이학박사 학위논문

Multi-Level Hidden Markov Model and ULSTM Network

(다층 히든 마코브 모델과 **ULSTM** 네트워크)

2017년 **8**월

서울대학교 대학원

수리과학부

이 준 석

Multi-Level Hidden Markov Model and ULSTM Network

(다층 히든 마코브 모델과 **ULSTM** 네트워크)

지도교수 최 형 인

이 논문을 이학박사 학위논문으로 제출함

2017년 **4**월

서울대학교 대학원

수리과학부

이 준 석

이 준 석의 이학박사 학위논문을 인준함

2017년 **6**월

위 원 장 _____ (인)

부 위 원 장 _____ (인)

위 원 _____ (인)

위 원 _____ (인)

위 원 _____ (인)

Multi-Level Hidden Markov Model and ULSTM Network

A dissertation

submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

to the faculty of the Graduate School of

Seoul National University

by

Junseok Lee

Dissertation Director : Professor Hyung-In Choi

Department of Mathematical Sciences
Seoul National University

August 2017

Abstract

Financial data is a representative example of time series data. In analyzing time series data, unlike other data types, observations at other points of times act primarily to interpret the current observations.Time series data have been studied for a long time using traditional methodologies. This thesis present methods analyzing time series data especially the financial data. Several experiments presented in this thesis will show the effectiveness of the introduced machine learning models. This thesis cover not only a classical machine learning techniques but also recently active techniques.

Time series data is one of important subjects of machine learning. Compared to classical methods, Machine Learning has had an remarkable effect in analyzing time series. We will describe some time series analysis methods that are typical for machine learning. We will also present more advanced models. The first one of them is a model that uses the Markov chain. Chapter 2 provide the basic knowledges about the Markov chains. In Chapter 3, we present an existing model whose base is on the Markov chains. In consequent chapter, a new model that we created will be introduced. The experimental results are also contained in the chapter.

The second part of this thesis start from explaining the deep learning architecture. Chapter 5 contains explanations about basic notations in deep learning and specific type of models in deep learning architecture. The models introduced in this chapter are often used when dealing with time series data in deep learning. In Chapter 6 we present an extended version of the model based on the models introduced in Chapter 5. In this chapter, we conduct an experiment to compare our model with the existing model.

Key words: Hidden Markov Models, Long Short Term Memory

Student Number: 2012-23027

Contents

Chapter 1

Introduction

Financial data has been studied by many researchers as an important part of time series data. Typically, black scholes models or Garch models attempted to explain the sequential structure of financial data. The statistical approach to the analysis of financial data has been successful not only in academic terms but also in practical terms. The irregular looking sequential structure of these financial data is of great interest in machine learning. The goal of this thesis is to propose models which can explain the financial circumstances and find a learning process of our models.

Hidden Markov models (HMMs) have long been used to analyze sequential data as a method of traditional machine learning. The HMM is a probabilistic model in which the distribution of an observation depends on underlying and unobservable state, called hidden state. Under this concept, HMM has been successful in many areas for decades. Scott[14], in his paper, proposed a method of detecting network intrusion using the Markov arrival process, a variant of HMM. Juang and Rabiner[8] suggested a method of speech recognition using HMM in their paper and it was proved that it has many effects. Liu et al[9], in their paper, used HMM to obtain genetic information. In their paper, Monte Carlo simulation methods such as Gibbs sampling are used to present the results. Romberg, Choi and Baraniuk used HMM in image analysis and pattern recognition in their work. In economics, Hamilton proposed a method analysing the business cycle with HMM architecture.

An inference algorithm for HMM was provided by Stratonovich[17]. The inference algorithm he created is a way to find the probability of a model based on recursive methods. An algorithm for decoding hidden states was invented by Viterbi[18]. Viterbi proposed, in his paper, an algorithm to find the most probable sequence of hidden states of HMM. The mathematical algorithm for learning the hidden Markov model was first developed by Baum, et al[1] with aid of Stratonovich's work. After the work done by Baum, et al, there were many variations in estimating HMM. Scott[13][15], in his paper, presented an algorithm to estimate the HMM using Bayesian methods.

The objective of this paper is to introduce a variation of HMM and give a method to estimate the parameters of the model. We named the model as multi-level hidden Markov model. We estimated the parameters for the observations according to the MLHMM estimation procedure presented in the paper using the KOSCOM data set. We obtained the simulation results with the estimated parameters and compared the simulation data with the real data to determine the suitability of our model in the financial analysis.

Our motivation example is a data set containing the information of every transaction in the KOSPI and KOSDAQ market. The data is recorded in milliseconds. Every observation contains at least 49 features, including stock price traded, traded volume, traded time, etc. Unlike normal data, which only contains information on the close price, this data set includes all the changes in price during the day. This data is therefore suitable for studying the market micro-structures. This data also has a data structure suitable for applying the basic idea of the model presented in this paper.

Chapter 2 provides basic knowledges about Markov chains. The provided knowledges are used to give conditions that are needed to use HMM well. In Chapter 3, we give a definition of HMM. The inferencing and learning algorithms for HMM will be presented also. The inference algorithm has recursive structure and it is called a *forward algorithm*. We introduce a backward algorithm that retrieves conditional probabilities through recursion in

the backward direction. After forwarding algorithm and backward algorithm for all steps, We inference the probability and we presented the learning algorithm HMM parameters through the algorithm introduced by Baum, et al. In Chapter 4, we define our new model, multi-level hidden Markov model (MLHMM), and give a description of our model. We present a recursive algorithm for probability inference of our model. Then, we give estimating algorithm for our model. We conduct the estimating process with actual data to determine if our model is suitable for analyzing financial markets. The experiment focuses on finding characteristics of financial data. Representatively, distributions of stock returns are our main concern. To find an adequate distribution of stock returns, common distributions like normal distribution have been used. However, because of the fundamental limitations of common distribution functions it is hard to find a desired distribution. Our goal is to solve the problem. We also conduct statistical tests to confirm that our model accomplish our motivating goal.

From Chapter 5, we investigate the deep learning architecture. Especially, we focus on the specific type of the deep learning techniques. The recurrent neural network (RNN) is the tool for the data with sequential structure. In Chapter 5, we provide basic knowledges about artificial neural networks. We define notations that will be used to construct RNN architecture. Then, we introduce popular RNN cells. Chapter 6 introduce a variation of RNN cell that we created. We will explain the methodology and purpose of our new model. We also conduct an experiment to compare how well a new model predicts financial phenomena compared to existing models. In the experiment, we use the KOSCOM data. We will use some more information as well as pricing data. Using multiple features makes our model can predict the financial circumstances.

The next chapter gives the necessary mathematical notations to define one of our model, MLHMM. We only provide content that includes only the minimum parts necessary to describe our model.

Chapter 2

Markov Chains

2.1 Basics

Definition 2.1.1. *Let* $\{X_t; t \in N\}$ *be a sequence of random variables. We say the sequence of random variables is a* Markov chain *if for every* $t \in N$*, it satisfies the following* Markov property

$$Pr[X_{t+1}|X_t, ..., X_1] = Pr[X_{t+1}|X_t]. \tag{2.1}$$

The conditional probability of state of next time conditioning on the past process up to presence is equivalent to that conditioning only on the presence. The graphical representation for conditional dependencies of Markov chains is described in Figure 2.1

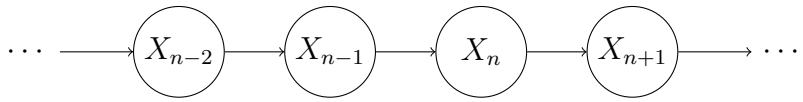

Figure 2.1: Directed acyclic graph for Markov chain dependencies

Definition 2.1.2. *The conditional probabilities,*

$$Pr[X_{s+t} = j|X_s = i]$$

for $i, j \in \Omega$, are called transition probabilities. *If the transition probabilities do not depend on s, we say that the Markov chain is* homogeneous.

Without any indication, we only consider the homogeneous cases. When a Markov chain is homogeneous, its transition probability is denoted by

$$a_{ij}(t) = Pr[X_{s+t} = j | X_s = j].$$

For a Markov chain with finite state space having n states, we define a matrix by

$$A(t) = \big(a_{ij}(t)\big)$$

for $t \in N$. For all $u, s, t \in N$ the following equation holds

$$Pr[X_{u+s+t} = j | X_u = i] = \sum_{k=1}^{n} Pr[X_{u+s+t} = j | X_{u+s} = k] Pr[X_{u+s} = k | X_u]. \tag{2.2}$$

We rewrite the above equation using $a_{ij}(t)$, and get

$$a_{ij}(s+t) = \sum_{k=1}^{n} a_{ik}(s) a_{kj}(t). \tag{2.3}$$

Therefore for all homogeneous Markov chains with finite sample space, the following *Champman-Kolmogorov equation* holds

$$A(s+t) = A(s)A(t)$$

for $s, t \in N$. The Champman-Kolmogorov equation also implies that

$$A(t) = A(1)^t$$

for every $t \in N$. We abbreviate the matrix $A(1)$ as A and $a_{ij}(1)$ as a_{ij}. The row sums of the square matrix A are equal to 1 since the probability that we move any states from a state is 1.

Definition 2.1.3. *When a Markov chain is homogeneous and has finite state space, we call the matrix A the* transition probability matrix *of the Markov chain.*

The state probability $Pr[X_t = i]$ of a Markov chain is one of our main concern. Let $\pi(t)$ denotes row vector representation of the probabilities of being in certain state at time t

$$\pi(t) = (\pi_1(t), ..., \pi_n(t)),$$

where $\pi_i(t)$ denotes the state probability $Pr[X_t = i]$ for each $i = 1, 2, ..., n$. Especially, we call the state probability $\pi(1)1$ as the *initial probability* of a Markov chain.

2.2 Properties of Markov Chains

Definition 2.2.1. *We say a state j of a Markov chain $\{X_t\}$ is* accessible *from a state i if the following holds*

$$Pr[X_{t+s} = j | X_s = i] \neq 0 \qquad \text{for some } t \in N \cup \{0\}.$$

That is, if we get to the state j from i some time or other, we say the state is accessible.

Definition 2.2.2. *If two states i and j are accessible from each other, then we say that the two states* communicate *with each other.*

The communicating relation form an equivalence relation on the state space of a Markov chain. Therefore the communication relation splits the state space into equivalence classes.

Definition 2.2.3. *We call the equivalence class described above as* communicating class. *If a communicating class is not accessible from any other state outside the communicating class we say the class is* closed. *We call a Markov chain is* irreducible *if it has only one communication class.*

Let $p_{ij}(t)$ denotes the conditional probability that the first time to reach j is t for given present state i. The conditional probability that the process will ever be in state j starting from the state i is given by

$$Pr[X_{t+s} = j \text{ for some } t \in N | X_s = i] = \sum_{t=1}^{\infty} p_{ij}(t). \tag{2.4}$$

We define a random variable T_i for each state i by

$$T_i = \min\{t \geq 1 | X_t = i \text{ given } X_0 = i\} \tag{2.5}$$

We call the random variable T_i as the first returning time of state i.

Definition 2.2.4. *We say a state i is* transient *if the following holds*

$$Pr[T_i < \infty] > 0. \tag{2.6}$$

That is, the probability that the process will never return to the state i given that process was initially in state i is not zero. On the other hand, if

$$Pr[T_i < \infty] = 1 \tag{2.7}$$

the state i is said to be recurrent.

Definition 2.2.5. *We say a state iR is* periodic *if there exists an integer $s > 0$ such that*

$$p_{ii}(t) = 0 \tag{2.8}$$

for all $t \in N$ other than the multiples of s. In case that $s = 1$, we say the state i is aperiodic.

Definition 2.2.6. *We say a state i is* positive recurrent *if the expected value of T_i is finite. That is*

$$E[T_i] < \infty. \tag{2.9}$$

A positive recurrent and aperiodic state is called ergodic.

2.3 Conclusion

This concludes the chapter on the basic knowledge about the Markov chains. Upon the knowledge, we will explain a model using the Markov chains. This chapter also gives the basic settings for the Markov chains that we will use in this thesis. If there is no indication, every Markov chain is assumed to be time homogeneous, ergodic and irreducible.

Chapter 3

Hidden Markov Models

HMMs are statistical models which are constructed to figure out the unobservable characteristic of hidden states from observed data. In the HMM architecture the underlying hidden state is assumed to follow the Markov property. Throughout this chapter we assume the hidden states have a finite state space of size n. Also we assume that the underlying Markov chain is irreducible and time homogeneous.

3.1 Construction of Models

There are many variations of HMM. We will introduce a basic form of the hidden Markov model to help understanding the concept of hidden Markov models.

3.1.1 Definitions

The HMMs are doubly stochastic processes. A HMM assumes an underlying process which represents unobservable characteristics of certain phenomenon or nature. The key assumption made in hidden Markov model is that observations are modulated by the underlying unobservable process.

We consider a sequence of observable random variables and denote it as $\{Y_t; t = 0, 1, ..., T\}$. We assume that the random variables Y_t's have the same

finite state space O with $|O| = m$. Let $\{X_t; t = 0, 1, ..., T\}$ be a Markov chain with transition probability matrix $A = \left(a_{ij}\right)$ and a state space $\Omega = \{1, 2, ..., n\}$ of size n. We assume that each random variable Y_t is modulated by corresponding hidden random variable X_t. For detailed description, we introduce some materials.

- $\pi = (\pi_i)$ denotes the initial probability density of hidden state. Since π is a probability density, the following holds

$$\sum_{i=1}^{n} \pi_i = 1. \tag{3.1}$$

- $A = (a_{ij})$ is the state transition probability matrix and a_{ij} is given by

$$a_{ij} = Pr[X_{t+1} = j | X_t = i]$$

  under constraint $\sum_{j=1}^{n} a_{ij} = 1$ for every $i \in \Omega$.

- $e = (e_x)_{x \in \Omega}$ is the emitting probability density functions and e_x is given by

$$e_x(y) = Pr[Y_t = y | X_t = x].$$

  for each $x \in \Omega$. Since e_x is the probability mass function, it satisfies the following

$$\sum_{y \in O} e_x(y) = 1 \tag{3.2}$$

  for each $x \in \Omega$.

Suppose we are given a sequence of observations $\mathbf{y} = \{y_0, ..., y_T\}$ and the parameters $\theta = (\pi, A, e)$. Then, the full likelihood function $L(\theta|\mathbf{y})$ of HMM is given by

$$L(\theta|\mathbf{y}) = \sum_{x_0, ..., x_T = 1}^{n} \pi_{x_0} e_{x_0}(y_0) a_{x_0 x_1} e_{x_1}(y_1) \cdots a_{x_{T-1} x_T} e_{x_T}(y_T), \tag{3.3}$$

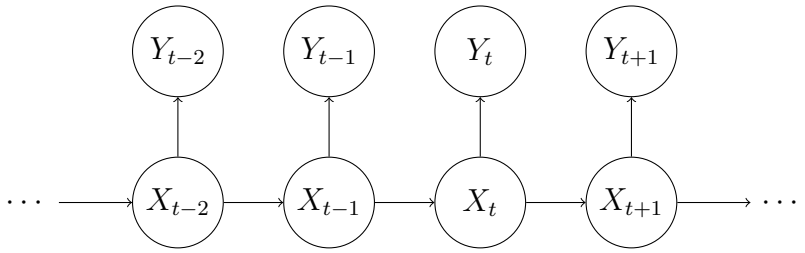

Figure 3.1: Directed acyclic graph for model dependencies

where x_t denote the state at time t.

3.1.2 Main Problems

There are three main problems in HMM architecture.

Evaluation Given the HMM model, the parameters $\theta = (\pi, A, e)$ and the observation sequence $\mathbf{y} = \{y_1, ..., y_T\}$, evaluating the probability that the observations follow the HMM model with given parameters.

Learning Given the HMM model structure and the observation sequence $\mathbf{y} = \{y_1, ..., y_T\}$, estimating the model parameters $\theta = (\pi, A, e)$ which maximize the probability of observations.

Decoding Given the HMM model structure, the parameters $\theta = (\pi, A, e)$ and the observation sequence $\mathbf{y} = \{y_1, ..., y_T\}$, finding the most likely hidden state sequence $\mathbf{x} = \{x_1, ..., x_T\}$ which generate the given observation sequence.

The evaluation problem can be solved by so called forward and backward recursions

3.2 Learning HMM

A traditional way to find the parameter which well fits the model is to maximize the likelihood function. However, there is some difficulty in calcu-

lating likelihood and finding the parameter maximizing the likelihood function. Therefore, in this section, we introduce a method to estimate HMMs.

3.2.1 Maximum-likelihood

Suppose that we have n i.i.d. samples $o_1, o_2, ..., o_n$. We assume that the samples came from a distribution belongs to a family of distributions $\{f(\cdot|\theta)|\theta \in \Theta\}$. We denote the true value of the parameter by θ^*. For i.i.d samples $x_1, x_2, ..., x_n$ with distribution $f(\cdot|\theta)$ the joint probability is given by

$$f(o_1, o_2, ..., o_n|\theta) = f(o_1|\theta) \times f(o_2|\theta) \times \cdots \times f(o_n|\theta). \tag{3.4}$$

The likelihood function is given by the joint probability of samples

$$\mathcal{L}(\theta|x_1, x_2, ..., x_n) = f(x_1, x_2, ..., x_n|\theta). \tag{3.5}$$

Under above construction, we regard the joint probability as a function of the parameter θ. For practical reason, we often use the log-valued function, so called log-likelihood

$$l(\theta) = l(\theta|o_1, o_2, ..., o_n) = \log \mathcal{L}(\theta|o_1, o_2, ..., o_n). \tag{3.6}$$

By maximizing the likelihood function or log-likelihood function, we can estimate the true parameter value $\hat{\theta}$. The method is called the maximum likelihood estimation.

Definition 3.2.1. *Let $\hat{\theta}$ be given by*

$$\hat{\theta} = \underset{\theta \in \Theta}{\operatorname{argmax}}\{\mathcal{L}(\theta)\} = \underset{\theta \in \Theta}{\operatorname{argmax}}\{l(\theta)\}. \tag{3.7}$$

Then we call the estimator $\hat{\theta}$ as maximum likelihood estimator (MLE) *and denote it by $\hat{\theta}_{MLE}$.*

3.2.2 Expectation-Maximizing Algorithm

The expectation maximization algorithm (EM algorithm) is introduced by Dempster, *et al.*[5]. The EM algorithm is an algorithm to find the local maximum of a likelihood function when it is hard to find with direct methods. Specifically, EM algorithm is an effective iterative procedure for computing optimal parameter when the underlying model has latent (in HMM, hidden) variables.

Definition 3.2.2. *Let P and Q be probability measures on a finite sample space Ω. Suppose P is absolutely continuous with respect to Q. Then the* Kullback-Leibler divergence *from Q to P is defined by*

$$KL(P, Q) = E_P[\log(P/Q)]. \tag{3.8}$$

Let $\{f_\theta | \theta \in \Theta\}$ be a family of probability mass functions with parameter space Θ. Assume that the support of f_θ's are identical regardless of θ and $f_\theta = f_{\theta^*}$ if and only if $\theta = \theta^*$. In this case, we define the Kullback-Leibler divergence by

$$KL(\theta_0, \theta) = E_{X|\theta_0}[\log(f_{\theta_0}(X)/f_\theta(X))]. \tag{3.9}$$

For all $x \in \mathbb{R}$ we have an inequality

$$-\log x \geq -x + 1.$$

We get, from above inequality, the following

$$\begin{aligned} KL(\theta_0, \theta) &= -E_{X|\theta_0}[\log(f_\theta(X)/f_{\theta_0}(X))] \\ &\geq E_{X|\theta_0}[-f_\theta(X)/f_{\theta_0}(X) + 1] \\ &= 0. \end{aligned}$$

Proposition 3.2.3. *The Kullback-Leibler divergence satisfies the following inequality*

$$KL(\theta_0, \theta) \geq 0 \tag{3.10}$$

and the equality holds if and only if $\theta = \theta_0$ *a.e. on* Θ.

Suppose we are given a statistical model with observable data $\mathbf{x}$. With unobservable hidden states H, we are given the task of optimizing parameter θ. The likelihood for the model with complete hidden states, $\mathbf{h}$, is given by

$$L(\theta|\mathbf{x}, \mathbf{h}) = p(\mathbf{x}, \mathbf{h}|\theta),$$

where $p(\mathbf{x}, \mathbf{h}|\theta)$ is the probability that we observe $\mathbf{x}$ underlying hidden states $\mathbf{h}$ under the parameter θ. The likelihood for observations $\mathbf{x}$ is given by marginalizing the probability over hidden states $\mathbf{h}$

$$L(\theta|\mathbf{x}) = \sum_{\mathbf{h}} p(\mathbf{x}, \mathbf{h}|\theta). \tag{3.11}$$

The maximum likelihood estimate of θ is given, by maximizing $L(\theta|X)$, as

$$\hat{\theta}_{MLE} = \underset{\theta \in \Theta}{\operatorname{argmax}}\, L(\theta|X). \tag{3.12}$$

The computation time grows exponentially as the number of hidden states increases.

By Beyes' rule, we have

$$p(\mathbf{h}|\mathbf{x}, \theta) = \frac{p(\mathbf{x}, \mathbf{h}|\theta)}{p(\mathbf{x}|\theta)} \tag{3.13}$$

where $p(\mathbf{h}|\mathbf{x}, \theta)$ denotes the probability that we are in states $\mathbf{h}$ given observations $\mathbf{x}$ and θ and $p(\mathbf{x}|\theta)$ denotes the probability that we observe $\mathbf{x}$ under θ. By taking logarithm on both sides, we get

$$\log p(\mathbf{x}|\theta) = \log p(\mathbf{x}, \mathbf{h}|\theta) - \log p(\mathbf{h}|\mathbf{x}, \theta). \tag{3.14}$$

Taking expectation over all possible hidden states under parameter θ^* gives

$$\begin{aligned}\log p(\mathbf{x}|\theta) &= \sum_{\mathbf{h}} p(\mathbf{h}|\mathbf{x},\theta^*)\log p(\mathbf{x}|\theta) \\ &= \sum_{\mathbf{h}} p(\mathbf{h}|\mathbf{x},\theta^*)\log p(\mathbf{x},\mathbf{h}|\theta) + \sum_{\mathbf{h}} p(\mathbf{h}|\mathbf{x},\theta^*)\log p(\mathbf{h}|\mathbf{x},\theta) \\ &= E_{H|X,\theta^*}[\log p(X,H|\theta)] - E_{H|X,\theta^*}[\log p(H|X,\theta)].\end{aligned} \tag{3.15}$$

Let us define $Q(\theta|\theta^*)$ by

$$Q(\theta|\theta^*) = E_{H|\mathbf{x},\theta^*}[\log p(\mathbf{x},H|\theta)]. \tag{3.16}$$

Then, by the proposition 3.2.3, the following inequality holds

$$\begin{aligned}&\log p(\mathbf{x}|\theta) - \log p(\mathbf{x}|\theta^*) \\ &\qquad = E_{H|\mathbf{x},\theta^*}[\log p(\mathbf{x},H|\theta)] - E_{H|\mathbf{x},\theta^*}[\log p(H|\mathbf{x},\theta)] \\ &\qquad - E_{H|\mathbf{x},\theta^*}[\log p(\mathbf{x},H|\theta^*)] + E_{H|\mathbf{x},\theta^*}[\log p(H|\mathbf{x},\theta^*)] \\ &\qquad \geq E_{H|\mathbf{x},\theta^*}[\log p(\mathbf{x},H|\theta)] - E_{H|\mathbf{x},\theta^*}[\log p(\mathbf{x},H|\theta^*)] \\ &\qquad = Q(\theta|\theta^*) - Q(\theta^*|\theta^*)\end{aligned} \tag{3.17}$$

and equality holds if and only if $\theta = \theta^*$.

Proposition 3.2.4. *Let $Q(\theta|\theta^*)$ be defined as in equation 3.16. Then the following inequality holds*

$$\log p(\mathbf{x}|\theta) - \log p(\mathbf{x}|\theta^*) \geq Q(\theta|\theta^*) - Q(\theta^*|\theta^*) \tag{3.18}$$

and the equality holds if and only if $\theta = \theta^*$.

Therefore, we can find the parameter θ such that $Q(\theta|\theta^*) \geq Q(\theta^*|\theta^*)$, the parameter that we found enhance the $p(\mathbf{x}|\theta)$ relative to the old parameter $p(\mathbf{x}|\theta^*)$. The goal of EM-algorithm is to find the parameter that maximizes $Q(\theta|\theta^*)$ rather than directly search the parameter that maximizes the value $\log p(X|\theta)$. The EM-algorithm consists of two steps, expectation step and maximization step. At expectation step, we calculate the expectation,

$$Q(\theta|\theta^*) = E_{H|\mathbf{x},\theta^*}[\log p(\mathbf{x},H|\theta)]$$

and then at maximization step, we find the parameter θ that maximizes $Q(\theta|\theta^*)$.

$$\hat{\theta} = \underset{\theta \in \Theta}{\operatorname{argmax}}\, Q(\theta|\theta^*). \tag{3.19}$$

By iterating the expectation and maximization steps until certain conditions satisfied, we can get a parameter that fits the model. The following is the corresponding pseudo code for the EM algorithm

Algorithm 1 EM algorithm

1: Initialize the parameter $\theta^{(0)}$ and $t = 0$.
2: **while** $|\theta^{(t+1)} - \theta^{(t)}| > \epsilon$ **do**
3: Calculate $Q(\theta|\theta^{(t)})$.
4: Find $\theta^{(t+1)}$ such that

$$\theta^{(t+1)} = \underset{\theta \in \Theta}{\operatorname{argmax}}\, Q(\theta|\theta(t))$$

5: $t \leftarrow t + 1$
6: **end while**
7: **return** $\theta^{(t)}$.

3.2.3 Baum-Welch Algorithm

The Baum-Welch Algorithm was designed by Baum, *et al*[1]. to implement the EM algorithm in HMM circumstances. Before we start, we define some variables and the methods to calculate the variables.

Definition 3.2.5. *Let $f_t(i)$ be the probability that we observe the observations $y_1, ..., y_t$ and we are in hidden state i at time t under given parameter θ. The variable f_t is defined by*

$$f_t(i) = Pr[Y_0 = y_0, ..., Y_t = y_t, X_t = i|\theta]. \tag{3.20}$$

Then we call f_t a forward variable *at time t.*

The forward variable at time $t = 0$ is clearly given by

$$\begin{aligned} f_0(i) &= Pr[X_0 = i|\theta] \\ &= \pi_i. \end{aligned} \tag{3.21}$$

When the forward variable at time t is given for every state i, we can calculate the forward variable at time $t+1$. The formula is given by

$$\begin{aligned} f_{t+1}(i) =& Pr[Y_0 = y_0, ..., Y_{t+1} = y_{t+1}, X_{t+1} = i|\theta] \\ =& \sum_{j=1}^{n} Pr[Y_0 = y_0, ..., Y_t = y_t, X_t = j|\theta] Pr[X_{t+1} = i|X_t = j, \theta] \\ & Pr[Y_{t+1} = y_{t+1}|X_{t+1} = i, \theta] \\ =& \sum_{j=1}^{n} f_t(j) a_{ji} e_i(y_{t+1}). \end{aligned} \tag{3.22}$$

Therefore, we get the following proposition.

Proposition 3.2.6. *The following recurrence relation between forward variables at time t and $t+1$ holds*

$$f_{t+1}(i) = \sum_{j=1}^{n} f_t(j) a_{ji} e_i(y_{t+1}) \tag{3.23}$$

for $t = 1, 2, ..., T-1$ and $i \in \Omega$.

By above recursion, we can calculate the forward variables for every $t = 1, 2, ..., T$. We call the calculation procedure as $forwardrecursion$. The pseudo code for forward recursion is as follows.

Definition 3.2.7. *Let $b_t(i)$ be the probability that we observe the observations $y_{t+1}, ..., y_T$ given being in the hidden state i at time t under given parameter θ. The variable b_t is defined by*

$$b_t(i) = Pr[Y_{t+1} = y_{t+1}, ..., Y_T = y_T|X_t = i, \theta]. \tag{3.24}$$

We call the variable b_t a backward variable at time t. For computational convenience, we let $b_T(i) = 1$ for $i = 1, 2, ..., n$.

Algorithm 2 Forward Recursion

1: Set $f_0(i) = \pi_i$ for $i = 1, 2, ..., n$.
2: **for** $t = 1$ to T **do**
3: **for** $i = 1$ to n **do**
4: $f_t(i) \leftarrow \sum_{j=1}^{n} f_{t-1}(j) a_{ji} e_i(y_t)$
5: **end for**
6: **end for**
7: **return** $f_t(i)$ for $t = 0, 1, ..., T$ and $i = 1, 2, ..., n$.

As we did in forward variables, we can calculate the backward variable at each time recursively. Suppose we have, for each hidden state $j = 1, 2, ..., n$, backward variable, $b_{t+1}(j)$, at time $t + 1$. Then,

$$\begin{aligned} b_t(i) =& Pr[Y_{t+1} = y_{t+1}, ..., Y_T = y_T | X_t = i, \theta] \\ =& \sum_{j=1}^{n} Pr[Y_{t+2} = y_{t+2}, ..., Y_T = y_T | X_{t+1} = j, \theta] Pr[X_{t+1} = j | X_t = i, \theta] \\ & Pr[Y_{t+1} = y_{t+1} | X_{t+1} = j, \theta] \\ =& \sum_{j=1}^{n} b_{t+1}(j) a_{ij} e_j(y_{t+1}) \end{aligned} \tag{3.25}$$

Proposition 3.2.8. *The following recurrence relation between backward variables holds*

$$b_t(i) = \sum_{j=1}^{n} b_{t+1}(j) a_{ij} e_j(y_{t+1}) \tag{3.26}$$

for $t = T - 1, T - 2, ..., 1$ *and* $i \in \Omega$.

The direction of calculation of backward variables is backward direction, different from that of forward variables. The procedure that calculate the backward variables is called *backward recursion*. The pseudo code is as follows

Algorithm 3 Backward Recursion

1: Set $b_T(i) = 1$ for all $i = 1, 2, ..., n$.
2: **for** $t = T - 1$ to 1 **do**
3: **for** $i = 1$ to n **do**
4: $b_t(i) \leftarrow \sum_{j=1}^{n} b_{t+1}(j) a_{ij} e_j(y_{t+1})$
5: **end for**
6: **end for**
7: **return** $b_t(i)$; $t = 0, 1, 2, ..., T$, $i = 1, 2, ..., n$.

The full likelihood, $L(\theta|Y)$, is given in equation 3.3.

$$L(\theta|Y) = \sum_{x_0,...,x_T=1}^{n} \pi_{x_0} e_{x_0}(y_0) a_{x_0 x_1} e_{x_1}(y_1) \cdots a_{x_{T-1} x_T} e_{x_T}(y_T),$$

We denote the probability that we observe the sequence of observations $\mathbf{y} = \{y_0, ..., y_T\}$ when underlying hidden states sequence $\mathbf{x} = \{x_0, ..., x_T\}$ under parameter θ by $p(\mathbf{x}, \mathbf{y}|\theta)$.

$$p(\mathbf{x}, \mathbf{y}|\theta) = Pr[Y_0 = y_0, ..., Y_T = y_T, X_0 = x_0, ..., X_T = x_T|\theta] \tag{3.27}$$

The summands in equation 3.3 is the same as $p(x_0, ..., x_T|\theta)$. Therefore, we can simplify the equation 3.3 by

$$L(\theta|\mathbf{y}) = \sum_{x_0,...,x_T=1}^{n} p(\mathbf{x}, \mathbf{y}|\theta) \tag{3.28}$$

Calculating $L(\theta|\mathbf{y})$ requires $O(T^n)$ computation time. To avoid the difficulty in computation, we adopt another approach to get the parameter. We denote the sequence of hidden random variables $\{X_0, ..., X_T\}$ by $\mathbf{X}$

Let $Q(\theta|\theta^*)$ be

$$Q(\theta|\theta^*) = \sum_{x_0,...,x_T=1}^{n} p(\mathbf{x}, \mathbf{y}|\theta^*) \log p(\mathbf{x}, \mathbf{y}|\theta). \tag{3.29}$$

We know, from the proposition 3.2.4, that if we update θ so that

$$Q(\theta|\theta^*) \geq Q(\theta^*|\theta^*), \tag{3.30}$$

then, we get an optimized parameter compared to the old one. The new parameter $\hat{\theta}$ is given by

$$\hat{\theta} = \underset{\theta\in\Theta}{\operatorname{argmax}}\, Q(\theta|\theta^*).$$

Suppose we are initially given a parameter θ^* with transition probabilities a_{ij}^* for $i, j = 1, 2, ..., n$, initial probabilities π_i^* for $i = 1, 2, ..., n$ and the probability distributions $e_i^*(y)$ for $i = 1, 2, ..., n$ and $y \in O$. Moreover, we are given the forward and backward variables at each time, under given parameter θ^*.

Our work is to find the parameter $\hat{\theta}$ which maximizes the expectation $Q(\theta|\theta^*)$ under some constraints.

$$\begin{aligned}
&\hat{\theta} = \underset{\theta\in\Theta}{\operatorname{argmax}}\, Q(\theta|\theta^*),\\
&\textit{under}\\
&\sum_{j=1}^{n} a_{ij} = 1 \qquad \textit{for } i = 1, 2, ..., n,\\
&\sum_{i=1}^{n} \pi_i = 1 \qquad \textit{and}\\
&\sum_{y\in\Omega_y} e_i(y) = 1 \qquad \textit{for } i = 1, 2, ..., n
\end{aligned} \tag{3.31}$$

Let $\mathcal{L}(\theta|\theta^*)$ be the Lagrangian of the problem

$$\mathcal{L}(\theta|\theta^*) = Q(\theta|\theta^*) - \lambda_\pi \sum_{i=1}^{n} \pi_i - \sum_{i=1}^{n} \lambda_a^i \sum_{j=1}^{n} a_{ij} - \sum_{i=1}^{n} \lambda_e^i \sum_{y\in O} e_i(y). \tag{3.32}$$

The differential of $\mathcal{L}$ with respect to π_i is given by

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \pi_i} =& \frac{\partial Q(\theta|\theta^*)}{\partial \pi_i} - \lambda_\pi \\ =& \frac{\partial}{\partial \pi_i}\Big\{ \sum_{x_0,...,x_T=1}^{n} \log\Big(\pi_{x_0} e_{x_0}(y_0) a_{x_0x_1} e_{x_1}(y_1) \cdots a_{x_{T-1}x_T} e_{x_T}(y_T)\Big) \\ & p(x_0, ..., x_T, \mathbf{y}|\theta^*)\Big\} - \lambda_\pi \\ =& \frac{1}{\pi_i} \sum_{x_1,...,x_T=1}^{n} p(i, x_1, ..., x_T, \mathbf{y}|\theta^*) - \lambda_\pi.\end{aligned} \tag{3.33}$$

The optimal $\hat{\pi}_i$ makes the differential zero, for each $i = 1, 2, ..., n$. Therefore, we get the following equations for $\hat{\pi}_i$.

$$\begin{aligned}&\sum_{x_1,...,x_T=1}^{n} p(i, x_1, ..., x_T|\theta^*) - \lambda_\pi \hat{\pi}_i = 0, \\ &\sum_{i=1}^{n} \hat{\pi}_i = 1\end{aligned} \tag{3.34}$$

Solving the equation 3.34 gives

$$\hat{\pi}_i = \frac{\sum_{x_1,...,x_T=1}^{n} p(i, x_1, ..., x_T, \mathbf{y}|\theta^*)}{p(\mathbf{y}|\theta^*)} \tag{3.35}$$

for each $i = 1, 2, ..., n$. The equation 3.35 can be rewritten by

$$\hat{\pi}_i = Pr[x_0 = i|\mathbf{y}, \theta^*]. \tag{3.36}$$

for each $i \in \Omega$. For each a_{ij}, we conduct similar procedure. The differential

of $\mathcal{L}$ with respect to a_{ij} is given by

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial a_{ij}} =& \frac{\partial Q(\theta|\theta^*)}{\partial a_{ij}} - \lambda_a^i \\ =& \frac{\partial}{\partial a_{ij}} \Big\{ \sum_{x_0,...,x_T=1}^{n} \log\Big(\pi_{x_0} e_{x_0}(y_0) a_{x_0x_1} e_{x_1}(y_1) \cdots a_{x_{T-1}x_T} e_{x_T}(y_T)\Big) \\ & p(x_0, ..., x_T, \mathbf{y}|\theta^*)\Big\} - \lambda_a^i \\ =& \frac{1}{a_{ij}} \sum_{t=0}^{T-1} \sum_{x_0,...x_{t-1}=1}^{n} \sum_{x_{t+2},...x_T=1}^{n} p(x_0, ..., x_{t-1}, x_t = i, x_{t+1} = j, ..., x_T, \mathbf{y}|\theta^*) \\ & - \lambda_a^i \end{aligned} \tag{3.37}$$

As in the case of π, the optimizing solution $\hat{a}_{ij}$ which maximizes $\mathcal{L}$ makes the differential zero for all $i, j = 1, 2, ..., n$. The equalities give the following equations

$$\begin{aligned}\lambda_a^i \hat{a}_{ij} - & \sum_{t=0}^{T-1} \sum_{x_0,...x_{t-1}=1}^{n} \sum_{x_{t+2},...x_T=1}^{n} p(x_0, ..., x_{t-1}, x_t = i, x_{t+1} = j, ..., x_T, \mathbf{y}|\theta^*) \\ & = 0 \end{aligned} \tag{3.38}$$

for all $i, j = 1, 2, ..., n$ and

$$\sum_{j=1}^{n} \hat{a}_{ij} = 1 \tag{3.39}$$

for all $i = 1, 2, ..., n$. The solution to the equations are given by

$$\hat{a}_{ij} = \frac{\sum_{t=0}^{T-1} \sum_{x_0,...x_{t-1}=1}^{n} \sum_{x_{t+2},...x_T=1}^{n} p(x_0, ..., x_{t-1}, x_t = i, x_{t+1} = j, ..., x_T, \mathbf{y}|\theta^*)}{\sum_{t=0}^{T-1} \sum_{x_0,...x_{t-1}=1}^{n} \sum_{x_{t+1},...x_T=1}^{n} p(x_0, ..., x_{t-1}, x_t = i, x_{t+1}, ..., x_T, \mathbf{y}|\theta^*)} \tag{3.40}$$

for each $i, j = 1, 2, ...n$. By marginalizing the probabilities, the equation 3.40 can be rewritten by

$$\hat{a}_{ij} = \frac{\sum_{t=0}^{T-1} Pr[X_t = i, X_{t+1} = j|\mathbf{y}, \theta^*]}{\sum_{t=0}^{T-1} Pr[X_t = i|\mathbf{y}, \theta^*]}, \tag{3.41}$$

for all $i, j = 1, 2, ..., n$.

Remaining calculation is for the parameter $e_i(y)$, the probability mass function of observation given hidden state i. Therefore, we also differentiate $\mathcal{L}$ with respect to $e_i(y)$ and get

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial e_i(y)} =& \frac{\partial Q(\theta|\theta^*)}{\partial e_i(y)} - \lambda_e^i \\
=& \frac{\partial}{\partial e_i(y)} \Big\{ \sum_{x_0,...,x_T=1}^{n} \log\Big(\pi_{x_0} e_{x_0}(y_0) a_{x_0 x_1} e_{x_1}(y_1) \cdots a_{x_{T-1} x_T} e_{x_T}(y_T)\Big) \\
& p(x_0, ..., x_T, \mathbf{y}|\theta^*) \Big\} - \lambda_e^i \\
=& \frac{1}{e_i(y)} \sum_{t=0}^{T} \sum_{x_0,...,x_{t-1}=1}^{n} \sum_{x_{t+1},...,x_T=1}^{n} \mathbf{1}_{\{y_t\}}(y) p(x_0, ..., x_t = i, ..., x_T, \mathbf{y}|\theta^*) - \lambda_e^i
\end{aligned} \tag{3.42}$$

where $\mathbf{1}_A$ denotes the indicator function of the set A.
The $\hat{e_i}$ we want will make the derivative of $\mathcal{L}$ against the e_i to zero. Therefore, we can obtain the following relations.

$$\begin{aligned}
&\sum_{t=0}^{T} \sum_{x_0,...,x_{t-1}=1}^{n} \sum_{x_{t+1},...,x_T=1}^{n} \mathbf{1}_{\{y_t\}}(y) p(x_0, ..., x_t = i, ..., x_T, \mathbf{y}|\theta^*) = \lambda_e^i \hat{e}_i(y), \\
&\sum_{y \in O} \hat{e}_i(y) = 1
\end{aligned} \tag{3.43}$$

for all $i = 1, 2, ..., n$. By solving the linear equations, we get the following

solution for each $e_i(y)$.

$$e_i(y) = \frac{\sum_{t=0}^{T} \sum_{x_0,...,x_{t-1}=1}^{n} \sum_{x_{t+1},...,x_T=1}^{n} \mathbf{1}_{\{y_t\}}(y) p(x_0, ..., x_t = i, ..., x_T, \mathbf{y}|\theta^*)}{\sum_{t=0}^{T} \sum_{x_0,...,x_{t-1}=1}^{n} \sum_{x_{t+1},...,x_T=1}^{n} p(x_0, ..., x_t = i, ..., x_T, \mathbf{y}|\theta^*)} \tag{3.44}$$

for $y \in O$. Marginalizing the probabilities and applying Bayes' theorem gives

$$e_i(y) = \frac{\sum_{t=0}^{T} \mathbf{1}_{\{y_t\}}(y) Pr[X_t = i|\mathbf{y}, \theta^*]}{\sum_{t=0}^{T} Pr[X_t = i|\mathbf{y}, \theta^*]}. \tag{3.45}$$

The discussion so far suggests the following proposition.

Proposition 3.2.9. *Suppose we are given a parameter $\theta^* = (\pi^*, A^*, e_i^*)$. Then, the new parameter $\hat{\theta} = (\hat{\pi}, \hat{A}, \hat{e}_i)$ which optimizes $Q(\theta|\theta^*)$ is given by*

$$\begin{aligned} \hat{\pi}_i &= Pr[x_0 = i|\mathbf{y}, \theta^*], \\ \hat{a}_{ij} &= \frac{\sum_{t=0}^{T-1} Pr[X_t = i, X_{t+1} = j|\mathbf{y}, \theta^*]}{\sum_{t=0}^{T-1} Pr[X_t = i|\mathbf{y}, \theta^*]} \text{ and} \\ \hat{e}_i(y) &= \frac{\sum_{t=0}^{T} \mathbf{1}_{\{y_t\}}(y) Pr[X_t = i|\mathbf{y}, \theta^*]}{\sum_{t=0}^{T} Pr[X_t = i|\mathbf{y}, \theta^*]} \end{aligned} \tag{3.46}$$

for $i, j \in \Omega$ and $y \in O$.

There are many other estimating algorithms. Scott[15] proposed Bayesian method for HMMs. The Markov chain Monte Carlo (MCMC) techniques enables estimation without the recursive methods. Also, by mixing the MCMC techniques and the recurrent formulas, we can enhance the efficiency of the estimation of HMMs.

3.3 Conclusion

This concludes the chapter on the hidden Markov models and its estimating procedure. The concepts of the HMM is used in the next chapter. A variant of

HMM will be introduced in the next chapter including an adjusted algorithm for estimating the model.

Chapter 4

Multi-Level Hidden Markov Model

HMM has been successful in many areas. When dealing with time series, HMM assumes the same probabilistic model at every step. However, when applying HMM to real phenomena, it is not appropriate to analyze all time steps with the same probability model. In this regard, we came to think of the weaknesses of HMM and suggested changes in the model to overcome them.

4.1 Model Construction

We consider a homogeneous Markov chain $\{X_t; t = 0, 1, ..., T\}$ with transition probabilities $A = (a_{xx'})$ for $x, x' \in \Omega_x$ with state space $\Omega_x = \{x^1, x^2, ..., x^n\}$. Each $a_{xx'}$ is defined by

$$a_{xx'} = Pr[X_{t+1} = x' | X_t = x] \tag{4.1}$$

for each $x, x' \in \Omega_x$. We call the hidden process $\{X_t\}$ as the second hidden state process or initial modulating state process. The probability that we move from $x \in \Omega_x$ to any other states at next time is 1. Therefore, we get

the following equality

$$\begin{aligned}\sum_{x' \in \Omega_x} a_{xx'} &= \sum_{x' \in \Omega_x} Pr[X_{t+1} = x' | X_t = x, \theta] \\ &= 1\end{aligned} \tag{4.2}$$

for each $x \in \Omega_x$.

For each $t = 0, 1, ..., T$, we construct homogeneous Markov chains $\{Y_t^{(s)}; s = 1, ..., S\}$ sharing a transition probabilities $B = (b_{yy'})$ so that every Markov chain has the same transition probability and state space $\Omega_y = \{y^1, y^2, ..., y^m\}$. For simplicity, we set $Y_t^{(0)}$ to be X_t. In order to clarify the explanation, we will sometimes use the notation $Y_t^{(0)}$ instead of X_t. Let $\mathbf{Y}_t = \{Y_t^{(1)}, ..., Y_t^{(S)}\}$ be a Markov chain whose initial state is modulated by X_t. We call each hidden state process $Y_t = \{Y_t^{(s)}; s = 1, 2, ..., S\}$ as the first hidden state. The probability that we move from a certain state $y \in \Omega_y$ to any other states is 1. Therefore, as in above, we get the following

$$\sum_{y' \in \Omega_y} b_{yy'} = 1 \tag{4.3}$$

for $y \in \Omega_y$. Let us denote the probability that $Y_t^{(1)} = y$ given $X_t = x$ where $x \in \Omega_x$ and $y \in \Omega_y$ by c_{xy}. Then c_{xy} is given by

$$c_{xy} = Pr[Y_t^{(1)} = y | X_t = Y_t^{(0)} = x] \tag{4.4}$$

for $x \in \Omega_x$ and $y \in \Omega_y$. For each $x \in \Omega_x$, the probability that we move from x to any state $y \in \Omega_y$ is 1. Therefore, we get the following constraint for the parameter c_{xy}.

$$\sum_{y \in \Omega_y} c_{xy} = 1 \tag{4.5}$$

for $x \in \Omega_x$.

For the Markov chain $\{X_t\}$, let $\pi = (\pi_x)$ be the initial state probability. That is, $\pi_x = Pr[X_0 = x]$. Since the probability that we are in any states in

Ω_x is 1, the sum of π_x is 1.

$$\sum_{x \in \Omega_x} \pi_x = 1. \tag{4.6}$$

For each y in the state space of Ω_y, we assign a probability density e_y. Each e_y is defined by

$$e_y(o) = Pr[O_t^{(s)} = o | Y_t^{(s)} = y] \tag{4.7}$$

for $y \in \Omega_y$, $t = 0, 1, ..., T$ and $s = 1, 2, ..., S$. Since for each $y \in \Omega_y$, e_y is probability density function the following holds

$$\sum_{o \in \Omega_o} e_y(o) = 1 \tag{4.8}$$

for each $y \in \Omega_y$. We set $\mathbf{o}_t$ to be a sequence of observations at time t, $(o_t^{(1)}, ..., o_t^{(S)})$, and $\mathbf{o}$ to be a sequence of $\mathbf{o}_t$, $(\mathbf{o}_0, ..., \mathbf{o}_T)$.

Definition 4.1.1. *Let O be a stochastic process satisfying*

$$Pr[O|A, B, C, \pi, \{e_y\}] = \sum_{x_0, ..., x_T \in \Omega_x} \sum_{y_t^{(s)} \in \Omega_y} \pi_{x_0} \Gamma_{x_0}(o_0) \prod_{t=1}^{T} a_{x_{t-1} x_t} \Gamma_{x_t}(o_t), \tag{4.9}$$

where

$$\Gamma_{x_t}(o_t) = c_{x_t y_t^{(1)}} e_{y_t^{(1)}}(o_t) \prod_{s=1}^{S-1} b_{y_t^{(s)} y_t^{(s+1)}} e_{y_{t+1}^{(s)}}(o_{t+1}^{(s+1)}).$$

Then the process $\{O\}$ is said to follow a Multi-level hidden Markov model (MLHMM) modulated by $\{X_t\}$ and $\{Y_t^{(s)}\}$.

A MLHMM is a variation of HMM. The graphical description of the model structure is described in figure 4.1. The hidden states and observations in the MLHMM can be viewed as nodes in a graph as in figure 4.1.

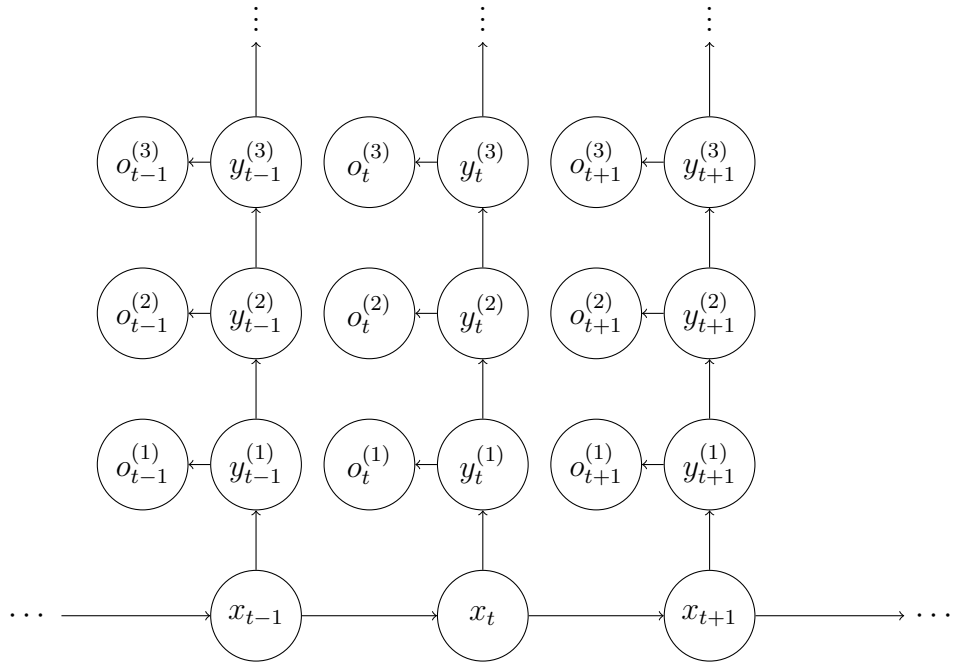

Figure 4.1: Directed acyclic graph for a specific type of MLHMM dependencies.

The figure 4.1 clearly reveals the characteristics of the MLHMM. We built a new hidden state in MLHMM that constructs hidden states that modulate each observation and modulates its hidden state. This model requires a challenge to apply the method used in existing HMMs. So, in the next section, we will make some tools to our model work well.

4.2 Estimation of MLHMM

Direct estimation of the parameters of the MLHMM is computationally very difficult. In order to avoid this difficulty, we propose some algorithms by extending the idea in HMM.

4.2.1 Probability Evaluating Process

Let $O_{X|Y}$ denotes the set of observations which are not conditionally independent with a random variable X given Y. When we are given in the situation as in Figure 4.2, we define the forward variable of X in direction X_i^+, $f_{X,X_i^+}(x)$, by

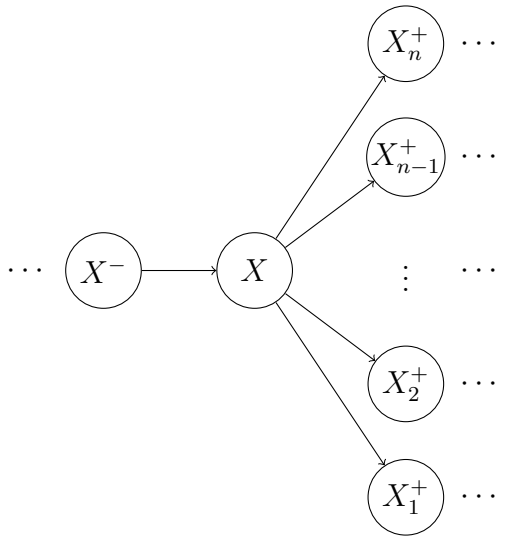

Figure 4.2: A directed acyclic graph having multiple children

$$f_{X,X_i^+}(x) = Pr[O_{X|X_i^+}, X = x] \tag{4.10}$$

Also we define the backward variable of X in direction X_i^+ by

$$b_{X,X_i^+}(x) = Pr[O_{X|\{X^-,X_i^+\}}|X = x] \tag{4.11}$$

and the backward variable of X by

$$b_X(x) = Pr[O_{X|X^-}|X = x]. \tag{4.12}$$

Let us denote the transition probability for each edge under the parameter

θ by $a_{X,Y}(x, y)$. That is,

$$a_{X,Y}(x, y) = Pr[Y = y|X = x]. \tag{4.13}$$

Let Ω^- be the state space of the random variable X^- and Ω_x be the state space of the random variable X. For each X_i^+, let Ω_i be the corresponding state space. Then, we have, for each X_i^+,

$$\begin{aligned} b_{X,X_i^+}(x) &= Pr[O_{X|\{X^-,X_i^+\}}|X = x] \\ &= \prod_{j\neq i} \sum_{x_j^+\in\Omega_j} Pr[O_{X_j^+|X}|X_j^+ = x_j^+]Pr[X_j^+ = x_j^+|X = x] \\ &= \prod_{j\neq i} \sum_{x_j^+\in\Omega_j} a_{X,X_j^+}(x, x_j^+)b_{X_j^+}(x_j^+). \end{aligned} \tag{4.14}$$

Also we have, for backward variable without direction, the following recurrence relation

$$\begin{aligned} b_X(x) &= Pr[O_{X|X^-}|X = x, \theta] \\ &= \prod_{i=1}^{n} \sum_{x_i^+\in\Omega_i} Pr[O_{X_i^+|X}|X_i^+ = x_i^+, \theta]Pr[X_i^+ = x_i^+|X = x] \\ &= \prod_{i=1}^{n} \sum_{x_i^+\in\Omega_i} a_{X,X_i^+}(x, x_i^+)b_{X_i^+}(x_i^+) \end{aligned} \tag{4.15}$$

for $x \in \Omega_x$. So, we can compute the backward variable at X when the backward variables at steps X_i^+ are given. The following computation gives

the recurrence relation for the backward variable at X.

$$\begin{aligned}
f_{X,X_i^+}(x) &= Pr[O_{X|X_i^+}, X = x|\theta] \\
&= \sum_{x^- \in \Omega_x} Pr[O_{X^-|X}, X^- = x^-|\theta] Pr[X = x|X^- = x^-, \theta] \\
&\prod_{j \neq i} \sum_{x_j^+ \in \Omega_j} Pr[O_{X_j^+|X}|X_j^+ = x_j^+, \theta] Pr[X_j^+ = x_j^+|X = x, \theta] \\
&= \sum_{x^- \in \Omega^-} Pr[O_{X^-|X}, X^- = x^-|\theta] Pr[X = x|X^- = x^-, \theta] \\
&\prod_{j \neq i} \sum_{x_j^+ \in \Omega_j} a_{X,X_j^+}(x, x_j^+) b_{X_j^+}(x_j^+) \\
&= b_{X,X_i^+}(x) \sum_{x^- \in \Omega_{x^-}} f_{X^-,X}(x^-) Pr[X = x|X^- = x^-, \theta] \\
&= b_{X,X_i^+}(x) \sum_{x^- \in \Omega_{x^-}} f_{X^-,X}(x^-) a_{X^-,X}(x^-).
\end{aligned} \tag{4.16}$$

By combining the discussion so far, we can get the following proposition.

Proposition 4.2.1. *For the variables f_{X,X_i^+}, b and $b_{X_i^+}$, we have the following recurrence relations*

$$\begin{aligned}
f_{X,X_i^+}(x) &= b_{X,X_i^+}(x) \sum_{x^- \in \Omega^-} f_{X^-,X}(x^-) a_{X^-,X}(x^-) \\
&= \prod_{j \neq i} \sum_{x_j^+ \in \Omega_j} b_{X_j^+}(x_j^+) a_{X,X_j^+}(x, x_j^+) \sum_{x^- \in \Omega^-} f_{X^-,X}(x^-) a_{X^-,X}(x^-) \\
b_X(x) &= \prod_{i=1}^{n} \sum_{y \in \Omega_i} a_{X,X_i^+}(x, y) b_{X_i^+}(y) \\
b_{X_i^+}(x) &= \prod_{j \neq i} \sum_{y \in \Omega_i} a_{X,X_j^+}(x, y) b(y),
\end{aligned} \tag{4.17}$$

where $a_{X,Y}(x, y)$ denotes the probability $Pr[Y = y|X = x, \theta]$.

Therefore, the probabilities of a node having multiple children nodes and

a parent node can be calculated with the probabilities of children nodes and a parent node.

We define some types of variables to apply the preceding discussion to the concrete MLHMM model we presented. We will also propose an algorithm for our model in a similar way to HMM. Different from the case of HMM, the four parameters to be presented in our MLHMM model should be calculated sequentially. Therefore, our discussion will be presented in the order of these calculations.

Definition 4.2.2. *Let $b_{Y_t^{(s)}}(y)$ be the probability that we observe the observations $o_{Y_t^{(s+1)}|Y_t^{(s)}}$ given $Y_t^{(s)} = y$. That is, $b_{Y_t^{(s)}}(y)$ is given by*

$$b_{Y_t^{(s)}}(y) = Pr[o_{Y_t^{(s+1)}|Y_t^{(s)}}|Y_t^{(s)} = y, \theta] \tag{4.18}$$

for $t = 0, 1, ..., T$, $s = 0, 1, ..., S$ and $y \in \Omega_y$. We call $b_{Y_t^{(s)}}$ as the backward variable of Y.

Let us set the each last state backward variable of Y, $b_{Y_t^S}$, to be 1 for every $y \in \Omega_y$ and $t = 0, 1, ..., T$ for computational convenience. Suppose we are given $b_{Y_t^{(s+1)}}(y)$ for all $y \in \Omega_y$. With the given values of $b_{Y_t^{(s+1)}}(y)$, we can compute $b_{Y_t^{(s)}}(y)$ as follows.

$$\begin{aligned} b_{Y_t^{(s)}}(y) =& Pr[o_{Y_t^{(s+1)}|Y_t^{(s)}}|Y_t^{(s)} = y, \theta] \\ =& \sum_{y' \in \Omega_y} Pr[o_{Y_t^{(s+2)}|Y_t^{(s+1)}}|Y_t^{(s+1)} = y', \theta] \\ & Pr[O_t^{(s+1)} = o_t^{(s+1)}|Y_t^{(s+1)} = y', \theta] Pr[Y_t^{(s+1)} = y'|Y_t^{(s)} = y, \theta] \\ =& \sum_{y' \in \Omega_y} b_{Y_t^{(s+1)}}(y') e_{y'}(o_t^{(s+1)}) b_{yy'} \end{aligned} \tag{4.19}$$

for $t = 0, 1, ..., T$, $s = 1, 2, ..., S-1$ and $y \in \Omega_y$. When $s = 0$, the above

equation is written in a slightly different form.

$$\begin{aligned} b_{Y_t^{(0)}}(y) =& Pr[O_{Y_t^{(1)}|Y_t^{(0)}}|Y_t^{(0)} = y, \theta^*] \\ &= \sum_{y' \in \Omega_y} Pr[O_{Y_t^{(2)}|Y_t^{(1)}}|Y_t^{(1)} = y', \theta^*] \\ &\quad Pr[O_t^{(1)} = o_t^{(1)}|Y_t^{(1)} = y', \theta] Pr[Y_t^{(1)} = y'|Y_t^{(0)} = y, \theta^*] \\ &= \sum_{y' \in \Omega_y} b_{Y_t^{(1)}}(y') e_{y'}(o_t^{(1)}) c_{yy'} \end{aligned} \tag{4.20}$$

for $t = 1, 2, ..., T$ and $y \in \Omega_x$. Based on the discussion so far, we can get the following proposition.

Proposition 4.2.3. *Suppose we are given a parameter θ and observations O. Assume that we are given $b_{Y_t^{(s+1)}}(y')$ for all $y' \in \Omega_y$. Then we can get the following recurrence relation.*

$$b_{Y_t^{(s)}}(y) = \sum_{y' \in \Omega_y} b_{Y_t^{(s+1)}}(y') e_{y'}(o_t^{(s+1)}) b_{yy'} \tag{4.21}$$

for $s = 1, 2, ..., S-1$ and $t = 0, 1, ..., T$. For $s = 0$, we get the equation

$$b_{Y_t^{(0)}}(y) = \sum_{y' \in \Omega_y} b_{Y_t^{(1)}}(y') e_{y'}(o_t^{(1)}) c_{yy'} \tag{4.22}$$

for $t = 0, 1, ..., T$.

In HMM, there was only one type of backward variable for the estimation, but another type of backward variable is required for MLHMM estimation. There are many ways to create a backward variable, but here we define the following type of backward variable.

Definition 4.2.4. *Let $b_{X_t}(x)$ denotes the probability that we observe the observations $O_{X_{t+1}|X_t}$ and given in the state $X_t = x$. Mathematically, $b_{X_t}(x)$ is defined by*

$$b_{X_t}(x) = Pr[O_{X_{t+1}|X_t}|X_t = x, \theta] \tag{4.23}$$

for $t = 0, 1, ..., T$ *and* $x \in \Omega_x$. *We call this variable as the* backward variable of X.

To find the value of $b_{X_t}(x)$ for all $t = 0, 1, ..., T$, we want to find the recurrent relation as before. This recurrence relation is made by backward step. We set, for computational convenience, the last values $b_{X_T}(x)$ to be 1.

$$b_{X_T}(x) = 1 \tag{4.24}$$

for all $x \in \Omega_x$. For the remaining steps, we do the calculations through the recurrence relation that will be given below. For $t = 0, 1, ..., T-1$, we can get the following backward recurrence relation.

$$\begin{aligned} b_{X_t}(x) =& Pr[O_{X_{t+1}|X_t}|X_t = x, \theta] \\ =& \sum_{x'=1}^{n} Pr[O_{X_{t+2}|X_{t+1}}|X_{t+1} = x', \theta] \\ & Pr[O_{Y_{t+1}^{(1)}|Y_{t+1}^{(0)}}|X_{t+1} = x', \theta] Pr[X_{t+1} = x'|X_t = x, \theta] \\ =& \sum_{x'=1}^{n} Pr[O_{X_{t+2}|X_{t+1}}|X_{t+1} = x', \theta] \\ & Pr[O_{Y_{t+1}^{(1)}|Y_{t+1}^{(0)}}|Y_{t+1}^{(0)} = x', \theta] Pr[X_{t+1} = x'|X_t = x, \theta] \\ =& \sum_{x'=1}^{n} b_{X_{t+1}}(x') b_{Y_{t+1}^{(0)}}(x') a_{xx'} \end{aligned} \tag{4.25}$$

for $x \in \Omega_x$. Through this process, we can calculate the backward variable of X at every step.

Definition 4.2.5. *Let* f_{X_t} *denotes the probability that we observe the observations* $O_{X_t|X_{t+1}}$ *and we are in state* $X_t = x$. *Mathematically,* f_{X_t} *is defined by*

$$f_{X_t}(x) = Pr[O_{X_{t-1}|X_t}, X_t = x|\theta] \tag{4.26}$$

for $t = 1, 2, ..., T$ *and* $x \in \Omega_x$. *We call this variable as the* forward variable of X.

In order to calculate the forward variables for X, we need to calculate the backward variables for Y that we made before.

$$\begin{aligned} f_{X_t}(x) =& Pr[O_{X_t|X_{t+1}}, X_t = x|\theta] \\ =& \sum_{x'=1}^{n} Pr[O_{X_{t-1}|X_t}, X_{t-1} = x'|\theta] Pr[X_t = x|X_{t-1} = x', \theta] \\ & Pr[O_{Y_t^{(1)}|X_t}|X_{t-1} = x', \theta] \\ =& \sum_{x'=1}^{n} f_{X_{t-1}}(x') a_{x'x} b_{Y_t^{(0)}}(x') \end{aligned} \tag{4.27}$$

for $t = 1, 2, ..., T$ and $x \in \Omega_x$. This result give the following proposition.

Proposition 4.2.6. *The forward variable f_{X_t} satisfies the following recurrence relation.*

$$f_{X_t}(x) = \sum_{x'=1}^{n} f_{X_{t-1}}(x') a_{x'x} b_{Y_{t-1}^{(0)}}(x') \tag{4.28}$$

for $t = 1, 2, ..., T$ and $x \in \Omega_x$. For $t = 0$, $f_{X_0}(x)$ is given by

$$f_{X_0}(x) = \pi_x \tag{4.29}$$

for all $x \in \Omega_x$, where $\pi_x = Pr[X_0 = x|\theta]$.

What remains for us is to define a forward variable for Y and find the calculation method for the variable.

Definition 4.2.7. *Let $f_{Y_t^{(s)}}$ denotes the probability that we observe the observations $O_{Y_t^{(s)}|X_t^{(s+1)}}$ and we are in state $Y_t^{(s)} = y$ for some $y \in \Omega_y$. Mathematically, $f_{Y_t^{(s)}}$ is defined by*

$$f_{Y_t^{(s)}}(y) = Pr[O_{Y_t^{(s)}|Y_t^{(s+1)}}, Y_t^{(s)} = y|\theta] \tag{4.30}$$

for $t = 0, 1, 2, ..., T$, $s = 0, 1, ..., S$ *and* $y \in \Omega_y$. *We call this variable as the* forward variable of Y.

Unlike previous calculations, we do not have to specify an initial value for this variable. We first define $f_{Y_t^{(0)}}$ by

$$f_{Y_t^{(0)}}(i) = Pr[Y_t^{(0)}, O_{Y_t^{(1)}|Y_t^{(0)}}|\theta] = \tag{4.31}$$

To calculate this step, the forward and the backward variables of X must be calculated in advance.

$$\begin{aligned} f_{Y_t^{(s)}}(y) =& Pr[O_{Y_t^{(s)}|Y_t^{(s+1)}}, Y_t^{(s)} = y|\theta] \\ =& \sum_{y'=1}^{m} Pr[O_{Y_t^{(s-1)}|Y_t^{(s)}}, Y_t^{(s-1)} = y'|\theta] \\ & Pr[Y_t^{(s)} = y|Y_t^{(s-1)}, \theta] Pr[O_t^{(s)} = o_t^{(s)}|Y_t^{(s)}, \theta] \\ =& \sum_{y'=1}^{m} f_{Y_t^{(s-1)}}(y') b_{y'y} e_y(o_t^{(s)}) \end{aligned} \tag{4.32}$$

for $s = 2, 3, ..., S$ and $t = 0, 1, ..., T$. For $s = 1$, we will give slightly different approach to get $f_{Y_t^{(s)}}$. The equation for $s = 1$ is given by

$$\begin{aligned} f_{Y_t^{(1)}}(y) =& Pr[O_{Y_t^{(1)}|Y_t^{(2)}}, Y_t^{(1)} = y|\theta] \\ =& \sum_{x=1}^{n} Pr[O_{Y_t^{(0)}=X_t|Y_t^{(1)}}, Y_t^{(0)} = X_t = x|\theta] \\ & Pr[Y_t^{(1)} = y|Y_t^{(0)}, \theta] Pr[O_t^{(1)} = o_t^{(1)}|Y_t^{(1)}, \theta] \\ =& \sum_{x=1}^{n} f_{Y_t^{(0)}}(x) c_{xy} e_y(o_t^{(1)}), \end{aligned} \tag{4.33}$$

for $y \in \Omega_y$. The above steps are the probability evaluating procedure for the MLHMM. Because of the presence of the nodes with multiple children nodes, the evaluating process of the MLHMM is slightly different from that of HMM. The probabilities that we computed will be used to update the

model parameters.

4.2.2 Updating process

Let $p(\mathbf{x}, \mathbf{y}, \mathbf{o}|\theta)$ be the probability that we are in hidden states $\mathbf{x} \in \Omega_{\mathbb{X}}$ and $\mathbf{y} \in \Omega_{\mathbb{Y}}$ and observe the observations $\mathbf{o}$ under the parameter θ. Therefore, $p(\theta, \mathbf{x}, \mathbf{y})$ is given by

$$p(\mathbf{x}, \mathbf{y}, \mathbf{o}|\theta) = \pi_{x_0}\Gamma_{x_0}(\mathbf{o}_0, \mathbf{y}_0)\prod_{t=1}^{T} a_{x_{t-1}x_t}\Gamma_{x_t}(\mathbf{o}_t, \mathbf{y}_t) \tag{4.34}$$

where $\mathbf{x} \in \Omega_{\mathbb{X}}$, $\mathbf{y} \in \Omega_{\mathbb{Y}}$ and Γ is as given in equation (4.9). If we let $L(\theta|\mathbf{o})$ be the likelihood function for the MLHMM under parameter θ, $L(\theta|\mathbf{o})$ is given by

$$\begin{aligned} L(\theta|\mathbf{o}) &= Pr[\mathbf{O} = \mathbf{o}|\theta] \\ &= \sum_{\mathbf{x}\in\Omega_{\mathbb{X}}}\sum_{\mathbf{y}\in\Omega_{\mathbf{Y}}} p(\mathbf{x}, \mathbf{y}, \mathbf{o}|\theta) \end{aligned} \tag{4.35}$$

The optimal parameter θ for MLHMM makes $L(\theta|\mathbf{o})$ maximized.

We conduct parameter estimation by using $Q(\theta|\theta^*)$ where $Q(\theta|\theta^*)$ is defined by

$$\begin{aligned} Q(\theta|\theta^*) =& E_{\mathbf{X},\mathbf{Y}|\theta^*}[\log Pr[\mathbf{X}, \mathbf{Y}, \mathbf{o}|\theta]] \\ &= \sum_{\mathbf{x}\in\Omega_{\mathbb{X}}}\sum_{\mathbf{y}\in\Omega_{\mathbb{Y}}} Pr[\mathbf{x}, \mathbf{y}, \mathbf{o}|\theta^*] \log Pr[\mathbf{x}, \mathbf{y}, \mathbf{o}|\theta] \end{aligned} \tag{4.36}$$

By proposition 3.2.4, we know that $Q(\theta|\theta^*) \geq Q(\theta^*|\theta^*)$ implies $L(\theta|O) \geq L(\theta^*|O)$. Therefore, our optimizing problem is converted into the following problem

$$\hat{\theta} = \underset{\theta\in\Theta}{\operatorname{argmax}}\, Q(\theta|\theta^*) \tag{4.37}$$

subject to

$$\begin{aligned}
&\sum_{x\in\Omega_x} \pi_x = 1,\\
&\sum_{x'\in\Omega_x} a_{xx'} = 1 \qquad \textit{for } x \in \Omega_x,\\
&\sum_{y'\in\Omega_y} b_{yy'} = 1 \qquad \textit{for } y \in \Omega_y,\\
&\sum_{y\in\Omega_y} c_{xy} = 1 \qquad \textit{for } x \in \Omega_x \textit{ and}\\
&\sum_{o\in\Omega_o} e_y(o) = 1 \qquad \textit{for } y \in \Omega_y.
\end{aligned}$$

Then our dual problem of optimization is in what follows. Let us define $\mathcal{L}$ by

$$\begin{aligned}
\mathcal{L}(\theta|\theta^*) =& Q(\theta|\theta^*) - \sum_{x\in\Omega_x} \lambda_a^x \sum_{x'\in\Omega_x} a_{xx'} - \sum_{y\in\Omega_y} \lambda_b^y \sum_{y'\in\Omega_y} b_{yy'}\\
&- \sum_{x\in\Omega_x} \lambda_c^x \sum_{y\in\Omega_y} c_{xy} - \lambda_\pi \sum_{x\in\Omega_x} \pi_x - \sum_{y\in\Omega_y} \lambda_e^y \sum_{o\in\Omega_o} e_y(o)
\end{aligned} \tag{4.38}$$

for $\theta, \theta^* \in \Theta$. Our task is to find the $\hat{theta}$ which makes the differential of $\mathcal{L}$ be zero. By differentiating $\mathcal{L}$ with respect to π_x, we get the following

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \pi_x} =& \frac{\partial Q(\theta|\theta^*)}{\partial \pi_x} - \lambda_\pi\\
=& \sum_{\mathbf{x}\in\Omega_{\mathbb{X}|X_0=x}} \sum_{\mathbf{y}\in\Omega_{\mathbb{Y}}} \frac{\partial p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta)}{\partial \pi_x} \frac{p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*)}{p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta)} - \lambda_\pi\\
=& \sum_{\mathbf{x}\in\Omega_{\mathbb{X}|X_0=x}} \sum_{\mathbf{y}\in\Omega_{\mathbb{Y}}} \frac{1}{\pi_x} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*) - \lambda_\pi
\end{aligned} \tag{4.39}$$

for each $x \in \Omega_x$. The optimizing value $\hat{\pi}_x$ makes the differential $\frac{\partial \mathcal{L}}{\partial \pi_x}$ be 0.

Therefore we get the following equalities.

$$\begin{aligned} &\sum_{\mathbf{x}\in\Omega_{\mathbb{X}|x_0=x}} \sum_{\mathbf{y}\in\Omega_{\mathbb{Y}}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*) - \lambda_\pi \pi_x = 0 \\ &\sum_{x\in\Omega_x} \pi_x = 1. \end{aligned} \tag{4.40}$$

Solving the equations gives

$$\hat{\pi}_x = \frac{\sum_{\mathbf{x}\in\Omega_{\mathbb{X}|x_0=x}} \sum_{\mathbf{y}\in\Omega_{\mathbb{Y}}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*)}{\sum_{x\in\Omega_x} \sum_{\mathbf{x}\in\Omega_{\mathbb{X}|x_0=x}} \sum_{\mathbf{y}\in\Omega_{\mathbb{Y}}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*)} \tag{4.41}$$

for $x \in \Omega_x$. By marginalizing the probability and using Bayes' theorem, we get

$$\begin{aligned} \sum_{\mathbf{x}\in\Omega_{\mathbb{X}|x_0=x}} \sum_{\mathbf{y}\in\Omega_{\mathbb{Y}}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*) =& Pr[X_0 = x, \mathbf{o}|\theta^*] \\ =& Pr[X_0 = x|\mathbf{o},\theta^*] Pr[\mathbf{o}|\theta^*] \end{aligned} \tag{4.42}$$

for $x \in \Omega_x$. We also get the following equation

$$\sum_{x\in\Omega_x} \sum_{\mathbf{x}\in\Omega_{\mathbb{X}|x_0=x}} \sum_{\mathbf{y}\in\Omega_{\mathbb{Y}}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*) = Pr[\mathbf{o}|\theta^*]. \tag{4.43}$$

Therefore, the following equation holds for the initial probability $\hat{\pi}_x$

$$\hat{\pi}_x = Pr[X_0 = x|\mathbf{o},\theta^*]. \tag{4.44}$$

The differential of $\mathcal{L}$ with respect to the transition probability $a_{xx'}$, for each

$x, x' \in \Omega_x$, is given by

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial a_{xx'}} =& \frac{\partial Q(\theta|\theta^*)}{\partial a_{xx'}} - \lambda_a^x \\ =& \sum_{t=0}^{T-1} \sum_{\mathbf{x}\in\Omega_{\mathbb{X}}} \sum_{\mathbf{y}\in\Omega_{\mathbb{Y}}} \frac{\partial p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta)}{\partial a_{xx'}} \frac{p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*)}{p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta)} - \lambda_a^x \\ =& \sum_{t=0}^{T-1} \sum_{\mathbf{x}\in\Omega_{\mathbb{X}|X_t=x,X_{t+1}=x'}} \sum_{\mathbf{y}\in\Omega_{\mathbb{Y}}} \frac{1}{a_{xx'}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*) - \lambda_a^x.\end{aligned} \tag{4.45}$$

The optimizing parameter $\hat{a}_{xx'}$ satisfies the following equations.

$$\begin{aligned}&\sum_{t=0}^{T-1} \sum_{\mathbf{x}\in\Omega_{\mathbb{X}|X_t=x,X_{t+1}=x'}} \sum_{\mathbf{y}\in\Omega_{\mathbb{Y}}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*) - \lambda_a^x a_{xx'} = 0, \\ &\sum_{x'\in\Omega_x} a_{xx'} = 1\end{aligned} \tag{4.46}$$

Solving the equation 4.46 gives us the following

$$\hat{a}_{xx'} = \frac{\sum_{t=0}^{T-1} \sum_{\mathbf{x}\in\Omega_{\mathbb{X}|X_t=x,X_{t+1}=x'}} \sum_{\mathbf{y}\in\Omega_{\mathbb{Y}}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*)}{\sum_{t=0}^{T-1} \sum_{x'\in\Omega_x} \sum_{\mathbf{x}\in\Omega_{\mathbb{X}|X_t=x,X_{t+1}=x'}} \sum_{\mathbf{y}\in\Omega_{\mathbb{Y}}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*)} \tag{4.47}$$

for each $x, x' \in \Omega_x$. By marginalization and Bayes' theorem, we get the following equations for the probability

$$\begin{aligned}\sum_{\mathbf{x}\in\Omega_{\mathbb{X}|X_t=x,X_{t+1}=x'}} \sum_{\mathbf{y}\in\Omega_{\mathbb{Y}}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*) =& Pr[X_t = x, X_{t+1} = x', \mathbf{o}|\theta^*] \\ =& Pr[X_t = x, X_{t+1} = x'|\mathbf{o},\theta^*] Pr[\mathbf{o}|\theta^*]\end{aligned} \tag{4.48}$$

for each $x, x' \in \Omega_x$ and we also get

$$\begin{aligned}\sum_{x' \in \Omega_x} \sum_{\mathbf{x} \in \Omega_{\mathbb{X}|X_t=x, X_{t+1}=x'}} \sum_{\mathbf{y} \in \Omega_{\mathbb{Y}}} p(\mathbf{x}, \mathbf{y}, \mathbf{o}|\theta^*) =& Pr[X_t = x, \mathbf{o}|\theta^*] \\ =& Pr[X_t = x|\mathbf{o}, \theta^*] Pr[\mathbf{o}|\theta^*].\end{aligned} \tag{4.49}$$

Therefore, we get the following equation for $\hat{a}_{ij}$

$$\hat{a}_{xx'} = \frac{\sum_{t=0}^{T-1} Pr[X_t = x, X_{t+1} = x'|\mathbf{o}, \theta^*]}{\sum_{t=0}^{T-1} Pr[X_t = x|\mathbf{o}, \theta^*]} \tag{4.50}$$

for $x, x' \in \Omega_x$. The differential of $\mathcal{L}$ with respect to the transition probability $b_{yy'}$, for $y, y' \in \Omega_y$, is given by

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial b_{yy'}} =& \frac{\partial Q(\theta|\theta^*)}{\partial b_{yy'}} - \lambda_b^y \\ =& \sum_{t=0}^{T} \sum_{s=1}^{S-1} \sum_{\mathbf{x} \in \Omega_{\mathbb{X}}} \sum_{\mathbf{y} \in \Omega_{\mathbb{Y}}} \frac{\partial p(\mathbf{x}, \mathbf{y}, \mathbf{o}|\theta)}{\partial b_{yy'}} \frac{p(\mathbf{x}, \mathbf{y}, \mathbf{o}|\theta^*)}{p(\mathbf{x}, \mathbf{y}, \mathbf{o}|\theta)} - \lambda_b^y \\ =& \sum_{t=0}^{T} \sum_{s=1}^{S-1} \sum_{\mathbf{x} \in \Omega_{\mathbb{X}}} \sum_{\mathbf{y} \in \Omega_{\mathbb{Y}|Y_t^s=y, Y_t^{(s+1)}=y'}} \frac{1}{b_{yy'}} p(\mathbf{x}, \mathbf{y}, \mathbf{o}|\theta^*) - \lambda_b^y.\end{aligned} \tag{4.51}$$

The optimizing parameter $\hat{b}_{yy'}$ satisfies the following equalities:

$$\begin{aligned}&\sum_{t=0}^{T} \sum_{s=1}^{S-1} \sum_{\mathbf{x} \in \Omega_{\mathbb{X}}} \sum_{\mathbf{y} \in \Omega_{\mathbb{Y}|Y_t^s=y, Y_t^{(s+1)}=y'}} p(\mathbf{x}, \mathbf{y}, \mathbf{o}|\theta^*) - \lambda_b^y b_{yy'} = 0, \\ &\sum_{y' \in \Omega_y} b_{yy'} = 1\end{aligned} \tag{4.52}$$

for all $y, y' \in \Omega_y$. Thus we get the following solution for the above equalities

$$\hat{b}_{yy'} = \frac{\sum_{t=0}^{T} \sum_{s=1}^{S-1} \sum_{\mathbf{x} \in \Omega_{\mathbb{X}}} \sum_{\mathbf{y} \in \Omega_{\mathbb{Y}|Y_t^s=y, Y_t^{(s+1)}=y'}} p(\mathbf{x}, \mathbf{y}, \mathbf{o}|\theta^*)}{\sum_{t=0}^{T} \sum_{s=1}^{S-1} \sum_{\mathbf{x} \in \Omega_{\mathbb{X}}} \sum_{\mathbf{y} \in \Omega_{\mathbb{Y}|Y_t^s=y}} p(\mathbf{x}, \mathbf{y}, \mathbf{o}|\theta^*)} \tag{4.53}$$

for $y, y' \in \Omega_y$. Marginalizing the probability over appropriate sequences and applying Bayes' theorem gives the following equation

$$\begin{aligned}\sum_{\mathbf{x}\in\Omega_{\mathbb{X}}}\sum_{\mathbf{y}\in\Omega_{\mathbb{Y}|Y_t^s=y,Y_t^{(s+1)}=y'}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*) =& Pr[Y_t^{(s)}=y, Y_t^{(s+1)}=y', \mathbf{o}|\theta^*] \\ =& Pr[Y_t^{(s)}=y, Y_t^{(s+1)}|\mathbf{o},\theta^*]Pr[\mathbf{o}|\theta^*]\end{aligned} \tag{4.54}$$

for $t = 0, 1, ..., T$, $s = 1, 2, ..., S-1$ and $y, y' \in \Omega_y$. Also we get the following

$$\begin{aligned}\sum_{\mathbf{x}\in\Omega_{\mathbb{X}}}\sum_{\mathbf{y}\in\Omega_{\mathbb{Y}|Y_t^s=y}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*) =& Pr[Y_t^{(s)}=y, \mathbf{o}|\theta^*] \\ =& Pr[Y_t^{(s)}=y|\mathbf{o},\theta^*]Pr[\mathbf{o}|\theta^*]\end{aligned} \tag{4.55}$$

for $t = 0, 1, ..., T$, $s = 1, 2, ..., S-1$ and $y \in \Omega_y$. Therefore, the equation (4.53) is converted into

$$\hat{b}_{yy'} = \frac{\sum_{t=0}^{T}\sum_{s=1}^{S-1} Pr[Y_t^{(s)}=y, Y_t^{(s+1)}=y'|\mathbf{o},\theta^*]}{\sum_{t=0}^{T}\sum_{s=1}^{S-1} Pr[Y_t^{(s)}=y|\mathbf{o},\theta^*]} \tag{4.56}$$

for $y, y' \in \Omega_y$. The differential of $\mathcal{L}$ with respect to the inter transition probability c_{xy}, where $x \in \Omega_x$ and $y \in \Omega_y$, is given by

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial c_{xy}} =& \frac{\partial Q(\theta|\theta^*)}{\partial c_{xy}} - \lambda_c^x \\ =& \sum_{t=0}^{T}\sum_{\mathbf{x}\in\Omega_{\mathbb{X}|X_t=x}}\sum_{\mathbf{y}\in\Omega_{\mathbb{Y}|Y_t^{(s)}=y}} \frac{\partial p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta)}{\partial c_{xy}}\frac{p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*)}{p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta)} - \lambda_c^x \\ =& \sum_{t=0}^{T}\sum_{\mathbf{x}\in\Omega_{\mathbb{X}|X_t=x}}\sum_{\mathbf{y}\in\Omega_{\mathbb{Y}|Y_t^{(s)}=y}} \frac{1}{c_{xy}}p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*) - \lambda_c^x.\end{aligned} \tag{4.57}$$

The optimizing parameter $\hat{c}_{xy}$ makes the differential zero. Therefore, we get

the following equations

$$\begin{aligned}\sum_{t=0}^{T}\sum_{\mathbf{x}\in\Omega_{\mathbb{X}|X_t=x}}\sum_{\mathbf{y}\in\Omega_{\mathbb{Y}|Y_t^{(s)}=y}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*) - \lambda_c^x c_{xy} = 0 \\ \sum_{y\in\Omega_y} c_{xy} = 1\end{aligned} \tag{4.58}$$

for $x \in \Omega_x$ and $y \in \Omega_y$. Solving the equation (4.58) gives us the following

$$\hat{c}_{xy} = \frac{\sum_{t=0}^{T}\sum_{\mathbf{x}\in\Omega_{\mathbb{X}|X_t=x}}\sum_{\mathbf{y}\in\Omega_{\mathbb{Y}|Y_t^{(s)}=y}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*)}{\sum_{t=0}^{T}\sum_{y\in\Omega_y}\sum_{\mathbf{x}\in\Omega_{\mathbb{X}|X_t=x}}\sum_{\mathbf{y}\in\Omega_{\mathbb{Y}|Y_t^{(s)}=y}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*)} \tag{4.59}$$

for $x \in \Omega_x$ and $y \in \Omega_y$. Applying Bayes' theorem after marginalizing the probability gives the following equation

$$\begin{aligned}\sum_{\mathbf{x}\in\Omega_{\mathbb{X}|X_t=x}}\sum_{\mathbf{y}\in\Omega_{\mathbb{Y}|Y_t^{(s)}=y}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*) &= Pr[X_t = x, Y_t^{(1)} = y, \mathbf{o}|\theta^*] \\ &= Pr[X_t = x, Y_t^{(1)} = y|\mathbf{o},\theta^*]Pr[\mathbf{o}|\theta^*]\end{aligned} \tag{4.60}$$

for $t = 0, 1, ..., T$, $x \in \Omega_x$ and $y \in \Omega_y$. Also we get the following

$$\begin{aligned}\sum_{y\in\Omega_y}\sum_{\mathbf{x}\in\Omega_{\mathbb{X}|X_t=x}}\sum_{\mathbf{y}\in\Omega_{\mathbb{Y}|Y_t^{(s)}=y}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*) &= Pr[X_t = x, \mathbf{o}|\theta^*] \\ &= Pr[X_t = x|\mathbf{o},\theta^*]Pr[\mathbf{o}|\theta^*]\end{aligned} \tag{4.61}$$

for $t = 0, 1, ..., T$ and $x \in \Omega_y$. Therefore, we get the following result for $\hat{c}_{xy}$,

$$\hat{c}_{xy} = \frac{\sum_{t=0}^{T} Pr[X_t = x, Y_t^{(1)} = y|\mathbf{o},\theta^*]}{\sum_{t=0}^{T} Pr[X_t = x|\mathbf{o},\theta^*]} \tag{4.62}$$

for $x \in \Omega_x$ and $y \in \Omega_y$ at each time $t = 0, 1, ..., T$. By differentiating $\mathcal{L}$ with

respect to $e_y(o)$, for each $y \in \Omega_y$, we get

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial e_y(o)} =& \frac{\partial Q(\theta|\theta^*)}{e_y(o)} - \lambda_e^y \\ =& \sum_{t=0}^{T}\sum_{s=1}^{S}\sum_{\mathbf{x}\in\Omega_{\mathbb{X}}}\sum_{\mathbf{y}\in\Omega_{\mathbb{Y}|Y_t^{(s)}=y}} \frac{\partial p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta)}{\partial e_y(o)}\frac{p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*)}{p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta)} - \lambda_e^y \\ =& \sum_{t=0}^{T}\sum_{s=1}^{S}\sum_{\mathbf{x}\in\Omega_{\mathbb{X}}}\sum_{\mathbf{y}\in\Omega_{\mathbb{Y}|Y_t^{(s)}=y}} \frac{1}{e_y(o)} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*) - \lambda_e^y.\end{aligned} \tag{4.63}$$

The optimal value $\hat{e}_y(o)$ makes the differential $\dfrac{\partial \mathcal{L}}{\partial e_y(o)}$ zero. Therefore, $\hat{e}_y(o)$ satisfies

$$\begin{aligned}&\sum_{t=0}^{T}\sum_{s=1}^{S}\sum_{\mathbf{x}\in\Omega_{\mathbb{X}}}\sum_{\mathbf{y}\in\Omega_{\mathbb{Y}|Y_t^{(s)}=y}} \mathbf{1}_{o_t^{(s)}}(o)p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*) - \lambda_y\hat{e}_y(o) = 0, \\ &\sum_{o\in\Omega_o}\hat{e}_y(o) = 1\end{aligned} \tag{4.64}$$

for all $y \in \Omega_y$ and $o \in \Omega_o$. Solving the equation (4.64) gives

$$\hat{e}_y(o) = \frac{\sum_{t=0}^{T}\sum_{s=1}^{S}\sum_{\mathbf{x}\in\Omega_{\mathbb{X}}}\sum_{\mathbf{y}\in\Omega_{\mathbb{Y}|Y_t^{(s)}=y}} \mathbf{1}_{o_t^{(s)}}(o)p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*)}{\sum_{t=0}^{T}\sum_{s=1}^{S}\sum_{\mathbf{x}\in\Omega_{\mathbb{X}}}\sum_{\mathbf{y}\in\Omega_{\mathbb{Y}|Y_t^{(s)}=y}} p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*)} \tag{4.65}$$

for each $y \in \Omega_y$ and $o \in \Omega_o$. As before, marginalization and applying Bayes' theorem gives

$$\sum_{\mathbf{x}\in\Omega_{\mathbb{X}}}\sum_{\mathbf{y}\in\Omega_{\mathbb{Y}|Y_t^{(s)}=y}} \mathbf{1}_{o_t^{(s)}}(o)p(\mathbf{x},\mathbf{y},\mathbf{o}|\theta^*) = \mathbf{1}_{o_t^{(s)}}(o)Pr[Y_t^{(s)} = y, \mathbf{o}|\theta^*] \tag{4.66}$$

for each $y \in \Omega_y$ and $o \in \Omega_o$ and at each times $t = 0, 1, ..., T$ and $s = 1, 2, ..., S$.

Therefore, we get the following equation

$$\hat{e}_y(o) = \frac{\sum_{t=0}^{T}\sum_{s=1}^{S} \mathbf{1}_{o_t^{(s)}}(o) Pr[Y_t^{(s)} = y|\mathbf{o}, \theta^*]}{\sum_{t=0}^{T}\sum_{s=1}^{S} Pr[Y_t^{(s)} = y|\mathbf{o}, \theta^*]} \tag{4.67}$$

for each $y \in \Omega_y$ and $o \in \Omega_o$. What remains is calculating the probabilities in the left-hand sides of the results. We denote the probability $Pr[X_t = x|\mathbf{o}, \theta^*]$ by $\zeta_t(x)$. For each $t = 1, 2, ..., T$, $\zeta_t(x)$ is given by

$$\begin{aligned} \zeta_t(x) =& \frac{Pr[X_t = x, \mathbf{o}|\theta^*]}{Pr[\mathbf{o}|\theta^*]} \\ =& \frac{f_{X_t}(x) b_{X_t}(x)}{\sum_{x\in\Omega_x} f_{X_t}(x) b_{X_t}(x)}, \end{aligned} \tag{4.68}$$

where $x \in \Omega_x$ and $t = 0, 1, ..., T$. Similarly, we denote the probability $Pr[Y_t^{(s)} = y|\mathbf{o}, \theta^*]$ by $\xi_t^s(y)$. For each $t = 0, 1, ..., T$ and $s = 1, 2, ..., S$, $\xi_t^s(y)$ is given by

$$\begin{aligned} \xi_t^s(y) =& \frac{Pr[Y_t^{(s)} = y, \mathbf{o}|\theta^*]}{Pr[\mathbf{o}|\theta^*]} \\ =& \frac{f_{Y_t^{(s)}}(y) b_{Y_t^{(s)}}(y)}{\sum_{y\in\Omega_y} f_{Y_t^{(s)}}(y) b_{Y_t^{(s)}}(y)}, \end{aligned} \tag{4.69}$$

where $y \in \Omega_y$. We let $\alpha_t(x, x')$ be the probability $Pr[X_t = x, X_{t+1} = x'|\mathbf{o}, \theta^*]$. Then $\alpha_t(x, x')$ is given by

$$\begin{aligned} \alpha_t(x, x') =& \frac{Pr[X_t = x, X_{t+1} = x', \mathbf{o}|\theta^*]}{Pr[\mathbf{o}|\theta^*]} \\ =& \frac{f_{X_t}(x) a^*_{xx'} b_{X_{t+1}}(x') b_{Y_{t+1}^{(0)}}(x')}{\sum_{x,x'\in\Omega_x} f_{X_t}(x) a^*_{xx'} b_{X_{t+1}}(x') b_{Y_{t+1}^{(0)}}(x')} \end{aligned} \tag{4.70}$$

for each $t = 0, 1, ..., T-1$ and $x, x' \in \Omega_x$. We denote the probability $Pr[Y_t^{(s)} =$

$y, Y_t^{(s+1)} = y'|\mathbf{o}, \theta^*]$ by $\beta_t^s(y, y')$. Then $\beta_t^s(y, y')$ is given by

$$\begin{aligned}\beta_t^s(y,y') =& \frac{Pr[Y_t^{(s)} = y, Y_t^{(s+1)} = y', \mathbf{o}|\theta^*]}{Pr[\mathbf{o}|\theta^*]} \\ =& \frac{f_{Y_t^{(s)}}(y) b^*_{yy'} b_{Y_t^{(s+1)}}(y') e^*_{y'}(e_t^{(s+1)})}{\sum_{y,y' \in \Omega_y} f_{Y_t^{(s)}}(y) b^*_{yy'} b_{Y_t^{(s+1)}}(y') e^*_{y'}(e_t^{(s+1)})}\end{aligned} \tag{4.71}$$

for each $t = 0, 1, ..., T$, $s = 1, 2, ..., S-1$ and $y, y' \in \Omega_y$. We let $\gamma_t(x, y)$ be the probability $Pr[X_t = x, Y_t^{(1)} = y|\mathbf{o}, \theta^*]$. Then $\gamma_t(x, y)$ is given by

$$\begin{aligned}\gamma_t(x,y) =& \frac{Pr[X_t = x, Y_t^{(1)} = y, \mathbf{o}|\theta^*]}{\sum_{x \in \Omega_x} \sum_{y \in \Omega_y} Pr[X_t = x, Y_t^{(1)} = y, \mathbf{o}|\theta^*]} \\ =& \frac{\sum_{x' \in \Omega_x} f_{X_{t-1}}(x') a^*_{x'x} b_{X_t}(x) c^*_{xy} b_{Y_t^{(1)}}(y) e^*_y(e_t^{(1)})}{\sum_{x \in \Omega_x} \sum_{y \in \Omega_y} \sum_{x' \in \Omega_x} f_{X_{t-1}}(x') a^*_{x'x} b_{X_t}(x) c^*_{xy} b_{Y_t^{(1)}}(y) e^*_y(e_t^{(1)})}\end{aligned} \tag{4.72}$$

for each $t = 0, 1, ..., T$, $x \in \Omega_x$ and $y \in \Omega_y$. With ζ, ξ, α, β and γ, we can calculate the updating parameters. The following proposition gives the results.

Proposition 4.2.8. *The optimizing parameter $\hat{\theta} = (\hat{A}, \hat{B}, \hat{C}, \hat{\pi}, \hat{e}_i)$ is given by*

$$\begin{aligned}\hat{a}_{xx'} &= \frac{\sum_{t=0}^{T-1} \alpha_t(x, x')}{\sum_{t=0}^{T-1} \zeta_t(x)}, \\ \hat{b}_{yy'} &= \frac{\sum_{t=0}^{T} \sum_{s=1}^{S-1} \beta_t^s(y, y')}{\sum_{t=0}^{T} \sum_{s=1}^{S-1} \xi_t^s(y)}, \\ \hat{c}_{xy} &= \frac{\sum_{t=0}^{T} \gamma_t(x, y)}{\sum_{t=0}^{T} \zeta_t(x)}, \\ \hat{\pi}_x &= \zeta_t(x) \ \textit{and} \\ \hat{e}_y(o) &= \frac{\sum_{t=0}^{T} \sum_{s=1}^{S} \mathbf{1}_{o_t^{(s)}}(o) \xi_t^s(y)}{\sum_{t=0}^{T} \sum_{s=1}^{S} \xi_t^s(y)}\end{aligned} \tag{4.73}$$

for $x, x' \in \Omega_x$, $y, y' \in \Omega_y$ and $o \in \Omega_o$.

The estimating process for a MLHMM is iterating algorithm. A pseudo algorithm for the estimation is given as follows.

Algorithm 4 Estimation of MLHMM

1: Initialize the parameter $\theta^{(0)}$ and set $t = 0$.
2: **while** $|\theta^{(t+1)} - \theta^{(t)}| > \epsilon$ **do**
3: Calculate $b_{Y_t^{(s)}}$.
4: Calculate f_{X_t} and b_{X_t}.
5: Calculate $f_{Y_t^{(s)}}$.
6: Update $a_{xx'}$, $b_{yy'}$, c_{xy}, π_x and $e_y(o)$.
7: **end while**
8: **return** $\theta^{(t)}$.

4.3 Application

4.3.1 Data Description

Our motivating example is a data set which contains every transactions in KOSPI market recorded by KOSCOM. This data set has 736 days of observation from June 2014 to May 2016. The data set was recorded every milliseconds and includes information about transaction time, traded volume, traded price, etc. Our data also contains data on over-the-counter transactions. However, we do not cover the over-the-counter transactions in this paper. We divide the time from 9:00 am to 3:00 pm into one minute. Then, we measured the close price of each minute. We take each one minute close price as an observation. So we have 350 observations per day.

4.3.2 Model Construction

For implication, we set the number of hidden states of X and Y to be 5. We let the random variable $Y_t^{(s)}$ denotes the hidden variable that modulates the observation at time s in a day t. We also let another random variable, X_t, be a variable which represents a hidden variable that modulates the initial

state of the hidden state sequence $\{Y_t^{(s)}; s = 1, 2, ..., S\}$. Roughly speaking, we consider $\{Y_t^{(s)}\}$ to be the hidden state which modules the short term behavior of stock price and $\{X_t\}$ to be the hidden state which modules the tendency of stock price movements between days. We set T to be 736 since our data has observations of 736 days. We also set S to be 350 because the regular market opens for 350 minutes.

4.3.3 Result

We conduct the learning process of MLHMM by the KOSCOM transaction data. Our work is based on the parameters resulting from this learning. Using the simulated data we present some results.

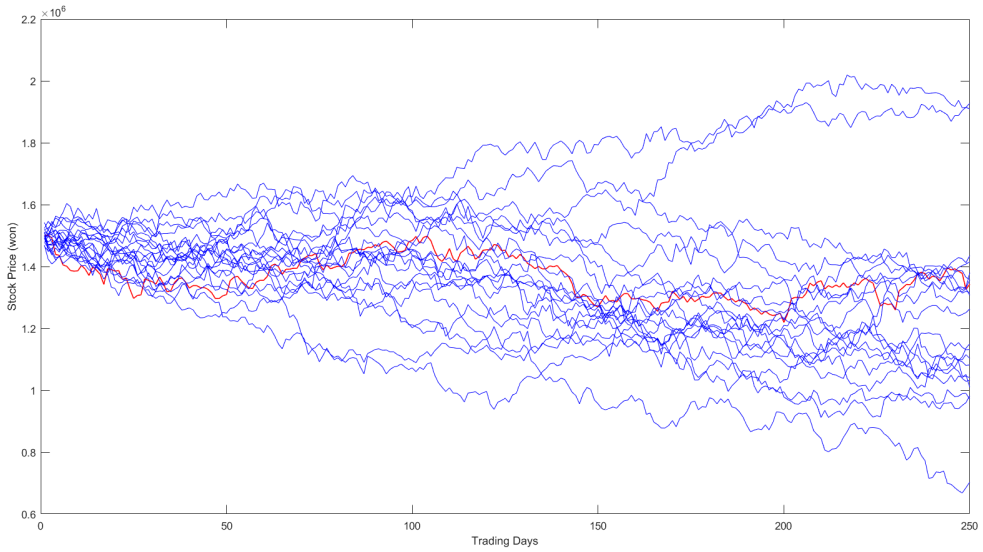

Figure 4.3: The stock price movement within 250 trading days. The red curve indicates the stock prices from real data and the other blue curves of the stock prices simulated by MLHMM

Figure 4.3 shows the stock price movement of Samsung electronics and

simulated stock price movements. In the graph, the red line represents the price of the Samsung electronics stock. The remaining blue lines represent the results simulated by MLHMM. We simulate the stock price with the parameters estimated using the Samsung electronics stock price. We can see that the simulated prices are around the actual price. This shows that our simulation data reflects the actual situation.

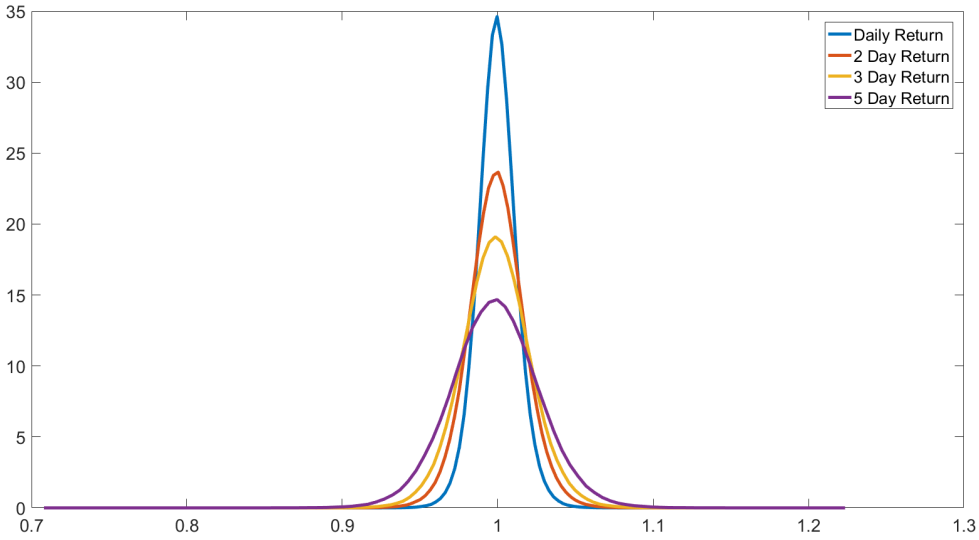

Figure 4.4: Daily, two-day, three-day, and one-week stock returns simulated. Each return was calculated by dividing the close price of the last day of the measurement period by the open price of the first day. In the case of the daily return, the first day and the last day are regarded as the same day.

Figure 4.4 shows how the distribution of the simulated returns varies as the return period varies. The longer the period, the greater the variance of the distribution of returns as we expected.

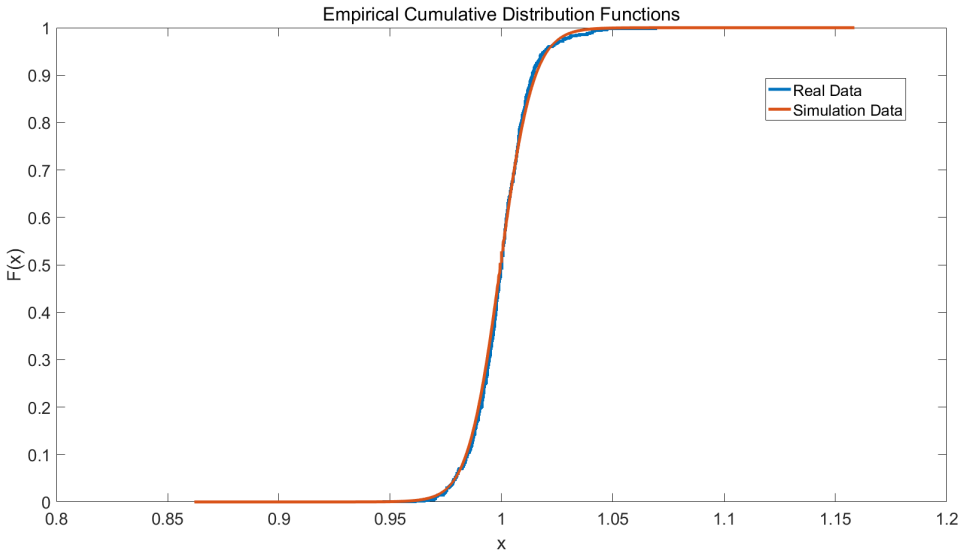

Figure 4.5: Empirical cumulative distributions of real and simulated data. The blue line represents the empirical cumulative distribution of daily returns from the real data and the red line represents the empirical cumulative distribution of the daily returns from the simulated data.

Figure 4.5 represents the empirical cumulative distributions of returns. The blue line holds for the distribution for the real data and the red one represent the distribution for the simulated data. As we can see, the blue and red lines almost coincide. This allows us to expect that simulated data and real data come from the same distribution. Based on our belief, we construct a statistical test. We set our testing hypothesis as follows

$$\begin{aligned} \mathcal{H}_0 \quad &: \quad \textit{The real and simulated data come from the same distribution} \\ \mathcal{H}_1 \quad &: \quad \textit{not } \mathcal{H}_0 \end{aligned}$$

We conduct two-sample Kolmogorov-Smirnov test with $\alpha = 0.05$. The test accept the null-hypothesis with p-value 0.128. This is what we expected. From these results, we can conclude that our simulated data well reflects the real market circumstances.

4.4 Conclusion

This concludes the chapters we presented until now. This thesis introduces MLHMM as a flexible model for time series data. We develop an algorithm to estimate the MLHMM models. The mathematical foundation of the estimating algorithm guarantees the convergence and the robustness of our estimation algorithm.

Chapter 5

Recurrent Neural Network

When analyzing time series data, not only information at a certain point in time but also information about data at an earlier point are important. Conventional neural networks are difficult to apply to this time series analysis. A RNN cell is a special type of neural network architecture. The RNN solved the problem described above by adding hidden state. In RNN architecture, the hidden state plays a role of storing the information that is previously published. In this chapter, we introduce a basic RNN network and advanced RNN architecture.

5.1 Neural Networks

In this section, we introduce knowledges about basic neural networks. Warren McCulloch and Walter Pitts[10] A description for a basic neural network is given in Figure 5.1. The input data $x = (x_1, ..., x_n) \in \mathbb{R}^n$ is a n-dimensional real vector. The output vector, $y = (y_1, ..., y_m)$, is a m-dimensional vector. Each element y_i is computed by

$$y_i = f(w_i^t x + b_i), \tag{5.1}$$

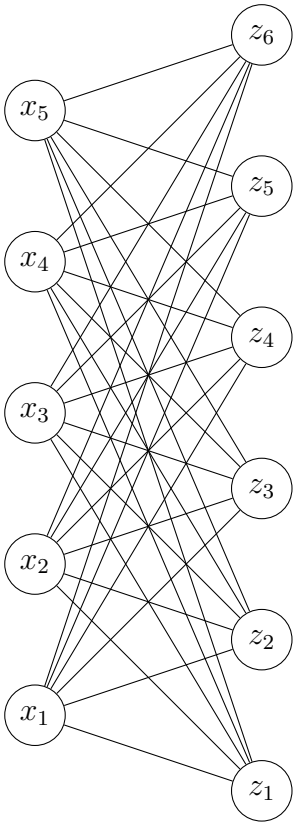

Figure 5.1: Description of a basic nueral network

where w is a n-dimensional vector and b_i is a constant. Abbreviating the results, we can get

$$y = f(Wx + b), \tag{5.2}$$

where W is a $m \times n$ matrix and b is m-dimensional vector. We call the f in equation 5.1 an activation function. There are various activations functions. We give some examples of the activations functions. A frequently used activation functions is the sigmoid function. In a wide range, the sigmoid function refers to a function which have S-shaped curve. We often call the logistic function a sigmoid function. The logistic function is given by

$$f(x) = \frac{1}{1 + e^{-x}} \tag{5.3}$$

for $x \in \mathbb{R}$. Without any notion, the sigmoid function implies a logistic function. The sigmoid function is denoted by $\sigma(x)$. The sigmoid function has range $(0, 1)$. Because of the range of sigmoid function, the sigmoid function give a probabilistic interpretation. We often use the hyperbolic tangent function as an activation function. The hyperbolic function is given below:

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \tag{5.4}$$

The hyperbolic tangent function has range $(-1, 1)$. Different from the sigmoid function, the hyperbolic tangent function can not give probabilistic interpretation. Although the hyperbolic tangent function is widely used because of its performance. The activations functions like sigmoid or hyperbolic tangent have a problem. The outputs go to zero when we apply the activation functions many times. It is because the range are restricted to $(-1, 1)$. The rectified linear unit (ReLU) is introduced to solve this problem. The ReLU function has the form

$$f(x) = \max(0, x). \tag{5.5}$$

Since the differential of ReLU function is 1 the differential does not goes to zero after applying the activations many times. Besides, there are various

activations functions. Variations of ReLU such as exponential linear unit (ELU)[4] and and S-shaped rectified linear units (S-ReLUs)[19].

To optimize loss function, we need to compute gradients of the loss function with respect to the weights of a model. The backpropagation algorithm (Rumelhart et al.[11]) is a method of computing the gradient of a loss function. The main idea is to apply the chain rule to the differential of loss function. The backpropagation algorithm make the computation of the gradients of stacked neural networks possible.

5.2 Recurrent Neural Networks

In this section, we give an introductory RNN models. A basic structure of RNN is given in Figure 5.2. A basic RNN cell needs hidden memory state and current state input. The current input data, combined with hidden state memory, produces new hidden state memory (and an output if necessary). Recurrently, the new hidden state is used to generate the next step hidden state. The hidden state at step t is given using the hidden state at prior step $t-1$ and the input at t.

$$h_t = f(W \cdot [h_{t-1}, x_t] + b), \tag{5.6}$$

where $\cdot$ represent the usual matrix multiplication and $[h_{t-1}, x_t]$ is a concatenated vector. The hidden state h_t is given to the next step RNN cell and goes through the same calculation. Usually in RNN cell, the sigmoid function is used as an activation function.

The main problem arises in RNN architecture is that RNN can not store the memory for long time. This problem is caused by so called vanishing gradient. The vanishing gradient problem was explored by Bengio, et al[2]. The vanishing gradient is caused by the activation function in a RNN cell. Activation functions like sigmoid and hyperbolic tangent map real value to $(-1, 1)$. As the time step increases, more activation functions are applied to

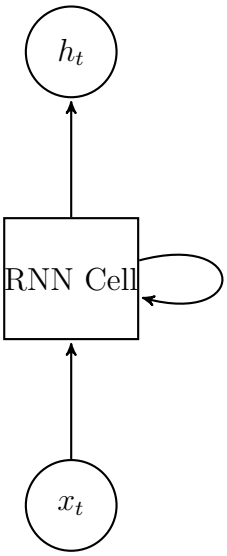

Figure 5.2: Description for the RNN architecture

the hidden state in RNN cells. This make the hidden state be zero.

Hochreiter and Schmidhuber [7] solved this problem by introducing long-short term memory (LSTM). They added so called memory cell to store long term memory cell (simply, memory cell). The memory cell allows the RNN network to store long term memory.

Let us denote the hidden state and memory cell at time step t by h_t and C_t respectively. The input at time t is given by x_t. A LSTM cell updates the information in the hidden state and the memory cell in following order. At first, the LSTM cell decide whether to abandon the prior information or not.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{5.7}$$

And the next step is to determine what are to be stored in long-term memory cell. Also, we decide the adequate amount of information. The computation

of this step is as follows

$$\begin{aligned} i_t =& \sigma(W_i \cdot [h_{t-1}, x_t] + b_f \\ c_t =& \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \\ C_t =& f_t * C_{t-1} + i_t * c_t. \end{aligned} \tag{5.8}$$

Now the long term memory cell is updated. Using the long-term memory, we can recognize the current state. The new hidden state is computed using the updated cell state. The new hidden state is calculated in following steps.

$$\begin{aligned} o_t =& \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\ h_t =& o_t * \tanh(C_t). \end{aligned} \tag{5.9}$$

Then the long term memory cell state and the short term hidden state are given to the next step LSTM cell and goes through the same calculation we described. There are many variants on RNN. Gers and Schmidhuber[6] suggested an extended LSTM model by adding "peephole connections". The gated recurrent unit (GRU), introduced by Cho, et al.[3] solved the vanishing gradient in a way slightly different from the method of LSTM. The GRU uses update gate rather than using forget and input gate separately. It has simpler form than the LSTM cell. Besides, like the bidirectional RNN (BRNN) invented by Schuster, et al[12], the depth gated RNN introduced by Yao,et al[20].

5.3 Conclusion

This concludes the chapter on RNN networks. The RNN network provides a way of analyzing time series data. After the invention of the RNN network, there were many variants of RNN cell to solve the problems arise in classical RNN cell. A representative variant of RNN is LSTM cell. This allows an RNN to store longer memory.
In the next section, we introduce a variant of LSTM cell and show experimental results.

Chapter 6

Unity Long Short Term Memory

The LSTM cell stores information in a hidden state and a memory cell separately. The LSTM cell stores the same information in duplicate in two different storage repositories. In this chapter, we propose a variation of LSTM architecture to prevent the problem of excessive storage of information.

6.1 Construction of Network

The separated storage of LSTM cell enables us to store long-term dependencies. However, because of the separated memory storage, the information is stored in duplicate. To avoid this problem, we need to restrict the total amount of the information. Computationally, the size of the information should be a constant. We adopt a different type of LSTM cell by introducing one way to enhance the forget gate of LSTM cell. The calculation of LSTM cell is largely divided into the following three stages.

- decoupling the information.

$$\begin{aligned} i_t^C &= W_d \cdot [h_{t-1}, x_t]' + b_d \\ i_t^h &= 1 - i_t^C \end{aligned}$$

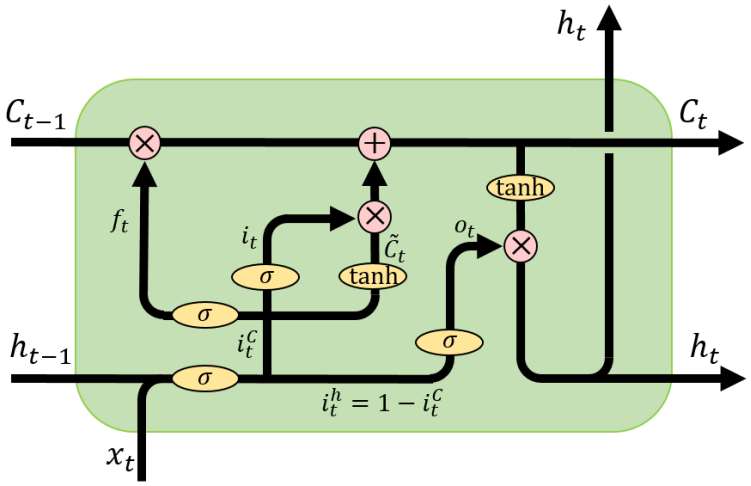

Figure 6.1: A graphical description of an ULSTM cell

- updating long term memory cell state.

$$
\begin{aligned}
f_t =& \sigma(W_f \cdot i_t^C + b_f) \\
i_t =& \sigma(W_i \cdot i_t^C + b_i) \\
\tilde{C}_t =& \tanh(W_C \cdot i_t^C + b_C) \\
C_t =& C_{t-1} * f_t + \tilde{C}_t * i_t
\end{aligned}
$$

- updating hidden state.

$$
\begin{aligned}
o_t =& \sigma(W_o \cdot i_t^h + b_o) \\
h_t =& o_t * \tanh(C_t)
\end{aligned}
$$

We add a node called decoupling node. In the node, we divide the information which is obtained from the old memories and input data into two parts. We adjust the divided parts to be 1 in addition. By adding the decoupling node, we prevent information from being stored redundantly. Figure 6.1 depicts the ULSTM architecture.

6.2 Experiment

In this section,

6.2.1 Data Description

The motivation example is a data set referred to as the *KOSCOM transaction* data. This data includes information on all transactions made in the KOSPI market in milliseconds. For each transaction, we can see 51 recorded features including transaction price, transaction time, transaction volume, and so on. This data contains information on all types of transactions in the KOSPI market, including the over the counter transactions. Among them, we only deal with regular market transactions.

Knowing the price of a stock at certain time or period does not mean we can trade at that price. We can not know the exact price that we can trade at a specific time. To solve this problem, we introduce an index so called value weight average price (VWAP). The definition of VWAP is as follows

$$VWAP = \frac{\sum Number\ of\ Shares\ Boutght \times Share\ price}{TotalSharesBought}. \tag{6.1}$$

We assume that we can always buy the stock at calculated VWAP.

6.2.2 Results

Our investment strategy is as follows. We determine it is a good time to buy stocks if you can sell them at a price 0.15% higher than the current price within 10 minutes after current time. Our goal is to predict future price movements with data on stock up to the present and classify current statues. There are two cases. One of them is the case that current time is good for buying stock. The other case indicates that our strategy can not make profit within 10 minutes.

We construct 1 layer models using different RNN cells. For all models we set the dimension of hidden state vector to be 128. We also set the maximum

time step to be 200. Each model use one of following RNN cells:

- Basic RNN cell
- GRU cell
- LSTM cell
- ULSTM cell

At each time step, a hidden state comes from each of our model. We decode the hidden state with 1 layer network. We also construct 3 layer models. The other settings are the same as the settings in 1 layer model. The only difference is that we decode the hidden state with 3 layered networks.

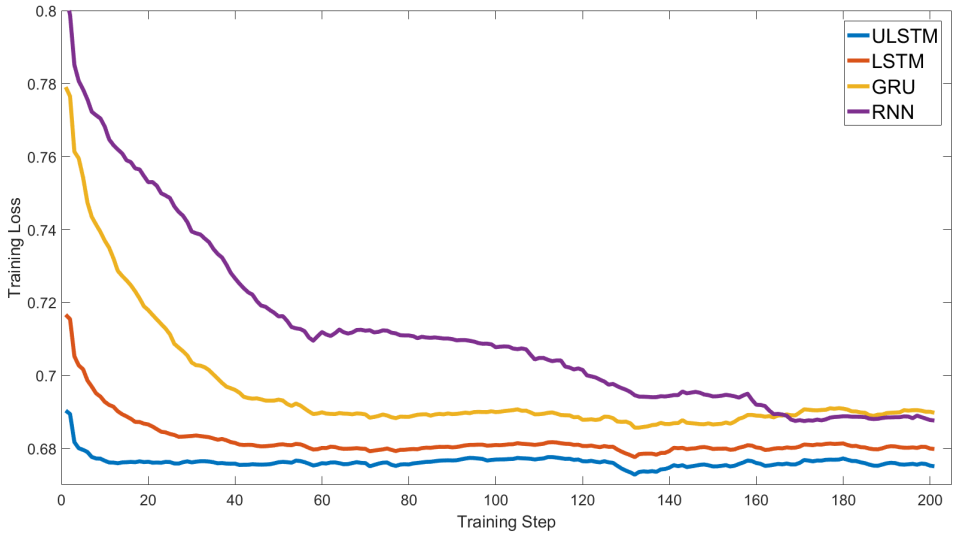

Figure 6.2: Training losses for 4 different RNN cells with 1 layer model

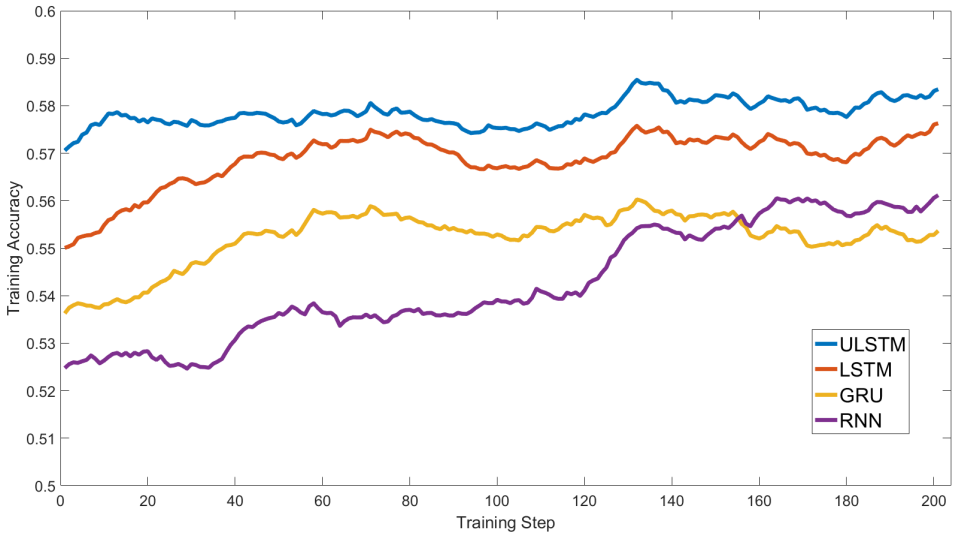

Figure 6.3: Training accuracies for 4 different RNN cells with 1 layer model

Figure 6.2 indicates the losses for the models with 1 layer structure using 4 different types of RNN cells. Each graph is averaged over 200 steps of loss. At every steps, the ULSTM network shows lower loss than any other RNN cells losses. Figure 6.3 represents the training accuracies of the models with 1 layer structure. At each step, the model using ULSTM cell shows higher accuracy than other models.

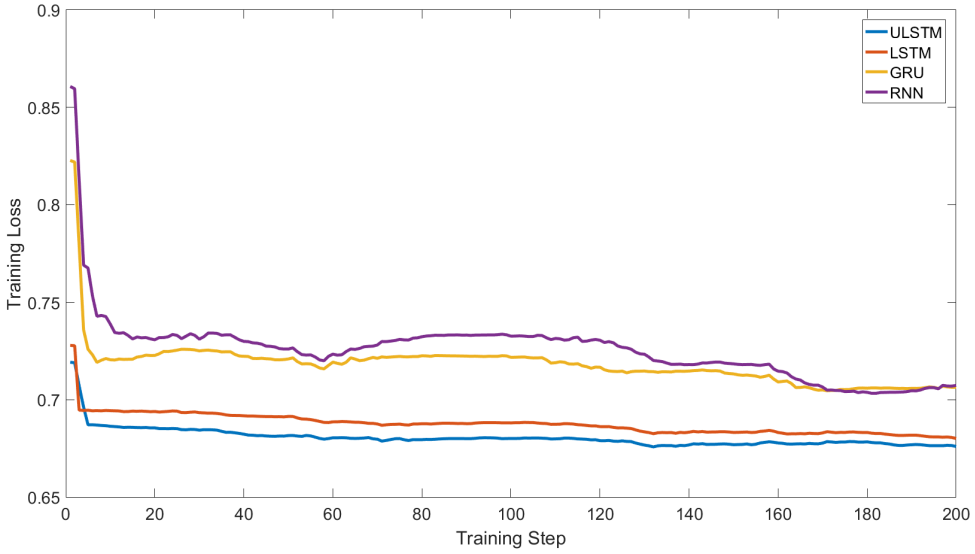

Figure 6.4: Training losses for 4 different RNN cells with 3 layer model

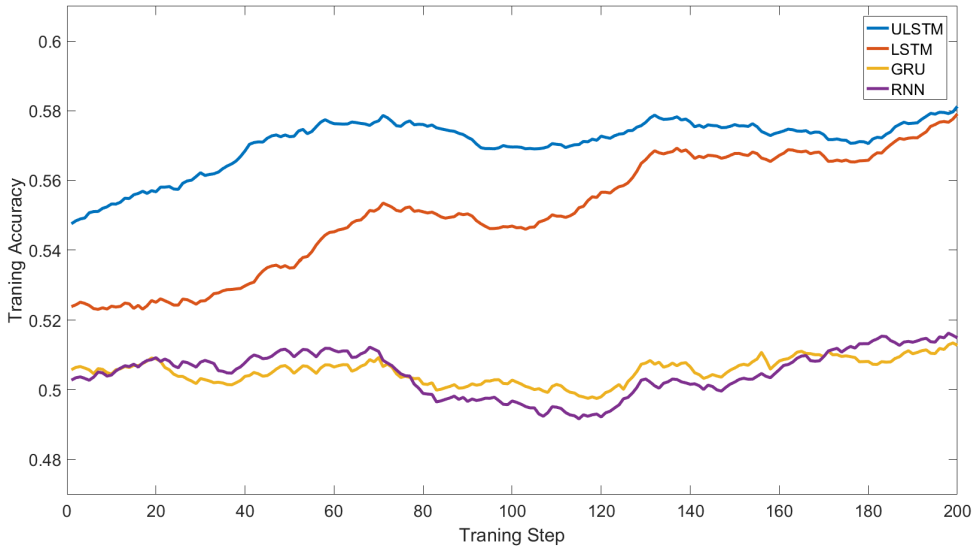

Figure 6.5: Training accuracies for 4 different RNN cells with 4 layer model

Figure 6.4 plots the losses for 4 different RNN cells with 3 layered neural network decoder. Similar to the 1 layer case, the ULSTM cell indicates lower loss than losses of other RNN cells except for first few steps.

Model (layer)	Average training accuracy (%)	Test accuracy (%)
ULSTM (1 L)	58.35	58.01
LSTM (1 L)	57.63	57.35
GRU (1 L)	55.37	54.94
Basic RNN (1 L)	56.12	55.81
ULSTM (3 L)	57.29	56.98
LSTM (3 L)	56.72	56.40
GRU (3 L)	54.73	54.06
Basic RNN (3 L)	54.31	53.88

Table 6.1: Training and test accuracies of models trained using KOSCOM Transaction data. The training and the test accuracies are presented in percent.

Table 6.1 shows the training and test accuracies of each model. We can see that among the models with the same number of layers, models using ULSTM cell have the highest accuracies. For 1 layer models, ULSTM cell show 58.01% accuracy. This result is 0.66% higher than the same layered model using LSTM cell. In 3 layer case, the ULSTM cell show better result than the LSTM cell which is best among the RNN cell types except the ULSTM cell.

6.3 Conclusion

This concludes the chapter on the ULSTM cell. In this chapter, we introduce the ULSTM cell to prevent duplicate storage of information. By adding decoupling node, we let the information not be duplicated. We also implemented experiments that uses various types of RNN cells including ULSTM. In the experiments, we used the KOSCOM data which includes every

transactions in KOSPI market. We gathered only the values which can be observed in real time in the market and reconstructed the data.

The result shows the effectiveness of the ULSTM cell compared to the other RNN cell types in analyzing financial data. We can assume that the ULSTM cell give an effective way of using RNN architecture.

Chapter 7

Conclusion

In this thesis, we present various methods analyzing time series data. Different from other data, the relations between data points have some relations among them. Clarifying the relations is important in analyzing time series data.

The goal of HMMs is to discover the relations using transition probabilities between the hidden state. Chapter 2 gives basic knowledges about the Markov chains. The explained properties are used to construct the HMM architecture. We present certain conditions which make the HMMs be implementable in practice. Under the settings made in Chapter 2, we introduce a basic HMM, in Chapter 3.

Chapter 4 introduces our new model, MLHMM. By using multiple hidden state sequences, the MLHMM analyzes the long term dependencies of the time series. The new hidden state sequence, which modulates the long term dependency, affects the entire process by adjusting the initial state of the short term hidden state sequence. As we add several hidden states, the graphical structure of our model have parts with multiple children nodes. For this reason, we can not directly apply the existing estimation algorithm to estimate this model. In this chapter, we present a new estimation method of MLHMM and provide a mathematical theory for it. In subsequent chapter, we adopt the MLHMM to analyze the real market data. The data, we used,

includes not only the close prices of stocks in KOSPI, but also includes the prices of every transactions in the day market. This allows us to use the stock price movement information. We divide a day by 1 minute and observe the closing price of each period. We estimated the MLHMM with the data. Using the estimated parameters, we generated simulation data sets. The data sets show us that the MLHMM behaves as the real stock prices do. Several statistical tests confirm this assertion.

In Chapter 5, we provide basics about deep neural networks. The deep neural network (or deep learning) is a specific tool of machine learning that has seen remarkable progress recently. The activation functions of a neural network adds non-linearity to the network. There are various activation functions. Each activation function has its own characteristic when applied. The backpropagation algorithm provide a method to compute the gradient of the loss function. This algorithm makes the deep learning architecture be implementable in practice. By stacking multiple hidden layers, we compose a deep learning model. Then we present a special type of deep learning architecture. RNN was created with the primary purpose of analyzing time series data among deep learning techniques. By using hidden state to store past memories, RNN made great improvement in time series analysis. After the success of RNN architecture a remarkable variation of RNN, LSTM, was developed. LSTM has been developed with the goal of solving the problem of RNN's long-term dependency and has been successful. LSTM sets two different types of memory storages. By storing long-term memory in a long-term memory cell, LSTM solved the vanishing gradient. This well posed idea makes RNN architecture be more effective than before.

Chapter 6 present the ULSTM cells. The motivation of ULSTM cell is to divide the information not to duplicate memories. The decoupling node is added to accomplish our motivating goal. We conduct experiments using the KOSCOM data. We reconstructed the data to be used in the experiment by extracting some features from the data. The results of experiments show that ULSTM cell shows higher accuracy than any other RNN cell types. Financial time series data is a field of interest among various time series data.

The effectiveness of ULSTM network on financial time series is shown in the experiments.

In this thesis, we showed two machine learning architectures. The MLHMM provide a probabilistically interpretable method. One of main subjects we are dealing with is calculating derivatives of financial markets. We solved this by estimating our model and making simulated data. The ULSTM provides a way of predicting the financial events of future time. Predicting the future in finance is regarded as an impossible thing. As the deep learning gets remarkable results, attempts to predict financial events like stock price movements are made. Our experiments on ULSTM cell have the same goal. We introduced ULSTM as a prediction model. The results showed that the ULSTM architecture has great effect on analyzing financial time series.

In this thesis, the purpose of our model is two-fold. One is to find the appropriate distributions of financial data especially the stock return data. Predicting the future from the past data is one of our main concerns. We find that our models worked well as we expected. However, we expect that a well organized model structure can improve our model efficiency. Finding the well structured model will be the successive work of this thesis.

Bibliography

[1] Baum, L. E., Petrie, T., Soules, G., and Weiss, N. (1970). "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains", *The Annals of Mathematical Statistic*, 41, 164-171.

[2] Bengio, Y., Simard, P., and Frasconi, P. (1994). "Learning long-term dependencies with gradient descent is difficult", *IEEE Transactions on Neural Networks*, 5(2), 157-166.

[3] Cho, K., Merrienboer, B. V., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. and Bengio, Y. (2014). "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).*

[4] Clevert, D.-A., Unterthiner, T. and Hochreiter, S. (2015). "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)". *arxiv*:1511.07289.

[5] Dempster, A.P., Laird, N. M., and Rubin, D.B.(1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm", *Journal of the Royal Statistical Society*, Ser. B, 39, 1-22.

[6] Gers, F. and Schmidhuber, J. (2000). "Recurrent nets that time and count". *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium.*

[7] Hochreiter, S. and Schmidhuber, J. (1997). "Long Short-Term Memory", *Neural Computation*, 9(8), 1735-1780.

[8] Juang, B. H. and Rabiner, L. R. (1991). "Hidden Markov Models for Speech Recognition". *Technometrics*, 33(3), 251.

[9] Liu, J. S., Neuwald, A. F. and Lawrence, C. E. (1995). "Bayesian Models for Multiple Local Sequence Alignment and Gibbs Sampling Strategies". *Journal of the American Statistical Association*, 90(432), 1156.

[10] McCulloch, W. S. and Pitts, W. (1943). "A Logical Calculus of Ideas Immanent in Nervous Activity". *Bulletin of Mathematical Biophysics*, 5 (4), 115–133.

[11] Rumelhart, D. E., Hinton, G. E. and Williams, R. J. (1986). "Learning representations by back-propagating errors". *Nature*, 323(6088), 533-536.

[12] Schuster, M. and Paliwal, K. (1997). "Bidirectional recurrent neural networks". *IEEE Transactions on Signal Processing*, 45(11), 2673-2681.

[13] Scott, S. L. (1999). "Bayesian Analysis of a Two-State Markov Modulated Poisson Process", *Journal of Computational and Graphical Statistics*, 8(3), 662.

[14] Scott, S. L. (2001). "Detecting network intrusion using a Markov modulated nonhomogeneous Poisson process". Available at www-rcf.usc.edu/-sls/research.html.

[15] Scott, S. L. (2002). "Bayesian Methods for Hidden Markov Models". *Journal of the American Statistical Association*, 97(457), 337-351.

[16] Socher, R. (2014) "Recursive Deep Learning for Natural Language Processing and Computer Vision". Phd thesis, Stanford University.

[17] Stratonovich, R.L.(1960). "Conditional Markov Processes", *Theory of Probability and its Applications*, 5, 156-178.

BIBLIOGRAPHY

[18] Viterbi, A. J. (1967). "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm", *IEEE Trans. Inform. Theory*, vol. IT–13, pp. 260–269

[19] Xiaojie Jin, Chunyan Xu, Jiashi Feng, Yunchao Wei, Junjun Xiong and Shuicheng Yan (2015). "Deep Learning with S-shaped Rectified Linear Activation Units". *arxiv*:1512.07030.

[20] Yao, K., Cohn, T., Vylomova, K., Duh, K. and Dyer, C. (2015). "Depth-gated recurrent neural networks". *arXiv preprint*.

국문초록

금융 데이터는 시계열 데이터의 대표적인 예이다. 시계열 데이터에서는, 다른 데이터 타입과는 다르게, 다른 시점의 관측치가 현재의 관측치를 해석하는데 주요하게 작용한다. 오랫동안 시계열 데이터는 고전적인 방법론으로 연구되어왔다. 우리는 이 논문에서 여러 시계열 데이터 중에서 금융 시계열 데이터를 분석하는 방법을 제시한다. 몇가지 실험들은 우리가 제시할 머신러닝 모델들의 효과를 입증할 것이다. 이 논문은 고전적인 방법의 머신런닝 기법을 다룰 뿐만 아니라 최근에 활발히 연구되는 머신러닝 기법들 또한 다룰 것이다.

시계열 데이터는 머신러닝의 중요한 주제 중에 하나이다. 기존의 방법들과 비교해서 머신 러닝 기법들은 시계열 데이터를 분석하는 데 뛰어난 효과를 보여왔다. 우리는 몇가지 머신러닝에서 기본적으로 쓰이는 몇가지 시계열 데이터 분석방법에 대해서 설명할 것이다. 또한 우리는 좀 더 발전될 모델을 제시할 것이다. 그 중에 한가지는 마코브 체인을 이용하는 모델이다. 챕터 2는 마코브 체인에 대한 기본적인 지식을 제공할 것이다. 챕터 3에는 기존의 모델을 설명할 것이다. 이어지는 챕터에서는 우리가 만들어낸 새로운 모델이 소개될 것이다. 이 챕터에는 실험적인 결과들도 포함될 것이다.

이 논문의 두번째 파트는 딥러닝 기술을 설명하는 것부터 시작한다. 챕터 5는 딥러닝의 기본적인 용어들과 딥러닝 분야의 특정한 모델에 대한 설명이 들어있다. 이 쳅터에서 제시되는 모델은 딥러닝에서 시계열 데이터를 다루는 데 주로 사용되는 방법이다. 쳅터 6에서 우리는 이전 챕터에서 소개된 모델들을 바탕으로 확장된 버전의 모델을 제시한다. 이 챕터에서 우리는 기존에 제시된 모델과 우리의 모델을 비교해보는 실험을 진행한다.

주요어휘**:** 히든 마코브 모델, 장단기 기억 네트워크

학번**:** 2012-23027