t(i) T_{ij} \phi_j(o_{t+1})). \]

If we store the argument that maximizes the above equation into a variable named \( \psi_t(j) \), we can decode the optimal state sequence using the following dynamic programming procedure:

(a) Initialization:

\[ \delta_1(i) = \pi_i \phi(i_1), \quad 1 \leq i \leq M \]
\[ \psi_1(i) = 0, \ 1 \leq i \leq M. \]

(b) Recursion:

\[ \delta_t(j) = \max_i (\delta_{t-1}(i)T_{i,j}) \phi_j(o_{t+1}) \]

\[ \psi_t(j) = \arg \max_i (\delta_{t-1}(i)T_{i,j}) \]

(c) Termination:

\[ P^* = \max_i \delta_T(i) \]

\[ s_T^* = \arg \max_i \delta_T(i). \]

(d) Backtracking (for decoding state sequence):

\[ s_t^* = \psi_{t+1}(s_{t+1}^*), \ t \in \{T - 1, T - 2, \ldots, 1\}. \]

3. **Baum-Welch algorithm**: Given the number of state \( M \), this algorithm estimates an \( \Lambda \) that maximizes \( P(O|\Lambda) \) using an E-M (expectation-maximization) procedure. Note that just like any other E-M algorithm, Baum-Welch only optimizes \( \Lambda \) locally and has no theoretical guarantee to reach a global optimum. Intermediate variables need to be defined for the ease of describing the algorithm:

\[ \xi_t(i, j) = P(s_t = i, s_{t+1} = j|O, \Lambda), \ t \in \{1, 2, \ldots, T - 1\}, i, j \in \{1, 2, \ldots, M\}. \]

\[ \gamma_t(i) = P(s_t = i|O, \Lambda), \ t \in \{1, 2, \ldots, T\}, i \in \{1, 2, \ldots, M\}. \]

which stand for the probability of being in state \( i \) at time \( t \) and being in state \( j \) at time \( t + 1 \), and the probability of being in state \( i \) at time \( t \) conditioned on observed sequence and the model respectively. These two variables can be
expressed using the forward- and backward-variable:

\[
\xi_t(i, j) = \frac{P(O, s_t = i, s_{t+1} = j | \Lambda)}{P(O | \Lambda)} = \frac{P(o_1 \ldots o_t, s_t = i | \Lambda) T_{i,j} \phi_j(o_{t+1}) P(o_{t+2} \ldots o_T, s_{t+1} = j | \Lambda)}{P(O | \Lambda)}
\]

\[
\gamma_t(i) = \frac{P(O, s_t = i | \Lambda)}{P(O | \Lambda)} = \frac{P(o_1 o_2 \ldots o_t, s_t = i | \Lambda) P(o_{t+1} o_{t+2} \ldots o_T, s_{t+1} = i | \Lambda)}{P(O | \Lambda)}
\]

And obviously, they are related by the following equation:

\[
\gamma_t(i) = \sum_{j=1}^{M} \xi_t(i, j).
\]

Since \(\sum_{t=1}^{T-1} \gamma_t(i)\) can be interpreted as the expected number of transitions from state \(i\) and \(\sum_{t=1}^{T-1} \xi_t(i, j)\) can be interpreted as the expected number of transitions from state \(i\) to state \(j\). Given the current estimation of \(\Lambda = (\pi, T, \{\phi_i\}_{i=1}^{M})\) and the observed sequence \(O\), we can re-estimate the HMM by the following procedures:

\[
\hat{\pi}_i = \gamma_1(i),
\]

\[
\hat{T}_{i,j} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)}.
\]

For a discrete sequence, let \(v_k\) denote the the \(k\)-th symbol:

\[
\hat{\phi}_j(k) = \frac{\sum_{t=1 \text{ s.t. } o_t = v_k}^{T-1} \gamma_t(i)}{\sum_{t=1}^{T-1} \gamma_t(i)}.
\]

For a continuous sequence the emission function of which is a Gaussian, the
newly estimated $\hat{\phi}_j$ can be fully specified by:

$$
\hat{\mu}_i = \frac{\sum_{t=1}^{T} \gamma_t(i) o_t}{\sum_{t=1}^{T} \gamma_t(i)},
$$

$$
\hat{\Sigma}_i = \frac{\sum_{t=1}^{T} \gamma_t(i)(o_t - \hat{\mu}_i)(o_t - \hat{\mu}_i)^T}{\sum_{t=1}^{T} \gamma_t(i)}.
$$

Then, a new estimation of the HMM $\hat{\Lambda} = (\hat{\pi}, \hat{T}, \{\hat{\phi}_i\}_{i=1}^{M})$, that has been proved to have a higher probability generating the observed sequence i.e., $P(O|\hat{\Lambda}) > P(O|\Lambda)$, is obtained. By iteratively repeating the above procedure, the estimated $\Lambda$ will converge to a local maximum.

Since a HMM can be seen as an abstraction of the sequence(s) from which it is estimated, measuring the dissimilarity of two HMMs is meaningful for sequence classification and clustering. Traditionally, Kullback-Leibler divergence (KLD)-based distance[43] has been the dominating approach:

$$
d(\Lambda^{(1)}, \Lambda^{(2)}) = \int_{O_T} \frac{1}{T} \log\left(\frac{P(O_T|\Lambda^{(1)})}{P(O_T|\Lambda^{(2)})}\right) dO_T
$$

$$
= \lim_{T \to \infty} \frac{1}{T} \left(\log(P(O_T|\Lambda^{(1)})) - \log(P(O_T|\Lambda^{(2)}))\right).
$$

where $O_T = o_1 o_2 \ldots o_T$. In the second line, $O_T$ is sampled from $\Lambda^{(1)}$ and $P(O_T|\Lambda)$ can be evaluated using the forward procedure.

### 1.1.3 Optimal Transport

Optimal transport is a set of theories and algorithms for realizing a transportation at minimal cost. As a motivating example, let us assume there is a pile of sand and a hole to be filled up with the sand, as illustrated in Fig. 1.3 (a). Of course, the pile of sand and the hole can be assumed to have the same volume. Without loss of generality, the pile of sand is assumed to have a mass of 1. Mathematically, we can formulate this simple sand transportation example as following: the pile and the hole shall be modeled by probability measures $\mu$ and $\nu$, defined on some measure space.
$X,$ and $Y,$ respectively. If we let $A$ and $B$ be any measurable subsets of $X$ and $Y$, $\mu(A)$ and $\nu(B)$ give the amount of sand located inside $A$, and the amount of sand to be transported toward $B$. The cost of such sand transportation can be modeled by a measurable cost function defined on $X \times Y$: $c(x, y): X \times Y \to \mathbb{R}_+ \cup +\infty$. The physical meaning of $c(x, y)$ could be interpreted as the point to point cost, i.e., the cost of transporting one unit of mass from location $x \in X$ to location $y \in Y$. Optimal transport aims at solving a transport plan $\Pi(x, y): X \times Y \to \mathbb{R}_+ \cup +\infty$, which represents how much sand is transported from location $x$ to location $y$.

![Diagram of moving sand](image)

(a) Moving Sand

![Diagram of moving bricks](image)

(b) Moving Bricks

**Figure 1.3.** Motivating examples for optimal transport. The figure (a) is reproduced from the Chapter 1 of [90].

Real world problems are often discrete. Even for a continuous problem, sometimes a discrete approximation is desired to make it computable. The ideal optimal transport problem formulation based on probability theory above can be easily extended to a discrete case. As a motivating example for the discrete optimal transport, Fig. 1.3 (b) illustrates a problem where we have to move stacks of bricks to fill an
abandoned swimming pool. Mathematically, we can formulate the discrete optimal transportation problem as follows: Let \( p \in \Delta_{m_1}, q \in \Delta_{m_2} \) where \( \Delta_m \) stands for the set of \( m \)-dimention simplex: 
\[
\Delta_m \overset{\text{def}}{=} \{ p \in \mathbb{R}_+^m : \langle p, \mathbb{1} \rangle = 1 \}.
\]
In the moving bricks example, \( p, q \) represents the distribution of the bricks and the distribution of vacancy of the swimming pool respectively, both of which normalized to 1; \( m_1 \) and \( m_2 \) can be regarded as the number of brick stacks and the number of vacant “columns” in the swimming pool to be filled with bricks. The set of transportation plans can be represented as 
\[
\Pi(p, q) \overset{\text{def}}{=} \{ Z \in \mathbb{R}^{m_1 \times m_2} : Z \cdot \mathbb{1}_{m_2} = p, Z^T \cdot \mathbb{1}_{m_1} = 1 \},
\]
often called the coupling set. Thus, the transportation cost can be represented as \( M \in \mathbb{R}^{m_1 \times m_2} \), where \( M_{i,j} \) represents the cost of moving a brick from stack \( i \) to vacant “column” \( j \). The goal is to solve the optimal transport cost between \( p \) and \( q \) with respect to \( M \):

\[
W(p, q) \overset{\text{def}}{=} \min_{Z \in \Pi(p,q)} \langle Z, M \rangle.
\]

For the above linear programming problem, there is a dual optimization problem that is more convenient to be solved sometimes. Let \( C_M > 0 \). And let \( g = [g_1, \ldots, g_{m_1}]^T \) and \( h = [h_1, \ldots, h_{m_2}]^T \) be the dual variable. Then for a sufficient large \( C_M \) (subject to \( p, q, \) and \( M \)) the dual optimization problem is:

\[
W(p, q) = \max_{f \in \Omega(M)} \langle p, g \rangle - \langle q, h \rangle,
\]

where

\[
\Omega(M) \overset{\text{def}}{=} \{ f = [g, h] \in \mathbb{R}^{m_1+m_2} \mid -C_M < g_i - h_j \leq M_{i,j}, 1 \leq i \leq m_1, 1 \leq j \leq m_2 \}.
\]

Finally, we will review Sinkhorn[26], a notable fast numerical algorithm for solving the optimal transport optimization up to some tolerable error. The basic idea is to smooth the original optimization objective with an entropy term. The objective now changes to:

\[
W(p, q) \overset{\text{def}}{=} \min_{Z \in \Pi(p,q)} \langle Z, M \rangle - \eta^{-1}H(Z),
\]

where \( \eta > 0 \), \( H(Z) \) is the entropy of the joint probability \( Z \). [26] proved that one
can solve the above modified problem up to an error threshold of \( \epsilon' \):

**Algorithm 1** Sinkhorn algorithm for optimal transportation

**Input:** \( A, \Pi_{r,c}, \epsilon' \)

**Output:** \( A^{(k)} \)

1. initialize \( k = 0 \)
2. \( A^{(0)} \leftarrow A/\|A\|_1, \ x^0 \rightarrow 0, \ y^0 \rightarrow 0 \).
3. \# \ dist(A^{(k)}, \Pi_{r,c}) = ||\cdot||_2 + ||\cdot||_2
4. **while** \( \text{dist}(A^{(k)}, \Pi_{r,c}) < \epsilon' \) **do**
5. \( k \rightarrow k + 1 \)
6. **if** \( k \) is odd **then**
7. \( x_i \rightarrow \log(\frac{r_i}{\exp(A^{(k-1)})}) \) for \( 1 \leq i \leq m_1 \) \# \( r_i(\cdot) \) is the sum of the \( i \)-th row.
8. \( x^k \rightarrow x^{k-1} + x, \ y^k \rightarrow y^{k-1} \)
9. **else**
10. \( y_i \rightarrow \log(\frac{c_i}{\exp(A^{(k-1)})}) \) for \( 1 \leq i \leq m_2 \) \# \( c_i(\cdot) \) is the sum of the \( i \)-th column.
11. \( x^k \rightarrow x^{k-1}, \ y^k \rightarrow y^{k-1} + y \)
12. **end if**
13. \( A^{(k)} \rightarrow D(\exp(x^k))AD(\exp(y^k)) \) \# \( D \) converts vector to a diagonal matrix
14. **end while**

Sinkhorn has been proved in [44] to reach convergence in \( \mathcal{O}(\rho(\epsilon')^{-2} \log(s/l)) \) iterations, where \( s = \sum_{i,j} A_{i,j}, \ l = \min_{i,j} A_{i,j} \) and \( \rho \geq r_i, c_j \forall i,j \). If we change the convergence criterion from \( L_2 \) to \( L_1 \), i.e. \( \text{dist}(A^{(k)}, \Pi_{r,c}) = ||\cdot||_1 + ||\cdot||_1 \), which is more appropriate for simplex constrained probabilities, Sinkhorn is able to converge in \( \mathcal{O}((\epsilon')^{-2} \log(s/l)) \) [8] to obtain a solution \( \hat{Z} \) satisfying:

\[
\langle \hat{Z}, M \rangle \leq \min_{Z \in \Pi_{\mathbb{P},\mathbb{Q}}} \langle Z, M \rangle + \frac{2\log(n)}{\eta} + 4\epsilon'\|M\|_\infty,
\]

where \( n = \max\{m_1, m_2\} \). If we let \( \eta = \frac{4\log(n)}{\epsilon'}, \ \epsilon' = \frac{\epsilon}{8\|M\|_\infty} \) in the above equation, it shows that Sinkhorn can output an solution \( \hat{Z} \) with \( \epsilon \) guarantee in the objective in \( \mathcal{O}(\epsilon'^{-2}(\log(n) + \eta\|M\|_\infty) \) Sinkhorn iterations, which is equivalent to \( \mathcal{O} \left( \frac{m_1 m_2 \log(\max\{m_1, m_2\})}{\epsilon^3 ||\cdot||_\infty^2} \right) \) time complexity.
1.1.4 Geometric Model – Why defining a distance is essential

A geometric model describes the observed data by characterizing a metric space in which we assume all data resides in. The characterization of the metric space is accomplished by providing a distance measure between any two data points, or in other words, two instances in the dataset. It contrasts with a probabilistic model, which assumes the datapoints are sampled probabilistically from an underlying distribution specified by the model, and a logical model, which uses logical expression to divide data points into homogeneous groups. Notable examples of geometric models include nearest-neighbor models and k-means clustering. Notable examples of probabilistic models include naive Bayes model. Notable examples of logical models include decision trees. The probabilistic model can be further categorized into discriminative models and generative models, where the former models the conditional distribution $P(Y|X)$ and the latter models the joint distribution $P(X,Y)$ if we denote the datapoint as $(X,Y)$ with $X$ being the feature and $Y$ being the label to be predicted. As an example, hidden Markov model is a probabilistic model since it assigns probability to sequences and is also a generative model.

1.1.5 Transfer Learning

The goal of transfer learning is to learn from related learning problems such that the representation of a model learned on a certain domain for a task can benefit another model of interest that trained on a different domain or for a different task. More formally, we assume the training data and test data are from the same domain $D = (\mathcal{X}, p(X))$, where $\mathcal{X}$ is the feature space and $p(X)$ is the distribution of the samples. And we define a task is defined by $T = (\mathcal{Y}, f(\cdot))$, where $\mathcal{Y}$ is the label space and $f : \mathcal{X} \rightarrow \mathcal{Y}$ is the mapping between the feature space to the label space that we are trying to learn. In transfer learning, we aim to benefit a target model $T$ on domain $D_T$ and for task $T_T$ from the representation of another so called source model $S$ on domain $D_S$ and for task $T_S$. The three most common scenarios are:

1. “Same domain, different tasks” where $D_T = D_S, T_D \neq T_S$. 
2. “Different domains, same task”, also known as domain adaptation, where $\mathcal{D}_T \neq \mathcal{D}_S, \mathcal{T}_D = \mathcal{T}_S$.

3. “Different task, different domains” where $\mathcal{D}_T \neq \mathcal{D}_S, \mathcal{T}_D \neq \mathcal{T}_S$.

For medical imaging problems, where large scale training data is expensive, and sometimes impossible to collect, transfer learning has attracted more attention to in recent years as a powerful way to improve the robustness and performance of the learned models [91, 70, 27, 4, 69].

1.1.6 Placenta Pathology and Imaging

![Figure 1.4. The Placenta. (The image source for (a) is National Institute of Diabetes and Digestive and Kidney Diseases, National Institutes of Health [3]. The photos in (b) and (c) are from the dataset we curated (see in Chapter 3.))](image-url)
The placenta is a temporary organ serving as the bridge between the mother and the fetus, allowing the nutrient delivery, waste elimination and gas exchange (both oxygen and carbon-dioxide) through the mother’s blood circulation. As illustrated in Fig. 1.4 (a), before the delivery, the maternal side of the placenta is attached to the womb of the mother and the fetal side of the placenta connects with the fetus through the umbilical cord. A placenta is made of both maternal and fetal cells, and is in the shape of a disc as illustrated in Fig. 1.4 (b) and (c). It has three main functions: metabolic function, endocrine function, and immune function. Hence, identifying placental disease can be tremendously useful for the short and long-term clinical care of both mother and child.

As described in [13], existing placenta imaging research can be roughly categorized into two types: those using microscopic images of the slices of the placentas and those using the macroscopic images of the placentas taken by cameras. Both types are useful. So the placenta diagnosis reports in many U.S. hospitals, including Northwestern Memorial Hospital, record indicators and measurements from both the image of placenta slices under a microscope and gross placenta. Past research on macroscopic image analysis of placentas mostly focus on manually measuring the shape, color, eccentricity, cord-related statistics, and other characteristics of the placenta from digital photographs [93, 92, 34, 77, 66, 65]. They support the use of diagnostic and research utility of placental photographs.

1.2 Organization of the Thesis

The thesis is organized as follows:

1. Chapter 2 presents the aggregated Wasserstein framework for defining a distance between two hidden Markov models. This chapter is based on [20], [21].

2. Chapter 3 proposes a novel pipeline for addressing automated morphological characterization of placentas. The main model consists of 3 encoder-decoder networks with a shared encoder. A transfer learning training strategy is employed to transfer the representation learned from the placental segmentation
task to disc side classification task and umbilical cord insertion point localization task. This chapter is based on [19], [22].

3. Chapter 4 concludes the thesis and discusses a number of interesting directions that can be explored in the future.
Aggregated Wasserstein Distance for Hidden Markov Models

2.1 Introduction

A hidden Markov model (HMM) with a Gaussian emission distribution for any given state is a widely used stochastic model for time series of vectors residing in an Euclidean space. It has been widely used in the pattern recognition literature, such as acoustic signal processing (e.g. [11, 24, 40, 61, 95, 68, 80, 29]) and computer vision (e.g. [72, 16, 81, 7]) for modeling spatial-temporal dependencies in data. Although the conditional distribution of the observation given the state in an HMM is not necessarily Gaussian, in our discussion, unless specified, the Gaussian distribution is assumed. The marginal distribution of an observation at any time spot thus follows a Gaussian mixture model (GMM). Our new distance between HMMs heavily exploits the GMM marginal distribution. In some literature, GMM (rather than the single Gaussian distribution) is assumed for the observation conditioned on any given state.

The work presented in this section is based on the following paper that has been accepted for publication: Yukun Chen, Jianbo Ye, Jia Li. “Aggregated Wasserstein Distance and State Registration for Hidden Markov Models”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2019
in HMM. This more general form of HMM is equivalent to an HMM containing an enlarged set of states with single Gaussian distributions. Hence, it poses no particular difficulty for our proposed framework. More detailed remarks are given in Section 2.6.

A long-pursued question is how to quantitatively compare two sequences based on the parametric representations of the HMMs estimated from them respectively. The HMM parameters lie on a non-linear manifold. Thus a simple Euclidean distance on the parameters is not proper. As argued in the literature (e.g. [25, 83]), directly comparing HMM in terms of the parameters is non-trivial, partly due to the identifiability issue of parameters in a mixture model. Specifically, a mixture model can only be estimated up to the permutation of states. Different components in a mixture model are actually unordered even though labels are assigned to them. In other words, the permutation of labels has no effect on the likelihood of the model. Some earlier solutions do not principally tackle the parameter identifiability issue and simply assume the components are already aligned based on whatever labels given to them [53]. Other more sophisticated solutions sidestep the issue to use model independent statistics including the KL divergence [43, 96] and probability product kernels [41, 42]. Those statistics however cannot be computed easily, requiring Monte Carlo samples or the original sequences [83, 57], which essentially serve as Monte Carlo samples. To overcome the computational intensity, various fast approximations for KL divergence between HMMs (and GMMs) have been proposed to tradeoff speed and accuracy. Notable examples include variational approximation [36, 37] and fast component matching approximation [32, 89].

Sometimes approaches that use the original sequence data give more reliable results than the Monte Carlo approaches. But such approaches require that the original sequences are instantly accessible at the phase of data analysis. Imagine a setting where large volumes of data are collected across different sites. Due to the communication constraints or the sheer size of data, it is possible that one cannot transmit all data to a single site. We may have to work on a distributed platform. The models are estimated at multiple sites; and only the models (much compressed
information from the original data) are transmitted to a central site. This raises the need of approaches requiring only the model parameters. Existing methods using only the model parameters typically rely on Monte Carlo sampling (e.g. KL-D based methods [43]) to calculate certain log-likelihood statistics. However, the rate of convergence in estimating the log-likelihoods is $O \left( \left( \frac{1}{n} \right)^{2/d} \right)$ [52, 62], where $n$ is the data size and $d$ the dimension. This can be slow for HMMs in high dimensions, not to mention the time to generate those samples.

In this paper, we propose a non-simulation parameter-based framework named Aggregated Wasserstein to compute the distance between HMMs. To address the state identifiability issue, the framework first solves a registration matrix between the states of two HMMs according to an optimization criterion. The optimization problem is essentially a fast approximation to the Wasserstein metric between two marginal GMMs. Once the registration matrix is obtained, we compute separately the difference between the two marginal GMMs and the difference between two transition matrices. Finally, we combine the two parts by a weighted sum. The weight can be cast as a trade-off factor balancing the importance between differentiating spatial geometries and stochastic dynamics of two HMMs.

Empirical results show that the advantages of the aggregated Wasserstein approach are not restricted to computational efficiency. In fact, the new distance outperforms KL divergence purely as a distance measure under some scenarios. We thus move one step further under this parameter-based framework for defining a distance between HMMs. Aiming at improving how the states are registered, we propose a second approach to calculate the registration matrix based on Monte Carlo sampling. The second approach overcomes certain limitations of the first approach, but at the cost of being more computationally expensive. Despite requiring Monte Carlo samples, the second approach has a rate of convergence asymptotically at $O \left( \sqrt{\frac{\log n}{n}} \right)$ — much faster than the rate of computing log-likelihood based statistics in high dimensions.

We investigate our methods in real world tasks and compare them with the KL divergence-type methods. Practical advantages of our approach have been demon-
strated in real applications. By experiments on synthetic data, we also make effort to discover scenarios when our proposed methods outperform the others.

**Related works.** KL divergence and Wasserstein distance are useful and popular for comparing GMMs. They are both computationally intensive to solve for GMMs; and fast approximation methods have been developed in each case. For instance, variational approximation methods [36] have been proposed to speed up the calculation of KL. Interestingly, closed form solutions exist for the KL divergence as well as Wasserstein distance between Gaussians. Exploiting this fact, for both distances, the idea of matching components in the GMMs and then aggregating the pairwise distances between the components has been used for fast approximation, e.g., [32, 89] for KL, and [54, 20, 18] for Wasserstein. In fact, variational approximation methods [36, 37] also implicitly perform component matching as the variational parameters can be viewed as matching weights. We discuss these works in details below.

1. For KL, [32] proposed both a fast hard component matching scheme and a more accurate but slower soft component matching scheme based on unscented transformation. In [89], the *Asymptotic Likelihood Approximation* is proposed, which uses the same hard matching scheme as [32], but more theoretical analysis is provided.

2. For Wasserstein distance, to the best of our knowledge, [20] by the same authors here is the first work that approximate the true Wasserstein distance between GMMs using a scheme of “optimal transport within optimal transport”. The work [18], which appeared slightly later, uses the same idea but also explores barycenter calculation based on the proposed distance. The idea of component matching similar to [20, 18] was considered earlier in [54]. However, Lie-group-based distance was used in that work instead of the closed form Wasserstein distance between two Gaussians. As a result, the distance of [54] is not guaranteed to be the upper bound of the true Wasserstein distance between GMMs. We note that there are other approximation methods for the Wasserstein distance without relying on component matching. For example, the sliced Wasserstein distance proposed in [49, 48]. The idea is to
randomly project high dimensional distributions onto one dimension, in which the Wasserstein distance is easy to compute. In order to compare HMMs, we need the component matching matrix not just for the sake of comparing GMMs, but also for comparing the state transition matrices.

For HMMs, which is more complex than GMMs, KL divergence [43] has been dominant for model comparison in the literature. Various fast approximation methods have been proposed, such as variational approximation [37] and a coarser but even faster approximation by an upper bound of KL in closed form [28].

**Our contributions.** We develop a parameter-based framework for computing a distance between HMMs. Under such framework, a registration matrix is computed for the states in two HMMs. Two methods have been proposed to compute the registration, resulting in two distances, named *Minimized Aggregated Wasserstein* and *Improved Aggregated Wasserstein*. Both distances are experimentally validated to be robust and effective, often outperform KL divergence-based methods in practice.

The rest of the paper is organized as follows. We introduce notations and preliminaries in Section 2.2. The main framework for defining the distance is proposed in Section 2.3. The second approach based on Monte Carlo to compute the registration between two sets of HMM states is described in Section 2.4. Finally, we investigate the new framework empirically in Section 2.5 based on synthetic and real data.

## 2.2 Preliminaries

In Section 2.2.1, we review HMM and introduce notations. Next, the definition for Wasserstein distance is provided in Section 2.2.2, and its difference from the KL divergence in the case of Gaussian distributions is discussed.

### 2.2.1 Notations and Definitions

Consider a sequence $O_T = \{o_1, o_2, ..., o_T\}$ modeled by an HMM. Suppose there are $M$ states: $S = \{1, \ldots, M\}$. An HMM under the stationary condition assumes the
following:

1. Each observation \( o_i \in O_T \) is associated with a hidden state \( s_i \in S \) governed by a Markov chain (MC).

2. \( T \) is the \( M \times M \) transition matrix of the MC \( T_{i,j} \overset{\text{def}}{=} P(s_{t+1} = j|s_t = i) \), \( 1 \leq i, j \leq M \) for any \( t \in \{1, \ldots, T\} \). The stationary (initial) state probability \( \pi = [\pi_1, \pi_2, \ldots, \pi_M] \) satisfies \( \pi T = \pi \) and \( \pi 1 = 1 \).

3. The Gaussian probabilistic emission function \( \phi_i(o_t) \overset{\text{def}}{=} P(o_t|s_t = i), \quad i = 1, \ldots, M, \) for any \( t \in \{1, \ldots, T\} \), is the p.d.f. of the normal distribution \( N(\mu_i, \Sigma_i) \), where \( \mu_i, \Sigma_i \) are the mean and covariance of the Gaussian distribution conditioned on state \( i \).

In particular, we use \( \mathcal{M}(\{\mu_i\}_{i=1}^M, \{\Sigma_i\}_{i=1}^M, \pi) \) to denote the corresponding mixture of \( M \) Gaussians ( \( \{\phi_1, \phi_2, \ldots, \phi_M\} \) ). As we assume the Markov chain has become stationary, \( \mathcal{M}'s \) prior probabilities of components, also known as the mixture weights, are determined by \( \pi \), the stationary distribution of \( T \). Therefore, one can summarize the parameters for a stationary HMM via \( \Lambda \) as \( \Lambda(T, \mathcal{M}) = \Lambda(T, \{\mu_i\}_{i=1}^M, \{\Sigma_i\}_{i=1}^M) \). In addition, the \( i \)-th row of the transition matrix \( T \) is denoted by \( T(i,:) \in \mathbb{R}^{1 \times M} \). And the next observation’s distribution conditioned on current state \( i \) is also a GMM: \( \mathcal{M}^{(i)}(\{\mu_i\}_{i=1}^M, \{\Sigma_i\}_{i=1}^M, T(i,:)) \), which we abbreviated as \( \mathcal{M}^{(i)}|_{T(i,:)} \).

### 2.2.2 The Wasserstein Distance and the Gaussian Case

In probability theory, Wasserstein distance [90, 47] is a geometric distance naturally defined for any two probability measures over a metric space. An enlightening introduction is provided by [33] on the usage of this distance in information theory and the many other names it has been known of (e.g., Monge-Kantorovich, Ornstein, d-bar, Mallows).

**Definition 1** (p-Wasserstein distance). *Given two probability distribution \( f, g \) defined
on Euclidean space $\mathbb{R}^d$, the $p$-Wasserstein distance $W_p(\cdot, \cdot)$ between them is given by

$$W_p(f, g) \overset{def}{=} \left( \inf_{\gamma \in \Pi(f, g)} \int_{\mathbb{R}^d \times \mathbb{R}^d} \|x - y\|^p d\gamma(x, y) \right)^{1/p}, \quad (2.1)$$

where $\Pi(f, g)$ is the collection of all distributions on $\mathbb{R}^d \times \mathbb{R}^d$ with marginal $f$ and $g$ on the first and second factors respectively. In particular, the $\Pi(\cdot, \cdot)$ is often called as the coupling set. The $\gamma^* \in \Pi(f, g)$ that takes the infimum in Eq. (2.1) is called the optimal coupling.

Because it takes cross-support relationship into consideration, it has shown strength in computer vision [55, 73], document classification [50, 38], data mining[9] etc.

**Remark 1.** By the Hölder inequality, one has $W_p \leq W_q$ for any $p \leq q < \infty$. In this paper, we focus on the practice of $W_p$ with $0 < p \leq 2$.

**Proof.** Let the solved optimal coupling for $W_p$ and $W_q$ be $\gamma_p$ and $\gamma_q$ respectively. To prove $W_p \leq W_q$, $\forall p \leq q$, i.e.:

$$\left( \int \|x - y\|^p d\gamma_p \right)^{1/p} \leq \left( \int \|x - y\|^q d\gamma_q \right)^{1/q}, \quad (2.2)$$
we can prove the stronger inequality below instead:

\[
\left[ \int \|x - y\|^p d\gamma_q \right]^{1/p} \leq \left[ \int \|x - y\|^q d\gamma_q \right]^{1/p} 
\]

(2.3)

Eq (2.3) is stronger than (2.2) because $\gamma_p$ is the optimal transport for $W_p$. Let $f = \|x - y\|^p$. Then Eq (2.3) becomes

\[
\int f d\gamma_p \leq \left[ \int f^{q/p} d\gamma_q \right]^{p/q} 
\]

(2.4)

Let $\mu = q/p \geq 1$,

\[
\Leftrightarrow \int f d\gamma_q \leq \left[ \int f^\mu d\gamma_q \right]^{1/\mu} 
\]

(2.5)

This holds because:

\[
\Leftrightarrow \int f d\gamma_q \leq \left[ \int f^\mu d\gamma_q \right]^{1/\mu} \left[ \int 1^{\mu/(\mu-1)} d\gamma_q \right]^{1-1/\mu} 
\]

(2.6)

by the Hölder inequality.

While Wasserstein distance between two multi-dimensional GMMs is unsolved, it has a closed formula for two Gaussian $\phi_1(\mu_1, \Sigma_1)$ and $\phi_2(\mu_2, \Sigma_2)$ [31] when $p = 2$:

\[
W_2^2(\phi_1, \phi_2) = \|\mu_1 - \mu_2\|^2 + \text{tr} \left[ \Sigma_1 + \Sigma_2 - 2 \left( \Sigma_1^{1/2} \Sigma_2 \Sigma_1^{1/2} \right)^{1/2} \right]. 
\]

(2.7)

Remark 2. The formula of Wasserstein distance between two Gaussians does not involve the inverse-covariance matrix, thus admits the case of singularity. In comparison, KL divergence between two Gaussian $KL(\phi_1, \phi_2)$ goes to infinity if one of the covariances of $\phi_1$ and $\phi_2$ becomes singular.

Remark 3. The Wasserstein distance could also be more statistically robust than KL divergence by comparing the variance of their estimations. To illustrate this point, we conduct two sets of toy experiments.

Our first toy experiment is shown in Fig. 2.1 (a) upper figure. First, we sample 100 batches of data, each of size 50, from the pre-selected Gaussian
\( \phi_0 = \mathcal{N} \left( [0,0], \begin{pmatrix} 1,0 \\ 0,1 \end{pmatrix} \right). \) Then, we re-estimate each batch’s Gaussian parameters \( \hat{\phi}_0 = \mathcal{N}(\hat{\mu}, \hat{\Sigma}) \approx \phi_0 \) and calculate \( W(\hat{\phi}_0, \phi_i) \) and \( KL(\hat{\phi}_0, \phi_i) \), in which
\[ \phi_i = \mathcal{N} \left( [0.5 \cdot i, 0.5 \cdot i], \begin{pmatrix} 1,0 \\ 0,1 \end{pmatrix} \right), \quad i = 1, \ldots, 10 \] is a sequence of Gaussians, both with closed forms. Ideally, a distance that can consistently differentiate \( \phi_i \) by computing its distance to the \( \hat{\phi} \) should have larger value as \( i \) grows. Also, its sample deviations of \( W_2(\phi_i, \hat{\phi}_0) \) or \( KL(\phi_i, \hat{\phi}_0) \) should be small enough to not mask out the change from \( i \) to \( i + 1 \). Fig. 2.1 (b) shows the performance of Wasserstein Distance and KL divergence on this toy experiment. Both the averaged distance to \( \phi_i \) and the 3\( \sigma \) confidence interval are plotted. It is clear that the Wasserstein distances based on estimated distributions have smaller variance and can overall better differentiate \( \{\phi_i\} \).

Likewise, in our second toy experiment, we change \( \phi_i \)’s variances rather than their means (See Fig. 2.1 (a) bottom figure). At this time, we set \( \phi_i = \mathcal{N} \left( [0,0], \exp(0.5 \cdot i) \cdot \begin{pmatrix} 1,0 \\ 0,1 \end{pmatrix} \right) \). The result is plotted in Fig. 2.1 (c). It shows that KL divergence can be more robust than Wasserstein distance if \( \hat{\phi}_0 \) is compared to \( \phi_i \) at \( i < 0 \), but the situation quickly becomes worse at \( i \geq 2 \). This is due the asymmetric nature of KL divergence. Informally speaking, we conclude from the two toy experiments that estimating \( KL(\phi_i, \phi_0) \) can be statistically stable if \( \phi_i \) is under the “umbrella” of \( \phi_0 \), and becomes inaccurate otherwise. On the other hand, Wasserstein distance, as a true metric [90], has consistent accuracy across these two settings.

Note that the subscript in \( \widetilde{R}_p, \widetilde{W}_p, \) and \( D_p \) (the latter two will be defined shortly) corresponds to the power term in Eq. (2.13) and does not indicate that we use the \( p \)-Wasserstein distance in their definitions. In fact, as shown by Eq. (2.13), we always use 2-Wasserstein distance \( W_2 \). For the MAW distance, formally defined later in Sec. 2.3.4, the matching weights in \( W \) are the immediate result of Eq. (2.11) and Eq. (2.12), no extra computation needed. Instead of solving the optimal transport
between any two points required by the true Wasserstein distance between GMMs, Eq.(2.13) is essentially a scheme of “optimal transport within optimal transport”. The pairwise distance between Gaussian components is obtained by one level of optimal transport between data points, as implied by $W^p_2(\phi_{1,i},\phi_{2,j})$, while the overall distance between GMMs is computed by another level of optimal transport between the components.

We also note that the registration matrix solved by a scheme other than Eq. (2.11) (to be introduced in Section 2.4) can also be plugged into this equation. We will later prove that $\tilde{R}_p(M_1, M_2; W)$ is a semi-metric (Theorem 2). Next, we present Theorem 1 stating that this semi-metric is an upper bound on the true Wasserstein metric.

**Theorem 1.** For any two GMMs $M_1$ and $M_2$, define $\tilde{R}_p(\cdot, \cdot : W)$ by Eq. (2.13). If $W \in \Pi(\pi_1, \pi_2)$, we have for $0 < p \leq 2$

$$\tilde{R}_p(M_1, M_2 : W) \geq W_p(M_1, M_2),$$

where $W_p(M_1, M_2)$ is the true Wasserstein distance between $M_1$ and $M_2$ as defined in Eq. (2.1).

**Proof.** By Remark 1, We have:

$$\tilde{R}_p^p(M_1, M_2; W) \geq \sum_{i=1}^{M_1} \sum_{j=1}^{M_2} w_{i,j} W^p_p(\phi_{1,i}, \phi_{2,j}) \quad (2.8)$$

We construct $\gamma \in \Pi(M_1, M_2)$ in the following way: Given a $W \in \Pi(\pi_1, \pi_2)$ and any $\gamma_{i,j} \in \Pi(\phi_{1,i}, \phi_{2,j})$ for $i = 1, \ldots, M_1$ and $j = 1, \ldots, M_2$, we let $\tilde{\Pi}(M_1, M_2) =$

$$\left\{ \gamma \overset{\text{def}}{=} \sum_{i=1}^{M_1} \sum_{j=1}^{M_2} w_{i,j} \gamma_{i,j} \left| W \in \Pi(\pi_1, \pi_2), \text{ and } \gamma_{i,j} \in \Pi(\phi_{1,i}, \phi_{2,j}), i = 1, \ldots, M_1, j = 1, \ldots, M_2 \right\} \quad (2.9)$$
and \( \sum_{i=1}^{M_1} \sum_{j=1}^{M_2} w_{i,j} W_p(\phi_{1,i}, \phi_{2,j}) \) is the exact infimum for all possible \( \gamma \in \tilde{\Pi}(\mathcal{M}_1, \mathcal{M}_2) \), where we see \( \tilde{\Pi}(\mathcal{M}_1, \mathcal{M}_2) \subseteq \Pi(\mathcal{M}_1, \mathcal{M}_2) \). Thus,

\[
\sum_{i=1}^{M_1} \sum_{j=1}^{M_2} w_{i,j} W_p(\phi_{1,i}, \phi_{2,j}) \geq W_p(\mathcal{M}_1, \mathcal{M}_2) \quad (2.10)
\]

By combining it with Eq. (2.8), the inequality is implied. \( \square \)

For the brevity of notation, if \( W \) is solved from Eq. (2.11), the resulting distance \( \tilde{R}_p(\mathcal{M}_1, \mathcal{M}_2; W) \) is denoted by \( \tilde{W}_p(\mathcal{M}_1, \mathcal{M}_2) \).

### 2.3 The Framework of Aggregated Wasserstein

In this section, we propose a framework to compute the distance between two HMMs, \( \Lambda_1(\mathcal{T}_1, \mathcal{M}_1) \) and \( \Lambda_2(\mathcal{T}_2, \mathcal{M}_2) \), where \( \mathcal{M}_l \), \( l = 1, 2 \) are marginal GMMs with pdf \( f_l(x) = \sum_{j=1}^{M_l} \pi_{l,j} \phi_{l,j}(x) \) and \( \mathcal{T}_1, \mathcal{T}_2 \) are the transition matrices of dimension \( M_1 \times M_1 \) and \( M_2 \times M_2 \) (recall notations in Section 2.2 ). Based on the registration matrix between states in two HMMs, to be described in Section 2.3.1, the distance between \( \Lambda_1 \) and \( \Lambda_2 \) consists of two parts: (1) the difference between \( \mathcal{M}_1 \) and \( \mathcal{M}_2 \) (Section 2.3.2); and (2) the difference between \( \mathcal{T}_1 \) and \( \mathcal{T}_2 \) (Section 2.3.3).

#### 2.3.1 The Registration of States

The registration of states is to build a correspondence between \( \Lambda_1 \)'s states and \( \Lambda_2 \)'s states. In the simplest case (an example is illustrated in Fig 2.2), if the two marginal GMMs are identical distributions but the states are labeled differently (referred to as permutation of states), the registration should discover the permutation and yield a one-one mapping between the states. We can use a matrix \( W = \{w_{i,j}\} \in \mathbb{R}^{M_1 \times M_2} \) whose elements \( w_{i,j} \geq 0 \) to encode this registration. In particular, \( w_{i,j} = \pi_{1,i}(= \pi_{2,j}) \) iff state \( i \) in \( \Lambda_1 \) is registered to state \( j \) in \( \Lambda_2 \). With \( W \) given, through matrix multiplications (details delayed in Section 2.3.3), the rows and columns of \( \mathcal{T}_1 \) can be permuted to become identical to \( \mathcal{T}_2 \).
Figure 2.2. A simple registration example about how $T_2$ in $\Lambda_2$ is registered towards $\Lambda_1$ such that it can be compared with $T_1$ in $\Lambda_1$. For this example, $W$ encodes a “hard matching” between states in $\Lambda_1$ and $\Lambda_2$.

Generally and more commonly, there may exist no state in $\Lambda_2$ having the same emission function as some state in $\Lambda_1$, and the number of states in $\Lambda_1$ may not equal that in $\Lambda_2$. The registration process becomes more difficult. We resort to the principled Kantorovich’s formulation of optimal transport [90],[47] as a tool to solve this problem and formulate the following linear programming problem. Recall Eq. (2.7) for how to compute $W_2(\phi_1,i,\phi_2,j)$. Let $0 < p \leq 2$. Consider

$$\min_{W \in \Pi(\pi_1,\pi_2)} \sum_{i=1}^{M_1} \sum_{j=1}^{M_2} w_{i,j} W_2^p(\phi_1,i,\phi_2,j)$$

(2.11)

where $\Pi(\pi_1,\pi_2) \triangleq \{ W \in \mathbb{R}^{M_1 \times M_2} : \sum_{i=1}^{M_1} w_{i,j} = \pi_{2,j}, j = 1, \ldots, M_2; \sum_{j=1}^{M_2} w_{i,j} = \pi_{1,i}, i = 1, \ldots, M_1; \text{and } w_{i,j} \geq 0, \forall i,j \}$.

(2.12)

The rationale behind this is that two states whose emission functions are geometrically close and in similar shape should be more likely to be matched. The solution $W \in \Pi(\pi_1,\pi_2)$ of the above optimization is called the registration matrix between $\Lambda_1$ and $\Lambda_2$. And it will play an important role in the comparison of both marginal GMMs and transition matrices of $\Lambda_1$ and $\Lambda_2$.

The solution of Eq. (2.11) is an extension of hard matching between states for the
simplest case to the general soft matching when the hard matching is impossible. For the aforementioned simple example (Fig. 2.2), in which the two Gaussian mixtures are in fact identical (thus hard matching is possible), the solution of Eq. (2.11) is indeed a permutation matrix $W$ that correctly maps the states in the two models. In general, there are more than one non-zero elements per row or per column.

For HMMs with a large number of states, the computational complexity to solve Eq. (2.11) using basic linear programming is high ($O(M^3)$). Assuming tolerance of small approximation error, optimization regularized by entropy penalty [26] can achieve a complexity at the order of $O(M\log M)$, and a nearly linear complexity is achieved by [8]. In our experiments, as the number of states is not large, linear programming is efficient.

### 2.3.2 The Distance between Two Marginal GMMs

Our aim in this subsection is to quantify the difference between $\Lambda_1$ and $\Lambda_2$’s marginal GMMs $\mathcal{M}_1$ and $\mathcal{M}_2$ with density functions $f_1(x) = \sum_{j=1}^{M_1} \pi_{1,j} \phi_{1,j}(x)$ and $f_2(x) = \sum_{j=1}^{M_2} \pi_{2,j} \phi_{2,j}(x)$ respectively.

Given the discussion on the advantages of the Wasserstein metric (especially the Gaussian case) in Section 2.2, one may ask why not to use Wasserstein distance $W(\mathcal{M}_1, \mathcal{M}_2)$ directly to measure the dissimilarity between $\mathcal{M}_1, \mathcal{M}_2$? Unfortunately, there is no closed form formula for GMMs except for the reduced case of single Gaussians. Monte Carlo estimation is usually used. However, similar to the estimation of KL divergence, the Monte Carlo estimation for the Wasserstein distance also suffers from a slow convergence rate. The rate of convergence is as slow as that of KL divergence, i.e., $O\left(\left(\frac{1}{n}\right)^{1/d}\right)$ [71], again posing difficulty in high dimensions. So instead of estimating the Wasserstein distance itself, we make use of the solved registration matrix $W \in \Pi(\pi_1, \pi_2)$ (from Eq. (2.11)) and the closed form Wasserstein distance between every pair of Gaussians to quantify the dissimilarity between two marginal GMMs $\mathcal{M}_1$ and $\mathcal{M}_2$. The rationale is that the matching weights in $W$ establish a correspondence between the components in the two GMMs, under which a straightforward summation (of course, with proper normalization) of the pairwise
distances between the components quantifies the dissimilarity between the GMMs. Specifically, we define the registered distance between $\mathcal{M}_1$ and $\mathcal{M}_2$ under $W$ as:

$$
\tilde{R}_p(\mathcal{M}_1, \mathcal{M}_2; W) \overset{\text{def}}{=} \left( \frac{1}{p} \sum_{i=1}^{M_1} \sum_{j=1}^{M_2} w_{i,j} W_p^p(\phi_{1,i}, \phi_{2,j}) \right)^{\frac{1}{p}} \quad (2.13)
$$

### 2.3.3 The Distance between Two Transition Matrices

Given the registration matrix $W$, our aim in this subsection is to quantify the difference between $\Lambda_1$ and $\Lambda_2$’s transition matrices, $T_1 \in \mathbb{R}^{M_1 \times M_1}$ and $T_2 \in \mathbb{R}^{M_2 \times M_2}$. Since the identifiability issue is already addressed by the registration matrix $W$, $T_2$ can now be registered towards $T_1$ by the following transform:

$$
\tilde{T}_2 \overset{\text{def}}{=} W_r T_2 W_c \in \mathbb{R}^{M_1 \times M_1}, \quad (2.14)
$$

where matrix $W_r$ and $W_c$ are row-wise and column-wise normalized $W$ respectively, a.k.a. $W_r = \text{diag}^{-1}(W \cdot 1) \cdot W$ and $W_c = W \cdot \text{diag}^{-1}(1^T \cdot W)$. A simple example of this process is illustrated in the right part of Fig. 2.2. Likewise, $T_1$ can also be registered towards $T_2$:

$$
\tilde{T}_1 \overset{\text{def}}{=} W_c^T T_1 W_r \in \mathbb{R}^{M_2 \times M_2}. \quad (2.15)
$$

Then, a discrepancy denoted by $D(T_1, T_2; W)$ to measure the dissimilarity of two transition matrices is defined:

$$
D_p^p(T_1, T_2; W) \overset{\text{def}}{=} d_T^p(T_1, \tilde{T}_2) + d_T^p(T_2, \tilde{T}_1) \quad (2.16)
$$

where $\tilde{T}_1$ and $\tilde{T}_2$ are calculated from Eq. (2.14) and Eq. (2.15) (with $W$ given) respectively and

$$
d_T^p(T_1, \tilde{T}_2) \overset{\text{def}}{=} \sum_{i=1}^{M_1} \pi_1,i \tilde{W}_p^p \left( M_1^{(i)} | t_{1(i, :)}, M_1^{(i)} | t_{2(i, :)} \right) \quad (2.17)
$$

and

$$
d_T^p(T_2, \tilde{T}_1) \overset{\text{def}}{=} \sum_{i=1}^{M_2} \pi_2,i \tilde{W}_p^p \left( M_2^{(i)} | t_{1(i, :)}, M_2^{(i)} | t_{1(i, :)} \right) \quad (2.18)
$$
We remind that by the notations in Section 2.2.1, \( \mathcal{M}^{(i)}_{1\mid T_1(i,:)} \) is the pdf of the next observation conditioned on the previous state being \( i \) (likewise for the other similar terms).

**Remark 4.** The distances \( d_T(T_1, \tilde{T}_2) \) and \( d_T(T_2, \tilde{T}_1) \), as defined in Eq. (2.17) and (2.18), are not determined totally by the transition matrices after registration has been applied. Take \( d^p_T(T_1, \tilde{T}_2) \) as an example. It is a weighted sum over \( \tilde{W}^p_p \left( \mathcal{M}^{(i)}_{1\mid T_1(i,:)} , \mathcal{M}^{(i)}_{1\mid \tilde{T}_2(i,:)} \right) \), which depends on both the \( i \)th rows of \( T_1 \) and \( \tilde{T}_2 \) and the Gaussian components. Apparent simpler alternatives to \( \tilde{W}^p_p \left( \mathcal{M}^{(i)}_{1\mid T_1(i,:)} , \mathcal{M}^{(i)}_{1\mid \tilde{T}_2(i,:)} \right) \) include \( L_p \) distances or KL divergence between the rows of \( T_1 \) and \( \tilde{T}_2 \). The motivation for our more sophisticated treatment is that the states in HMMs often have no actual physical meaning and can suffer from artifacts of parameterization. To measure the similarity between two states, it is more robust to compare the conditional distributions of the observations at the next time position given the states (specified by \( \mathcal{M}^{(i)}_{1\mid T_1(i,:)} \) and \( \mathcal{M}^{(i)}_{1\mid \tilde{T}_2(i,:)} \)) than the conditional distributions of the next states (specified by the rows of \( T_1 \) and \( \tilde{T}_2 \)).

### 2.3.4 A Semi-metric between HMMs — Minimized Aggregated Wasserstein (MAW)

In summary, the dissimilarity between HMMs \( \Lambda_1, \Lambda_2 \) comprises two parts: the first is the discrepancy between the marginal GMMs \( \mathcal{M}_1, \mathcal{M}_2 \), and the second is the discrepancy between two transition matrices after state registration. A weighted sum of these two terms is taken as the final distance. We call this new distance the **Minimized Aggregated Wasserstein (MAW)** between HMMs. The matching weight matrix \( W \) is solved by Eq. (2.11).

\[
\text{MAW}(\Lambda_1, \Lambda_2) \overset{\text{def}}{=} (1 - \alpha) \tilde{R}_p(\mathcal{M}_1, \mathcal{M}_2; W) + \alpha D_p(T_1, T_2; W).
\] (2.19)

Theorem 2 states that MAW is a semi-metric. A semi-metric shares all the properties of a true metric (including separation axiom) except for the triangle inequality.
Theorem 2. MAW defined by Eq. (2.19) is a semi-metric for HMMs if $0 < \alpha < 1$.

Proof. Since Wasserstein Distance is a metric\(^1\),

\[
\begin{align*}
\tilde{R}_p(M_1, M_2; W) &\geq 0, \quad (2.20) \\
\tilde{R}_p(M_1, M_2; W) &= \tilde{R}_2(M_2, M_1; W), \quad (2.21) \\
d_T(T_1, \tilde{T}_2)^p &= \sum_{i=1}^{M_1} \pi_{1,i} \tilde{W}_2 \left( M_1^{(i)} \mid T_1(i,:) \right) \left( M_1^{(i)} \mid \tilde{T}_2(i,:) \right)^p \geq 0, \quad (2.22) \\
d_T(T_2, \tilde{T}_1)^p &= \sum_{i=1}^{M_2} \pi_{2,i} \tilde{W}_2 \left( M_2^{(i)} \mid T_2(i,:) \right) \left( M_2^{(i)} \mid \tilde{T}_1(i,:) \right)^p \geq 0. \quad (2.23)
\end{align*}
\]

By Eq. (2.20), (2.22) and (2.23),

\[
\begin{align*}
MAW(\Lambda_1, \Lambda_2) &\overset{\text{def}}{=} (1 - \alpha) \tilde{R}_p(M_1, M_2; W) + \alpha D_p(T_1, T_2 : W) \quad (2.24) \\
&\geq 0. \quad (2.25)
\end{align*}
\]

And

\[
\begin{align*}
D_p(T_1, T_2 : W)^p &= d_T(T_1, \tilde{T}_2)^p + d_T(T_2, \tilde{T}_1)^p \\
&= d_T(T_2, \tilde{T}_1)^p + d_T(T_1, \tilde{T}_2)^p \\
&= D(T_2, T_1 : W)^p. \quad (2.27)
\end{align*}
\]

By, Eq. (2.21), (2.27),

\[
\begin{align*}
MAW(\Lambda_1, \Lambda_2) &= MAW(\Lambda_2, \Lambda_1). \quad (2.28)
\end{align*}
\]

So we have proved MAW is symmetric, greater or equal than zero. And it is obvious that MAW has zero distance between two HMMs who represent the same

distribution. The remaining part is to prove if two HMMs have zero MAW distance, their distributions are the same.

If $MAW(\Lambda_1, \Lambda_2) = 0$, because $0 < \alpha < 1$ and by Eq. (2.20), (2.22) and (2.23),

$$\tilde{R}_p(M_1, M_2; W) = 0 \quad (2.29a)$$
$$D_p(T_1, T_2 : W) = 0 \quad (2.29b)$$

By Eq. (2.29a) and the fact that Wasserstein distance for Gaussian is a true metric, $M_1$ and $M_2$ should be identical.

By Eq. (2.22), (2.23), and the fact that Wasserstein distance for Gaussian is a true metric,

$$\tilde{W}_p \left( M_1^{(i)} |_{T_1(i,:)} ; M_1^{(i)} |_{T_2(i,:)} \right) = 0, \quad (2.30)$$
$$\tilde{W}_p \left( M_2^{(i)} |_{T_2(i,:)} ; M_2^{(i)} |_{T_1(i,:)} \right) = 0. \quad (2.31)$$

That is $M_1^{(i)} |_{T_1(i,:)} ; M_1^{(i)} |_{T_2(i,:)}$ are identical and $M_2^{(i)} |_{T_2(i,:)} ; M_2^{(i)} |_{T_1(i,:)}$ are identical.

So $T_1 = T_2$. Then $\Lambda_1(M_1, T_1)$ and $\Lambda_2(M_2, T_2)$ should be identical. So, we have proved $MAW$ is a semi-metric.

Note that by Eq. (2.20), (2.21) and the fact that $\tilde{R}_2(M_1, M_2; W) = 0$ iff $M_1$ and $M_2$ are identical, we also proved that $\tilde{R}_2(M_1, M_2; W)$ is a metric for GMM. (We mentioned this at Section 2.3.2)

For clarity, we summarize MAW’s computation procedure in Algorithm 2.

---

**Algorithm 2** Minimized Aggregated Wasserstein (MAW)

**Input:** Two HMMs $\Lambda_1 \left( T_1, M_1 \left( \{\mu_{1,i}\}_{i=1}^{M_1}, \{\Sigma_{1,i}\}_{i=1}^{M_1} \right) \right)$ and $\Lambda_2 \left( T_2, M_2 \left( \{\mu_{2,i}\}_{i=1}^{M_2}, \{\Sigma_{2,i}\}_{i=1}^{M_2} \right) \right)$

**Output:** $MAW(\Lambda_1, \Lambda_2) \in \{0\} \cup \mathbb{R}^+$

1: Compute registration matrix $W$ by Eq. (2.11)
2: Compute $\tilde{R}_p(M_1, M_2; W)$ by Eq. (2.13)
3: Compute $D_p(T_1, T_2)$ by Eq. (2.16), Eq. (2.17) and Eq. (2.18)
4: Compute and return $MAW(\Lambda_1, \Lambda_2)$ defined by Eq. (2.19).

---

**Choosing $\alpha$.** The choice of $\alpha$ can depend on the problem in consideration. In the
context of classification, we aim at the best capability of distinguishing the classes. In particular, $\alpha$ can be determined by maximizing the classification accuracy obtained by the 1-nearest neighbor classifier on a set of small but representative training HMMs with ground truth labels. More complex options such as maximizing the area under the receiver operating characteristics curve (AUC) can also be used. For simplicity, we choose $\alpha$ by maximizing 1-nearest neighbor accuracy in this paper. An example for choosing $\alpha$ in our Motion Retrieval experiment is shown in the Appendix. A.1.

**Arbitrary Initial State Distribution.** Until now we have assumed that the initial state distribution of the HMM is the stationary distribution of the Markov chain governing the states. It is actually straightforward to generalize MAW to HMMs with arbitrary initial state distributions. We will denote $\pi^{(0)}$ as the probability mass function (pmf) of the initial state, that is, $\pi^{(0)}$ is a row vector containing the probability for each possible value of the state. Then the pmf of the states at time $t$ is $\pi^{(t)} = \pi^{(0)} T^t$. The associated GMM marginal distribution of the observation $o_t$ at time $t$ for HMM model $\Lambda_1$ or $\Lambda_2$ is respectively $\mathcal{M}_1^{(t)} (\{\mu_{1,i}\}_{i=1}^M, \{\Sigma_{1,i}\}_{i=1}^M, \pi_1^{(t)})$, $\mathcal{M}_2^{(t)} (\{\mu_{2,i}\}_{i=1}^M, \{\Sigma_{2,i}\}_{i=1}^M, \pi_2^{(t)})$. Denote the registration matrix solved from Eq. (2.11) at time $t$ by $W^{(t)}$. (The only change will be to replace the marginal distribution constraints in Eq. (2.12) by $\pi_1^{(t)}$ and $\pi_2^{(t)}$). Then the MAW with any given initial state distribution can be expressed as:

$$\text{MAW}(\Lambda_1, \Lambda_2) \overset{\text{def}}{=} \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^T \left( (1 - \alpha) \tilde{R}_p(\mathcal{M}_1^{(t)}; \mathcal{M}_2^{(t)}; W^{(t)}) + \alpha D_p(T_1; T_2; W^{(t)}) \right).$$  

Under mild conditions on the transition matrices of the Markov chains, the state distributions $\pi_1^{(t)}$ and $\pi_2^{(t)}$ will converge to the stationary distributions of the Markov chains when $t \to \infty$ (Theorem 4.3.3 of [76]). Then it is easy to see that Eq. (2.32) will also converge to Eq. (2.19) as $t \to \infty$ because the optimal transport distance is continuous with respect to the marginal constraints [90]. For large $T$, Eq. (2.19) is a reasonable approximation to Eq. (2.32).
On the other hand, for finite-length sequences (small $T$), we can define MAW by Eq. (2.32) without taking $\lim_{T \to \infty}$. Because $W^{(t)}$ is computed at each $t$, the amount of computation will increase by a factor of $T$ (complexity linear in $T$). Another possibility is to compute the Wasserstein distance for the joint probability density of the whole sequence of observations. This joint density is a GMM with $M^T$ mixture components. Similarly, we can approximate this Wasserstein distance by the registered distance (see Eq. (2.13)). But even with the approximation in Eq. (2.13), the order of complexity is exponential (specifically, $M^T$), rendering the approach impractical.

### 2.4 Improved State Registration

The main disadvantage of estimating the matching matrix $W$ by Eq. (2.11) and then computing $R_p(\cdot, \cdot)$ by Eq. (2.13) is that $W$ can be sensitive to the parametrization of GMMs. Two GMMs whose distributions are close can be parameterized very differently, especially when the components are not well separated, resulting in a substantially larger value of $\tilde{W}$ than the true Wasserstein metric. In contrast, the real Wasserstein metric $W$ only depends on the underlying distributions $M_{1,2}$, and thus does not suffer from the artifacts caused by the GMM parameterization.

So a key question is “Can we propose a meaningful interpretation of component-wise matching such that the optimal coupling from $\Pi(M_1, M_2)$ for the Wasserstein distance between the two mixture distributions can be realized or approximated arbitrarily well by simulation?” Interestingly, the answer is yes, but we must revise the notion of matching between two individual components. The resulting new approach is called Improved Aggregated Wasserstein (IAW), which approximates the true Wasserstein metric by Monte Carlo simulation.

In the proof of Theorem 1, we see that in the construction of $\tilde{\Pi}$ (Eq. (2.9)) we introduced coupling $\gamma_{i,j} \in \Pi(\phi_{1,i}, \phi_{2,j})$ for all pairs of components $\phi_{1,i}$ and $\phi_{2,j}$. The overall coupling between the two distributions is

$$
\gamma = \sum_{i=1}^{M_1} \sum_{j=1}^{M_2} w_{i,j} \gamma_{i,j},
$$

(2.33)
which decomposes coupling $\gamma$ into two stages. The first stage contains a set of couplings $\gamma_{i,j}$ between component densities $\phi_{1,i}$ and $\phi_{2,j}$ (matching at the level of data points), while the second stage is the component-wise (or component-level) matching specified by $\mathbf{W}$. To maintain a proper meaning of the registration matrix $\mathbf{W}$, we would like to keep such a two stage decomposition for $\gamma$. However, when MAW distance is defined, we have a rigid requirement on $\gamma_{i,j}$: $\gamma_{i,j} \in \Pi(\phi_{1,i}, \phi_{2,j})$, which is at the cost of not being able to approach the true Wasserstein metric. Here, we relax this constraint to the extent that the component densities $\phi_{1,i}$ and $\phi_{2,i}$ are still respected.

**Definition 2.** Given $\mathbf{W} \in \Pi(\pi_i, \pi_j)$, we say densities $\{\tilde{\phi}_{1,i,j}, \tilde{\phi}_{2,i,j}\}$ couple with $\mathbf{W}$ subject to $(\mathcal{M}_1, \mathcal{M}_2)$ if for all $i = 1, \ldots, M_1$ and $j = 1, \ldots, M_2$,

$$
\sum_{j=1}^{M_2} \frac{w_{i,j}}{\pi_{1,i}} \tilde{\phi}_{1,i,j} = \phi_{1,i}, \quad \sum_{i=1}^{M_1} \frac{w_{i,j}}{\pi_{2,j}} \tilde{\phi}_{2,i,j} = \phi_{2,j}.
$$

(2.34)

We denote these conditions collectively by

$$
\{\tilde{\phi}_{1,i,j}, \tilde{\phi}_{2,i,j}\} \in \Gamma(\mathbf{W} \in \Pi(\pi_i, \pi_j)|\mathcal{M}_1, \mathcal{M}_2).
$$

(2.35)

Now when we match component $i$ with $j$, instead of treating $\phi_{1,i}$ and $\phi_{2,j}$, we treat $\tilde{\phi}_{1,i,j}$ and $\tilde{\phi}_{2,i,j}$. Figuratively speaking, we divide $\phi_{1,i}$ into $M_2$ parts $\tilde{\phi}_{1,i,j}$, $j = 1, \ldots, M_2$, and $\phi_{2,j}$ into $M_1$ parts $\tilde{\phi}_{2,i,j}$, $i = 1, \ldots, M_1$ subject to the constraints in Eqs. (2.34) and then match $\tilde{\phi}_{1,i,j}$ and $\tilde{\phi}_{2,i,j}$. Obviously, MAW is a special case where we have identical $\tilde{\phi}_{1,i,j}$’s over all the $j$’s: $\tilde{\phi}_{1,i,j} \equiv \phi_{1,i}$ and similarly $\tilde{\phi}_{2,i,j} \equiv \phi_{2,j}$ for all the $i$’s. This also shows that $\Gamma(\mathbf{W} \in \Pi(\pi_i, \pi_j)|\mathcal{M}_1, \mathcal{M}_2)$ is nonempty for any $\mathbf{W} \in \Pi(\pi_i, \pi_j)$. Fig. 2.3 illustrates the intuition for this improved state registration scheme in comparison with that used for computing MAW. The main difference is the granularity of the solved transportation plan.

Let $\gamma_{i,j}$ be any coupling measure from $\Pi(\tilde{\phi}_{1,i,j}, \tilde{\phi}_{2,i,j})$, that is, $\tilde{\phi}_{1,i,j}$ and $\tilde{\phi}_{2,i,j}$ are marginals of $\gamma_{i,j}$. If $\{\tilde{\phi}_{1,i,j}, \tilde{\phi}_{2,i,j}\}$ couple with $\mathbf{W}$ subject to $(\mathcal{M}_1, \mathcal{M}_2)$, we immedi-
ately have

\[ \gamma = \sum_{i=1}^{M_1} \sum_{j=2}^{M_2} w_{i,j} \gamma_{i,j} \in \Pi(\mathcal{M}_1, \mathcal{M}_2). \]

Alternatively, instead of defining \( \gamma_{i,j} \in \Pi(\phi_{1,i,j}, \phi_{2,i,j}) \), we can bypass the introduction of \( \tilde{\phi}_{1,i,j} \) and \( \tilde{\phi}_{2,i,j} \) and simply impose the following constraints:

\[
\sum_{i=1}^{M_1} w_{i,j} \int x \ d\gamma_{i,j}(x,y) = \pi_{2,j} \phi_{2,j}(y), \quad (2.36)
\]

\[
\sum_{j=1}^{M_2} w_{i,j} \int y \ d\gamma_{i,j}(x,y) = \pi_{1,i} \phi_{1,i}(x). \quad (2.37)
\]

In fact, any \( \gamma \in \Pi(\mathcal{M}_1, \mathcal{M}_2) \) can be represented in the form of Eq. (2.33) with \( \gamma_{i,j} \) satisfying Eqs. (2.36) and (2.37), and the corresponding \( \mathbf{W} \) is given by

\[
\mathbf{W} = \int \pi(x; \mathcal{M}_1) \cdot \pi(y; \mathcal{M}_2)^T d\gamma(x,y), \quad (2.38)
\]
where $\pi(x; \cdot)$ (a column vector) denotes the posterior mixture component probabilities at point $x$ inferred from a given GMM.

We now have the following theorem.

**Theorem 3.** For any $\gamma \in \Pi(M_1, M_2)$, let $W$ be defined by Eq. (2.38), then there exist $\gamma_{i,j}$, $i = 1, \ldots, M_1$ and $j = 1, \ldots, M_2$ satisfying constraints in Eq. (2.36) and (2.37) such that $\gamma = \sum_{i=1}^{M_1} \sum_{j=1}^{M_2} w_{i,j} \gamma_{i,j}$.

**Proof.** For the ease of notation, we assume $p = 2$. The proof also applies to any $0 < p \leq 2$ under trivial modification (implied by the Hölder inequality). We let

$$
\hat{\Pi}(M_1, M_2) \overset{\text{def}}{=} \left\{ \gamma \overset{\text{def}}{=} \sum_{i=1}^{M_1} \sum_{j=1}^{M_2} w_{i,j} \gamma_{i,j} \right\},
$$

$$
\gamma_{i,j} \in \Pi(\tilde{\phi}_{1,i,j}, \tilde{\phi}_{2,i,j}), \{\tilde{\phi}_{1,i,j}, \tilde{\phi}_{2,i,j}\} \in \Gamma(W|\mathcal{M}_1, \mathcal{M}_2),
$$

$$
W \in \Pi(\pi_1, \pi_2) \right\}.
$$

From the definitions, one can verify that $\hat{\Pi}(M_1, M_2) \subseteq \Pi(M_1, M_2)$. Therefore, optimizing transportation cost over $\gamma \in \hat{\Pi}(M_1, M_2)$ gives a tighter upper bound of $W(M_1, M_2)$ than $\widetilde{W}(M_1, M_2)$. Moreover, Theorem 3 is proved if we have

$$
\hat{\Pi}(M_1, M_2) = \Pi(M_1, M_2).
$$

To prove this, we only need to show that for any $\gamma \in \Pi(M_1, M_2)$, there exist $w_{i,j} \in \Pi(\pi_1, \pi_2)$, $\{\tilde{\phi}_{1,i,j}, \tilde{\phi}_{2,i,j}\} \in \Gamma(\{w_{i,j}\}|\mathcal{M}_1, \mathcal{M}_2)$ and $\gamma_{i,j} \in \Pi(\tilde{\phi}_{1,i,j}, \tilde{\phi}_{2,i,j})$ with $i = 1, \ldots, M_1$ and $j = 1, \ldots, M_2$ such that

$$
\gamma = \sum_{z_1=1}^{M_1} \sum_{z_2=1}^{M_2} w_{z_1,z_2} \gamma_{z_1,z_2}.
$$

The constructive proof goes in two steps. First, given any random variables $(x_1, x_2) \sim \gamma \in \Pi(M_1, M_2)$, we can induce component membership random variables $(z_1, z_2)$ by

$$
p(z_1, z_2) = \int_{\mathbb{R}^d \times \mathbb{R}^d} p(z_1, z_2|x_1, x_2) d\gamma(x_1, x_2) \overset{\text{def}}{=} w_{z_1,z_2}, \quad (2.39)
$$
where the condition probability is defined multiplicatively by

\[ p(z_1, z_2|x_1, x_2) \overset{\text{def}}{=} \frac{\pi_{1,z_1} \phi_{1,z_1}(x_1)}{f_1(x_1)} \cdot \frac{\pi_{2,z_2} \phi_{2,z_2}(x_2)}{f_2(x_2)}. \]  

(2.40)

One can verify that \( \{w_{i,j}\} \in \Pi(\pi_1, \pi_2) \) by the definition of Eq. (2.39): for any \( i = 1, \ldots, M_1 \)

\[
\sum_{j=1}^{M_2} w_{i,j} = \int_{\mathbb{R}^d \times \mathbb{R}^d} \sum_{j=1}^{M_2} p(z_1 = i, z_2 = j|x_1, x_2) d\gamma(x_1, x_2) \\
\overset{\text{by the Bayes rule}}{=} \int_{\mathbb{R}^d \times \mathbb{R}^d} \frac{\pi_{1,i} \phi_{1,i}(x_1)}{f_1(x_1)} d\gamma(x_1, x_2) \\
\overset{\text{integrate out } x_2, \text{ since } \gamma \in \Pi(M_1, M_2)}{=} \pi_{1,i}.
\]

Likewise, \( \sum_{i=1}^{M_1} w_{i,j} = \pi_{2,j} \) for any \( j = 1, \ldots, M_2 \). It is obvious that \( W \) defined by Eqs. (2.39) and (2.40) is the same as defined by Eq. (2.38).

Second, consider the conditional measure

\[ \gamma(x_1, x_2|z_1, z_2) = \frac{p(z_1, z_2|x_1, x_2) \gamma(x_1, x_2)}{w_{z_1,z_2}} \]

(by the Bayes rule), its marginals are

\[
d\gamma(x_1|z_1, z_2) = \frac{1}{w_{z_1,z_2}} \int_{x_2 \in \mathbb{R}^d} p(z_1, z_2|x_1, x_2) d\gamma(x_1, x_2),
\]

\[
d\gamma(x_2|z_1, z_2) = \frac{1}{w_{z_1,z_2}} \int_{x_1 \in \mathbb{R}^d} p(z_1, z_2|x_1, x_2) d\gamma(x_1, x_2).
\]

By definition, we know \( \gamma(x_1, x_2|z_1, z_2) \in \Pi(\gamma(x_1|z_1, z_2), \gamma(x_2|z_1, z_2)) \). One can validate that \( \{\gamma(x_1|z_1, z_2), \gamma(x_2|z_1, z_2)\} \in \Gamma(\{w_{i,j}\}|M_1, M_2) \): for \( z_1 = 1, \ldots, M_1 \) and
\[ z_2 = 1, \ldots, M_2, \]

\[
\sum_{z_2=1}^{M_2} \frac{w_{z_1, z_2}}{\pi_{1, z_1}} d\gamma(x_1|z_1, z_2) = \int_{x_2 \in \mathbb{R}^d} \sum_{z_2=1}^{M_2} \frac{\phi_{1, z_1}(x_1)}{f_1(x_1)} \cdot \frac{\pi_{2, z_2} \phi_{2, z_2}(x_2)}{f_2(x_2)} d\gamma(x_1, x_2) = \int_{x_2 \in \mathbb{R}^d} \frac{\phi_{1, z_1}(x_1)}{f_1(x_1)} d\gamma(x_1, x_2) = \phi_{1, z_1}(x_1) dx_1.
\]

Likewise, we can show

\[
\sum_{z_1=1}^{M_1} \frac{w_{z_1, z_2}}{\pi_{2, z_2}} d\gamma(x_2|z_1, z_2) = \phi_{2, z_2}(x_2) dx_2.
\]

Let \( \tilde{\phi}_{i, j} \) be the p.d.f. of \( \gamma(x_1|z_1 = i, z_2 = j) \) and \( \gamma_{i, j} \) \( \overset{\text{def}}{=} \gamma(x_1, x_2|z_1 = i, z_2 = j) \), we see that \( \gamma \in \Pi(M_1, M_2) \). Therefore, \( \Pi(M_1, M_2) \subseteq \Pi(M_1, M_2) \). Combined with the fact that \( \Pi(M_1, M_2) \subseteq \Pi(M_1, M_2) \), the proof is complete. \( \square \)

**Remark 5.** The optimal coupling in Eq. (2.1) can be factored as a finite mixture model with \( M_1 \cdot M_2 \) components, whose proportion vector is \( W^* \). \( W^* \) is taken from the minimizer \( \gamma^* = \sum_{i,j} w_{i,j}^* \gamma_{i,j}^* \in \hat{\Pi}(M_1, M_2) \) of the following problem

\[
\inf_{\gamma \in \Pi(M_1, M_2)} \int_{\mathbb{R}^d \times \mathbb{R}^d} \|x - y\|^2 d\gamma(x, y). \tag{2.41}
\]

Suppose the Wasserstein distance between two GMMs \( M_1 \) and \( M_2 \) are pre-solved such that the inference for their optimal coupling \( \gamma^* \) (referring to Definition 1) is at hand. We denote the new state registration matrix induced from \( \gamma^* \) by Eq. (2.38) as \( W^* \).

A Monte Carlo method to estimate \( W^* \) is hereby given. Two sets \( \{x_1, \ldots, x_n\} \) and \( \{y_1, \ldots, y_n\} \) of equal size i.i.d. samples are generated from \( M_1 \) and \( M_2 \) re-
spectively. The $W^*$ is then empirically estimated by

$$\tilde{W}_n^* \overset{\text{def}}{=} \left[ \pi(x_1; M_1), \ldots, \pi(x_n; M_1) \right] \cdot \Pi_n \cdot \left[ \pi(y_1; M_2), \ldots, \pi(y_n; M_2) \right]^T$$

where $\Pi_n \in \mathbb{R}^{n \times n}$ is the $p$-th optimal coupling solved for the two samples (essentially a permutation matrix). We use the Sinkhorn algorithm (Discussed in Sec. 1.1.3) to approximately solve the optimal coupling [26]. $\tilde{W}_n^*$ converges to $W^*$ with probability 1, as $n \to \infty$. Consequently, IAW is defined similarly as MAW in Eq. (2.19) but with a different $W$ computed from Eq. (2.42).

**Remark 6 (Convergence Rate).** The estimation of $W^*$ follows the mixture proportion estimation setting [14, 79], whose rate of convergence is $O \left( \sqrt{\frac{V_{\tilde{H}} \log n}{n}} \right)$. Here $V_{\tilde{H}} = V_{\tilde{H}}(d, M_1, M_2)$ is the VC dimension of the geometric class induced by the family $\tilde{H}(M_1, M_2)$ (Check the proof of Theorem 3 and [5] for related definitions).

### 2.5 Experiments and Results

We conduct experiments to quantitatively evaluate the proposed MAW and IAW. In particular, regarding $\tilde{R}_p$, $\tilde{W}_p$, and $D_p$, we set $p = 1$. Our comparison baseline is the KL-based distance [43] since it is the most widely used one (e.g., [83], [25]).

Roughly speaking, this is the KL rate for HMM sequences computed by Monte Carlo simulations. In Section 2.5.1, we use synthetic data to evaluate the sensitivity of MAW and IAW to the perturbation of $\mu$, $\Sigma$, and $T$. Similar synthetic experiments have been done in related work (e.g. [25]). In Section 2.5.2, we present results of more extensive experiments to confirm the robustness of the findings in Section 2.5.1 when HMMs are of different numbers of states, dimensions and levels of difficulty to differentiate. We also compare MAW and IAW against the variational approximation of KL between GMMs proposed by [36] (abbreviated as KL_var_gmm) and the variational approximation of KL between HMMs proposed by [37]² (abbreviated as

---

²Since the KL_var_hmm proposed in [37] are originally defined for a more generalized version of HMM which also models the length of associated sequences (while our focus is on traditional
KL_var_hmm) in both Section 2.5.1 and Section 2.5.2. In Section 2.5.3, we compare MAW and IAW with KL using the Mocap data under both retrieval and classification settings. In Section 2.5.4, we compare MAW and IAW with KL using TIMIT speech data under the settings of t-SNE visualization and $k$ nearest neighbor classification. Note that for KL and IAW, for both of which the sampling size has to be determined, we make sure the sample size is large enough such that the value of distance value has converged.

### 2.5.1 Sensitivity to the Perturbation of HMM Parameters.

Three sets of experiments are conducted to evaluate MAW and IAW’s sensitivity to the perturbation of HMM parameters $\{\mu_j\}_{j=1}^M$, $\{\Sigma_j\}_{j=1}^M$, and $T$ respectively. In each set of experiments, we have five pre-defined 2-state HMM models $\{\Lambda_j (\{\mu_{i,j}\}_{j=1}^2, \{\Sigma_{i,j}\}_{j=1}^2, T_i)\}_{i=1}^5$, among which the only difference is GMM means $\{\mu_{i,1}, \mu_{i,2}\}$, covariances $\{\Sigma_{i,1}, \Sigma_{i,2}\}$, or transition matrices $T_i$. For example, in the first experiment, we perturb $\{\mu_{i,1}, \mu_{i,2}\}$ by setting the parameters $\{\mu_{i,1}, \mu_{i,2}\}$ to be $8 \leq 5 \pm \Delta \mu \leq 9$ respectively. $\{\Sigma_{i,1}, \Sigma_{i,2}\}$ for $i = 1, ..., 5$ are the same: $\left( \begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array} \right)$, $\left( \begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array} \right)$. And the transition matrices are also the same: $\left( \begin{array}{cc} 0.8 & 0.2 \\ 0.2 & 0.8 \end{array} \right)$. $\Delta \mu$ here is a parameter to control the difference between the 5 models. The smaller the value, the more similar are the 5 models and the more challenging will retrieval be. We choose $\Delta \mu$ to be 0.2, 0.4, 0.6, and compare KL, KL_var_gmm, KL_var_hmm, MAW and IAW under each setting. Please refer to Table 2.1 for detailed experiment setup for the HMMs, we make certain simplifications on the computation of KL_var_hmm, which are detailed in Appendix A.6.
other two experiments. For each of the five models, 10 sequences of dimension 2 and of length 100 are generated. The models estimated from the 10 sequences by the well-known Baum-Welch algorithm become instances that belong to one class. To summarize, we have 50 estimated models in total which belong to 5 classes. Then we use every model as a query to retrieve other models using each type of the distances. A retrieved model is considered a match if it is in the same class as the query model (this applies to all the retrieval experiments in the sequel). The precision recall plot for the retrieval is shown in Fig. 2.4.
Table 2.1. Summary of the parameters setup for parameter perturbation experiments. \textit{rand}(2) here means random matrix of dimension 2 by 2. \textit{Dirichlet}(\vec{x}) here means generating samples from Dirichlet distribution with parameter \vec{x}.

<table>
<thead>
<tr>
<th>Exp. index</th>
<th>deviation step</th>
<th>( \tilde{\mu} )</th>
<th>( \tilde{\Sigma} )</th>
<th>( \mathbf{T} )</th>
</tr>
</thead>
</table>
| (a) | \( \Delta \mu = 0.2 \)
  , 0.4, 0.6 |
  \( \left\{ \begin{pmatrix} 2 + i\Delta \mu \\ 2 + i\Delta \mu \end{pmatrix}, \begin{pmatrix} 5 + i\Delta \mu \\ 5 + i\Delta \mu \end{pmatrix} \right\}_{i = 1, 2, 3, 4, 5} \) |
  \( \left\{ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right\} \) |
  \( \begin{pmatrix} 0.8 & 0.2 \\ 0.2 & 0.8 \end{pmatrix} \) |
| (b) | \( \Delta \sigma = 0.2 \)
  , 0.4, 0.6 |
  \( \left\{ \begin{pmatrix} 2 \\ 2 \end{pmatrix}, \begin{pmatrix} 5 \\ 5 \end{pmatrix} \right\} \) |
  \( \left\{ \begin{pmatrix} 0.2 \cdot \exp(i\Delta \sigma \cdot \mathbf{S}), \\ 0.2 \cdot \exp(i\Delta \sigma \cdot \mathbf{S}) \right\}_{i = 1, 2, 3, 4, 5}, \right\} |
  \( \mathbf{S} = \text{rand}(2) \) |
  \( \begin{pmatrix} 0.8 & 0.2 \\ 0.2 & 0.8 \end{pmatrix} \) |
| (c) | \( \Delta t = 0.2 \)
  , 0.4, 0.6 |
  \( \left\{ \begin{pmatrix} 2 \\ 2 \end{pmatrix}, \begin{pmatrix} 5 \\ 5 \end{pmatrix} \right\} \) |
  \( \left\{ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right\} \) |
  \( \left\{ \Delta t \cdot \mathbf{S} + (1 - \Delta t) \cdot \mathbf{T}_i \right\} |
  \( \mathbf{T}_i[j, :] \sim \text{Dirichlet}(10 \cdot \mathbf{S}[j, :]) \) |
  \( \left\{ \begin{pmatrix} 0.8 & 0.2 \\ 0.2 & 0.8 \end{pmatrix} \right\}_{i = 1, 2, 3, 4, 5}, \mathbf{S} = \begin{pmatrix} 0.8 & 0.2 \\ 0.2 & 0.8 \end{pmatrix} \) |
Figure 2.4. Precision-recall plot for the study to compare KL, KL_var_gmm, KL_var_hmm, MAW and IAW’s sensitivity to the perturbation of HMM’s parameters. (best viewed in color)
As shown by sub-figure (a) of Fig. 2.4, IAW performs better than KL, and KL performs better than MAW in the task of differentiating HMMs under the perturbation of \( \{\mu_j\}_{j=1}^M \). Sub-figure (c) of Fig. 2.4 shows that MAW and IAW perform better than KL to differentiate the perturbation of \( \mathbf{T} \). From Sub-figure (b), we see that for the task of differentiating perturbation of \( \{\Sigma_j\}_{j=1}^M \), KL performs better than IAW, and both IAW and KL perform much better than MAW. But for the less challenging cases, IAW has comparable performance with KL. As KL$_\text{var}_\text{gmm}$ is an approximation of KL between GMMs, when applied to HMMs, the comparison of models is restricted to the marginal GMMs, while the transition matrices are ignored. As a result, we only test KL$_\text{var}_\text{gmm}$ in the experiments with perturbation of \( \mu \) and \( \Sigma \) but not \( \mathbf{T} \) (same for Section 2.5.2). Under both settings, as shown by sub-figure (a) and (b), KL$_\text{var}_\text{gmm}$ performs slightly better than KL. This is reasonable since the settings in this subsection are purely hand crafted and all Gaussian components overlap with each other. In the next subsection, where the models are all randomly generated, the performance of KL$_\text{var}_\text{gmm}$ is usually inferior to KL. As for KL$_\text{var}_\text{hmm}$, its performance is in between those of KL and KL$_\text{var}_\text{gmm}$ under both the perturbation of \( \{\mu_j\}_{j=1}^M \) and perturbation of \( \{\Sigma_j\}_{j=1}^M \); and is similar with KL's under perturbation of \( \mathbf{T} \).

**Table 2.2.** Synthetic data per distance computation time comparison. (KL sample size: 2000, IAW sample size: 500)

<table>
<thead>
<tr>
<th></th>
<th>KL</th>
<th>MAW</th>
<th>IAW</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
<td>5.8ms</td>
<td>3.8ms</td>
<td>57.9ms</td>
</tr>
</tbody>
</table>
more retrieval experiments similar to Section 2.5.1. These experiments are under different settings of 1) number of states, 2) dimension, 3) level of differentiation between HMM classes/categories.

The HMMs used for retrieval are still drawn from a two-stage scheme similar to that described in Section 2.5.1. In the first stage, however, we design procedures to automatically generate seed HMMs instead of manually setting them as in Section 2.5.1. In the second stage, for each seed HMM, we generate $m$ sequences $o_{1:T}$ of length $T$ and estimate an HMM for each sequence. Those estimated HMMs will deviate somewhat from the original seed HMMs, and the amount of deviation can be controlled by $T$ (roughly speaking, the larger $T$ is, the less the deviation, and the easier the retrieval task). Again, each seed HMM is treated as one class. Specifically, the models estimated from each of the $m$ sequences generated by this seed HMM are instances in one class.

The difficulty of retrieval is determined by both the difference between seed HMMs and the variation among the estimated HMMs within one class. The two stage scheme to generate HMMs allows us to control the process of generating a collection of HMMs for retrieval and to facilitate experiments in a larger space of hyper-parameters. For each collection of HMMs generated, we conduct 1-nearest neighbor retrieval as we have done in Section 2.5.1 to test the differentiation ability of KL, MAW and IAW.

The procedure for generating seed HMMs for $\mu$ perturbation experiments is specified in Algorithm 3. Similar procedures for generating seed HMMs with $\Sigma$ perturbation and transition matrix perturbation are presented in Appendix. A.2. Briefly speaking, 1) for $\Sigma$ perturbation, a perturbed $\Sigma$ is generated by linear combination of a given $\Sigma_{base}$ and a random matrix $\Sigma_{pert}$ drawn from a Wishard distribution; 2) for transition matrix perturbation, a perturbed transition matrix is generated by linear combination of a given $T_{base}$ and a random matrix $P$, each row of which is drawn from a Dirichlet distribution. In both cases, the magnitude of perturbation can be controlled by the coefficients in the linear combinations.

We explored a range of hyper-parameters for the HMMs. In particular, we let the
Algorithm 3 Seed HMMs generation for \( \mu \) perturbation experiment.

**Input:** State number: \( M \), dimension: \( D \), transition matrix \( T \in \mathbb{R}^{M \times M} \), number of seed HMMs: \( S \), perturbation magnitude control variable: \( \gamma \)

**Output:** \( \{ \Lambda_i \left( T, M \left( \{ \mu_{j,i} \}_{j=1}^{M} \right), \Sigma \right) \}_{i=1}^{S} \)

1. Draw \( M \) seed \( \mu \)s, i.e. \( \{ \mu_1, ..., \mu_M \} \), from \( \mathcal{N}(\mathbf{0}, \mathbf{I}) \).
2. For each seed \( \mu_i \) \( i \in 1, ..., M \), we sample \( S \) \( \mu \)s, i.e. \( \{ \mu_{i,1}, \mu_{i,2}, ..., \mu_{i,S} \} \) from \( \mathcal{N}(\mu_i, \gamma \cdot \mathbf{I}) \) as the \( i \)-th state of each HMM’s \( \mu \) parameters. Where \( \gamma \) is a knob to control the magnitude of the perturbation.
3. Set every seed HMM the same \( \Sigma = \{ \Sigma_1, ..., \Sigma_M \}, \forall i \in \{1, ..., M\}, \Sigma_i \) is a \( D \times D \) matrix \( \beta_i \times \mathbf{I} \), where \( \beta_i \sim \mathcal{N}(1, 0.1) \).
4. Set every seed HMM the same transition matrix \( T \).

Number of states be 3, 5, or 8, the dimension be 3, 5, or 8, and set the extent of differentiation between HMMs to 4 levels. Any combination of choices is experimented with, resulting in performance evaluations for \( 3 \times 3 \times 4 = 36 \) data sets. The number of seed HMMs in each data set, denoted by \( S \), is 6; and the number of sequences (and hence estimated HMMs) generated from each seed HMM is set to 6.

The experimental results are displayed in Figure 2.5. For each case of perturbation, results are shown in a panel of 3 by 3 sub-figures. The dimension of HMMs is in increasing order from top rows to bottom rows, and the number of states is in increasing order from left columns to right columns. Within each sub-figure, the dimension and the number of states for HMMs are fixed, while the horizontal axis is for the value of \( \gamma \), the so-called “scale”, by which we control the extent of differentiation between the HMMs. The vertical axis is for value of Area Under the receiver operating characteristic Curve (AUC), which is a widely used evaluation metric for retrieval. The higher the value, the better the retrieval performance. Since we use every generated HMM as a query in each data set, we have obtained an error bar computed from all the queries under any setting, which is also shown in the sub-figures.
Figure 2.5. Nearest neighbor retrieval experiment Scale v.s. AUCs. under (a) the perturbation of $\mu$, (b) the perturbation of $\Sigma$ and (c) the perturbation of transition matrix. (best viewed in color)
Figure 2.6. Analysis of how the accuracy of the estimated HMMs affects results of the NN Retrieval experiment (perturbation of $\mu$). $T = M \times d \times \tau$, where $M = 3$, $d = 3$. The case $\tau = \infty$ means the true parameters of the seed HMMs are used (instead of estimated parameters).

In general, we can see from Figure 2.5 that the conclusion we made in Section 2.5 holds across different numbers of states, dimensions, and levels of differentiation between HMMs. More specifically,

1. For $\mu$ perturbation experiments, IAW and KL perform similarly, both better than MAW across different numbers of states and dimensions.

2. For $\Sigma$ perturbation experiments, KL performs better than IAW and IAW performs better than MAW.

3. And for transition matrix perturbation experiments, MAW and IAW perform similarly and they both perform much better than KL. The advantage of IAW and MAW becomes more significant when the number of states or the dimension increases.

However, different from Section 2.5.1, KL_var_gmm and KL_var_hmm perform worse than KL in most settings under the perturbation of $\mu$ and the perturbation of $\Sigma$, as shown by sub-figure (a) and (b) in Figure 2.5. Under the perturbation of $T$, the performance of KL_var_hmm is similar with or in some cases (when the number of state is 8) better than KL, but still not as good as MAW or IAW.

**Discussion on the effect of $T$.** As discussed above, in our two stage sampling scheme of HMMs, we use the sampling length $T$ to control how much a re-estimated HMMs deviates from the seed HMM. In our experiments, we empirically set $T$ to $\tau \times M \times d$, where $\tau$ is the length factor (we set $\tau = 10$ in experiments with results
shown in Fig. 2.5), \( M \) is the number of states and \( d \) is the dimension. As a larger value of \( M \) or \( d \) makes the estimation of HMM harder, we expect that a larger \( T \) is needed to achieve the same level of accuracy in estimation. If the estimation is not accurate, the estimated HMMs tend to deviate more from the seed HMMs and hence be harder to discern, making retrieval more difficult. We also explore values of \( \tau \) other than 10 to see how \( T \) affects the retrieval experiments. Fig. 2.6 shows the retrieval results across \( \tau = \{2, 5, 10, 30, 100, 500, \infty\} \) for the experiment with perturbation on \( \mu \). Specifically, 3-state and 3-dimensional HMMs are used, the same setup as the experiment with results shown in the top-left sub-figure (boxed with dashed red lines) in Fig. 2.5 (a). Corresponding to the experimental setup of each sub-figure in Fig. 2.5, we have examined the effect of \( T \) on retrieval. The complete results are shown in Appendix. A.5 (see Fig. A6, Fig. A7 and Fig. A8). We observe that the relative rank of performance for KL, KL\_var\_gmm, KL\_var\_hmm, MAW and IAW generally stays the same at different \( \tau \)’s.

### 2.5.3 Real Data: Motion Time Series

![CMU motion capture data visualization](image)

**Figure 2.7.** Visualization of CMU motion capture data. Top: Jump. Bottom: Walk.

In this section, we use the Carnegie Mellon Motion Capture Dataset (Mocap) to evaluate MAW and IAW and make comparison with KL-based approach, which [57] takes. To improve the stability of evaluation, we only select motion categories 1) whose sequences contain only 1 motion, and 2) which contain more than 20 sequences. In total, there are 7 motion categories, i.e. *Alaskan vacation, Jump, Story, clean, salsa dance, walk, and walk on uneven terrain* that meet this criterion and they contain
a total of 337 motion sequences. Since the sequence data is of high dimension (62), following the practice of [57], we split the 62 dimension data to 6 joint-groups. And we conduct both motion retrieval based on every joint-group separately and motion classification using Adaboost based on all the joint-groups together.

2.5.3.1 Motion Retrieval

![Precision Recall Plot for Motion Retrieval](image)

**Figure 2.8.** Precision Recall Plot for Motion Retrieval. The plot for 6 joint-groups, i.e. root$_{12}$, head\_neck\_thorax$_{12}$, rbody$_{12}$, lbody$_{12}$, rleg$_6$, lleg$_6$, are displayed separately.

For each motion time series, we first estimate a 3-state HMM for each joint-group. Given any joint-group, the model estimated from every sequence is used as a query to retrieve models estimated from other sequences using KL, MAW and IAW distances respectively. The parameter $\alpha$ for MAW and IAW is chosen such that the 1-nearest neighbor classification accuracy on a small set-aside evaluation set is maximized. The set-aside evaluation set contains 25% of all motion time series. The remaining 75% motion time series are used for the nearest neighbor

---

$^3$root$_{12}$, head\_neck\_thorax$_{12}$, rbody$_{12}$, lbody$_{12}$, rleg$_6$, lleg$_6$. (The subscript number denotes the dimension of the group)
retrieval. We do not split them further into training and testing sets in the retrieval experiment as our focus is on evaluating the distance. How to choose $\alpha$ is illustrated in Appendix. A.1 (Fig. A.1 shows how to choose $\alpha$ for rleg and the values of all chosen $\alpha$’s are in Table A1). The precision-recall plots for the motion retrieval are provided in Fig. 2.8. Any point on the precision-recall curve is an average over all the motion sequences, each served as query once. We can see that MAW and IAW yield consistently better retrieval results on all the joint-groups. For comparison, we also present results when $\alpha = 0$ and $\alpha = 1$ for MAW and IAW based retrieval in Appendix. A.1(Fig. A2 and Fig. A3). We observe that all the selected $\alpha$’s are small and the performance enhancement from using a non-zero value of $\alpha$ is marginal (similar results for experiments in Section. 2.5.4). The reasons could be 1) the differentiation information in the geometry of the steady state distributions of the estimated HMMs is much more dominant than that in the transition matrices; 2) the imprecision in the estimation of the transition matrices further reduces the effect of the distance measure between those matrices.

2.5.3.2 Retrieval with Missing Dimensions

One important advantage of MAW or IAW over KL is that it can handle HMMs with degenerate support. Imagine a scenario that a proportion of collected time series may have missing dimensions for various reasons such as malfunctioning sensors. The missing variables can differ among the collected series, so we cannot simply remove certain variables across the board. This situation can arise frequently in real world data collection when sensor monitoring expands over a long period in a highly dynamic environment. If we are restricted to use complete data, it is possible that only a small fraction of instances qualify.

For simplicity of experiments, we still conduct retrieval experiments using the motion time series and simulate the scenario when 1, 2 or 3 sensors malfunctioned so we failed to record the data for the corresponding dimensions. Specifically, each motion sequence is used as query once. For any motion sequence, we randomly select $k = 1, 2, \text{ or } 3$ dimensions to assume missing and set the values of those
dimensions to 0. For clarity of the results, in our experimental setup, the same $k$ is used for all the sequences in each reported case. However, the variables assumed missing are randomly chosen per sequence. An HMM is estimated for each sequence based on the reduced dimensions. We then map this HMM to the full dimension by setting the mean of the missing variables to 0 and the variances and covariances involving any missing variable to 0. Although the HMMs for both the query and the sequences to be retrieved (referred to as retrieval sequences below) are degenerate in the full dimension, the MAW distance is well defined and can be computed without numerical issues. Despite the fact that it is straightforward to marginalize out a missing dimension in an HMM trained on a larger set of dimensions, we cannot simply compare the HMM of a query and the HMM of a retrieval sequence based on the shared non-missing dimensions. Given any query, the set of shared non-missing dimensions varies with the retrieval sequence. As a result, relying only on commonly available dimensions will lead to model comparison in different spaces, which is not acceptable for retrieval. On the other hand, the full dimensional space provides a common ground for comparison. As described in Section 2.5.3, we retrieve sequences based on the MAW distance between the models estimated from the sequences. The precision recall curves are plotted in Fig. 2.9. The performance drops as the number of missing dimensions increases. However, the degradation is small in comparison with the difference between MAW and KL. If we compare Fig. 2.9 and 2.8, we see that even when every query has 3 missing dimensions, the most part of the precision-recall curve is clearly better than that achieved by KL in every sub-figure (each corresponding to one joint-group). Similar experiments using the IAW distance have also been done with results shown in the Fig. 2.10. We can draw the same conclusion for IAW.

2.5.3.3 Motion Classification

First, we split the 337 motion sequences randomly into two sets, roughly half for training and half for testing. In the training phase, for each of the 7 motion categories, we train one HMM for every joint-group. For each sequence, we also estimate one HMM for every joint-group. We then compute its distances (either KL, MAW
Figure 2.9. Precision Recall Plot for MAW based Motion Retrieval under data missing for certain dimension(s) setting. The plot for 6 joint-groups, i.e. root, head\_neck\_thorax, rbody, lbody, rleg, lleg, are displayed separately.

or IAW) to all the 7 HMMs (one for each motion category) on the same joint-group data. We repeat this for every joint-group. These distance values are treated as features. The dimension of the feature vector of one motion sequence is thus the number of joint-groups multiplied by 7. For MAW and IAW, we avoid the selection of $\alpha$ by treating distances $R_p(M_1, M_2; W)$ and $D_p(T_1, T_2; W)$ as separate input features to the classifier, thus doubling the input dimension for MAW and IAW. Finally, we use Adaboost with depth-two decision trees to obtain a classification accuracy on the test data. We plot the classification accuracy with respect to the iteration number in Adaboost in Fig. 2.11 (a). According to the mapping between dimension indexes and sensor locations on the body, the variables of Mocap data can also be split into 27 more refined joint-groups. Under the 27 joint-group split scheme, we run the same classification experiments again and summarize the results in Fig. 2.11 (b). Overall, the results show that under both the 6 joint-group scheme and the 27 joint-group scheme, MAW (92.90% for 6 joint scheme and 94.67% for 27 joint scheme) and IAW (93.49% for 6 joint scheme and 98.22% for 27 joint scheme) achieve considerably
better classification accuracy than KL (90.53% for 6 joint scheme and 95.86% for 27 joint scheme). The confusion matrices are also drawn in the Appendix. A.3. To demonstrate the positive effect on classification obtained by the distance term $D_p(T_1, T_2; W)$ between transition matrices, we conduct the same MAW and IAW based Adaboost classification experiments without using $D_p(T_1, T_2; W)$. The classification results become significantly worse, an evidence that $D_p(T_1, T_2; W)$ helps distinguish those motion sequence HMMs. Please see Appendix. A.4 for the detailed results.

The computation time of Mocap data with 6 joint-groups is 21ms by MAW, 158ms by IAW (1000 samples), and 8ms by KL (1000 samples). And that of Mocap data with 27 joint-groups is 17ms by MAW, 160ms by IAW (1000 samples), and 7ms by KL (1000 samples). Again, the MAW and IAW are implemented in MATLAB, and KL-D is implemented in C.
Figure 2.11. Testing accuracies with respect to the iteration number in Adaboost (number of weak classifiers selected). (a) Motion Classification by Adaboost on 6 joints. (b) Motion Classification by Adaboost on 27 joints. The iteration number means the number of features incrementally acquired in Adaboost.

2.5.4 Real Data: TIMIT speech data

The TIMIT\(^4\) data set contains 6300 spoken utterances, each of which is segmented based on 61 phonemes. Following the standard regrouping of phoneme labels described in \([51]\), we select 48 phonemes from 61 phonemes for modeling and these 48 phonemes are merged into 39 phoneme categories. For the stability of HMM estimation, we (randomly) group phoneme segments within each phoneme category into a set of subgroups, each containing 20 phonemes. We also force all phoneme segments within each subgroup to either belong to training set or test set. We concatenate phonemes within each subgroup (Obviously, they have the same phoneme label) and treat it as a single instance for further visualization and classification task. We call such concatenated sequences 20-concat phoneme segments. In the training set, we have 2218 such 20-concat phoneme segments and in the test set, we have 795 such 20-concat phoneme segments. The histogram for the number of 20-concat phoneme segments in each phoneme category is shown in the Fig. 2.12.
Figure 2.12. Histogram of 20-concat phoneme segments

Table 2.3. Phoneme k-NN classification accuracy comparison.

<table>
<thead>
<tr>
<th>k</th>
<th>KL</th>
<th>MAW</th>
<th>IAW</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>α* = 0.13</td>
<td>α = 0</td>
<td>α = 1</td>
</tr>
<tr>
<td>1</td>
<td>0.492</td>
<td>0.525</td>
<td>0.521</td>
</tr>
<tr>
<td>2</td>
<td>0.487</td>
<td>0.502</td>
<td>0.486</td>
</tr>
<tr>
<td>3</td>
<td>0.525</td>
<td>0.536</td>
<td>0.528</td>
</tr>
<tr>
<td>4</td>
<td>0.522</td>
<td>0.536</td>
<td>0.526</td>
</tr>
<tr>
<td>5</td>
<td>0.548</td>
<td>0.553</td>
<td>0.543</td>
</tr>
<tr>
<td>6</td>
<td>0.537</td>
<td>0.545</td>
<td>0.537</td>
</tr>
<tr>
<td>7</td>
<td>0.545</td>
<td>0.553</td>
<td>0.536</td>
</tr>
<tr>
<td>8</td>
<td>0.545</td>
<td>0.553</td>
<td>0.536</td>
</tr>
<tr>
<td>9</td>
<td>0.558</td>
<td>0.551</td>
<td>0.533</td>
</tr>
<tr>
<td>10</td>
<td>0.560</td>
<td>0.530</td>
<td>0.523</td>
</tr>
<tr>
<td>11</td>
<td>0.550</td>
<td>0.552</td>
<td>0.537</td>
</tr>
<tr>
<td>12</td>
<td>0.551</td>
<td>0.547</td>
<td>0.525</td>
</tr>
</tbody>
</table>

2.5.4.1 Phoneme k-NN classification.

We first conduct phoneme classification experiments on the TIMIT database using $k$-nearest neighbor based on MAW, IAW and KL distances. The split into training and testing data is typical in the literature (see [39]). As mentioned previously,
we use 20-concat phoneme segments as instances for classification. We preprocessed Mel-frequency cepstral coefficients (MFCC features) with sliding window size 4 ms and frame rate 2 ms (and Δ and ΔΔ of MFCC, i.e. the first order and second order difference of MFCC). Following [64], we estimate an HMM with 3 states for each segmented phoneme which is a sequence of 39-dimensional MFCC feature vectors (13 static coefficients, Δ, and ΔΔ). Following common practices in speech research [30], we force the Σ’s of HMM to be diagonal. For each HMM estimated from a 20-concat phoneme segment in the test set, we compute its KL, MAW and IAW distances to all the HMMs estimated from 20-concat phoneme segments in the training set. For MAW and IAW, α is selected such that 1-nearest neighbor accuracy is maximized on the training set. We set the sample size for KL to 5000 and sample size for IAW to 500. Then we perform k-nearest neighbor classification. The accuracies with respect to k are shown in Table 2.3. The best accuracy for KL, MAW or IAW respectively is underscored. IAW achieves significantly better accuracy than MAW and KL, while MAW and KL yield similar results. For comparison, we also show the results at α = 0 and α = 1 for MAW and IAW.
Figure 2.13. KL-based, MAW-based and IAW-based t-SNE visualization comparison.
Table 2.4. Quantitative comparison of t-SNE visualization based on KL, MAW and IAW. The best of each row (in terms of one measure) is underscored.

<table>
<thead>
<tr>
<th>Measure</th>
<th>KL</th>
<th>MAW</th>
<th>IAW</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-NN accuracy</td>
<td>0.522403</td>
<td>0.511118</td>
<td>0.535015</td>
</tr>
<tr>
<td>within category variance</td>
<td>0.299801</td>
<td>0.286023</td>
<td>0.231758</td>
</tr>
<tr>
<td>total variance</td>
<td>0.286023</td>
<td>0.286023</td>
<td>0.231758</td>
</tr>
</tbody>
</table>

2.5.4.2 Phoneme t-SNE visualization.

We also apply t-SNE visualization [87] to all the 20-concat phoneme segments (including both training set and test set). The t-SNE visualization method only relies on a pair-wise distance matrix $D$ for all the instances. We first use KL, MAW and IAW to compute $D_{KL}$, $D_{MAW}$, $D_{IAW}$ respectively, and then input each of them to t-SNE. The t-SNE visualization results are compared in Figure 2.13. The visualization by t-SNE based on MAW or IAM seems slightly better than that based on KL because the clusters appear more distinct and compact. Manual inspection is of course subjective. We compare the 2-D embeddings by t-SNE based on KL, MAW, IAW distances using two quantitative measures:

1. 1-Nearest Neighbor classification accuracy based on the embedded vectors. The higher the accuracy, the better the embedding is. (The same measure has been used in Sec. 3.2 of [88] to compare embeddings obtained by different visualization methods.)

2. Ratio between the weighted average (weights proportional to the number of data points in each category) of within category variance and the total variance. This measure reflects the relative compactness of the categories, which is invariant to scaling in the embedded space. The lower the ratio, the better the embedding is.

The comparison results are shown in Table 2.4.

From the table, we can see that, according to the ratio between the average within category variance and the overall variance, IAW yields the tightest embedding for points in the same category, while KL yields the least tight embedding. We want to emphasize that we use HMMs estimated from 20-concat phoneme segments
as visualization instances, whereas [87] uses raw MFCC features of each phoneme segment frame as a visualization instance (specifically, D is obtained from Euclidean distances between per frame features). Our visualization for the HMMs takes into account both the difference in the marginal distributions of the feature vectors and the difference in stochastic transition, whereas [87] (Fig. 7 in that paper) only captures the first kind of difference.

2.6 Discussions and Conclusions

Although we focus on HMM whose emission function is Gaussian in this paper, the same methodology extends readily to:

1. HMM whose emission function is GMM but not single Gaussian. (Each state with GMM emission function consists of k Gaussians can be split into k states. Our current method can be applied directly then.)

2. Other Hidden Markov Models with non-Gaussian state emission functions, provided that a distance between any two state conditional distributions can be computed. For instance, an HMM with discrete emission distributions can be handled by using the Wasserstein metric between discrete distributions.

In conclusion, we have developed the MAW and IAW distances between HMMs that are invariant to state permutation. These new distances are computationally efficient, especially MAW. Comparisons with the KL divergence have demonstrated stronger retrieval and classification performance and improved t-SNE visualization. In the future, it is interesting to explore how to reasonably group HMMs into a number of clusters based on our proposed MAW and IAW. The HMM clustering has been studied under the context of KL divergence [25], and the clustering under Wasserstein distance has been studied for empirical distributions [94].
Automated Morphological Characterization of Placenta from Photos

3.1 Introduction

The placenta is a window into the events of a pregnancy and the health of the mother and baby [74]. Yet, a very small percentage of placentas around the world are ever examined by a pathologist. Even in developed countries like the U.S., placentas are examined and characterized by a pathologist only when it is considered necessary and resources are available. Full pathological examination is expensive and time consuming. Pathologists or pathologist assistants perform a macroscopic or gross examination and select sections for microscopic examination. After processing, they examine sections under a microscope and produce a written report (e.g., Fig. 3.1(b-c)) that contains various measurements (e.g., the weight, the disc diameter) and diag-

noses (e.g., completeness or retained placenta, cord insertion type, shape category, meconium, chorioamnionitis etc.). In some specialty centers, including Northwestern Memorial Hospital, the gross examination includes photography using a special equipment (Fig. 3.1(a)). These measurements and placental diagnoses can be useful for the short- and long-term clinical care of the mother and baby.

Automated placental assessment based on photographic imaging can potentially allow more placentas to be examined, reduce the number of normal placentas sent for full pathological examination, and provide more accurate and timely morphological and pathological measurements or analyses. Typical photographs of the placentas capture the umbilical cord inserting into the fetal side of the disc, as well as the maternal side appearance. Two example images of placentas can be found later in Fig. 3.1(d). This chapter focuses on a fully automated system for morphological characterization of placentas. Such systems will be the cornerstone for automated pathological analyses because segmentation of disc and cord, location of cord insertion point, and determination of fetal/maternal side are important first steps before further analyses can be done.

### 3.1.1 Related Work

Existing placenta imaging research can be roughly categorized into two types: those using microscopic images of sections of the placenta [85, 46] and those using macroscopic images of the placenta taken by cameras [92] or by MRI [6]. A comprehensive overview of both microscopic and macroscopic placenta pathology can be found in a book by [12]. Microscopic assessment is more established, but requires equipment and personnel to make slides and microscopes and microphotography to make images. The second category, in contrast, only requires an ordinary camera or even a camera phone, and thus has much more potential to be widely adopted. To our knowledge, there has not been an automated approach to analyze placenta photographs. We believe such an approach has the potential to be adopted widely because today’s smartphones have high-quality cameras and highly capable CPU, GPU, and/or AI chips.
In this chapter, we present a two-stage pipeline (as illustrated in Fig. 3.2) for automated placental assessment and examination using photos. In the first stage, we take a transfer learning (TL) approach to tackle the associated tasks of morphological characterization rather than employing an independent model for each task. Transfer learning promises performance gain and robustness enhancement through representation sharing for closely related tasks [63] and has become popular in medical imaging applications in recent years [23]. [23] summarizes the use cases of transfer learning into three categories: “same domain, different tasks”, “different domains, same task” and “different domains, different tasks”. Our method is closest to the “same domain, different tasks” category but is not an exact match. More precisely, our method should fall into a category described as “similar/overlapped domains, different tasks” because the source and target domains have overlap but are not the same (see Sec. 3.3 for more detailed discussions). Specifically, we transfer the learned representation of the encoder from the segmentation task to the other two tasks, i.e. disc side classification and insertion point localization. Our network architecture design takes inspiration from the recent deep learning advances on classification [35], image segmentation [56, 75], and key-point localization [60]. In particular, the design of our segmentation module follows the practice of concatenating feature maps in encoder with feature maps in decoder, such as performed in the U-Net [75]; and the design of our insertion point module follows the practice of regressing a Gaussian heat map, rather than using the coordinate values, as the ground truth, which has been shown to be successful in human key-point/joint localization tasks [86, 12, 60, 67]. [86] first showed the importance of intermediate supervision to improving localization accuracy. We take their idea in our design by considering two heat map predictions in the final loss — one from the final feature layer and one from the intermediate feature layer. In the second stage, we employ independent models each tailored for an individual task for a few important placental assessment tasks including but not limited to detection of retained placenta (i.e., incomplete placenta), umbilical cord knot, meconium, abruption, chorioamnionitis, and hypercoiled cord; and categorization of umbilical cord insertion type.
3.1.2 Rationale for a Two-stage Pipeline

We chose to pursue a two-stage pipeline based on the following observations, both of which make it difficult to build an end-to-end model for all tasks:

- Almost all of our second-stage tasks only apply to either the fetal side or the maternal side of a placenta or only to the disc/cord/ruler region.

- A relatively small fraction of all images bear the abnormalities we attempt to detect for the tasks in the second stage. And the sets of images bearing different abnormalities often have little overlap.

The first observation makes it natural for the second-stage tasks to take in the segmentation and disc-side predictions from the first stage to narrow down the region of interest and eliminate irrelevant information. Also, this means the input feature space for these tasks is rather different from the first stage or other second-stage tasks, and it is difficult, if not impossible, to apply transfer learning here to let those tasks benefit from the representations learnt from other tasks. In contrast, tasks in the first stage are more closely related and have larger overlapped input feature space. The second observation makes it sometimes impractical to use the same training/testing set for all tasks. Each task may have its own training/testing set such that the model will not be dominated by negative cases (i.e., without abnormalities).

3.1.3 Contributions

We summarize the primary contributions as follows.

- We introduce a novel pipeline for comprehensive, automated placental assessment and examination using photos. The design of the pipeline, which has two stages, takes the relationship and the similarity of the tasks into consideration. Specifically, we use transfer learning to boost performance and robustness for closely related tasks with significant overlapped input space in the first stage. And we use separate models to address distinct tasks in the second stage by using the first-stage prediction to determine if an image is relevant (through side
classification) and to provide the region of interest (through segmentation). Our method is explainable by design and achieves highly promising results. We believe isolating the models for irrelevant tasks and enforcing strong priors on the information flow between sub-models are critical under a limited label and robustness-prioritized setting, which is typical for medical image analysis. The reason is that such isolation could significantly reduce the possibility of learning signals/correlations that do not hold true for the general distribution but just happen to be the case in our collected data based on our prior domain knowledge. Additionally, distinct sub-models in the second stage can be developed in parallel and can be upgraded without worrying that it will affect performance for other tasks.

- Our use of transfer learning for the first-stage tasks can be categorized into the “similar/overlapped domains, different tasks” type, which is novel and can be applicable to other medical image analysis problems.

- We curated a first-of-its-kind large-scale dataset with hand-labeled segmentation maps, umbilical cord insertion point location and diagnoses extracted from the associated pathology reports. This dataset enabled us to develop our computational pipeline addressing automated placental assessment and examination tasks. We believe the dataset will also be highly beneficial to future research on the placenta and adverse prenatal and postpartum outcomes.

3.2 Dataset

We collected a dataset consisting of 18,400 placenta photos taken at Northwestern Memorial Hospital, a large urban academic medical center as well as the associated pathology reports written in natural English language by the pathologist who originally examined the placenta, spanning the years of 2016 to 2018. These photos were taken by on-site pathologists and pathologist assistants using a camera installed on a fixed height arm against standardized blue background as illustrated in Fig. 3.1(a).
Figure 3.1. Data curation process. (a) Placenta photography equipment at Northwestern Memorial University. This equipment is used to collect high and consistent quality placenta photos for the curation of our dataset. (b-c) Extracting diagnoses and measurements from unidentified pathological report in p.d.f format. (d-f) Collecting pixel level segmentation map for cord, disc and ruler, insertion point location and whether a image captures fetal or maternal side placenta through our web-based labeling tool.

Figure 3.2. Schematic diagram of our proposed two-stage framework for automated placental assessment and examination using photos. Details of Stage I models and Stage II models will be described in Sec. 3.3 and Sec. 3.4 respectively.

Pathology classification is standardized, and the pathologists have perinatal training and expertise. From the 18,400 placenta photos (of about 9,000 placentas), 1,370 photos were selected, of which 665 are fetal-side images and 705 are maternal-side
images, to be hand labeled.\footnote{The numbers of fetal-side and maternal-side images are uneven because some of the collected images did not meet our image quality standard (e.g. disc occluded by irrelevant object such as scissors) and we had to discard them from the dataset. We plan to release our dataset in the future after substantial expansion.} Fig. 3.1 shows our data curation process. We developed a web-based tool (Fig. 3.1(e)) to collect the following data: i) the pixel-wise segmentation maps, ii) the side-type label as fetal side or maternal side, and iii) the cord insertion point (only for fetal side, visualized as a Gaussian heat map centered at the marked coordinate in (Fig. 3.1(f))) so that multiple trained labelers could annotate this dataset concurrently. We also extract diagnoses from the pathology reports (Fig. 3.1(b-c)). A complete list of diagnoses we extracted from the pathology reports are listed in Appendix A. For those placentas being diagnosed with being retained/incomplete the pixel-wise incomplete area was annotated by a highly-trained pathologist who is a research member (J.A.G.). For true knot in the cord, trained research members placed a bounding box around the knot with expert review as needed.

We divide the fully-labeled dataset into training and testing sets with the ratio of 0.8 : 0.2. Because the insertion point can only be observed from the fetal side, we only use the 665 fetal-side images for insertion point prediction, with the same training-testing ratio as aforementioned.

\subsection{3.3 Stage I: Morphological Characterization}

The proposed model for morphological characterization, as illustrated in Fig. 3.3, consists of an \textbf{Encoder} for feature pyramid extraction (blue), which is shared among all tasks, a fully convolutional \textbf{SegDecoder} for placenta image segmentation on both fetal- and maternal-side images (red), a \textbf{Classification Subnet} for fetal/maternal-side classification (purple), and a fully convolutional \textbf{IPDecoder} for insertion point localization.
Figure 3.3. The architecture of our model for morphological characterization: a multi-head convolutional neural network for placenta image segmentation, cord insertion point localization, and placenta disc side classification. “Up sample & Conv” is implemented by a transposed convolution layer. “Res conv blocks” are residual blocks with two convolutional layers with stride 2 and 1, respectively, and the same kernel size $3 \times 3$. “Score blocks” are convolutional layers with kernel size $1 \times 1$ and the number of output channel 1. The soft-max layers are omitted. We use dice loss, binary cross entropy (BCE) loss and mean square error (MSE) loss for the segmentation, classification, and insertion point localization, respectively.

3.3.1 Encoder as Feature Pyramid Extractor

The Encoder takes a placenta image $x$ (either the fetal side or the maternal side) as the input and outputs a pyramid of feature maps $\{f_1, f_2, f_3, f_4, f_5\}$ (represented as blue rectangles). Depending on the tasks, all or part of the feature maps are used by further task modules. Specifically, SegDecoder takes $\{f_1, f_2, f_3, f_4, f_5\}$ as input; Classification Subnet takes $\{f_5\}$ as input; and IPDecoder takes $\{f_3, f_4, f_5\}$ as input. The Conv-1 and Conv-2 blocks both consist of a Conv-BatchNorm-Relu layer. The difference, however, is that the Conv layer in Conv-1 block has stride 1, while the Conv layer in Conv-2 block has stride 2. The Res conv blocks are residual blocks with two convolutional layers with stride 2 and 1, respectively, and the same
kernel size $3 \times 3$, each of which spatially downsamples the input feature maps to half of its size and doubles the number of feature channels. The residual structure has been shown especially helpful for training deep architectures by [He_2016_CVPR].

### 3.3.2 SegDecoder for Segmentation

Our SegDecoder module consists of four expanding fully convolutional blocks, each of which takes the concatenation of a copy of the corresponding feature map $f_i, i \in \{1, 2, 3, 4\}$, and transposes a convoluted (up-scaling factor 2) output feature map of the last layer. Finally, we apply soft-max to predict the probability of pixel $(i, j)$ being of class $k$, denoted as $p(i, j, k)$. To overcome the problem of highly imbalanced number of pixels for different categories, we use dice loss [59] instead of the common cross entropy loss. Since we have four classes rather than two classes in [59], we adjust the dice loss to suit the 4-class scenario:

$$L_{\text{seg}} = 1 - \frac{\sum_{i,j} \sum_{k=0}^{3} p(i, j, k) \cdot g(i, j, k)}{\sum_{i,j} \sum_{k=0}^{3} (p^2(i, j, k) + g^2(i, j, k))},$$

where $i, j$ run over the row and column indexes of an image, respectively; $p(i, j, k)$ and $g(i, j, k)$ denote the predicted probability of the pixel at location $(i, j)$ and the 0/1 ground truth of that pixel belonging to class $k$, respectively.

### 3.3.3 Classification Subnet for Fetal/Maternal Side Classification

Because the fetal/maternal side can be inferred from the “disc” region of a placenta alone, we crop the full placenta image $x$ by a rectangle including the region of disc and resize the cropped image to $512 \times 512$ pixels as the input to the Encoder, which we denote as $x_c$. The cropping is based on the ground truth segmentation map during training and on the predicted segmentation map at inference. Our Classification Subnet consists of a Res conv block, two fully connected layers, and a soft-max layer. At the end, a binary cross entropy (BCE) loss is applied to supervise the network.
3.3.4 IPDecoder for Insertion Point Localization

Because the insertion point is always located within or adjacent to the “disc” region, we use cropped disc region image \( x_c \), by the same way as we perform cropping in Classification Subnet, as the input to the Encoder. Our IPDecoder is also fully convolutional and consists of two expanding fully convolutional blocks, the structure of which are the same as in the first two convolutional blocks in SegDecoder. The similarity of IPDecoder’s structure with SegDecoder’s helps us to ensure that the shared encoder representation could also be readily utilized here. Inspired by the success of intermediate supervision \([60]\), we predict the insertion point localization heat map after each expanding convolutional block by a convolutional layer with kernel size \( 1 \times 1 \) (denoted as “Score block” in Fig. 3.3) and use the MSE loss to measure the prediction error:

\[
L^\text{ip}_k = \sum_{i,j} ||h(i,j) - \hat{h}(i,j)||^2, \quad k \in \{1, 2\},
\]

where \( h(i,j) \) and \( \hat{h}(i,j) \) are the ground truth (Gaussian) heat map and the predicted heat map, respectively. And the final loss for insertion point is \( L^\text{ip} = L^\text{ip}_1 + L^\text{ip}_2 \).

During inference, the predicted insertion point location is determined by \( (i,j) = \arg \max_{i,j} \hat{h}(i,j) \).
Figure 3.4. The insertion type categorization and related automated measurements procedures consists of steps 1) to 5). (a), (b), and (c) illustrate the detailed procedure for steps 1), 2), and 3), respectively. (For (b) ruler scale extraction, Ms. Zhuomin Zhang helped with the coding and the illustration.)
3.3.5 Training and Testing

We use mini-batched stochastic gradient descent (SGD) with learning rate 0.1, momentum 0.9, and weight decay 0.0005 for all training. We use a batch size of 2 for all segmentation training and a batch size of 10 for all insertion point localization and fetal/maternal side classification training. The procedures of training are as follows. We first train the SegDecoder + Encoder from scratch with parameters initialized to zero. Next, we fix the learned weights for the Encoder and train Classification Subnet and IPDecoder subsequently (in other words, the Encoder only acts as a fixed feature pyramid extractor at this stage). The rationale for making such choices is that the training for segmentation task consumes all images we have gathered and makes use of pixel-wise dense supervision, which is much less likely to lead to an overfitting problem. In contrast, the training of Classification Subnet takes binary value as ground truth for each image and the training of IPDecoder only uses around half of the whole dataset (only fetal-side images). To alleviate the lack of labels and to make the model more robust, we use common augmentation techniques including random rotation (±30°), and horizontal and vertical flipping for all training images.

3.3.6 Implementation

We implemented the proposed pipeline in PyTorch [84] and ran experiments on an NVIDIA TITAN Xp GPU. For segmentation training, all images are first resized to 768 × 1024, which is of the same aspect ratio as the original placenta images. For insertion point localization and fetal/maternal side classification training, we resize all cropped “disc” region images to 512 × 512, which is natural because the cropped “disc” regions often have a bounding box close to a square.
3.4  Stage II: Placenta Feature Analysis

3.4.1  Umbilical Cord Insertion Type Categorization

Abnormal cord insertion is a feature of fetal vascular malperfusion [45]. Based on the segmentation, the predicted insertion point location, and the scale we extracted from the ruler, we can measure the distance from the insertion point to the nearest margin of the disc, the length of the long- and short-axis of the disc (all in centimeters). Further, we classify the cord insertion type into “centrally”, “eccentrically”, and “marginally”, based on the ratio between the distance from the insertion point to its closest disc margin and the average length of the long- and short-axis. The thresholds of the ratio between different category is selected by optimizing classification accuracy on the training set. As illustrated in Fig. 3.4, the detailed procedures for insertion type categorization and related automated measurements are as follows.

1. We recover the occluded disc area by merging the originally predicted disc area with the polygon defined by vertices (red) adjacent with both disc area and cord area. Here, erosion and dilation image processing operations is used to remove small holes sometimes appearing in the disc region given by the raw segmentation prediction.

2. We extract the scale information from ruler. Since the ruler in the image could be of any orientation, we first rectify the orientation of the ruler and fit a rectangle from the predicted ruler region. Next, we binarize the pixels within the ruler region such that the scale marker is more distinct. Thirdly, we use kernel density estimation to fit a distribution of the marker pixels (white after binarization) along the long edge of the ruler. And finally we read the number of pixels corresponding to one centimeter as the number of pixels between the two adjacent crests of the estimated distribution.

3. We estimate the long- and short-axis of a placenta by simulating how a pathologists measure those from a 2-D shape by using a vernier caliper.
4. We estimate the distance from the insertion point to its nearest point on disc margin.

5. We calculate the ratio between the distance from the insertion point to its closest disc margin and the average length of the long- and short-axis and conduct the classification based on pre-selected thresholds based on optimizing training set classification accuracy.

### 3.4.2 Other Placental Feature Analysis Tasks

In [22], we also addressed a number of other placental feature analysis tasks in the second stage of our pipeline. Those tasks include: detection of retained placenta (i.e., incomplete placenta), umbilical cord knot, meconium, abruption, chorioamnionitis, irregular shaped disc, and hypercoiled cord. We designed an individual sub-model for each of them. And each sub-model makes use of our segmentation and disc side prediction results from stage I. Since the methods and experiments for these placental feature analysis tasks involve work done by a larger research team, we will abbreviate the details about them and ask the readers to refer to [22] for their complete problem definition and methods.

### 3.5 Experiments and Results

In this section, we summarize the experimental results using our dataset. The results are organized by the two stages and then by the individual tasks within each stage.

#### 3.5.1 Morphological Characterization

##### 3.5.1.1 Segmentation

We compared our approach with two fully convolutional encoder-decoder architectures, the U-Net [75] and the SegNet [10]. The results are shown in Table. 3.1, Fig. 3.5, and Fig. 3.6.
Figure 3.5. (a), (b), and (c) are confusion matrices of our approach, U-Net, and SegNet, respectively.

<table>
<thead>
<tr>
<th>Model</th>
<th>pixel acc.</th>
<th>class acc.</th>
<th>mean IoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>U-Net</td>
<td>98.10</td>
<td>92.98</td>
<td>88.21</td>
</tr>
<tr>
<td>SegNet</td>
<td>96.51</td>
<td>94.56</td>
<td>84.57</td>
</tr>
<tr>
<td>ours</td>
<td><strong>98.73</strong></td>
<td><strong>97.26</strong></td>
<td><strong>93.93</strong></td>
</tr>
</tbody>
</table>

We report the segmentation performance using standard segmentation metrics: pixel accuracy, mean accuracy, and mean IoU. The definition of those metrics are as follows: suppose we have counted how many pixels are predicted to class $j$ but with their ground truth being class $i$ (for every $i, j \in \{0, 1, \ldots, k-1\}$, $k$ is the number of classes) and we store it as the term $C_{ij}$ in a $k \times k$ matrix $C$. We also denote the (ground truth) total number of pixels for class $i$ as $T_i$. It is easy to see that $T_i = \sum_{j=0}^{k-1} C_{ij}$. The pixel accuracy, mean class accuracy, and mean IoU are then defined as follows.

- **Pixel accuracy:**

  \[
  \frac{\sum_{i=0}^{k-1} C_{i,i}}{\sum_{i=0}^{k-1} T_i}. \tag{3.3}
  \]

- **Mean class accuracy:**

  \[
  \frac{1}{k} \sum_{i=0}^{k-1} \frac{C_{i,i}}{T_i}. \tag{3.4}
  \]
**Figure 3.6.** Examples of segmentation results. We show both fetal-side results (top two rows) and maternal-side results (bottom two rows).

**Figure 3.7.** Fetal/maternal side classification confusion matrices comparison. (a) Without shared encoder weights. (b) Ours.
Figure 3.8. Quantitative evaluation of insertion point localization with percentage of correct keypoints (PCK) curves.

- Mean IoU:

\[
\frac{1}{k} \sum_{i=0}^{k-1} \frac{C_{i,i}}{T_i + \sum_{j \neq i} C_{i,j}}.
\]  

(3.5)

In Fig. 3.5(a), (b) and (c), we compare pixel-wise prediction confusion matrices of our approach, U-Net, and Segnet, respectively, which reflects more details about segmentation performance for different categories. We also show a few segmentation examples in Fig. 3.6 for qualitative comparison. Our approach yields the best segmentation results, especially for differentiating the cord and the ruler classes.

3.5.1.2 Fetal/Maternal Side Classification

We achieved an overall fetal/maternal side classification accuracy of 97.51% on our test set. Without the shared encoder representation, we can only achieve 95.52% by training Encoder + Classification Subnet from scratch. We also compare their
Figure 3.9. Qualitative examples of insertion point heat map predictions.

confusion matrices in Fig. 3.7.

3.5.1.3 Insertion Point Localization

We choose Percentage of Correct Keypoints (PCK) as the evaluation metric. PCK measures the percentage of the predictions fall within a circle of certain radius centered at the ground truth location. More formally, PCK at normalized distance $x$ ($x \in [0, 1]$) is defined as:

$$PCK@x = \frac{\{p : \frac{\sqrt{||\hat{p} - p||^2}}{d} < x \land p \in \{p_i\}_{i=1}^n\}}{n},$$  \hspace{1cm} (3.6)

where $\{p_i\}_{i=1}^n$ are the $n$ keypoints we are trying to predict; $\hat{p}$ stands for our prediction for $p$; $||.||_2$ stands for the $L$-2 Euclidean distance, being used to measure the error.
Figure 3.10. Quantitative and qualitative evaluation for insertion point type categorization. (a) The confusion matrix for insertion type categorization. (b) Quantitative evaluation of our estimation on the distance from the insertion point to the nearest disc margin.

of the prediction $\mathbf{\hat{p}}$ from the ground truth $\mathbf{p}$; and $|.|$ stands for the cardinality of a set. In our method, we choose the diameter of the disc\(^2\) as the normalizing factor $d$. We compare our approach (both with and without shared encoder weights) to the Hourglass model (with number of stacks 1 and 2), which shows competitive results in human keypoint localization [60]. Fig. 3.8 shows the PCK curves, with the $x$ axis being the radius normalized by the diameter of the placenta. Each curve in Fig. 3.8 is the average of the results for five models trained with different seeds, and the light-colored band around each curve (view-able when the figure is enlarged) shows the standard deviation of the results. Our approach with shared Encoder consistently gives the best results, especially when the normalized distance is from 0.2 to 0.6. We also show a few qualitative examples of the insertion point heat maps predicted by each model, along with the ground truth in Fig. 3.9(b).

\(^2\)In practice, we approximate the diameter of the disc by the distance between the right most and left most pixel of the “disc” area in the segmentation map.
3.5.2 Placenta Feature Analysis

The predictions of the Stage I models enable us to conduct automatic placenta feature analysis by subsequent models/procedures.

3.5.2.1 Umbilical Cord Insertion Type Categorization

We achieved an overall 88% test accuracy and we show the classification confusion matrix in Fig. 3.10(a). Because the ground truth distance from the insertion point to its nearest point on the disc margin can be extracted from the pathology reports, as shown in B.1, we are able to conduct evaluation on our prediction for this important intermediate value. Fig. 3.10(b) shows the evaluation for our estimation of the distance from the insertion point to its nearest point on the disc margin on
the test set. The $x$-axis represents the threshold of the normalized error (absolute error normalized by the ground truth) and the $y$-axis shows the percentage of our estimation, the error of which is below such threshold. It can be seen that we have a 60% prediction accuracy if we set the threshold to 0.2. Qualitative examples of our insertion type categorization and associated automated categorization could be found in Fig. 3.11. Insertion type predictions are displayed in the upper right corner of each image, along with the ground truth in brackets. The success cases are green boxed and the failed cases are red boxed. For each image, the predicted insertion point location are marked with a green dot; a transparent green mask is overlaid on the image representing the predicted whole disc region; a (green) line is drawn between the insertion point and its nearest point on the disc margin. The predicted length of such line is displayed next to it, along with the ground truth length extracted from the pathology report (in brackets). The predicted long and short axes are also displayed, along with their predicted length in centimeters. We can see that the results for both the umbilical cord insertion type categorization and its related measurements are very appealing such that our method is already very promising to replace the current approach based on the manual measurement and naked-eye inspection.

3.5.2.2 Other Placental Feature Analysis Tasks

In summary, for the tasks of the detection of retained placenta (i.e., incomplete placenta), umbilical cord knot, meconium, abruption, chorioamnionitis, irregular shaped disc, and hypercoiled cord, we achieved very promising results:

1. **Detection of retained placenta**: We achieved a 0.836 area under the curve (AUC) for the receiver operating characteristic (ROC) curve our “retained or not” classification network. And we achieve an 0.636 intersection over union (IOU) for our “retaine region” localization network. As a demonstration of the usefulness of the segmentation map, we show that the “retained or not” classification network can only achieve a 0.827 AUC for the ROC curve. Finally, we note that this module also depends on a placental image being classified as
the maternal side.

2. Detection of meconium, abruption and chorioamnionitis: We achieved 0.97/0.98, 0.72/0.72 and 0.70/0.69 in terms of sensitivity and specificity for the detection of abruption, meconium, and chorioamnionitis, respectively, under the selected operating point.

3. Irregular shape detection: We achieved a 0.87 sensitivity and 0.97 specificity for shape classification.

4. Hypercoiled cord identification: We achieved a 0.85 sensitivity and 0.93 specificity for hypercoiled cord classification.

5. Knot detection: We achieved a 0.81 mean average precision (MAP) for the true knot detection. And from an image classification perspective, our detection network can be used as a classifier for determining if a image contains a “true knot”, with AUC = 0.93 for the ROC curve.

Again, since the methods and experiments for these placental feature analysis tasks involve work done by a larger research team, we will abbreviate the details and ask the readers to refer to [22] for the complete experimental results.

3.6 Discussions and Conclusions

We proposed a two-stage pipeline to address the tasks for automated placental assessment and examination. In the first stage, we designed a novel, compact multi-head encoder-decoder CNN to jointly solve placenta morphological characterization tasks by employing a transfer learning training strategy. We showed that our approach can achieve better performance than competitive baselines for each task. We also showed that the representation learned from the segmentation task can benefit insertion point localization and fetal/maternal side classification task. In the second stage, we took the output from the first stage, as well as the original placenta photo,
as the input and employed multiple independent models for a few noteworthy placental assessment tasks. Through ablation experiments, we demonstrated that the predictions from the first stage models help us achieve better performance for tasks in this stage. For second-stage placenta feature analysis tasks, though our results still have room to be improved, especially when more placental images diagnosed with those abnormalities are available in the future, our current approaches are already useful for triage purpose, which could significantly alleviate the workload for pathologists.

In the future, it will be interesting to explore if some of these tasks can benefit from both fully-labeled (small fraction) and unlabeled placenta (large fraction) photos by using semi-supervised learning techniques. Automated prediction of additional and potentially more fine-grained pathological indicators beyond the ones tackled in this chapter is also a direction we will pursue. We believe the prediction of some of those indicators could benefit from the predictions from the two stages of our current pipeline, such that a multi-stage pipeline becomes feasible.
Conclusions and Future Work

4.1 Conclusions

To conclude, this thesis makes two contributions for advancing data science. First, it presents a fundamental tool potentially useful for data science under edge computing setting. Specifically, we proposed an optimal transport based framework for computing a distance between two hidden Markov models. We demonstrated a number of our frameworks’ advantages through extensive experiments on both synthetic and real data. Specifically, Minimized Aggregated Wasserstein (MAW) is very fast to compute and has better performance than Kullback-Leibler divergence (KLD)-based distance does in certain synthetic data settings and in a few real data classification and retrieval problems; Improved Aggregated Wasserstein is slower but achieves better performance than KLD based distance in more synthetic data settings and in all real data settings. The nature of our framework makes it more appealing under the retrieval with missing dimension setting and edge computing setting.

Second, this thesis proposes a data-driven solution for automated placenta image analysis. Specifically, we designed a two-stage pipeline for automated placental assessment and examination from photos with a focus on the morphological characterization tasks. Through extensive experiments, we demonstrate that our method is to be able to produce accurate morphological characterization with the help of transfer learning. And the morphological characterization results is shown to be
useful for a number of placental feature analysis tasks in the second stage of our pipeline. We’d like to emphasize that pathological diagnosis of placenta is a complex process. Our proposed approach and its future variations are just the first step towards automating it and they need to be further improved and validated before deploy them in a clinical setting, which is mission-critical and could be a matter of life and death. That being said, our proposed approach is still valuable and useful because our results suggest it could at least be used for triage purpose, which could alleviate the workload of physicians and pathologists. Moreover, our approach could potentially democratize the pathological diagnosis of placenta in the sense that it could allow more people in under-developed regions to be diagnosed with this automated method. This could be meaningful because the shortage of pathologists in those regions cannot be mitigated in the near future.

4.2 Future Work

4.2.1 Future Work for Aggregated Wasserstein Framework

In the future, it would be interesting to extend our distances to more general hidden Markov models, such as those with emission functions other than a Gaussian distribution, and to more variants of hidden Markov models, e.g., those factoring in the length of the sequence [37]. It is also worth to test our distance on the task of clustering hidden Markov models. Though we focus on hidden Markov models in this thesis, the idea of aggregated Wasserstein distance could also be readily applied to Gaussian mixture models and other mixture models where the Wasserstein distance between a pair of the components is in closed form. Thus, we should definitely explore applying aggregated Wasserstein idea to those mixture models and their applications in the future.
4.2.2 Future Work for Automated Morphological Characterization of Placenta

In the future, we hope to make our approach massively scalable to potentially hundreds of hospitals across multiple countries and to create and accelerate the “more data – better model – benefit more people” loop. To accomplish that goal, we must first tackle the two most important factors that potentially hinder the scaling of our tool for automated placental assessment:

1. **Learning from unlabeled data.** It is very expensive and time consuming to obtain large scale fully labeled placenta data, especially for the segmentation maps. For placenta diagnosis predictions, the EHR in different hospitals may have inconsistencies such that one diagnosis appearing in one hospital is absent from EHR in another hospital. Besides, the marginal benefit brought by more fully labeled data can decay quickly. It is hence beneficial to pursue a semi-supervised learning approach for parts of or the entire pipeline we have proposed in Chapter 3, such that the vast, much cheaper unlabeled or missing labeled data can contribute to the training. Existing approaches for this direction include co-training [15], where the main idea is to predict labels for unlabeled data using existing models (with good enough performance) and to train the new model on both labeled data as well as those “machine-labeled” data.

2. **Training in a federated learning setting.** Several hospitals and individuals from the United States and internationally have shown interest in our tool following our publication [19]. As we deploy the tool, we will have the opportunity to leverage data from various sources to improve the AI system. Because information in the EHR is sensitive and private, it is impossible to gather all data to a centralized location in order for us to train more accurate models following a conventional paradigm. Hence, another meaningful future direction is to develop new algorithms to further improve our diagnosis pipeline under the federated learning setting, where raw data are constrained to stay in dis-
tributed “federations” and only “small focused, sporadic, updates” are allowed to transmit to a central location. Google has applied a model specifically designed under this federated setting into mobile applications [17, 58], where each mobile device is treated as a federation. In our application, the definition of “federation” could be a country, a region, a hospital or an individual. In short, this endows us a unique opportunity for exploring federated learning settings more realistic to medical applications.
Bibliography


[34] E Haeussner, C Schmitz, F Von Koch, and H-G Frank. “Birth weight correlates with size but not shape of the normal human placenta”. *Placenta* 34.7 (2013), pages 574–582.


T Yee Khong et al. “Sampling and definitions of placental lesions: Amsterdam placental workshop group consensus statement”. _Archives of Pathology & Laboratory Medicine_ 140.7 (2016), pages 698–713.


Clayton Scott. “A rate of convergence for mixture proportion estimation, with application to learning from noisy labels.” In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*. 2015.


Appendices

A Supplementary Material for Aggregated Wasserstein Framework

A.1 An Example of Choosing $\alpha$ for MAW and IAW

An example of how $\alpha$ for $rleg$ is chosen is illustrated in Fig. A1 and the values of all chosen $\alpha$’s are in Table A1. For comparison, MAW- and IAW-based retrieval results for the motion time series at $\alpha = 0$ and $\alpha = 1$ are shown in Fig. A2 and Fig. A3 respectively. By comparing MAW ($\alpha = 1$) in Fig. A2 and IAW ($\alpha = 1$) in Fig. A3, we can see the $D_p(T_1, T_2 : W)$ part in IAW yields much better retrieval than that in MAW. This might be attributed to the better state registration matrix $W$ produced by IAW.

![Figure A1. overall accuracy for rleg. w.r.t MAW’s $\alpha$](image)

---

**Figure A1.** overall accuracy for $rleg$. w.r.t MAW’s $\alpha$
Table A1. Choice of $\alpha$ for MAW and IAW in Motion Retrieval Experiments.

<table>
<thead>
<tr>
<th>Distance</th>
<th>root</th>
<th>head</th>
<th>neck</th>
<th>rbody</th>
<th>lbody</th>
<th>rleg</th>
<th>lleg</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAW</td>
<td>0.03</td>
<td>0.01</td>
<td>0.25</td>
<td>0.07</td>
<td>0.13</td>
<td>0.21</td>
<td></td>
</tr>
<tr>
<td>IAW</td>
<td>0.15</td>
<td>0.05</td>
<td>0.16</td>
<td>0.08</td>
<td>0</td>
<td>0.08</td>
<td></td>
</tr>
</tbody>
</table>

Figure A2. Precision Recall Plot for MAW based Motion Retrieval at $\alpha = 0$ and $\alpha = 1$. The plot for 6 joint-groups, i.e. $root_{12}$, $head\_neck\_thorax_{12}$, $rbody_{12}$, $lbody_{12}$, $rleg_{6}$, $lleg_{6}$, are displayed separately. (Best view in color.)
Figure A3. Precision Recall Plot for IAW based Motion Retrieval at $\alpha = 0$ and $\alpha = 1$. The plot for 6 joint-groups, i.e. root12, head_neck_thorax12, rboby12, lbody12, rleg6, lleg6, are displayed separately. (Best view in color.)
A.2 Seed HMMs Generation for \( \Sigma \) Perturbation Experiments and Transition Matrix Perturbation

The seed HMM generation procedures for \( \Sigma \) perturbation and transition matrix perturbation experiments are specified in Algorithm 4 and Algorithm 5 respectively.

**Algorithm 4** Seed HMMs generation for \( \Sigma \) perturbation experiment.

**Input:** State number: \( M \), dimension: \( D \), transition matrix \( T \in \mathbb{R}^{M \times M} \), number of seed HMMs: \( S \), base sigma \( \Sigma_{\text{base}} \), perturbation magnitude control variable: \( \gamma \).

**Output:** \( \{ \Lambda_i(T, \mathcal{M}(\bar{\mu}, \{\Sigma_{1,i} \ldots \Sigma_{M,i}\}))\}_{i=1}^{S} \)

1. Set every seed HMM the same \( \bar{\mu} = \{\mu_1, \ldots, \mu_M\} \), \( \forall i \in \{1, \ldots, M\} \), \( \mu_i \sim \mathcal{N}(0, 5 \cdot I) \)

2. randomly generate \( \{\Sigma_i\}_{i=1}^{S} \) as \((1-\gamma) \cdot \Sigma_{\text{base}} + \gamma \Sigma_{\text{pert}}^i\) where \( \Sigma_{\text{pert}}^i \sim \mathcal{W}_D(\Sigma_{\text{base}}, df) \).

(\( \mathcal{W}_D \) is the Wishard distribution. We set \( df \) to 10 in our experiments)

3. randomly generate \( \{\beta_j\}_{j=1}^{M} \) where \( \beta_j \sim \mathcal{N}(1, 0.1) \).

4. Set \( \{\Sigma'_{j,i}|\Sigma'_{j,i} = \beta_j \Sigma_i, i = 1 \ldots S, j = 1 \ldots M\} \)

5. Set Every seed HMM the same transition matrix \( T \).

**Algorithm 5** Seed HMMs generation for transition matrix perturbation experiment.

**Input:** State number: \( M \), dimension: \( D \), base transition matrix \( T_{\text{base}} \in \mathbb{R}^{M \times M} \), number of seed HMMs: \( S \), perturbation magnitude control variable: \( \gamma \).

**Output:** \( \{ \Lambda_i(T_i, \mathcal{M}(\bar{\mu}, \{\Sigma_{1,i} \ldots \Sigma_{M,i}\}))\}_{i=1}^{S} \)

1. Set every seed HMM the same \( \bar{\mu} = \{\mu_1, \ldots, \mu_M\} \), \( \forall i \in \{1, \ldots, M\} \), \( \mu_i \sim \mathcal{N}(0, 5 \cdot I) \)

2. Set every seed HMM the same \( \bar{\Sigma} = \{\Sigma_1, \ldots, \Sigma_M\} \), \( \forall i \in \{1, \ldots, M\} \), \( \Sigma_i \) is a \( D \times D \) matrix \( \beta_i \cdot I \), where \( \beta_i \sim \mathcal{N}(1, 0.1) \).

3. randomly generate transition matrices: \( \{T_i\}_{i=1}^{S} \) where \( T_i = \gamma \cdot T_{\text{base}} + (1-\gamma) \cdot P_i \), each row of perturbation matrix \( P \) is sampled by: \( P_{i,:} \sim \text{Dir}([1,1,\ldots,1]_{1 \times M}) \)
A.3 Confusion Matrices for Motion Classification

Confusion matrices for the motion classification experiment in Section 5.3.3 are plotted in Fig. A4.

Figure A4. (a) 6 joints, KL, corresponding to the blue dot in the left sub-figure of Fig. 8, (b) 6 joints, MA W, corresponding to the red dot in the right sub-figure of Fig. 8, (c) 6 joints, IA W, corresponding to the yellow dot in the left sub-figure of Fig. 8, (d) 27 joints, KL, corresponding to the blue dot in the right sub-figure of Fig. 8, (e) 27 joints, MA W, corresponding to the red dot in the left sub-figure of Fig. 8, (f) 27 joints, IA W, corresponding to the yellow dot in the right sub-figure of Fig. 8.

A.4 Adaboost Motion Classification Not Considering Transition Matrix

In order to further demonstrate the effectiveness of the term $D_p(T_1, T_2; W)$, we have conducted new Adaboost classification experiments based only on MAW (and...
IAW) distances between the steady states marginal distributions. Our existing results were obtained by inputting to Adaboost both $D_p(T_1, T_2; W)$ and the distances between the marginal distributions (in Section 5.3.3). We think such a supervised classification setting can demonstrate the usefulness of $D_p$ more directly than the unsupervised retrieval setting. The results are displayed in Figure. 2.11. We can see that for both MAW and IAW, the Adaboost classification results become worse if we don’t consider the term $D_p(T_1, T_2; W)$. And the performance difference is much more significant than the unsupervised retrieval experiments for Mocap and TIMIT data.

Figure A5. Testing accuracies with respect to the iteration number in Adaboost (number of weak classifiers selected). (a) Motion Classification by Adaboost on 6 joints. (b) Motion Classification by Adaboost on 27 joints. The iteration number means the number of features incrementally acquired in Adaboost. MAW for GMM and IAW for GMM here refers to the experiment settings where we solely consider $R_p(M_1, M_2; W)$ when we compute distance matrix by MAW or IAW.

A.5 Analysis of the Effect of $T$

Fig. A6, Fig. A7 and Fig. A8 in the following pages show the results of the experiments that investigate how $T$ affects the retrieval accuracy.
Figure A6. Analyzing the effect of the accuracy of the re-estimated HMMs on the NN retrieval results. In the experiment, $\mu$ is perturbed. Each column corresponds to a different length factor $\tau$. Note that $T = M \times d \times \tau$. The rightmost column with $\tau = \infty$ means that we used the true parameters of the seed HMMs.
Figure A7. Analyzing the effect of the accuracy of the re-estimated HMMs on the NN retrieval results. In the experiment, $\Sigma$ is perturbed. Each column corresponds to a different length factor $\tau$. Note that $T = M \times d \times \tau$. The rightmost column with $\tau = \infty$ means that we used the true parameters of the seed HMMs.
Figure A8. Analyzing the effect of the accuracy of the re-estimated HMMs on the NN retrieval results. In the experiment, the transition matrix is perturbed. Each column corresponds to a different length factor $\tau$. Note that $T = M \times d \times \tau$. The rightmost column with $\tau = \infty$ means that we used the true parameters of the seed HMMs.
A.6 Implementation Details of KL_var_hmm

First, we want to clarify a small mistake in Section 4 of [22] (page 4) for the derivation of $L^\hat{\phi}_t(a_{t-1}, b_{t-1})$ and $L^\hat{\phi}_n(a_{n-1}, b_{n-1})$. The correct recursion equation should be:

$$
L^\hat{\phi}_t(a_{t-1}, b_{t-1}) = \sum_{a_t} \pi_{at|a_{t-1}} p_{n-t}(a_t) \log \sum_{b_t} \omega_{bt|b_{t-1}} e^{L(fa_t||gb_t)} e^{L^\hat{\phi}_{t+1}(a_{t}, b_{t})/p_{n-t}(a_t)}
$$

and the recursion should begin with:

$$
L^\hat{\phi}_n(a_{n-1}, b_{n-1}) = \sum_{a_t} \pi_{an|a_{n-1}} p_0(a_n) \log \sum_{b_n} \omega_{bn|b_{n-1}} e^{L(fa_n||gb_n)}
$$

and should end with:

$$
L_{VA}(f(x_{1:n})||g(x_{1:n})) = \sum_{a_1} \pi_{a_1|I} p_{n-1}(a_1) \log \sum_{b_1} \omega_{b_1|I} e^{L(fa_1||gb_1)} e^{L^\hat{\phi}_{2}(a_{1}, b_{1})/p_{n-1}(a_1)}
$$

We find the result of the above recursion for $L^\hat{\phi}_t(a_{t-1}, b_{t-1})$ and $L^\hat{\phi}_n(a_{n-1}, b_{n-1})$ provides the same results as unsimplified equations (the last 3 equations in page 3 of [22]) only after we added the red terms above, which are missing in the original paper.

We note that variational KL for HMM in [22] are defined for a generalized version of hidden Markov model which also models the length of associated sequences by adding two artificial states: an initial state $I$ and a final state $F$. In contrast, traditional hidden Markov models can be seen as a model for infinite length sequence only. We acknowlege that it could be benificial by considering the length factor when comparing sequences/HMMs, especially for real data. However, our paper only focus on traditional HMMs so we made some simplification of KL_var_hmm to make it work under our experimental setting (without introducing new states: $I$ and $F$).

Specifically,

1. we assume that all HMMs in our experiments are generating the same length sequences (of length $N$). Thus, $L_{VA}(f(x_{1:n})||g(x_{1:n})) = 0$ for any $n \neq N$ since
the term \( p_{n-1}(a_1) = 0 \) under such assumption. And we have:

\[
L_{VA}(f || g) = \sum_{n=0}^{\infty} L_{VA}((f(x_{1:n})||g(x_{1:n}))) = L_{VA}((f(x_{1:N})||g(x_{1:N}))
\]

2. We also assume that all states at time \( N \) have the same probability generating the next \( F \) state. (It's reasonable since all HMMs used in Section 5.1 and Section 5.2 of our paper are purely artificially crafted and we do not have a prior (or a clue from real data) which states tend to end the sequence in the next time step.) This means \( p_0(a_N) = \frac{1}{M} \) for any \( a_N \) (\( M \) is the number of states). Since \( \sum_{a_{t+1}} \pi_{a_{t+1}|a_t} = 1 \), following the equation (17) of [22], we have \( p_{N-t}(a_t) = \frac{1}{M} \) for any \( t = \{1, 2, \ldots, N\} \).

To make the comparison fair, in both Section 5.1 and Section 5.2, we explore different \( N \) and choose the one that leads to the best performance in each set of experiments. We found that for \( \mu \) perturbation and \( \sigma \) perturbation experiments, relatively small \( N \) leads to slightly better results. So We choose \( N = 10 \) for those two sets of experiments in both Section 5.1 and Section 5.2. On the other hand, for transition matrix perturbation experiments, relatively large \( N \) leads to better results. We choose \( N = 100 \) for it. (Further increasing \( N \) will not leads to better results.) In short, we have selected \( N \) in favor of \( KL\_var\_hmm \). Figure A9 shows the analysis of cutoff sequence length \( N \)'s effect on \( KL\_var\_hmm \)'s performance for Section 5.1.

We also note that we use standard log-sum-exp trick in our implementation to avoid numerical errors when computing the term

\[
\log \sum_{b_t} \omega_{b_t|b_{t-1}} e^{L(f_{at}|g_{b_t})} e^{\hat{\phi}_{t+1}(a_t,b_t)/p_{n-t}(a_t)}
\]

in the above recursion equations.
Figure A9. The analysis of cutoff sequence length $N$'s effect on the recall-precision performance of KL_var_hmm.
B Supplementary Material for Automated Morphological Characterization of Placenta

B.1 Information Extracted from Pathology Reports

Table B2 summarizes the information we extracted from a pathology report.
<table>
<thead>
<tr>
<th>Category</th>
<th>Item</th>
<th>Description in the Pathology Report</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Information</td>
<td>Image ID</td>
<td>Unique ID given to the image</td>
</tr>
<tr>
<td></td>
<td>Maternal Age</td>
<td>Age of the mother at the time of delivery</td>
</tr>
<tr>
<td></td>
<td>Gestational Age</td>
<td>Age of fetus at the time of delivery</td>
</tr>
<tr>
<td></td>
<td>Placental Weight</td>
<td>Weight of Placenta in grams</td>
</tr>
<tr>
<td></td>
<td>Delivery Type</td>
<td>Cesarean or Vaginal</td>
</tr>
<tr>
<td></td>
<td>Freshness</td>
<td>Fresh or Fixed</td>
</tr>
<tr>
<td></td>
<td>Shape</td>
<td>Age of the mother at the time of delivery</td>
</tr>
<tr>
<td>Cord</td>
<td>Knot</td>
<td>True, False or None</td>
</tr>
<tr>
<td></td>
<td>Coils</td>
<td>Number of twists/coils in the segment of cord</td>
</tr>
<tr>
<td></td>
<td>Diameter</td>
<td>Average diameter, Minimum and Maximum diameter</td>
</tr>
<tr>
<td></td>
<td>Insertion Type</td>
<td>Central, Eccentric,Margina, Velamentous, Furcate</td>
</tr>
<tr>
<td></td>
<td>Insertion Distance</td>
<td>Distance from the cord insertion to the nearest margin</td>
</tr>
<tr>
<td>Membrane</td>
<td>Color</td>
<td>Opaque, Green, Yellow, Clear, Pink, Pruple, Tan, Gray</td>
</tr>
<tr>
<td></td>
<td>Insertion Type</td>
<td>Circummarginate, Circumvallate, Marginal</td>
</tr>
<tr>
<td>Fetal Side</td>
<td>Color</td>
<td>Opaque, Green, Red, Blue, Yellow, Clear, Pink, Purple, Tan, Gray, White, Variegated</td>
</tr>
<tr>
<td>Maternal Side</td>
<td>Completeness</td>
<td>Complete, Incomplete, Fragmented, Disrupted, Ragged, Roughened</td>
</tr>
<tr>
<td></td>
<td>Adherent Blood Clots</td>
<td>Adherent Blood Clot observed in the gross description</td>
</tr>
<tr>
<td>Final Diagnosis</td>
<td>Meconium</td>
<td>Meconium is detected in the Final Diagnosis</td>
</tr>
<tr>
<td></td>
<td>Amnion Nodosum</td>
<td>Amnion Nodosum is detected in the Final Diagnosis</td>
</tr>
<tr>
<td></td>
<td>Accessory Lobe</td>
<td>Accessory Lobe is detected in the Final Diagnosis</td>
</tr>
<tr>
<td></td>
<td>Chorangioma</td>
<td>Chorangioma is detected in the Final Diagnosis</td>
</tr>
<tr>
<td></td>
<td>Abruption</td>
<td>Abrupton is detected in the Final Diagnosis</td>
</tr>
<tr>
<td></td>
<td>Chorioamnionitis</td>
<td>Chorioamnionitis is detected in the Final Diagnosis</td>
</tr>
<tr>
<td></td>
<td>Placental Hypoplasia</td>
<td>Placental Hypoplasia is detected in the Final Diagnosis</td>
</tr>
</tbody>
</table>
Vita

Yukun Chen

Yukun Chen was born in Xi’an, Shannxi province, People’s Republic of China on March, 1992. He entered the Ph.D. program in Information Sciences and Technology at Penn State University in August 2014. Prior to his Ph.D. study, he received the B.Sc. degree in Physics from University of Science and Technology of China in June 2014. He has been a research assistant in Department of Physics, Oxford University in summer 2013. He worked as a Ph.D. software engineering intern at Google Inc. in 2017 and Facebook Inc. in 2018, 2019. His research interests lie primarily in the area of machine learning, computer vision and data mining. In Jan 2020, He joined Facebook Inc. as a research scientist after graduation.

Selected publications

- Yukun Chen, Jianbo Ye, Jia Li. “A Distance for HMMs based on Aggregated Wasserstein Metric and State Registration”, European Conference on Computer Vision (ECCV), 2016.