UNIVERSITÀ DEGLI STUDI DI PADOVA

UNIVERSITY OF PADUA

BACHELOR'S THESIS

An overview of Hidden Markov Models

Author:
Amirreza Soltani

Supervisor:
Lorenzo Finesso

A thesis submitted in fulfilment of the requirements for the degree of Information Engineering

in the

Department of Information Engineering (DEI)

September 2024

Declaration of Authorship

Candidate's declarations:

I, Amirreza Soltani, hereby certify that this thesis submitted in partial fulfilment of the requirements for the award of Information Engineering, University of Padua, is wholly my own work unless otherwise referenced or acknowledged. This work has not been submitted for any other qualification at any other academic institution.

Signed:

Date:

Supervisor's declaration:

I, Lorenzo Finesso, hereby certify that the candidate has fulfilled the conditions of the Resolution and Regulations appropriate for the degree of Information Engineering at University of Padua and that the candidate is qualified to submit this thesis in application for that degree.

Signed:

Date:

Certificate of Approval:

I certify that this is a true and accurate version of the thesis approved by the examiners, and that all relevant ordinance regulations have been fulfilled.

Supervisor:

Date:

UNIVERSITY OF PADUA

Department of Information Engineering (DEI)

Abstract

Information Engineering

An overview of Hidden Markov Models

by Amirreza Soltani

Hidden Markov Models (HMM's) are mathematical models of uncertain phenomena, well suited to describe complex dynamical behaviours. HMMs find applications in a variety of fields in the areas of digital signal processing, control, and pattern recognition. Popular as they have become, HMM's still offer a wealth of challenging theoretical problems. HMM's are an ongoing research topic in engineering, statistics, and probability.

The simplest probabilistic models of uncertain phenomena are independent processes. The basic feature of these models is their lack of memory: the present value of the process is not influenced by its past or future values. Independent processes are adequate to represent the randomness of games of chance or other simple physical phenomena, but are of limited use in describing true dynamic behaviors. One step above independent processes, in the hierarchy of probabilistic models, one finds Markov chain (MC) models. The present value of a MC depends on a finite (bounded) number of its most recent past values. If their memory size is correctly adjusted, MC's can be used to describe approximately phenomena with complex dynamical behaviour. Hidden Markov Models are at the top of this hierarchy. Roughly speaking, HMM's are processes that can be represented as functions of MC's with a finite number of states: if X(n) is a (finitely valued) MC and h(.) a given function, then Y(n) = h(X(n)) is an HMM.

In a HMM Y(n), in general the transformation h(.) destroys the Markov property of the process X(n), and as a result the HMM process Y(n)=h(X(n)) can exhibit infinite memory. That explains why HMMs are so popular: they are very simple to describe, as functions of MCs, and at the same time they can

capture complex dynamical behaviors thanks to their infinite memory.

The most basic probabilistic question is the characterization of HMM's. Is it possible to decide whether a process Y(t) is an HMM, knowing all of its finite dimensional distribution functions? The positive answer to this question was provided, independently, by Heller and by Furstenberg in the early sixties and later rederived, within the framework of System Theory, by Picci in 1976. Another probabilistic problem is the realization of HMM's, i.e. given the probabilistic description of the HMM Y(n), construct the parameters X(t) and h(.), such that Y(n)=h(X(n)). This problem has very recently been generalized to the construction of an approximate realization where, given Y(n) one constructs h(.) and X(n) such that the information theoretical criterion D(Y(n) | | h(X(n)) (relative entropy) is minimized.

Statistical inference problems for HMM's have also been intensively studied since the early sixties, when Baum and his coworkers presented two related algorithms (the Forward-Backward and a form of what is today known as the EM algorithm) that helped solve the Maximum Likelihood parameter estimation problem in the case of finitely valued HMM's.

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my supervisor, Lorenzo Finesso, for the continuous support of my Bachelor's thesis, for his patience, motivation, and immense knowledge. His guidance helped me in the duration of my research and writing of this thesis. I could not have imagined a better advisor and mentor for my study.

I would also like to thank the Department of Information Engineeering at University of Padua for providing me with the resources and facilities necessary to complete this bachelor degree.

I am also grateful to my thesis committee members, for their insightful comments and encouragements, but also for the hard questions which incented me to widen my research from various perspectives.

I thank my friends and classmates, for the stimulating discussions, for the sleepless nights we were working together before the exams and deadlines, and for all the fun we had in the last three years.

Last but not least, I would like to thank my family, for supporting me spiritually throughout these three years and my life in general.

This accomplishment would not have been possible without them.

Thank you.

Contents

List of Figures

List of Tables

List of Abbreviations

HMM	**H**idden **M**arkov **M**odel
POS	**P**art **Of** **S**peech
NLP	**N**atural **L**anguage **P**rocessing
TTS	**T**ext **To** **S**peech
ML	**M**aximum **L**ikelihood
E-M Algorithm	**E**xpectation-**M**aximization Algorithm
QMS	Observation probabilities
S	**S**tart State
E	**E**nd State
N	**N**oun
V	**V**erb
M	**M**odel

List of Symbols

S	Set of hidden states in the HMM
O	Set of observations in the HMM
π	Initial probability distribution over states
A	State transition probability matrix
B	Observation probability matrix
$P(O \mid \lambda)$	Probability of observation sequence given the model
$\alpha_t(i)$	Forward probability at time t in state i
$\beta_t(i)$	Backward probability at time t in state i
$\delta_t(i)$	Probability of the most probable state sequence ending in state i at time t
$\psi_t(i)$	Backpointer to the most probable state at time t-1 for state i at time t
$\gamma_t(i)$	Probability of being in state i at time t given the observation sequence
$\xi_t(i, j)$	Probability of being in state i at time t and state j at time t+1
T	Length of the observation sequence
N	Number of states in the HMM
M	Number of distinct observation symbols
θ	Parameters of the HMM
λ	Set of all HMM parameters (A, B, and π)
$P(O \mid S)$	Emission probability
a_{ij}	Probability of transitioning from state i to state j
$b_j(O_t)$	Probability of observing O_t given state j

Dedicated to . . .

Chapter I

Introduction

In the study of dynamic systems and sequential data, Hidden Markov Models (HMMs) have emerged as a powerful tool for modeling and analysis. Originally developed in the 1960s, HMMs have found widespread application in various fields such as speech recognition, bioinformatics, finance, and more. Their ability to represent systems with hidden states and probabilistic transitions makes them particularly suited for complex tasks where direct observation of all system components is not possible.(Rabiner (1989))

HMMs are characterized by a set of hidden states, each associated with a probability distribution. Transitions between these states are governed by a set of probabilities, and observations are generated according to the probability distributions of the current hidden state. This dual structure allows HMMs to capture both the observed data and the underlying state sequences, making them a versatile and robust modeling framework.(Rabiner (1989))

The significance of HMMs lies in their flexibility and the richness of the models they can produce. From decoding genetic sequences to predicting stock market trends, HMMs have demonstrated their capability to handle a wide range of applications, each with its own set of complexities and requirements.(Rabiner (1989))

This thesis aims to provide a comprehensive overview of Hidden Markov Models, exploring their theoretical foundations, key algorithms, and practical applications. By delving into the details of HMMs, this work seeks to highlight their importance and utility in modern data analysis and decision-making processes.(Rabiner (1989))

1.1 Background and Motivation

1.1.1 Historical Context

The development of Hidden Markov Models (HMMs) dates back to the 1970s, with significant contributions from Leonard E. Baum and his colleagues at the Institute for Defense Analyses. Initially, HMMs were introduced as a mathematical model for a specific class of stochastic processes. Baum's pioneering work laid the foundation for the theoretical underpinnings of HMMs, including the Forward-Backward algorithm, the Viterbi algorithm, and the Baum-Welch algorithm. Prior to this, in the 1960s, early work on Markov chains was conducted by other researchers, setting the stage for the later development of HMMs.(Baum and Petrie (1970)

In the early stages, HMMs were primarily theoretical constructs with limited practical applications. However, the potential of HMMs to model sequences with hidden states soon attracted interest from various research fields. One of the first significant applications was in speech recognition. Researchers at IBM in the 1970s and 1980s demonstrated that HMMs could effectively model the temporal variability of speech signals, leading to substantial improvements in speech recognition systems.(Baker (1975))

The adoption of HMMs in speech recognition marked a pivotal moment, showcasing their practical utility and sparking further research into their capabilities. As computational power increased and algorithms were refined, HMMs found applications in other domains. In bioinformatics, HMMs became invaluable for sequence alignment and gene prediction, helping to decode the complexities of genetic information. In finance, HMMs were employed to model market behaviors and predict stock prices, providing insights into economic trends.(Baker (1975))

Throughout the 1990s and 2000s, advancements in machine learning and artificial intelligence further enhanced the capabilities of HMMs. Researchers developed more sophisticated versions of HMMs, such as Hierarchical HMMs and Coupled HMMs, expanding their applicability to more complex systems. These innovations addressed some of the limitations of traditional HMMs, such as handling multiple interacting sequences and incorporating hierarchical structures.(Fine *et al.* (1998))

Today, HMMs continue to be a vital tool in various fields, benefiting from ongoing research and technological advancements. They have evolved from a theoretical concept to a practical framework widely used in industry and academia. The historical development of HMMs underscores their enduring relevance and the continuous efforts to improve their performance and extend their applications.(Rabiner (1989))

1.1.2 Importance of HMMs

The importance of Hidden Markov Models (HMMs) in modern data analysis and modeling cannot be overstated. Their ability to capture and represent complex sequential data with hidden states has made them an essential tool in a variety of fields. Here are some key reasons why HMMs are so significant:(Rabiner (1989))

1.Versatility in Modeling Sequential Data

HMMs are uniquely suited for modeling time series data where the system's underlying states are not directly observable. This makes them ideal for applications where the data sequence depends on hidden variables, such as in speech recognition, handwriting recognition, and biological sequence analysis.(Jurafsky and Martin (2009))

2. Robustness in Handling Noise and Uncertainty

One of the strengths of HMMs is their probabilistic nature, which allows them to handle noise and uncertainty effectively. By modeling the system as a stochastic process, HMMs can make robust predictions even in the presence of noisy or incomplete data.(Fine *et al.* (1998))

3. Powerful Predictive Capabilities

HMMs excel in predicting future states of a system based on observed data. This predictive power is leveraged in various applications, from forecasting stock prices in finance to predicting weather patterns in meteorology.(Hassan and Nath (2009))

4. Efficient Algorithms for Inference and Learning

The development of efficient algorithms, such as the Forward-Backward algorithm, the Viterbi algorithm, and the Baum-Welch algorithm, has made HMMs computationally feasible for real-world applications.These algorithms enable effective inference and learning of the model parameters from observed data, making HMMs practical for a wide range of tasks.(Baum and Petrie (1970))

5. Applications Across Diverse Domains

The applicability of HMMs spans numerous fields:

- **Speech Recognition**: HMMs have revolutionized automatic speech recognition systems by modeling the temporal variability in speech signals.
- **Bioinformatics**: In genomics, HMMs are used for gene prediction, sequence alignment, and identifying functional regions in DNA sequences.
- **Finance**: HMMs help in modeling and predicting financial markets, providing valuable insights for investment strategies.
- **Natural Language Processing**: HMMs are employed in tasks such as part-of-speech tagging, language modeling, and machine translation.
- **Robotics and Control Systems**: HMMs aid in the navigation and decision-making processes of autonomous systems.(Rabiner (1989))

6. Ongoing Research and Innovation

HMMs continue to be an active area of research, with ongoing efforts to enhance their capabilities and address existing limitations. Innovations such as hierarchical HMMs, coupled HMMs, and deep learning-based extensions are expanding the frontiers of what HMMs can achieve.(Ghahramani and Jordan (1997))

In summary, the importance of HMMs lies in their versatility, robustness, and powerful modeling capabilities, making them indispensable in both theoretical research and practical applications. Their ability to model hidden states in sequential data provides a framework for understanding and predicting complex systems, solidifying their role as a cornerstone in the field of probabilistic modeling.

1.1.3 Current Research Trends

The field of Hidden Markov Models (HMMs) is vibrant and continues to evolve, driven by advances in computational power, algorithmic innovation, and the expanding scope of applications. Current research trends in

HMMs focus on enhancing their efficiency, improving accuracy, and integrating them with other advanced machine learning techniques. Here are some of the prominent research directions:

1. Scalability and Computational Efficiency

As datasets grow in size and complexity, scalability becomes a critical factor. Researchers are developing more efficient algorithms and computational techniques to handle large-scale HMMs. Parallel processing, distributed computing, and optimization of the Baum-Welch and Viterbi algorithms are some of the key areas of focus. These advancements aim to make HMMs more feasible for real-time applications and large-scale data analysis.(Gales and Young (2007))

2. Hybrid Models

Combining HMMs with other machine learning models and techniques is a growing trend. Hybrid models, such as integrating HMMs with neural networks, have shown promise in improving performance. For example, combining HMMs with deep learning architectures allows for more nuanced feature extraction and state representation, enhancing the model's ability to handle complex data patterns.(Mohamed *et al.* (2011))

3. Deep Learning Integration

The integration of HMMs with deep learning methods has opened new avenues for research. Deep Hidden Markov Models (DHMMs) leverage the strengths of both HMMs and deep learning, allowing for more powerful sequence modeling. Techniques such as Long Short-Term Memory (LSTM) networks and convolutional neural networks (CNNs) are being used in conjunction with HMMs to capture long-range dependencies and spatial features in sequential data.(Graves *et al.* (2013))

4. Advances in Algorithmic Development

There is ongoing work to refine and develop new algorithms that improve the performance of HMMs. Innovations such as variational inference methods, approximate Bayesian computation, and improved expectation-maximization techniques are being explored. These advancements aim to enhance the accuracy and efficiency of parameter estimation and state inference in HMMs.(Blei *et al.* (2003))

5. Application-Specific Enhancements

Research is also focusing on tailoring HMMs to specific applications to improve their performance. For instance, in bioinformatics, efforts are being made to adapt HMMs for more accurate gene prediction and protein structure modeling. In finance, HMMs are being refined to better capture market volatility and investor behavior. Each application area drives specific enhancements and adaptations of the HMM framework.(Eddy (1998))

6. Robustness and Adaptability

Enhancing the robustness and adaptability of HMMs to changing environments and data variability is another critical area of research. Techniques such as adaptive HMMs, which can adjust their parameters in response to new data, and robust HMMs, designed to handle outliers and noise, are being developed. These models aim to maintain high performance even in dynamic and uncertain environments.(Bishop (2006))

7. Multimodal and Multisensor Fusion

Combining data from multiple sensors or modalities using HMMs is an emerging trend. Multimodal HMMs integrate information from various sources, such as audio and visual data, to improve the overall model's performance. This approach is particularly useful in fields like robotics, autonomous systems, and human-computer interaction, where multiple data streams need to be analyzed simultaneously.(Atrey *et al.* (2010))

8. Interpretability and Explainability

With the growing use of HMMs in critical applications, there is a push towards making these models more interpretable and explainable. Researchers are developing methods to provide insights into the decision-making processes of HMMs, helping users understand how and why certain predictions are made. This trend is crucial for applications in healthcare, finance, and other domains where transparency is essential.(Lipton (2018))

In conclusion, the current research trends in HMMs reflect a dynamic and innovative field. By addressing scalability, integrating with other machine learning techniques, refining algorithms, and enhancing application-specific performance, researchers are continually pushing the boundaries of what HMMs can achieve. These efforts ensure that HMMs remain a relevant and powerful tool in the ever-evolving landscape of data science and artificial intelligence.

1.2 Objectives of the Thesis

1.2.1 Primary Objectives

The primary objectives of this thesis are to explore the theoretical foundations of Hidden Markov Models (HMMs), investigate their practical applications across various domains, and address current challenges in the field. By achieving these objectives, this thesis aims to provide a comprehensive understanding of HMMs and demonstrate their significance in modern data analysis and modeling.

1. Understanding the Theoretical Foundations

- **Objective**: To delve into the core principles and mathematical underpinnings of HMMs.
- **Approach**: Reviewing key literature, including foundational papers and advancements, covering the basic structure of HMMs, the Forward-Backward algorithm, the Viterbi algorithm, and the Baum-Welch algorithm.

2. Investigating Practical Applications

- **Objective**: To explore and document the various applications of HMMs in fields such as speech recognition, bioinformatics, and finance.
- **Approach**: Case studies and reviews of existing applications, highlighting how HMMs are utilized to solve specific problems in different fields.

3. Addressing Current Challenges

- **Objective**: To identify and discuss the main challenges faced by researchers and practitioners working with HMMs.
- **Approach**: Reviewing recent research trends to identify common issues like computational complexity and parameter estimation, and proposing potential solutions and areas for future research.

1.2.2 Research Questions

This thesis seeks to address several key research questions that are fundamental to understanding and advancing the field of Hidden Markov Models (HMMs). These questions aim to explore the theoretical aspects, practical applications, and current challenges associated with HMMs.

1. What are the fundamental components and algorithms of HMMs?

This question aims to identify and explain the core elements that make up HMMs, including states, observations, transition probabilities, emission probabilities, and initial state distribution. It also covers the essential algorithms like the Forward-Backward algorithm, the Viterbi algorithm, and the Baum-Welch algorithm.

2. How do HMMs effectively model hidden states and sequential data?

This question focuses on understanding the mechanisms by which HMMs capture and represent hidden states in sequential data. It explores the process of modeling time-dependent data where underlying states are not directly observable, and how HMMs manage to infer these states from observable sequences.

3. What are the practical applications of HMMs in various fields?

This question aims to highlight the diverse applications of HMMs across different domains. It seeks to provide examples of how HMMs are applied in fields such as speech recognition, bioinformatics, finance, and more, demonstrating their versatility and utility in solving real-world problems.

4. What are the current challenges and limitations in the use of HMMs?

This question addresses the difficulties and limitations encountered when working with HMMs. It involves identifying issues such as computational complexity, parameter estimation, and adaptability to large datasets, and discussing how these challenges impact the effectiveness and efficiency of HMMs

By addressing these research questions, this thesis aims to provide a thorough exploration of HMMs, from their theoretical foundations to their practical applications and future prospects. This comprehensive approach ensures a well-rounded understanding of both the strengths and limitations of Hidden Markov Models.

1.3 Structure of Thesis

1.3.1 Chapter Overview

This thesis is organized into five chapters, each focusing on different aspects of Hidden Markov Models (HMMs) to provide a comprehensive understanding of their theoretical foundations, practical applications, and ongoing research challenges.

Chapter 2: Basics of Hidden Markov Models

This chapter introduces the fundamental concepts and components of HMMs, including states, observations, transition probabilities, emission probabilities, and initial state distribution. It also explains the basic structure and functionality of HMMs.

Chapter 3: Theoretical Framework

This chapter delves into the mathematical foundations of HMMs, discussing key algorithms such as the Forward-Backward algorithm, the Viterbi algorithm, and the Baum-Welch algorithm. It provides a detailed examination of how these algorithms work and their significance in the context of HMMs.

Chapter 4: Extensions and Variants of HMMs

This chapter explores various extensions and variants of traditional HMMs, including Hierarchical HMMs, Coupled HMMs, and other advanced models.

It also discusses how these variants improve upon the limitations of standard HMMs and their applications in more complex systems.

Chapter 5: Applications of Hidden Markov Models

This chapter explores the practical applications of HMMs across various fields. It presents case studies and examples from domains such as speech recognition, bioinformatics, finance, and more, demonstrating the versatility and effectiveness of HMMs in solving real-world problems.

Chapter 6: Part of Speech (POS) Tagging with HMM

This chapter focuses on the application of HMMs in Natural Language Processing, specifically in Part of Speech (POS) tagging. It covers the methodology, implementation, and optimization techniques such as the Viterbi algorithm for improving POS tagging accuracy.

Chapter 7: Conclusion

The final chapter summarizes the key findings and contributions of the thesis. It reflects on the significance of the research, discusses the implications of the results, and suggests potential directions for future research. This chapter concludes the thesis by highlighting the advancements made and identifying areas that warrant further investigation.

Chapter II

Basics of Hidden Markov Models

2.1 Introduction to Hidden Markov Models

Hidden Markov Models (HMMs) are powerful statistical tools used to model systems with observable outputs that depend on hidden internal states. The concept of HMMs extends the basic theory of Markov chains, which have been known and utilized by mathematicians and engineers for decades. While Markov chains are effective for modeling processes with observable states, HMMs allow for modeling systems where the states are not directly observable (hidden), but can only be inferred through observable outputs.(Rabiner (1989))

The basic theory of Markov chains is built on the principle that the future state of a process depends only on the present state, not on the sequence of events that preceded it. This is known as the Markov property. In an HMM, this property is extended to include hidden states, which makes HMMs particularly suitable for modeling complex systems where the underlying process is not directly visible.(Rabiner (1989))

Figure 2.1 shows the structure of a Hidden Markov Model (HMM). The hidden states (S_1, S_2, S_3) are represented as circles, and the transitions between these states are indicated by arrows labeled with transition probabilities (a_{ij}). The observable outputs (O_1, O_2, O_3) are shown as circles connected to the hidden states by arrows representing emission probabilities. This diagram helps in visualizing how the hidden states generate observable outputs and how the model transitions between states.

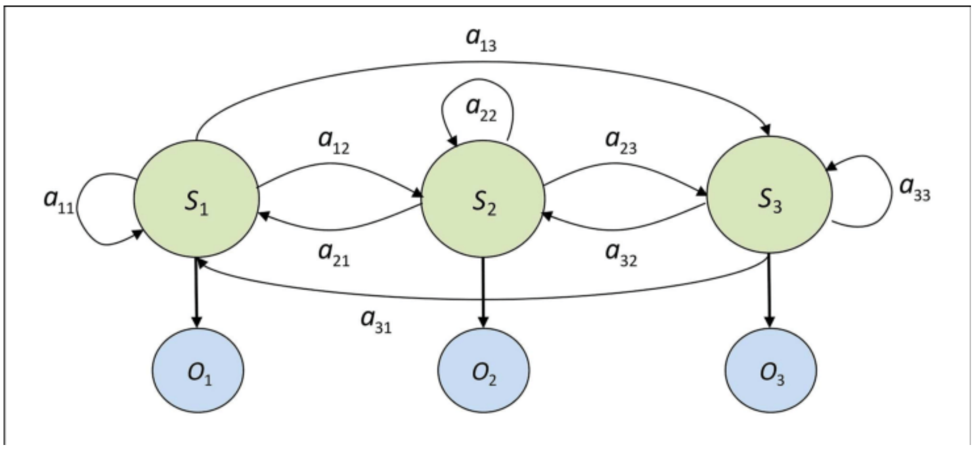

FIGURE 2.1: Structure of a Hidden Markov Model(Mariano *et al.* (2015))

2.2 Components of HMMs

To understand HMMs, it is essential to grasp the basic components and definitions:

States

States are the hidden conditions or configurations of the system. In an HMM, the states are not directly observable. Each state represents a distinct scenario or situation that the system can be in. The number of states is finite and denoted by N . The states are typically represented as $S_1, S_2, \ldots, S_N$. (Rabiner (1989))

Observations

Observations are the visible outputs generated by the hidden states. These observations form the data used to infer the hidden state sequence. Observations can be discrete symbols from a finite alphabet or continuous values. The number of possible observations is denoted by M , and they are typically represented as $O_1, O_2, \ldots, O_M$. (Rabiner (1989))

Transition Probabilities

Transition probabilities define the likelihood of transitioning from one state to another. These probabilities are represented in a matrix form, known as the state transition matrix A . Each element a_{ij} in the matrix represents the probability of transitioning from state S_i to state S_j : (Rabiner (1989))

$$A = \{a_{ij}\} \quad \text{where} \quad a_{ij} = P(S_{t+1} = S_j \mid S_t = S_i)$$

The transition probabilities must satisfy the following conditions:

$a_{ij} \geq 0$ and $\sum_{j=1}^{N} a_{ij} = 1$

Emission Probabilities

Emission probabilities describe the likelihood of observing a particular output given a specific state. These probabilities are represented in an observation probability matrix B . Each element $b_j(o)$ represents the probability of observing o when the system is in state S_j :

B = { $b_j(o)\}$ where $b_j(o) = P(O_t = o \mid S_t = S_j)$

For discrete HMMs, o is a symbol from a finite set of observations. For continuous HMMs, $b_j(o)$ is often modeled using probability density functions such as Gaussians. (Rabiner (1989))

Initial State Distribution

The initial state distribution defines the probabilities of the system starting in each possible state. This is represented by a vector π , where each element π_i represents the probability of starting in state S_i : (Rabiner (1989))

$\pi = \{\pi_i\}$ where $\pi_i = P(S_1 = S_i)$

The initial state distribution must satisfy the condition:

$\sum_{i=1}^{N} \pi_i = 1$

Example: Coin Toss Experiment (Rabiner (1989))

To illustrate these components, consider a simple coin toss experiment modeled as an HMM.

Hidden States:

- S_1 : Represents the state when the coin shows heads.
- S_2 : Represents the state when the coin shows tails.

Observations:

- O_1 : The announcement "Heads".
- O_2 : The announcement "Tails".

State Transitions:

- The probability of transitioning from heads to heads (a_{11}).
- The probability of transitioning from heads to tails (a_{12}).

• The probability of transitioning from tails to heads (a_{21}).

• The probability of transitioning from tails to tails (a_{22}).

For a fair coin:

$$a_{11} = a_{12} = a_{21} = a_{22} = 0.5$$

Emission Probabilities:

• The probability of observing "Heads" given that the hidden state is heads ($b_1(O_1)$).

• The probability of observing "Tails" given that the hidden state is heads ($b_1(O_2)$).

• The probability of observing "Heads" given that the hidden state is tails ($b_2(O_1)$).

• The probability of observing "Tails" given that the hidden state is tails ($b_2(O_2)$).

For a fair coin:

$$b_1(O_1) = b_1(O_2) = b_2(O_1) = b_2(O_2) = 0.5$$

Initial State Distribution:

• The initial state distribution vector π represents the probabilities of starting in heads or tails. For a fair coin, this might be:

$$\pi_1 = \pi_2 = 0.5$$

2.3 Types of HMMs

Hidden Markov Models (HMMs) come in various forms, each suited for different types of data and applications. Understanding these types helps in selecting the appropriate model for a given problem. Here, we explore the primary types of HMMs: Discrete HMMs, Continuous HMMs, and Hidden Semi-Markov Models (HSMMs). (Rabiner (1989))

2.3.1 Discrete HMMs

Discrete HMMs deal with discrete observations, meaning the outputs come from a finite set of symbols. These models are commonly used in applications where data can be categorized into distinct classes, such as in speech recognition, where each sound can be represented by a phoneme. (Rabiner (1989))

In a discrete HMM, observations are discrete and finite. The emission probabilities are represented by discrete probability distributions, making this type of HMM suitable for tasks like speech and language processing, where the data is naturally categorical. The observation probability matrix B consists of probabilities $b_j(o_k)$, where o_k is one of the possible discrete observations. Example applications include speech recognition, mapping sequences of sound waves to phonemes, and part-of-speech tagging, assigning parts of speech to words in a sentence.

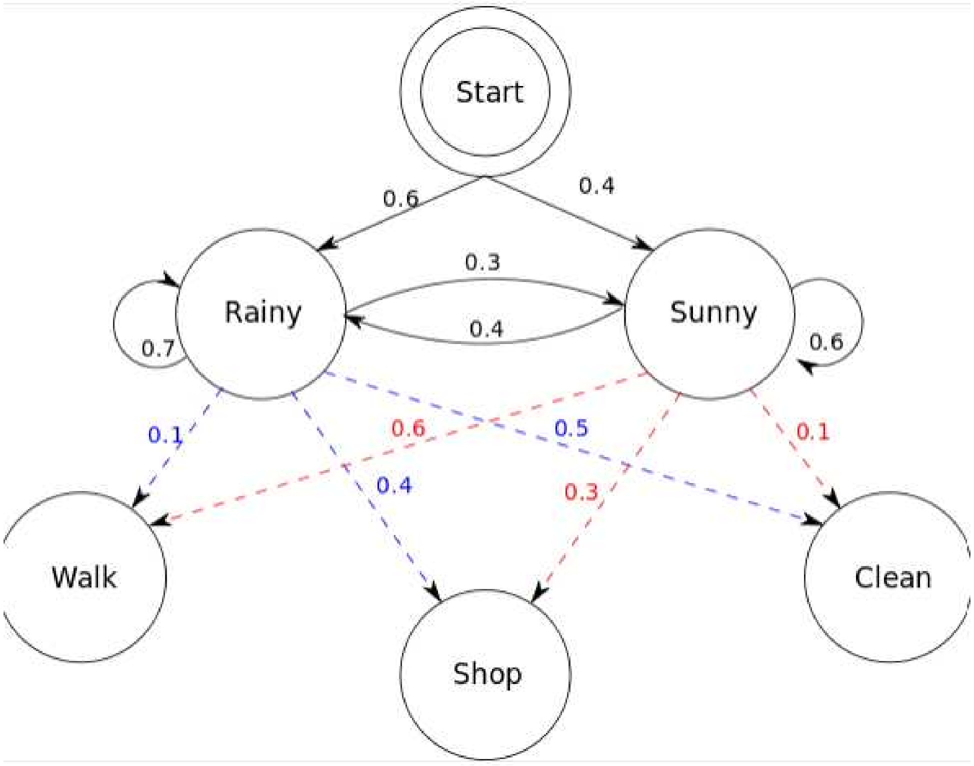

FIGURE 2.2: Example of Discrete Hidden Markov Model (Kumar (2023))

Figure 2.2 illustrates a Discrete Hidden Markov Model (HMM) with two hidden states and three observable activities. The hidden states in this model represent the weather conditions: "Rainy" and "Sunny". The observable activities are "Walk", "Shop", and "Clean".

States:

• $S_1(Rainy)$

• $S_2(Sunny)$

Transition Probabilities:

• The arrows between "Rainy" and "Sunny" indicate the probabilities of transitioning between these states. For example, the probability of transitioning from "Rainy" to "Sunny" is 0.3, while the probability of transitioning from "Sunny" to "Rainy" is 0.4.

• The self-loops on the states represent the probabilities of staying in the same state. For example, the probability of remaining "Rainy" is 0.7, and the probability of remaining "Sunny" is 0.6.

Initial State Probabilities:

• The model starts with a probability of 0.6 for "Rainy" and 0.4 for "Sunny".

Emission Probabilities:

• The dashed lines connect the states to the observable activities, indicating the probabilities of each activity given the current state. For instance, when the state is "Rainy", the probabilities of observing "Walk", "Shop", and "Clean" are 0.1, 0.4, and 0.5, respectively.

• When the state is "Sunny", the probabilities of observing "Walk", "Shop", and "Clean" are 0.6, 0.3, and 0.1, respectively.

This figure exemplifies how a discrete HMM models a system with hidden states (weather conditions) and observable outputs (activities). The transition probabilities govern the changes between hidden states, while the emission probabilities determine the likelihood of different observations given the current state.

2.3.2 Continuous HMMs

Continuous HMMs handle continuous observations, which can take any value within a range. These models are suitable for applications involving real-valued data, such as in signal processing or financial modeling.

In continuous HMMs, observations are continuous and can take any value within a range. The emission probabilities are often modeled using probability density functions (PDFs), such as Gaussians. This makes continuous HMMs suitable for tasks involving real-valued data like sensor readings or financial time series. For state j with observation o , the emission probabilities $b_j(o)$ are represented by PDFs, typically Gaussian distributions, as shown in the formula:

$$\mathrm{b}_j(o) = \frac{1}{\sqrt{2\pi\sigma_j^2}} \exp\left(-\frac{(o-\mu_j)^2}{2\sigma_j^2}\right)$$

where μ_j and σ_j^2 are the mean and variance of the Gaussian distribution for state j . Example applications include speech recognition, modeling the continuous acoustic features of speech, and financial modeling, predicting stock prices based on historical continuous data. (Rabiner (1989))

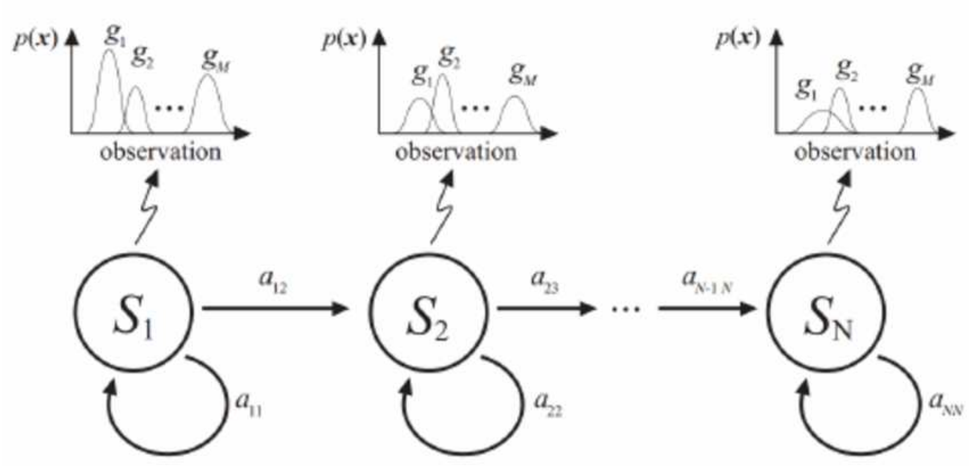

FIGURE 2.3: Example of Continuous Hidden Markov Model (Vidhya (????))

Figure 2.3 illustrates a Continuous Hidden Markov Model (HMM) with multiple hidden states and continuous observations. Each hidden state, represented as $S_1, S_2, \ldots, S_N$, is associated with a probability density function (PDF) that models the continuous observations. The arrows between the states show the transition probabilities, such as a_{12} for transitioning from state S_1 to state S_2, and self-loops like a_{11} represent the probability of remaining in the same state.

In continuous HMMs, emission probabilities are modeled using PDFs, typically Gaussian Mixture Models (GMMs). Each state has an associated PDF, depicted as graphs above the states, which describe the distribution of observations when the system is in that state. These PDFs, consisting of Gaussian components $g_1, g_2, \ldots, g_M$, capture the characteristics of continuous data. This model illustrates how continuous HMMs manage hidden states and continuous observations, with transition probabilities governing state changes and PDFs describing the likelihood of observations given the current state.

2.3.3 Hidden Semi-Markov Models

Hidden Semi-Markov Models extend traditional HMMs by allowing the duration of time spent in each state to follow an arbitrary distribution, rather than being geometrically distributed as in standard HMMs. This flexibility makes HSMMs useful for modeling more complex temporal patterns.

HSMMs allow the duration of time in each state to follow any specified distribution, which is particularly useful for applications where the time spent in states varies significantly and is not memoryless. These models are more complex and computationally intensive than standard HMMs. The duration distribution $D_j(d)$ specifies the probability of remaining in state j for d time steps. Transition and emission probabilities are similar to those in standard HMMs but account for the extended duration in states. Example applications include biological sequence analysis, modeling the time between genetic events, and customer behavior modeling, understanding the duration of time a customer spends in different stages of a buying process. (Zen *et al.* (2004))

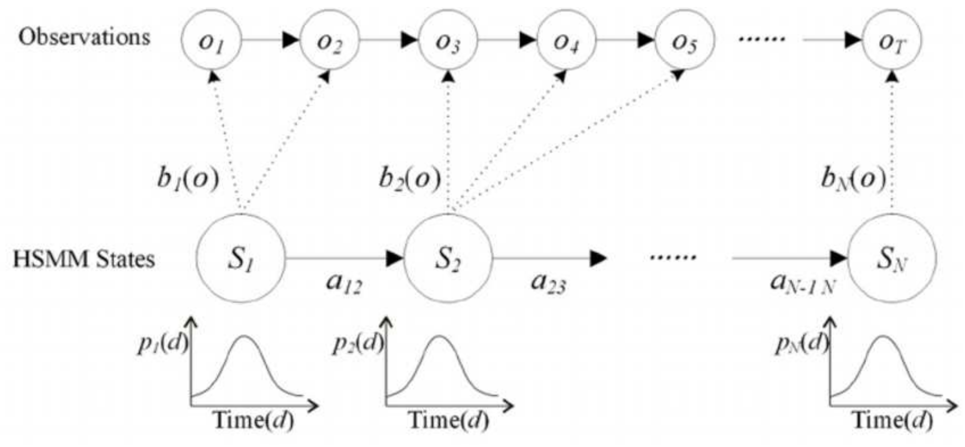

FIGURE 2.4: Example of Hidden semi-Markov Model (Anzhi (2023))

Figure 2.4 illustrates a Hidden Semi-Markov Model (HSMM) with multiple hidden states and observations over time. Each hidden state, represented as $S_1, S_2, \ldots, S_N$, has an associated duration distribution p(d) that models the time spent in each state. The arrows between the states, such as a_{12} for transitioning from state S_1 to state S_2, represent the transition probabilities.

Observations $O_1, O_2, \ldots, O_T$ are generated based on the hidden states. Emission probabilities, denoted as $b_1(o), b_2(o), \ldots, b_N(o)$, describe the likelihood of different observations given the current state. Unlike standard HMMs, HSMMs allow for non-geometric duration distributions, shown as $p_1(d), p_2(d), \ldots, p_N(d)$, providing more flexibility in modeling the time spent in each state.

This model demonstrates how HSMMs manage hidden states and observations over time, with transition probabilities governing state changes, emission probabilities describing observations, and duration distributions modeling the time spent in each state.

Figure 2.5 compares a standard Hidden Markov Model (HMM) with a Hidden Semi-Markov Model (HSMM).

In the HMM (Figure 2.5a), the hidden states S_1, S_2, S_3 are connected by transition probabilities a_{ij}, indicating the likelihood of transitioning from one state to another. Each state generates observations based on emission probabilities $b_1(\cdot), b_2(\cdot), b_3(\cdot)$. The initial state distribution π_1 specifies the starting probabilities of each state. The

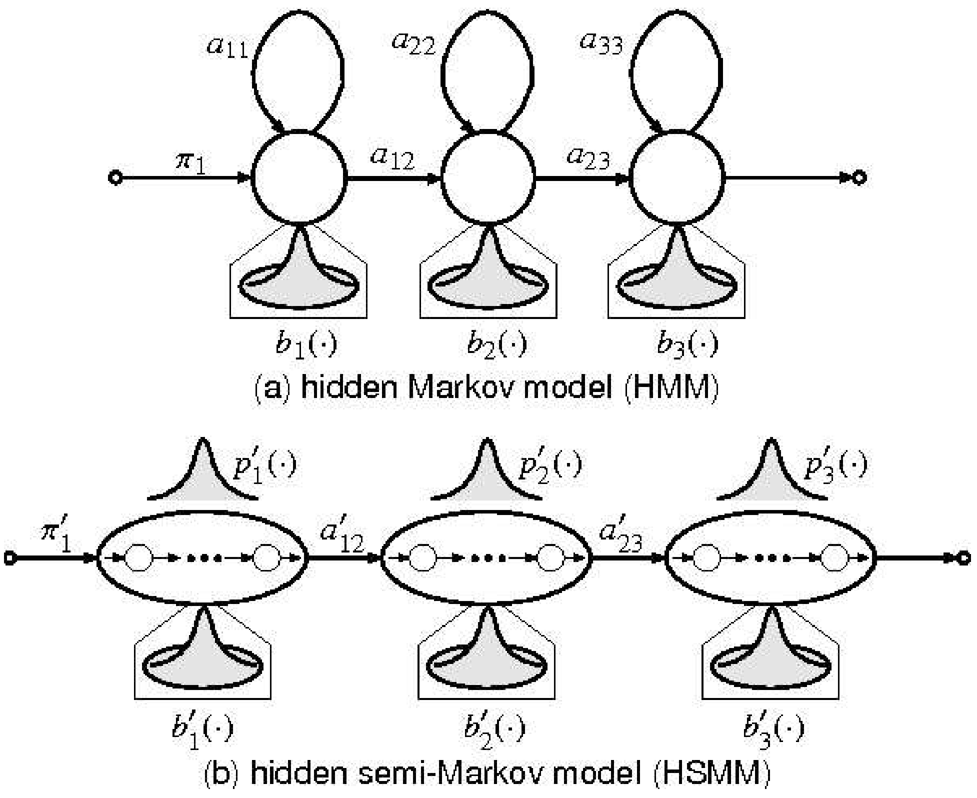

FIGURE 2.5: Comparision between HMM and HSMM (Zen *et al.* (2004))

time spent in each state follows a geometric distribution, as indicated by the loop arrows.

In the HSMM (Figure 2.5b), the structure is similar, with hidden states and transition probabilities $a\prime_{ij}$, but it includes explicit duration distributions $p\prime_1(\cdot), p\prime_2(\cdot), p\prime_3(\cdot)$ that model the time spent in each state. The emission probabilities $b\prime_1(\cdot), b\prime_2(\cdot), b\prime_3(\cdot)$ describe the likelihood of observations given the current state. Unlike the HMM, the HSMM allows for more flexible, non-geometric state durations, providing a more accurate representation of processes with variable state durations.

This comparison highlights the additional complexity and flexibility of HSMMs in modeling systems where the duration in each state is not memoryless, making them more suitable for applications requiring precise duration modeling.

2.4 Summary

Chapter 2 provided an essential foundation for understanding Hidden Markov Models (HMMs). It began with an introduction to HMMs, highlighting their significance in modeling complex dynamical systems with hidden states. The chapter covered the basic concepts and components of HMMs, including hidden states, observations, state transition probabilities, observation probabilities, and initial state probabilities.

We explored the different types of HMMs, such as discrete, continuous, and Hidden Semi-Markov Models (HSMMs), providing examples and figures to illustrate their structures and functions. The discrete HMM was exemplified through a weather model with states like "Rainy" and "Sunny" and observations such as "Walk," "Shop," and "Clean." The continuous HMM was discussed with reference to Gaussian Mixture Models (GMMs) for observation probabilities, and the HSMM was introduced with its ability to model state durations more flexibly.

Additionally, the chapter touched upon the importance of HMMs in various applications, setting the stage for more detailed discussions in later chapters. Through this overview, readers gained a comprehensive understanding of the fundamental elements and different types of HMMs, laying the groundwork for more advanced topics in subsequent chapters.

In the next chapter, we will delve into the detailed algorithms that underpin HMMs. We will explore the Forward-Backward Algorithm, Viterbi Algorithm, and Baum-Welch Algorithm in depth, providing step-by-step explanations and examples. This will enable us to understand how to evaluate, decode, and learn HMMs effectively, setting the stage for their practical applications.

Chapter III

Theoretical Framework

3.1 Introduction

In this chapter, we delve into the theoretical foundations and core algorithms that are fundamental to the practical use of Hidden Markov Models (HMMs). These algorithms enable us to evaluate, decode, and learn from HMMs, making them powerful tools for a wide range of applications. Understanding these algorithms is essential for anyone looking to apply HMMs to real-world problems.

We will cover the following key areas: probability theory in HMMs, detailed algorithms for HMMs, estimation techniques, model selection and evaluation, and extensions and variants of HMMs. Probability theory provides the essential background, introducing basic concepts and the principles of Markov processes. The detailed algorithms section introduces the Forward-Backward Algorithm, Viterbi Algorithm, and Baum-Welch Algorithm, providing step-by-step explanations and examples. Estimation techniques such as Maximum Likelihood Estimation and the Expectation-Maximization (EM) Algorithm will be discussed, alongside Bayesian Estimation Methods. The model selection and evaluation section will cover model complexity, cross-validation, and performance metrics to aid in selecting and evaluating HMMs. Finally, we will explore extensions and variants of HMMs to understand how these models can be adapted and extended for more complex scenarios. (Rabiner (1989))

3.2 Markov Chains

Markov chains are sequences of random variables that exhibit a specific type of dependency structure. A sequence of random variables $(X_n)n \geq 0$ taking values in a state space $X = \{1, 2, \ldots, r\}$ is a Markov chain if, for all $n \geq 0$ and all $x_0, x_1, \ldots, x_{n+1}$ in X, the probability of transitioning to the next state depends only on the current state and not on the sequence of events that preceded it. Formally, this property is expressed as:

$$\mathrm{P}(\mathrm{X}_{n+1} = j \mid X_n = i, X_{n-1} = x_{n-1}, \ldots, X_0 = x_0) = P(X_{n+1} = j \mid X_n = i)$$

This property implies that the future state depends only on the present state, making Markov chains memoryless. (Blackwell and Koopmans (1957))

3.2.1 Homogeneous Markov Chains

A Markov chain is said to be homogeneous if the transition probabilities are independent of time n. For a homogeneous Markov chain, the transition probability from state i to state j is given by the entry a_{ij} of the transition probability matrix A, where $a_{ij} = P(X_{n+1} = j \mid X_n = i)$. (Blackwell and Koopmans (1957))

The transition probability matrix A is a square matrix with nonnegative elements where each row sums to 1. This property ensures that the matrix is stochastic, meaning that it represents valid probability distributions over the states. Mathematically, a matrix A is stochastic if:

$$\textstyle\sum_j a_{ij} = 1 \quad \text{for all } i.$$

3.2.2 Dependence Structure

The dependence structure of a Markov chain is such that the Markov property holds even for non-consecutive indices. This is captured by the property that, for any sequences of times $0 \leq s_1 < s_2 < \cdots < s_N < t_1 < t_2 < \cdots < t_M$ and any sequences of states $x_{s_1}, x_{s_2}, \ldots, x_{s_N}$ and $x_{t_1}, x_{t_2}, \ldots, x_{t_M}$,

$$\mathrm{P}(\mathrm{X}_{t_1} = x_{t_1}, \ldots, X_{t_M} = x_{t_M} \mid X_{s_N} = x_{s_N}, X_{s_{N-1}} = x_{s_{N-1}}, \ldots, X_{s_1} = x_{s_1}) = P(X_{t_1} = x_{t_1}, \ldots, X_{t_M} = x_{t_M} \mid X_{s_N} = x_{s_N}).$$

This property emphasizes that the states of a Markov chain are conditionally independent of each other given the present state. (Blackwell and Koopmans (1957))

3.2.3 Probability of a Sequence

In a Markov chain, the probability of a sequence of states $x_0, x_1, \ldots, x_n$ can be computed as the product of the initial state probability and the transition probabilities along the path:

$$\mathrm{P}(\mathrm{X}_0 = x_0, X_1 = x_1, \ldots, X_n = x_n) = \pi_{x_0} \prod_{t=0}^{n-1} a_{x_t x_{t+1}}$$

where π_{x_0} is the initial state distribution. (Blackwell and Koopmans (1957))

Stationary Distributions

An important aspect of Markov chains is their long-term behavior, which is often studied through the concept of stationary distributions. A stationary distribution π is a probability distribution over the states that remains unchanged under the dynamics of the Markov chain, meaning that if the chain starts in the stationary distribution, it stays in that distribution. Formally, π is a stationary distribution if:

$$\pi = \pi A.$$

This means that π is a left eigenvector of the transition matrix A corresponding to the eigenvalue 1. If a Markov chain has a unique stationary distribution and is irreducible and aperiodic, the chain will converge to this stationary distribution regardless of the initial state distribution. (Blackwell and Koopmans (1957))

3.2.4 Ergodic Theorem for Markov Chains

The ergodic theorem for Markov chains states that for a finite state space X and a primitive transition probability matrix A (where A^p is strictly positive for some p), the chain will converge to a unique stationary distribution π. This is expressed as:

$$\lim_{n\to\infty} A^n = \mathbf{1}\pi$$

where 1 is a column vector of ones. This theorem implies that, in the long run, the state distribution of the chain becomes independent of the initial state distribution and is governed by $\pi.(\mathit{BlackwellandKoopmans}$ (1957))

Example: Weather Model

Consider a simple weather model with two states: Sunny and Rainy. Let the transition probability matrix be:

$$\mathrm{A} = \begin{pmatrix} 0.8 & 0.2 \\ 0.4 & 0.6 \end{pmatrix},$$

where the entry a_{ij} represents the probability of transitioning from state i to state j. If the initial state distribution is $\pi_0 = (0.5, 0.5)$, the state distribution after n steps can be computed as $\pi_n = \pi_0 A^n$.

In summary, this section provided an in-depth look at Markov chains, focusing on their definition, properties, and the concept of stationary distributions. Markov chains form the foundation for understanding hidden Markov models (HMMs), which will be explored in subsequent sections. In the next section, we will delve into the detailed algorithms that underpin HMMs, including the Forward-Backward Algorithm, Viterbi Algorithm, and Baum-Welch Algorithm. We will explore these algorithms step-by-step to understand how to evaluate, decode, and train HMMs, laying the groundwork for their practical applications. (Blackwell and Koopmans (1957))

3.3 Algorithms for HMMs

Hidden Markov Models (HMMs) rely on several fundamental algorithms for their practical use. These algorithms enable us to evaluate the probability of observation sequences, decode the most likely sequence of hidden states, and learn the parameters of the model from observed data. The three primary algorithms we will discuss are the Forward-Backward Algorithm, the Viterbi Algorithm, and the Baum-Welch Algorithm. (Rabiner (1989))

3.3.1 The Forward-Backward Algorithm

The Forward-Backward Algorithm is used to calculate the probability of a sequence of observations given a model, addressing the evaluation problem. This algorithm employs dynamic programming to compute the likelihood of the observed sequence efficiently.

Forward Procedure

The forward procedure calculates the probability of being in a particular state at a specific time given the observed sequence up to that time. The forward variable, $\alpha_t(i)$, is defined as the probability of the partial observation sequence $O_1, O_2, \ldots, O_t$ and state S_i at time t :

$$\alpha_t(i) = P(O_1, O_2, \ldots, O_t, S_t = S_i \mid \lambda)$$

Steps:

1. Initialization:

$$\alpha_1(i) = \pi_i b_i(O_1) \quad \text{for } 1 \le i \le N$$

2. Recursion:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}) \quad \text{for } 1 \le t \le T-1,\ 1 \le j \le N$$

3. Termination:

$$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

Backward Procedure

The backward procedure calculates the probability of the ending part of the observation sequence given a particular state at a specific time. The backward variable, $\beta_t(i)$, is defined as the probability of the partial observation sequence from t+1 to the end, given state S_i at time t and the model λ :

$$\beta_t(i) = P(O_{t+1}, O_{t+2}, \ldots, O_T \mid S_t = S_i, \lambda)$$

Steps:

1. Initialization:

$$\beta_T(i) = 1 \quad \text{for } 1 \le i \le N$$

2. Recursion:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \quad \text{for } t = T-1, T-2, \ldots, 1;\ 1 \le i \le N$$

Combining the forward and backward variables, the probability of being in state S_i at time t given the observation sequence can be calculated. (Rabiner (1989))

Figure 3.1 provides a visual representation of the key components and steps involved in the algorithm.

• Panel (a):The forward procedure, which computes the forward variables $\alpha_t(i)$. These variables represent the probability of being in state S_i at time t , given the observed sequence up to that time. The arrows show the transitions from state S_i at time t to state S_j at time t+1 , with associated transition probabilities a_{ij} .

• Panel (b): The backward procedure, which computes the backward variables $\beta_t(i)$. These variables represent the probability of the ending part of the observation sequence given state S_i at time t . The arrows indicate the transitions from state S_i at time t to state S_j at time t+1 , again with associated transition probabilities a_{ij} .

• Panel (c): A combined view showing the relationship between states and observations over time. The arrows between states $S_1, S_2, \ldots, S_N$ and observations $o_1, o_2, \ldots, o_T$ represent the emission probabilities, indicating how likely a state is to produce a particular observation.

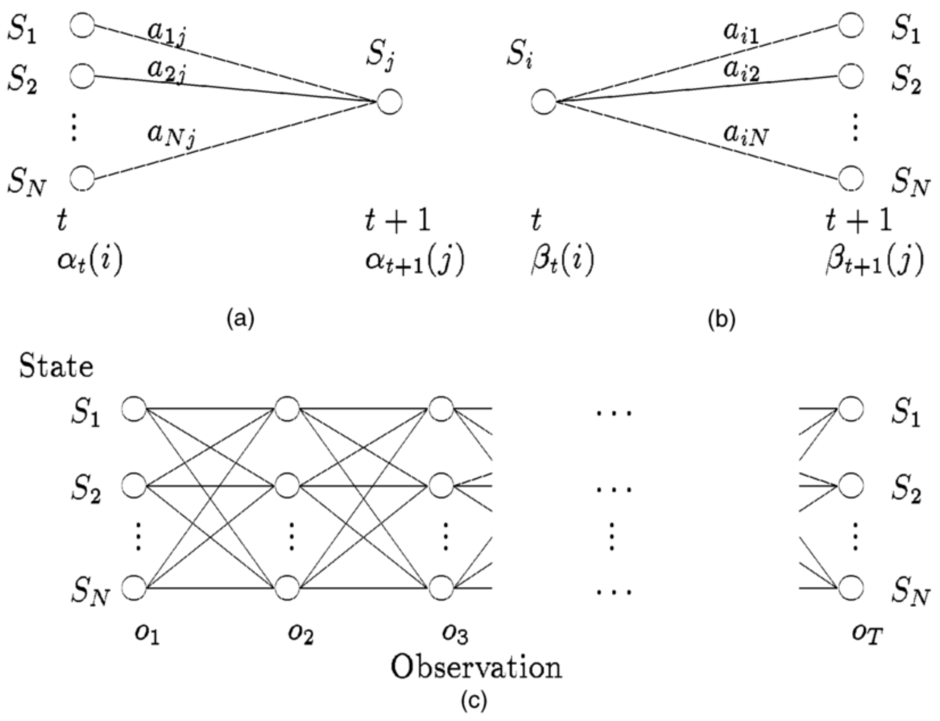

FIGURE 3.1: Illustration of Forward-Backward Algorithm (Li (2024))

3.3.2 The Viterbi Algorithm

The Viterbi Algorithm is used to find the most likely sequence of hidden states given a sequence of observations, addressing the decoding problem. This algorithm employs dynamic programming to ensure computational efficiency and is crucial for tasks that require the identification of the underlying state sequence.

The Viterbi variable, $\delta_t(i)$, represents the highest probability of any path that ends in state S_i and accounts for the first t observations:

$$\delta_t(i) = \max_{S_1,S_2,\ldots,S_{t-1}} P(S_1, S_2, \ldots, S_{t-1}, S_t = S_i, O_1, O_2, \ldots, O_t \mid \lambda)$$

The steps of the Viterbi Algorithm are as follows:

1. Initialization:

$$\delta_1(i) = \pi_i b_i(O_1) \quad \text{for } 1 \le i \le N$$

$$\psi_1(i) = 0$$

• The algorithm starts by initializing the probabilities for the first observation. $\delta_1(i)$ is the probability of being in state S_i at time 1, multiplied by the probability of observing O_1 given that state.

2. Recursion:

$$\delta_{t+1}(j) = \max_{1\le i\le N}[\delta_t(i)a_{ij}]b_j(O_{t+1}) \quad \text{for } 1 \le t \le T-1,\ 1 \le j \le N$$

$$\psi_{t+1}(j) =_{1\leq i\leq N} [\delta_t(i)a_{ij}]$$

• For each subsequent observation, the algorithm updates the probabilities $\delta_t(i)$ by considering all possible paths that could lead to each state and selecting the most probable one. The variable $\psi_{t+1}(j)$ keeps track of the state that provided the maximum probability for each step.

3. Termination:

$P^* = \max_{1\leq i\leq N} \delta_T(i)$

$$S_T^* =_{1\leq i\leq N} \delta_T(i)$$

• The algorithm identifies the maximum probability of the final observation sequence, P^* , and the corresponding final state S_T^* .

4. Path Backtracking:

$S_t^* = \psi_{t+1}(S_{t+1}^*) \quad \text{for } t = T-1, T-2, \ldots, 1$

Finally, the algorithm traces back through the states stored in ψ to reconstruct the most likely sequence of hidden states. (Rabiner (1989))

3.3.3 The Baum-Welch Algorithm

The Baum-Welch Algorithm, an Expectation-Maximization (EM) algorithm, is used to estimate the parameters of a Hidden Markov Model (HMM), addressing the learning problem. This algorithm iteratively adjusts the parameters to maximize the likelihood of the observed data.

The Baum-Welch Algorithm involves the following steps:

1. Initialization:

• Initialize the model parameters (transition probabilities a_{ij} , emission probabilities $b_j(k)$, and initial state distribution π_i) randomly or based on prior knowledge.

2. Expectation Step:

• Compute the forward variable $\alpha_t(i)$ and the backward variable $\beta_t(i)$.

Forward Variable Calculation:

$\alpha_t(i) = P(O_1, O_2, \ldots, O_t, S_t = S_i \mid \lambda)$

Backward Variable Calculation:

$\beta_t(i) = P(O_{t+1}, O_{t+2}, \ldots, O_T \mid S_t = S_i, \lambda)$

• The forward variable $\alpha_t(i)$ represents the probability of the partial observation sequence $O_1, O_2, \ldots, O_t$ and state S_i at time t . The backward variable $\beta_t(i)$ represents the probability of the partial observation sequence from t+1 to the end, given state S_i at time t and the model λ .

3. Maximization Step:

• Update the transition probabilities a_{ij} , emission probabilities $b_j(k)$, and initial state distribution π_i using the expected values from the forward and backward variables.

Transition Probabilities:

$$a_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

Emission Probabilities:

$$b_j(k) = \frac{\sum_{t=1,O_t=k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

Initial State Distribution:

$$\pi_i = \gamma_1(i)$$

where:

$$\gamma_t(i) = P(S_t = S_i \mid O, \lambda)\xi_t(i,j) = P(S_t = S_i, S_{t+1} = S_j \mid O, \lambda)$$

• The transition probabilities a_{ij} are updated based on the expected number of transitions from state S_i to state S_j . The emission probabilities $b_j(k)$ are updated based on the expected number of times observation k is generated from state S_j . The initial state distribution π_i is updated based on the probability of starting in state S_i .

4. Convergence Check:

• Check for convergence of the algorithm by evaluating the change in the likelihood of the observed data. If the change is below a certain threshold, the algorithm has converged. Otherwise, repeat the Expectation and Maximization steps. (Rabiner (1989))

The Figure 3.2 visualizes the key components and steps:

The horizontal axis represents time with states S_i and S_j at times t and t+1 . Observations $O_{t-1}, O_t, O_{t+1}, O_{t+2}$ are shown at each time step.

The forward variable $\alpha_t(i)$ represents the probability of the partial observation sequence up to time t and state S_i at time t . Arrows into S_i from previous observations show $\alpha_t(i)$.

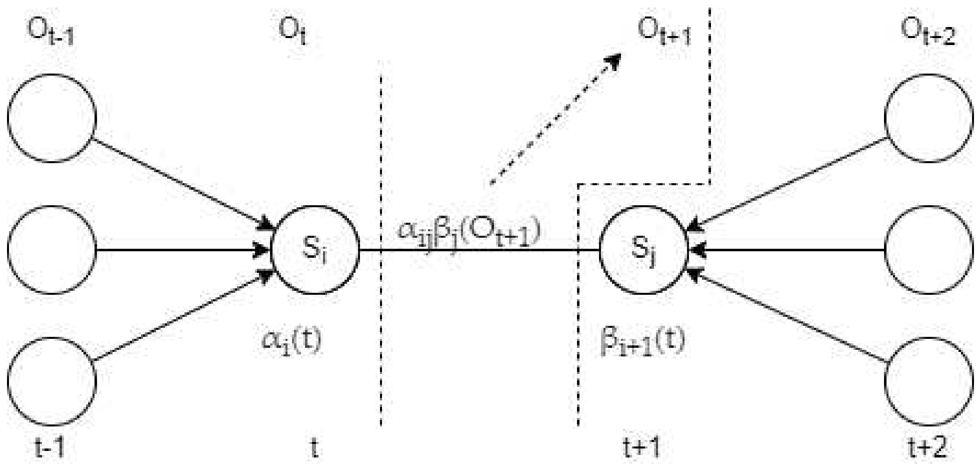

FIGURE 3.2: Illustration of Baum-Welch Algorithm (med (2024))

The backward variable $\beta_t(i)$ represents the probability of the partial observation sequence from time t+1 to the end, given state S_i at time t . Arrows out of S_j to future observations show $\beta_t(i)$.

The transition from S_i at time t to S_j at time t+1 with transition probability a_{ij} and emission probability $b_j(O_{t+1})$ is depicted.

The expectation step combines forward and backward variables to calculate expected transitions and emissions, used to update model parameters.

The maximization step updates parameters a_{ij} and $b_j(k)$ based on expected values, improving the likelihood of the observed data.

3.3.4 Maximum Likelihood Estimation

Maximum Likelihood Estimation (MLE) is a fundamental method for estimating the parameters of a statistical model. In the context of Hidden Markov Models (HMMs), MLE aims to find the model parameters that maximize the likelihood of the observed data. The Baum-Welch Algorithm, previously discussed, is a practical implementation of MLE for HMMs.

The likelihood function for an HMM is defined as the probability of the observed sequence $O = \{O_1, O_2, \ldots, O_T\}$ given the model parameters $\lambda = (\pi, A, B)$:

$$L(\lambda \mid O) = P(O \mid \lambda)$$

The goal of MLE is to find the parameter set λ that maximizes this likelihood function:

$$\hat{\lambda} =_{\lambda} L(\lambda \mid O)$$

Steps in Maximum Likelihood Estimation:

1. Define the Likelihood Function:

• The likelihood function for HMMs involves summing over all possible state sequences, which can be computationally intensive. This is where the forward-backward algorithm becomes useful.

2. Initialize Parameters:

• Start with an initial guess for the parameters $\lambda = (\pi, A, B)$. These can be randomly chosen or based on prior knowledge.

3. Expectation-Maximization (EM) Algorithm:

• Use the Baum-Welch algorithm to iteratively update the parameters. This involves two steps in each iteration:

Expectation Step (E-Step):

• Calculate the expected values of the sufficient statistics using the current parameters. This involves computing the forward and backward variables, $\alpha_t(i)$ and $\beta_t(i)$, as well as the state occupancy probabilities $\gamma_t(i)$ and transition probabilities $\xi_t(i,j)$.

Maximization Step (M-Step):

• Update the parameters to maximize the expected likelihood. This involves updating the transition probabilities A , emission probabilities B , and initial state distribution π based on the expected values computed in the E-step.

4. Check for Convergence:

• Evaluate the change in the likelihood function. If the change is below a certain threshold, the algorithm has converged, and the current parameters are considered the MLE estimates.

5. Final Parameters:

• After convergence, the final parameter set $\hat{\lambda}$ represents the maximum likelihood estimates for the HMM. (Rabiner (1989))

Example of MLE in HMMs

Consider a simple HMM with two states (Rainy and Sunny) and observations (Walk, Shop, Clean). The goal is to estimate the transition and emission probabilities from the observed sequence of activities.

Initialization:

• Transition probabilities A and emission probabilities B are initialized randomly.

E-Step:

• Compute the forward and backward variables for the observed sequence using the current parameters.

M-Step:

• Update the transition and emission probabilities based on the expected state transitions and state-occupation counts.

Convergence:

• Check the likelihood of the observed sequence given the updated parameters. If the likelihood has not changed significantly, stop; otherwise, repeat the E-step and M-step.

Final Parameters:

• The resulting transition and emission probabilities maximize the likelihood of the observed sequence under the HMM. (Rabiner (1989))

3.4 Bayesian Estimation

The Bayesian estimation approach for Hidden Markov Models (HMMs) provides an alternative to the traditional Maximum Likelihood Estimation (MLE), which is often computationally intensive and may suffer from convergence issues. Bayesian estimators offer a robust framework by incorporating prior information into the estimation process.

In the context of HMMs, the objective is to estimate the parameters $\theta = (A, B)$, where A represents the state transition probabilities and B represents the observation probabilities. The Bayesian estimation method involves specifying a prior distribution over the parameter space and updating this distribution based on observed data. (Di Masi and Finesso (1996))

3.4.1 Bayesian Framework

The Bayesian approach starts with the definition of a prior distribution $\nu(\cdot)$ on the parameter space Θ. The prior encapsulates our initial beliefs about the parameters before observing any data. As data becomes available, the prior is updated to a posterior distribution using Bayes' theorem.

Given observations $y_1, y_2, \ldots, y_n$, the posterior distribution $p(\theta \mid y_n)$ can be computed. The Bayesian estimator $\hat{\theta}_n$ is the expected value of the parameter with respect to the posterior distribution: (Di Masi and Finesso (1996))

$_n = \mathbb{E}[\theta \mid y_n].$

Dirichlet Priors

A common choice for the prior distribution in HMMs is the Dirichlet distribution, which is conjugate to the multinomial distribution. This choice simplifies the posterior updates. The Dirichlet prior for the transition probabilities A and the observation probabilities B can be specified as:

$$\nu_D(\theta) = \prod_i \left[\frac{\Gamma(\frac{k}{2})}{\Gamma(\frac{1}{2})^k} \prod_j a_{ij}^{\frac{1}{2}-1} \right] \prod_l \left[\frac{\Gamma(\frac{q}{2})}{\Gamma(\frac{1}{2})^q} \prod_y b_{ly}^{\frac{1}{2}-1} \right]$$

where $\Gamma(\cdot)$ is the Gamma function, k is the number of states, and q is the number of observation symbols. (Di Masi and Finesso (1996))

Posterior Distribution

Using the Dirichlet prior, the posterior distribution after observing n data points can be computed. The Bayesian estimator $\hat{a}ij$ for the transition probabilities and $\hat{b}ly$ for the observation probabilities are given by:

$$\hat{a}ij = \mathbb{E}[aij \mid y_n] = \frac{N_{ij}(x_n)+\frac{1}{2}}{N_i(x_n)+\frac{k}{2}}$$

where $N_{ij}(x_n)$ is the number of transitions from state i to state j in the observed sequence, and $N_i(x_n)$ is the number of times state i is visited.

Similarly, the estimator for the observation probabilities is:

$$\hat{b}ly = \mathbb{E}[bly \mid y_n] = \frac{M_{ly}(y_n)+\frac{1}{2}}{M_l(y_n)+\frac{q}{2}}$$

where $M_{ly}(y_n)$ is the number of times observation y is observed in state l, and $M_l(y_n)$ is the number of times state l is visited. (Di Masi and Finesso (1996))

3.4.2 Consistency and Convergence

Under certain regularity conditions, the Bayesian estimators are consistent, meaning that they converge to the true parameter values as the number of observations increases. Specifically, if θ_0 is the true parameter value, and the prior distribution is strictly positive everywhere, then the Bayesian estimator $\hat{\theta}_n$ converges to θ_0 almost surely. (Di Masi and Finesso (1996))

Theorem 2.1 from the reference states

$\hat{\theta}n \to \theta_0 \quad \text{a.s. } P\theta_0$

where the convergence is derived using a Laplace expansion and the Shannon-McMillan-Breiman theorem, ensuring that the posterior distribution concentrates around the true parameter values. (Di Masi and Finesso (1996))

3.4.3 Practical Computation

The practical computation of Bayesian estimators involves discretizing the parameter space and evaluating the posterior distribution over a grid. For a fine enough discretization, the Bayesian estimator can be approximated efficiently.

In summary, the Bayesian approach provides a powerful alternative to MLE for HMM parameter estimation, leveraging prior information and ensuring consistent estimates under appropriate conditions. This approach is particularly useful when the traditional EM algorithm faces convergence issues or when prior knowledge about the parameters is available.

This section provided an overview of Bayesian estimation for HMMs, highlighting the use of Dirichlet priors and the consistency of Bayesian estimators. In the next sections, we will delve into model selection and evaluation techniques, as well as explore extensions and variants of HMMs to address more complex scenarios. (Di Masi and Finesso (1996))

3.5 Model Selection and Evaluation

Selecting and evaluating the best Hidden Markov Model (HMM) for a given problem is crucial for achieving accurate and reliable results. This section discusses techniques for model selection, evaluating model performance, and ensuring that the chosen model generalizes well to new data.

3.5.1 Model Complexity

Choosing the appropriate complexity for a Hidden Markov Model (HMM) is critical to its performance and generalization capabilities. The complexity of an HMM primarily depends on the number of hidden states, the transition and emission probabilities, and the potential use of regularization techniques.

1. Number of States:

The number of hidden states in an HMM is a crucial factor in its complexity. If the model has too few states, it may not capture the underlying structure and dynamics of the data. Conversely, if the model has too many states, it may overfit the training data, leading to poor generalization on new data. Selecting the optimal number of states involves balancing these two aspects.

2. Transition and Emission Probabilities:

The complexity of the HMM is also influenced by the number of parameters in the transition and emission matrices. A model with fewer parameters may be more robust and easier to estimate but might miss important patterns in the data. On the other hand, a model with more parameters can capture detailed patterns but may overfit the data.

3. Regularization Techniques:

Regularization can help prevent overfitting by adding a penalty to the likelihood function for more complex models. Techniques such as L1 (Lasso) or L2 (Ridge) regularization can be applied to the parameters of the transition and emission matrices to discourage overly complex models and improve generalization. (Rabiner (1989))

Example of Model Complexity in HMMs

Consider an HMM used for part-of-speech tagging in natural language processing. The goal is to determine the optimal number of hidden states representing different parts of speech (e.g., nouns, verbs, adjectives). (Rabiner (1989))

- **Too Few States**: With only two states (e.g., noun and verb), the model might fail to distinguish between other parts of speech like adjectives and adverbs, leading to poor tagging accuracy.
- **Too Many States**: With too many states (e.g., 20 states), the model might overfit the training data, capturing noise rather than meaningful patterns, resulting in poor generalization to new sentences.
- **Optimal Number of States**: Through experimentation and evaluation, an optimal number of states (e.g., six states representing noun, verb, adjective, adverb, preposition, and conjunction) can be determined. This balance captures the necessary linguistic distinctions without overfitting.

3.5.2 Cross-Validation

Cross-validation is a robust technique used to assess the generalizability and performance of a Hidden Markov Model (HMM) on independent data sets. It helps in evaluating how well the model will perform on unseen data, thus preventing overfitting. Cross-validation involves partitioning the data into subsets, training the model on some subsets, and validating it on the remaining subsets.

1. K-Fold Cross-Validation

In k-fold cross-validation, the data set is divided into k equally-sized folds. The model is trained on k-1 folds and validated on the remaining fold. This process is repeated k times, with each fold serving as the validation set once. The performance metrics are then averaged over all k trials to provide a comprehensive evaluation of the model.

Steps in K-Fold Cross-Validation:

- Divide the data into k folds.
- For each fold i (from 1 to k):
- Train the HMM on k-1 folds (excluding fold i).
- Validate the HMM on fold i .
- Record the performance metric (e.g., log-likelihood, accuracy).
- Average the performance metrics over all k folds to obtain the final evaluation.

Example:

If you have a data set with 100 sequences and you choose k = 5 , each fold will contain 20 sequences. The model is trained on 80 sequences and validated on the remaining 20 sequences. This is repeated five times, and the results are averaged.

2. Leave-One-Out Cross-Validation (LOOCV)

LOOCV is a special case of k-fold cross-validation where k equals the number of data points. Each data point is used once as a validation set while the remaining points form the training set. LOOCV is computationally intensive but provides an unbiased estimate of model performance.

Steps in LOOCV:

- For each data point i :
- Train the HMM on all data points except i .
- Validate the HMM on data point i .
- Record the performance metric.
- Average the performance metrics over all data points to obtain the final evaluation.

Example:

If you have 100 sequences, the model is trained on 99 sequences and validated on the remaining one sequence. This is repeated 100 times, and the results are averaged.

Example of Cross-Validation in HMMs:

Consider an HMM used for speech recognition. The goal is to evaluate how well the HMM generalizes to new speech data.

• K-Fold Cross-Validation:

• Divide the speech data into 10 folds. Train the HMM on 9 folds and validate it on the remaining fold. Repeat this process 10 times. Average the log-likelihoods or accuracy scores to assess the model's performance.

• LOOCV:

• Train the HMM on all but one speech sequence and validate it on the excluded sequence. Repeat this process for each sequence in the data set. Average the results to obtain the final performance metric.

Performance Metrics:

- **Log-Likelihood**: Measure the log-likelihood of the validation set given the trained model.
- **Prediction Accuracy**: Compare the predicted hidden states with the true states if available.

Cross-validation provides a comprehensive way to evaluate the performance of HMMs, ensuring that the model selected performs well on unseen data. By using techniques like k-fold cross-validation and LOOCV, one can robustly assess and compare different models, leading to better generalization and more reliable predictions. (Hastie *et al.* (2009))

3.5.3 Performance Metrics

Evaluating the performance of Hidden Markov Models (HMMs) involves using a variety of metrics to assess their accuracy, reliability, and generalizability. These metrics help in understanding how well the model fits the data and how it is likely to perform on unseen data. Common performance metrics for HMMs include log-likelihood, Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), prediction accuracy, and measures like precision, recall, and F1-score.

1. Log-Likelihood

The log-likelihood of the observed data given the model is a fundamental measure of model fit. It quantifies how well the model explains the observed data. Higher log-likelihood values indicate a better fit.

$$\log L(\lambda \mid O) = \log P(O \mid \lambda)$$

Example: In a speech recognition task, the log-likelihood can be used to measure how well the HMM predicts the sequence of observed speech features.

2. Akaike Information Criterion (AIC)

AIC is used to compare models with different numbers of parameters. It penalizes models with more parameters to prevent overfitting.

$$\text{AIC} = 2k - 2\log L(\lambda \mid O)$$

where k is the number of parameters in the model.

Example: Comparing HMMs with different numbers of states for a part-of-speech tagging task, where the model with the lowest AIC is preferred.

3. Bayesian Information Criterion (BIC)

BIC is similar to AIC but includes a stronger penalty for models with more parameters, especially when the sample size is large.

$$\text{BIC} = k\log n - 2\log L(\lambda \mid O)$$

where n is the number of data points and k is the number of parameters. Example: In a bioinformatics application, BIC can help select the optimal HMM for modeling protein sequences.

4. Prediction Accuracy

For classification tasks, prediction accuracy measures the proportion of correctly predicted hidden states or observations.

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

Example: In part-of-speech tagging, the accuracy of the predicted tags compared to the true tags.

5. Precision, Recall, and F1-Score

These metrics are useful for evaluating model performance on imbalanced data sets.

• Precision: The proportion of true positive predictions among all positive predictions.

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

• Recall: The proportion of true positive predictions among all actual positive instances.

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

• F1-Score: The harmonic mean of precision and recall.

• F1-Score $= 2 \times \frac{\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}}$

Example: In medical diagnosis using HMMs, precision, recall, and F1-score can be used to evaluate the model's ability to correctly identify disease states.

Example of Performance Metrics in HMMs:

Consider an HMM used for handwriting recognition, where the goal is to evaluate the model's performance in recognizing handwritten characters.

- **Log-Likelihood**: Measure the log-likelihood of the observed sequences of pen strokes given the model.
- **AIC and BIC**: Compare different HMMs with varying numbers of states and transitions to select the model with the best balance of fit and complexity.
- **Prediction Accuracy**: Calculate the accuracy of the predicted characters compared to the true characters in a test set.
- **Precision, Recall, and F1-Score**: Use these metrics to evaluate the model's performance in correctly identifying specific characters, especially if some characters are less frequent than others.

By employing these performance metrics, one can comprehensively assess the effectiveness of HMMs in various applications, ensuring that the chosen model provides accurate and reliable predictions. (Burnham and Anderson (2004))

3.6 Summary

Chapter 3 delves into the theoretical foundations and core algorithms essential for the practical application of Hidden Markov Models (HMMs). It begins with an introduction to the importance of these algorithms, highlighting their roles in evaluating, decoding, and learning from HMMs.

The chapter covers the fundamental probability theory underlying HMMs, including basic probability concepts and Markov processes. This section explains how random variables, probability distributions, conditional probabilities, joint probabilities, and the properties of Markov chains form the bedrock of HMMs.

The detailed algorithms section provides an in-depth look at the Forward-Backward Algorithm, the Viterbi Algorithm, and the Baum-Welch Algorithm. The Forward-Backward Algorithm is crucial for evaluating the probability of a sequence of observations given a model. The Viterbi Algorithm is used to decode the most likely sequence of hidden states, while the Baum-Welch Algorithm addresses the learning problem by estimating the parameters of the HMM using an iterative Expectation-Maximization approach.

Estimation techniques such as Maximum Likelihood Estimation (MLE) and Bayesian Estimation Methods are discussed next. MLE seeks to find the parameters that maximize the likelihood of the observed data, while Bayesian methods incorporate prior information to refine parameter estimates. The chapter also covers model selection and evaluation, emphasizing the need to balance model complexity to avoid overfitting and ensuring that the chosen model generalizes well to new data. Techniques like cross-validation and metrics such as log-likelihood, AIC, BIC, prediction accuracy, precision, recall, and F1-score are explained.

Overall, Chapter 3 provides a comprehensive theoretical framework for understanding and applying HMMs, equipping researchers and practitioners with the knowledge to effectively use these models in various complex applications.

Chapter IV

Extensions and Variants of HMMs

4.1 Introduction

Hidden Markov Models (HMMs) have proven to be powerful tools for modeling sequential data across various domains. However, the standard HMM framework has limitations, particularly when dealing with complex systems that exhibit hierarchical structures or interactions between multiple processes. To address these limitations, several extensions and variants of HMMs have been developed.

This chapter focuses on two significant extensions of HMMs: Hierarchical Hidden Markov Models (HHMMs) and Coupled Hidden Markov Models (CHMMs). These advanced models provide greater flexibility and can capture more intricate dependencies and interactions in the data.

Hierarchical Hidden Markov Models introduce a multi-level structure, allowing the modeling of nested temporal dependencies. This hierarchical approach is particularly useful for systems where states themselves can be composed of other states, such as in speech recognition or activity recognition.

Coupled Hidden Markov Models, on the other hand, allow for the interaction between multiple HMMs. This coupling enables the modeling of systems where multiple dynamic processes influence each other, such as in multi-sensor fusion or human-computer interaction.

By exploring these extensions, we aim to understand how they enhance the modeling capabilities of HMMs and broaden their applicability to more complex real-world problems. The following sections will delve into the structure, key components, and applications of HHMMs and CHMMs, providing detailed explanations and examples to illustrate their utility. (Rabiner (1989))

Next, we will discuss the structure of Hierarchical Hidden Markov Models (HHMMs) in detail.

4.2 Hierarchical Hidden Markov Models

Hierarchical Hidden Markov Models (HHMMs) extend the traditional HMM framework by introducing a multi-level structure. This hierarchical approach allows for the modeling of systems where states can themselves be composed of other states, capturing more complex and nested temporal dependencies. This section delves into the structure, key components, and applications of HHMMs.

4.2.1 Structure of HHMMs

The structure of HHMMs is designed to capture hierarchical relationships within the data. In an HHMM, states are organized into a hierarchy where higher-level states can invoke sequences of lower-level states. This hierarchical structure enables the model to represent different levels of granularity in the data.

In a typical HHMM, the top-level states represent high-level activities or processes, while the lower-level states represent sub-activities or sub-processes. Transitions can occur both within a level and across levels, allowing the model to capture dependencies at multiple temporal scales. (Fine *et al.* (1998))

The main components of an HHMM structure include:

- **Levels of Hierarchy**: Each level contains states that represent a specific level of granularity.
- **State Transitions**: Transitions can occur between states at the same level or between different levels.
- **Observation Emission**: Observations can be emitted by states at any level, providing flexibility in modeling the data.

4.2.2 Key Components of HHMMs

The key components of HHMMs include the state hierarchy, transition mechanisms, and observation emission processes. Understanding these components is crucial for implementing and utilizing HHMMs effectively. (Bui *et al.* (2002))

- **State Hierarchy**: The states in an HHMM are organized into a hierarchical structure. Higher-level states can invoke sequences of lower-level states, allowing the model to capture nested dependencies. Each level in the hierarchy represents a different granularity of the data.
- **Transition Mechanisms**: Transitions in an HHMM can occur within a level or across levels. Intra-level transitions capture dependencies within the same

level, while inter-level transitions capture dependencies across different levels. This flexibility allows the HHMM to model both short-term and long-term dependencies in the data.

- **Observation Emission**: Observations can be generated by states at any level in the hierarchy. This feature allows the HHMM to model observations that can occur at different temporal scales. The emission probability distribution for each state specifies the likelihood of observing a particular observation given the state.

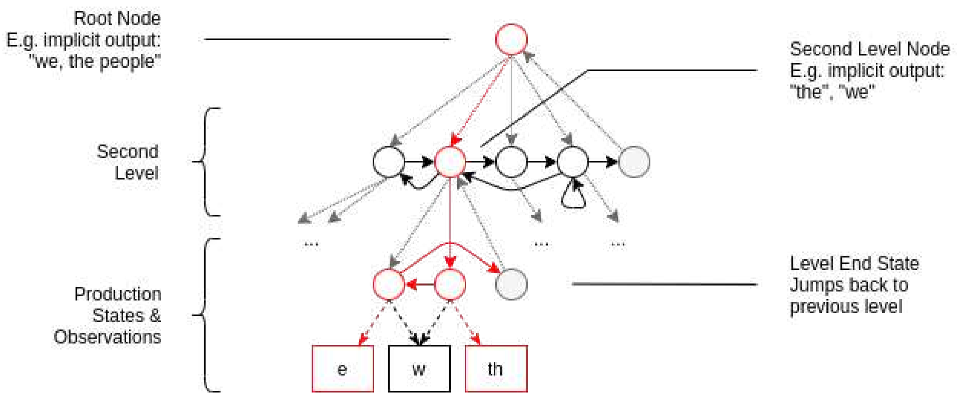

FIGURE 4.1: Hierarchical structure of HHMM with different levels of states. (Appel (2021))

4.2.3 Applications of HHMMs

HHMMs are particularly useful in applications where the data exhibits hierarchical or nested structures. Some common applications of HHMMs include: (Murphy and Paskin (2001))

- **Speech Recognition**: In speech recognition, HHMMs can model the hierarchical structure of speech, where phonemes form syllables, syllables form words, and words form sentences. This hierarchical modeling improves the accuracy of recognizing spoken language.

- **Activity Recognition**: HHMMs can be used to recognize complex activities composed of simpler sub-activities. For example, in human activity recognition, an HHMM can model activities such as “making coffee” as a sequence of sub-activities like “boiling water,” “grinding coffee beans,” and “pouring coffee”.

- **Natural Language Processing**: In natural language processing, HHMMs can model the hierarchical structure of text, where words form phrases, phrases form sentences, and sentences form paragraphs. This approach improves the ability to understand and generate natural language.

4.3 Coupled Hidden Markov Models

Coupled Hidden Markov Models (CHMMs) are an extension of the standard HMM framework designed to model multiple interacting processes. In CHMMs, several HMMs are coupled together, allowing the state of one process to influence the state transitions of other processes. This capability is particularly useful for systems where multiple dynamic processes interact with each other, such as in multi-sensor fusion or human-computer interaction. (Brand *et al.* (1997))

4.3.1 Structure of CHMMs

The structure of CHMMs is built to capture the interactions between multiple HMMs. Each HMM, referred to as a chain, represents a distinct process with its own states and observations. The chains are coupled through interaction terms, enabling the state of one chain to affect the state transitions of another.

In a typical CHMM, the overall state space is the Cartesian product of the state spaces of individual chains. This structure allows the model to represent complex dependencies between different processes. The main components of a CHMM structure include: (Brand *et al.* (1997))

- **Multiple Chains**: Each chain represents a separate HMM with its own states and transition probabilities.
- **State Dependency**: The state transitions in one chain can depend on the states of other chains, capturing the interdependencies between processes.
- **Joint State Space**: The combined state space is the product of the state spaces of all chains, representing all possible combinations of states across the chains

4.3.2 Key Components of CHMMs

Understanding the key components of CHMMs is essential for their implementation and application. These components include the state spaces, transition mechanisms, and observation processes for the coupled chains. (Brand *et al.* (1997))

- **State Spaces**: Each chain in a CHMM has its own state space, representing the possible states of the process it models. The joint state space of the CHMM is the Cartesian product of the individual state spaces.
- **Transition Mechanisms**: State transitions in a CHMM are influenced by the states of other chains. The transition probability for a state in one chain may depend on the current states of all chains, capturing the interactions between processes.

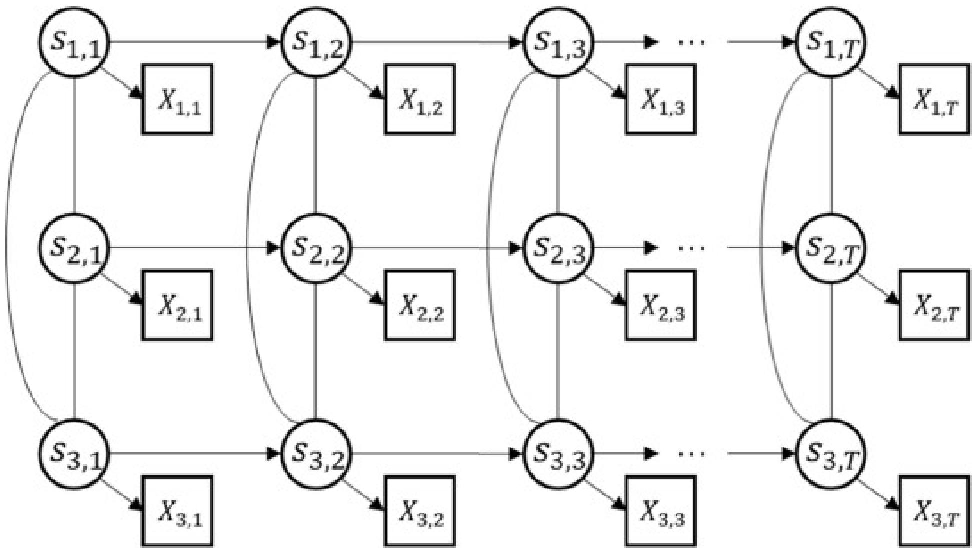

FIGURE 4.2: Coupled structure of CHMM with interacting HMMs (Zou *et al.* (2022))

- **Observation Processes**: Each chain generates observations based on its current state. The observation probability distributions for each chain are specified independently, allowing the CHMM to model complex observation patterns.

4.3.3 Applications of CHMMs

CHMMs are particularly useful in applications where multiple interacting processes need to be modeled. Some common applications of CHMMs include:

- **Multi-Sensor Fusion**: In multi-sensor fusion, CHMMs can integrate data from multiple sensors to improve the accuracy and reliability of recognition systems. For example, in a surveillance system, different sensors (e.g., cameras, microphones) provide complementary information that, when combined, offer a more comprehensive understanding of the environment. (Zhang and Lessmann (2008))

- **Human-Computer Interaction**: CHMMs can model interactions between different modalities, such as speech and gesture, in human-computer interaction. This capability enhances the system's ability to understand and respond to user inputs. (Oliver *et al.* (2000))

- **Financial Modeling**: In financial modeling, CHMMs can capture the dependencies between different financial indicators, such as stock prices, interest rates, and economic indicators. This enables more accurate predictions and insights for investment strategies and risk management. (Elliott *et al.* (2008))

4.4 Summary

In this chapter, we explored advanced extensions of the traditional Hidden Markov Model (HMM) framework, focusing on Hierarchical Hidden Markov Models (HHMMs) and Coupled Hidden Markov Models (CHMMs). These models enhance the flexibility and applicability of HMMs by addressing complex scenarios involving hierarchical structures and interacting processes.

Hierarchical Hidden Markov Models (HHMMs) introduce a multi-level structure that allows states at higher levels to invoke sequences of states at lower levels. This hierarchical approach is particularly effective in modeling nested temporal dependencies, making HHMMs suitable for applications such as speech recognition, activity recognition, and natural language processing. By capturing different levels of granularity, HHMMs improve the model's ability to represent and interpret complex patterns in sequential data.

Coupled Hidden Markov Models (CHMMs) extend HMMs by allowing multiple HMMs to interact with each other. This coupling enables the modeling of systems where multiple dynamic processes influence one another. CHMMs are particularly useful in applications like multi-sensor fusion, human-computer interaction, and financial modeling. By capturing the dependencies between different processes, CHMMs enhance the model's ability to integrate and analyze data from diverse sources, leading to more accurate and robust predictions.

The chapter provided detailed explanations of the structures and key components of both HHMMs and CHMMs. Examples and figures illustrated how these models can be applied to real-world problems, demonstrating their versatility and effectiveness.

In summary, the extensions and variants of HMMs discussed in this chapter offer powerful tools for modeling complex systems with hierarchical structures and interacting processes. These advanced models expand the applicability of HMMs and provide more nuanced and accurate representations of sequential data.

Chapter V

Applications of HMMs

5.1 Introduction to Applications

Hidden Markov Models (HMMs) have found widespread application across various domains due to their versatility and effectiveness in modeling sequential data. This chapter explores the practical applications of HMMs in several key areas: speech recognition, bioinformatics, finance, natural language processing (NLP), and robotics and control systems. By examining these applications, we aim to highlight the significant impact of HMMs and illustrate how the theoretical concepts discussed in previous chapters are implemented in real-world scenarios.

In each section, we will provide an overview of the application area, discuss how HMMs are utilized within that context, and present a case study to demonstrate the practical use of HMMs. These examples will showcase the flexibility and power of HMMs in addressing complex problems and improving the accuracy and efficiency of various systems. (Rabiner (1989))

5.2 Speech Recognition

Hidden Markov Models (HMMs) have revolutionized the field of speech recognition by providing a robust framework for modeling the variability and sequential nature of speech signals. This section explores how HMMs are applied in speech recognition systems, improving their accuracy and efficiency. (Rabiner (1989))

5.2.1 Overview

Speech recognition involves converting spoken language into text. The primary challenge lies in the variability of speech due to differences in accent, speed, intonation, and background noise. Traditional methods struggled to cope with these variations, but HMMs have proven to be highly effective in capturing the temporal dynamics of speech.

HMMs model speech as a sequence of observations (features extracted from the speech signal) and hidden states (underlying phonetic units). By leveraging the probabilistic nature of HMMs, speech recognition systems can handle variations in speech and accurately predict the most likely sequence of words corresponding to a given speech input. (Rabiner (1989))

5.2.2 HMMs in Speech Recognition

HMMs are used in speech recognition systems through several key steps: (Rabiner (1989))

1. Feature Extraction

The first step in speech recognition is to convert the audio signal into a sequence of feature vectors. This is typically done using techniques such as Mel-Frequency Cepstral Coefficients (MFCCs), which capture the essential characteristics of the speech signal.

2. Model Training

HMMs are trained on a large dataset of labeled speech to learn the transition and emission probabilities. This involves estimating the parameters of the HMM using algorithms such as the Baum-Welch algorithm. The training process results in an HMM for each phoneme or word in the vocabulary.

3. Decoding

During recognition, the trained HMMs are used to decode the sequence of feature vectors into the most likely sequence of words. The Viterbi algorithm is commonly used for this purpose, as it efficiently finds the optimal path through the state space that maximizes the likelihood of the observed sequence.

4. Language Modeling

In addition to the acoustic model provided by HMMs, a language model is used to incorporate syntactic and semantic constraints. This combination enhances the accuracy of the recognition system by favoring more likely word sequences.

5.2.3 Case Study: Speech-to-Text Systems

Speech-to-text systems convert spoken language into written text. These systems are widely used in applications such as virtual assistants, transcription services, and voice-controlled interfaces. (Rabiner (1989))

HMM Implementation:

- **Feature Extraction**: The audio signal is processed to extract MFCCs, which serve as the input to the HMM.
- **Model Training**: HMMs are trained on a large corpus of speech data. Each HMM represents a phoneme, and the models are trained using the Baum-Welch algorithm to estimate the transition and emission probabilities.
- **Decoding**: During recognition, the Viterbi algorithm is used to find the most likely sequence of phonemes given the sequence of MFCCs. This sequence is then mapped to words using a dictionary.
- **Language Modeling**: A language model is integrated to improve the accuracy by incorporating syntactic and semantic rules.

Example

Consider a virtual assistant like Siri or Google Assistant. When a user speaks a command, the system processes the audio signal, extracts features, and uses HMMs to decode the speech into text. The language model helps ensure the output text is coherent and contextually appropriate. This process allows the assistant to understand and respond to user commands accurately.

5.3 Bioinformatics

Hidden Markov Models (HMMs) have become essential tools in bioinformatics for analyzing biological sequences. By providing a probabilistic framework to model sequence data, HMMs enable researchers to identify patterns, predict structural elements, and infer evolutionary relationships within biological sequences. (Durbin *et al.* (1998))

5.3.1 Overview

Bioinformatics involves the application of computational techniques to understand and analyze biological data. DNA, RNA, and protein sequences are prime examples of sequential data in bioinformatics. These sequences exhibit complex patterns and dependencies that can be effectively modeled using HMMs.

HMMs are particularly useful in bioinformatics for tasks such as gene prediction, sequence alignment, and protein structure prediction. The probabilistic nature of HMMs allows for the modeling of biological variability and the identification of conserved motifs and domains within sequences. (Durbin *et al.* (1998))

5.3.2 HMMs in Bioinformatics

HMMs are applied in bioinformatics through several key processes: (Durbin *et al.* (1998))

1. Model Construction

HMMs are constructed to represent specific biological features or domains. For example, an HMM can be built to model a gene structure, including exons, introns, and regulatory regions.

2. Training

The parameters of the HMM are estimated using training data, which typically consists of known sequences annotated with the biological features of interest. The Baum-Welch algorithm is commonly used for parameter estimation.

3. Decoding

Given a new biological sequence, the HMM can be used to decode the sequence and identify regions that correspond to the modeled features. The Viterbi algorithm is often used to find the most likely path through the states of the HMM.

4. Multiple Sequence Alignment

HMMs can be employed to perform multiple sequence alignment by aligning sequences to a profile HMM. This approach captures the conserved regions across multiple sequences and highlights evolutionary relationships.

Multiple sequence alignment (MSA) is a critical task in bioinformatics that involves aligning three or more biological sequences—DNA, RNA, or proteins—to identify regions of similarity that may indicate functional, structural, or evolutionary relationships. Hidden Markov Models (HMMs) play an instrumental role in this process by providing a probabilistic framework that can account for the inherent variability and noise in biological data. In MSA, the objective is not only to find the optimal alignment of sequences but also to handle gaps and mismatches that may occur due to mutations or evolutionary divergence. HMMs are particularly well-suited for this task because they can model the insertion and deletion events through their state transition mechanisms, providing a robust means to generate alignments that reflect the true evolutionary history of the sequences.

One of the key advantages of using HMMs in MSA is their ability to capture conserved regions across sequences while also highlighting more variable or divergent sections. The HMM-based profile approach, known as a profile HMM, is often employed to align sequences to a consensus model, allowing researchers to align sequences even if they vary significantly from the template. This technique is especially valuable in cases where there is limited sequence homology, and traditional alignment methods might struggle. By leveraging the probabilistic nature of HMMs, MSA can reveal hidden patterns within sequences that are critical for understanding biological functions, identifying domains, and inferring evolutionary relationships. Thus, HMM-based MSAs provide a powerful and flexible tool in bioinformatics, enabling researchers to draw meaningful conclusions from large and complex datasets.

5.3.3 Case Study: Gene Prediction

Gene prediction involves identifying the regions of a DNA sequence that correspond to genes. This is a fundamental task in genomics, as genes encode the proteins essential for cellular functions. (Durbin *et al.* (1998))

HMM Implementation:

- **Model Construction**: An HMM is constructed to model the gene structure, including promoter regions, exons, introns, and termination sites. Each state in the HMM represents a different part of the gene structure.
- **Training**: The HMM is trained using a dataset of known gene sequences with annotated gene structures. The Baum-Welch algorithm is used to estimate the transition and emission probabilities.
- **Decoding**: For a new DNA sequence, the Viterbi algorithm is used to decode the sequence and identify the most likely gene regions. This involves finding the optimal path through the states that represents the gene structure.

Example

Consider the task of predicting genes in a newly sequenced genome. The HMM is applied to the DNA sequence, and the Viterbi algorithm identifies the regions that correspond to genes. This process allows researchers to annotate the genome with predicted gene locations, facilitating further analysis and functional studies.

5.4 Financial Modeling

Hidden Markov Models (HMMs) are extensively used in the field of finance for modeling and predicting time series data. Their ability to capture the temporal dependencies and underlying states of financial systems makes them invaluable for tasks such as stock market prediction, risk management, and algorithmic trading. (Elliott *et al.* (2008))

5.4.1 Overview

Financial markets are complex systems characterized by dynamic and often unpredictable behavior. Modeling such systems requires tools that can handle the inherent noise and variability. HMMs provide a probabilistic framework that captures the stochastic nature of financial time series, allowing for better understanding and prediction of market movements.

HMMs are particularly effective in identifying hidden states in financial data, such as bull and bear markets, and predicting future trends based on these states. By modeling the sequence of observed financial data and the underlying hidden states, HMMs offer a robust approach to financial analysis and decision-making.(Elliott *et al.* (2008))

5.4.2 HMMs in Financial Modeling

HMMs are applied in financial modeling through several key processes: (Elliott *et al.* (2008))

1. Model Construction

An HMM is constructed to represent the different states of the financial market. These states could represent various market conditions, such as high volatility, low volatility, uptrend, or downtrend.

2. Training

The parameters of the HMM are estimated using historical financial data. This involves identifying the transition probabilities between different market states and the emission probabilities of observed financial metrics within each state.

3. Decoding

Given new financial data, the HMM is used to decode the sequence and identify the most likely current state of the market. The Viterbi algorithm is often employed to find the optimal state sequence.

4. Prediction

Based on the identified states, the HMM can predict future market movements. This involves forecasting the probability of transitioning to different states and the expected financial metrics associated with those states.

5.4.3 Case Study: Stock Market Prediction

Stock market prediction involves forecasting future stock prices or market indices based on historical data. Accurate predictions can significantly enhance investment strategies and risk management. (Elliott *et al.* (2008))

HMM Implementation:

- **Model Construction**: An HMM is constructed with states representing different market conditions, such as bullish, bearish, and neutral markets. Each state has associated emission probabilities for stock price changes.
- **Training**: The HMM is trained using historical stock price data. The Baum-Welch algorithm is used to estimate the transition and emission probabilities.
- **Decoding**: For current market data, the Viterbi algorithm is used to decode the most likely sequence of market states. This helps in identifying the current market condition.
- **Prediction**: The HMM predicts future stock prices by forecasting the probability of transitioning to different market states and the expected price changes in those states.

Example

Consider a stock market prediction model for a major stock index. The HMM is applied to historical price data to identify patterns corresponding to different market conditions. Based on the current market state, the model predicts the likelihood of future price movements, helping investors make informed decisions about buying or selling stocks.

5.5 Natural Language Processing

Hidden Markov Models (HMMs) have been widely adopted in the field of Natural Language Processing (NLP) due to their effectiveness in modeling sequential data. They are particularly useful for tasks such as part-of-speech tagging, named entity recognition, and language modeling, where understanding the sequence and structure of words is crucial. (Manning and Schütze (1999))

5.5.1 Overview

Natural Language Processing involves the interaction between computers and human languages. It encompasses a range of tasks that require the understanding, interpretation, and generation of human language. HMMs provide a probabilistic framework that captures the dependencies between words and their parts of speech, making them suitable for various NLP applications.

HMMs in NLP help in disambiguating words, predicting the next word in a sequence, and identifying entities in text. By modeling the sequence of words and their underlying states, HMMs enhance the accuracy of language processing tasks. (Manning and Schütze (1999))

5.5.2 HMMs in NLP

HMMs are applied in NLP through several key processes: (Manning and Schütze (1999))

1. Model Construction

An HMM is constructed to represent the sequence of words and their associated parts of speech or named entities. Each state in the HMM corresponds to a part of speech or entity type.

2. Training

The parameters of the HMM are estimated using a labeled corpus of text. The Baum-Welch algorithm is commonly used for parameter estimation, which involves learning the transition probabilities between states and the emission probabilities of words given the states.

3. Decoding

Given a new sequence of words, the HMM is used to decode the sequence and identify the most likely sequence of parts of speech or named entities. The Viterbi algorithm is often employed for this purpose.

4. Prediction

Based on the identified sequence, the HMM can predict the next word or the part of speech for a given word in a sentence. This involves calculating the probability of different sequences and choosing the most likely one.

5.5.3 Case Study: Part-of-Speech Tagging

Part-of-speech tagging involves assigning a part of speech to each word in a sentence. This is a fundamental task in NLP that helps in understanding the grammatical structure of a sentence. (Manning and Schütze (1999))

HMM Implementation:

- **Model Construction**: An HMM is constructed with states representing different parts of speech, such as nouns, verbs, adjectives, and prepositions. Each state has associated emission probabilities for the words that commonly belong to that part of speech.
- **Training**: The HMM is trained using a labeled corpus where each word is annotated with its part of speech. The Baum-Welch algorithm is used to estimate the transition and emission probabilities.
- **Decoding**: For a new sentence, the Viterbi algorithm is used to decode the most likely sequence of parts of speech. This helps in tagging each word with its appropriate part of speech.
- **Prediction**: The HMM predicts the part of speech for each word in the sentence, enhancing the understanding of the sentence structure.

Example

Consider the sentence "The quick brown fox jumps over the lazy dog." The HMM is applied to this sentence, and the Viterbi algorithm decodes the sequence as follows:

- The (Determiner)
- quick (Adjective)
- brown (Adjective)
- fox (Noun)
- jumps (Verb)
- over (Preposition)
- the (Determiner)
- lazy (Adjective)
- dog (Noun)

This tagging helps in understanding the grammatical structure of the sentence, which can be useful for further NLP tasks such as parsing and machine translation.

5.6 Robotics and Control Systems

Hidden Markov Models (HMMs) are widely used in robotics and control systems to model and predict the behavior of dynamic systems. Their ability to handle sequential data and incorporate probabilistic reasoning makes them ideal for applications such as autonomous navigation, fault detection, and human-robot interaction. (Thrun *et al.* (2005))

5.6.1 Overview

Robotics and control systems involve the interaction between computational algorithms and physical processes. These systems require accurate modeling and prediction of states to function effectively. HMMs provide a robust framework for representing the uncertainty and variability inherent in these systems.

In robotics, HMMs can be used to model the states of a robot and predict its movements based on sensor data. In control systems, HMMs help in monitoring system states and detecting anomalies, ensuring safe and efficient operation.(Thrun *et al.* (2005))

5.6.2 HMMs in Robotics and Control

HMMs are applied in Robotics and Control through several key processes: (Thrun *et al.* (2005))

1. Model Construction

An HMM is constructed to represent the states of the system. In robotics, states can include different positions and orientations of the robot. In control systems, states can represent various operational modes.

2. Training

The parameters of the HMM are estimated using historical data. This involves learning the transition probabilities between states and the emission probabilities of observed signals or sensor readings.

3. Decoding

Given new sensor data, the HMM is used to decode the sequence and identify the most likely current state of the system. The Viterbi algorithm is often used for this purpose.

4. Prediction and Control

Based on the identified states, the HMM can predict future states and help in making control decisions. This involves calculating the probability of transitioning to different states and selecting the optimal control actions.

5.6.3 Case Study: Autonomous Navigation

Autonomous navigation involves the ability of a robot or vehicle to navigate through an environment without human intervention. Accurate state estimation and prediction are crucial for successful navigation. (Thrun *et al.* (2005))

HMM Implementation:

- **Model Construction**: An HMM is constructed with states representing different positions and orientations of the robot. Each state has associated emission probabilities for sensor readings, such as distance measurements from LIDAR or cameras.
- **Training**: The HMM is trained using data collected from the robot navigating in different environments. The Baum-Welch algorithm is used to estimate the transition and emission probabilities.
- **Decoding**: For new sensor data, the Viterbi algorithm is used to decode the most likely sequence of states. This helps in determining the robot's current position and orientation.
- **Prediction**: The HMM predicts future states based on current sensor data and helps in making navigation decisions. For example, if the HMM predicts an obstacle ahead, the control system can adjust the robot's path to avoid it.

Example

Consider an autonomous robot navigating through a building. The HMM is applied to sensor data from LIDAR and cameras to estimate the robot's position and orientation. The Viterbi algorithm decodes the most likely state sequence, helping the robot navigate through corridors and avoid obstacles. The HMM also predicts potential obstacles and guides the robot in real-time, ensuring safe and efficient navigation.

5.7 Summary

In this chapter, we explored the diverse applications of Hidden Markov Models (HMMs) across several key domains, demonstrating their versatility and effectiveness in handling sequential data and making predictions under uncertainty.

Speech Recognition

HMMs have significantly advanced speech recognition technology by providing a framework to model the temporal dynamics of speech. They enable the conversion of spoken language into text with high accuracy, handling variations in speech patterns, accents, and background noise. Through feature extraction, model training, decoding, and language modeling, HMMs form the backbone of modern speech-to-text systems.

Bioinformatics

In bioinformatics, HMMs are instrumental in analyzing biological sequences, such as DNA, RNA, and proteins. They facilitate tasks like gene prediction, sequence alignment, and protein structure prediction by capturing the complex patterns and dependencies within biological data. HMMs help in identifying conserved motifs and structural elements, aiding in the understanding of genetic information and evolutionary relationships.

Finance

HMMs are widely used in finance for modeling and predicting time series data. They capture the stochastic nature of financial markets and identify hidden states, such as bullish or bearish conditions. By constructing models, training on historical data, decoding current market states, and making predictions, HMMs enhance stock market prediction, risk management, and algorithmic trading strategies.

Natural Language Processing (NLP)

In NLP, HMMs are employed for tasks such as part-of-speech tagging, named entity recognition, and language modeling. They effectively model the sequence of words and their parts of speech, improving the understanding and generation of human language. HMMs enable accurate tagging of words in a sentence, facilitating further NLP tasks like parsing and machine translation.

Robotics and Control Systems

HMMs are crucial in robotics and control systems for modeling and predicting the behavior of dynamic systems. They support autonomous navigation, fault detection, and human-robot interaction by representing the states of a robot and predicting its movements based on sensor data. HMMs help in making control decisions and ensuring safe and efficient operation of robots and control systems.

Through detailed examples and case studies, we illustrated how HMMs are implemented and utilized in each domain. These applications highlight the power of

HMMs in addressing complex problems, enhancing prediction accuracy, and improving decision-making processes.

In conclusion, the applications of HMMs span a wide range of fields, showcasing their adaptability and robustness. By capturing temporal dependencies and handling uncertainties, HMMs provide valuable insights and solutions in speech recognition, bioinformatics, finance, NLP, and robotics. As technology and methodologies continue to evolve, the role of HMMs in these and other domains is expected to expand further, driving innovation and advancing our understanding of dynamic systems.

Chapter VI

Part of Speech tagging with HHM

6.1 What is Part of Speech (POS) Tagging?

Part of Speech (POS) tagging involves assigning each word in a sentence its corresponding part of speech, such as noun, verb, adjective, or adverb. This process, known as POS annotation, categorizes words into lexical tags or word classes. Traditionally, POS tagging was a manual task performed by human annotators. However, due to its labor-intensive nature, modern POS tagging is largely automated, with manual annotation reserved for creating training datasets for new automatic taggers.

POS tags provide significant information about words and their context, facilitating various NLP tasks. These tags are essential for applications such as information retrieval, parsing, text-to-speech systems, information extraction, and linguistic research. Additionally, POS tagging serves as a crucial intermediate step for higher-level NLP functions like parsing, semantic analysis, and machine translation, making it a foundational component in advanced natural language processing applications. (Great Learning Team (2022)

6.2 Techniques for POS Tagging

There are several techniques utilized for POS tagging, each offering unique advantages. Here are some common methods:

1. Rule-based POS Tagging

This technique applies a set of manually crafted rules that use contextual information to assign POS tags to words. These context frame rules, such as "If an ambiguous/unknown word ends with 'ing' and follows a verb, label it as a verb," guide the tagging process. Rule-based systems are straightforward but can struggle with exceptions and nuances in language.

2. Transformation-based Tagging

This method combines predefined rules with rules automatically generated during training. The system iteratively refines its rules to improve accuracy. Transformation-based approaches, also known as Brill tagging, start with a rough guess and iteratively correct mistakes by applying transformation rules learned from the training data.

3. Deep Learning Models

Advanced models like Meta-BiLSTM have been employed for POS tagging, achieving impressive accuracy rates of around 97%. These models leverage large datasets and deep neural networks to capture complex patterns in the data. Deep learning approaches can generalize well from large amounts of data, capturing subtleties in word usage and context that simpler models might miss.

4. Stochastic (Probabilistic) Tagging

This approach uses statistical methods to assign POS tags based on frequency and probability. The simplest stochastic method tags words according to their most frequent tag in the training data. More sophisticated methods, such as Hidden Markov Models (HMMs), calculate the probabilities of different tag sequences and select the sequence with the highest probability, ensuring grammatical consistency. This probabilistic approach balances between observed frequencies and contextual probabilities to produce the most likely tag sequences.

Hybrid Approaches:

Some systems combine rule-based, probabilistic, and machine learning methods to enhance tagging accuracy and robustness. By leveraging the strengths of multiple techniques, hybrid approaches can address the limitations of individual methods.

Contextual Tagging:

Leveraging contextual cues, these techniques can improve the accuracy of tagging homonyms and polysemous words by considering the surrounding words in a sentence. This is particularly useful in disambiguating words that can serve multiple grammatical functions.

These techniques have evolved significantly, providing robust tools for accurately tagging large corpora, thereby facilitating more complex NLP tasks. (Great Learning Team (2022)

6.3 POS Tagging with Hidden Markov Model

Hidden Markov Model (HMM) is a probabilistic technique widely used for POS tagging. HMMs are notable for their applications in various fields, including reinforcement learning and temporal pattern recognition such as speech, handwriting, gesture recognition, musical score following, and bioinformatics.

To illustrate how HMM selects an appropriate tag sequence for a sentence, consider the example by Dr. Luis Serrano. The process involves calculating the transition and emission probabilities to determine the likelihood of different sequences of POS tags.

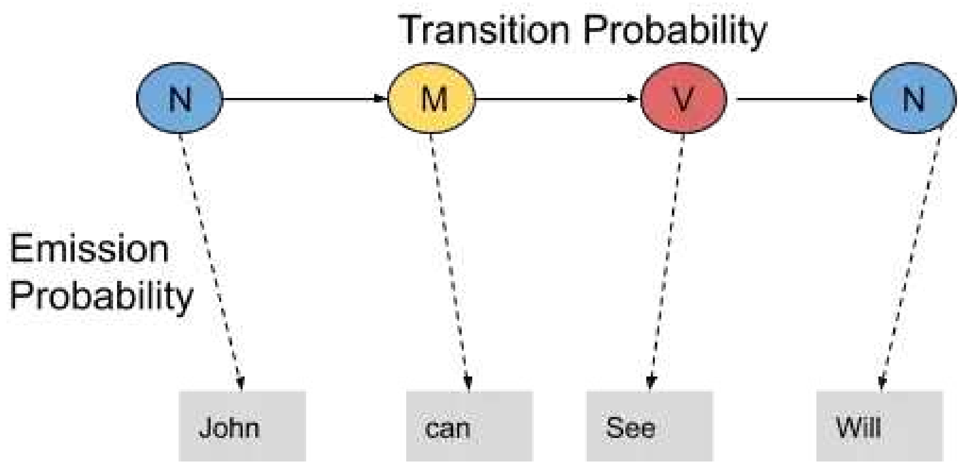

FIGURE 6.1: Transition and Emission Probabilities for HMM (Great Learning Team (2022)

In this example, we consider only three POS tags: noun, model, and verb. For the sentence "Ted will spot Will," tagged as noun, model, verb, and noun, we need to calculate the probability associated with this sequence of tags using their Transition and Emission probabilities.

Transition probability refers to the likelihood of a particular sequence, such as a noun followed by a model, a model followed by a verb, and a verb followed by a noun. This probability is crucial for determining if a sequence is likely to be correct.

Next, we determine the probability that each word in the sentence "Ted will spot Will" corresponds to its respective POS tag. These probabilities are known as Emission probabilities and are important for ensuring accurate tagging.

Let's calculate these probabilities for the following sentences:

- Mary Jane can see Will
- Spot will see Mary
- Will Jane spot Mary?

- Mary will pat Spot

Note that Mary, Jane, Spot, and Will are all names.

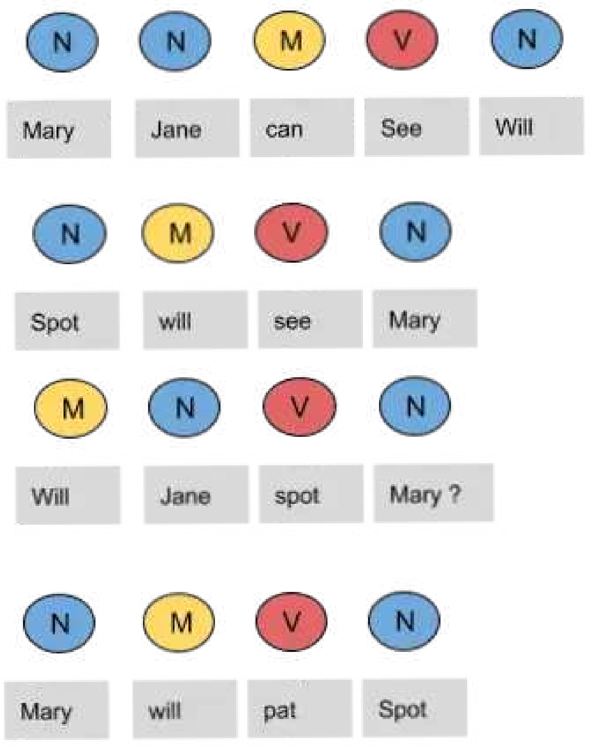

FIGURE 6.2: Example Senteces with POS Tags (Great Learning Team (2022)

In the provided sentences, the word "Mary" appears four times as a noun. To calculate the emission probabilities, we can construct a counting table as follows:

TABLE 6.1: Counting Table for Emission Probabilities (Great Learning Team (2022)

Words	Noun	Model	Verb
Mary	4	0	0
Jane	2	0	0
Will	1	3	0
Spot	2	0	1
Can	0	1	0
See	0	0	2
Pat	0	0	1

Next, we divide each column by the total number of their occurrences. For instance, the word "noun" appears nine times in the sentences above, so we divide each value in the noun column by 9. The resulting table after this operation is shown below:

From the table, we can infer the following probabilities:

TABLE 6.2: Normalized Counting Table for Emission Probabilities (Great Learning Team (2022)

Words	Noun	Model	Verb
Mary	$\frac{4}{9}$	0	0
Jane	$\frac{2}{9}$	0	0
Will	$\frac{1}{9}$	$\frac{3}{4}$	0
Spot	$\frac{2}{9}$	0	$\frac{1}{4}$
Can	0	$\frac{1}{4}$	0
See	0	0	$\frac{2}{4}$
Pat	0	0	1

- The probability that "Mary" is a Noun is $\frac{4}{9}$
- The probability that "Mary" is a Model is 0
- The probability that "Will" is a Noun is $\frac{1}{9}$
- The probability that "Will" is a Model is $\frac{3}{4}$

Similarly, we can determine the rest of the emission probabilities.

Next, we need to calculate the transition probabilities. For this, we define two additional tags: < S > and < E > . The tag < S > is placed at the beginning of each sentence, and < E > is placed at the end, as shown in the figure below.

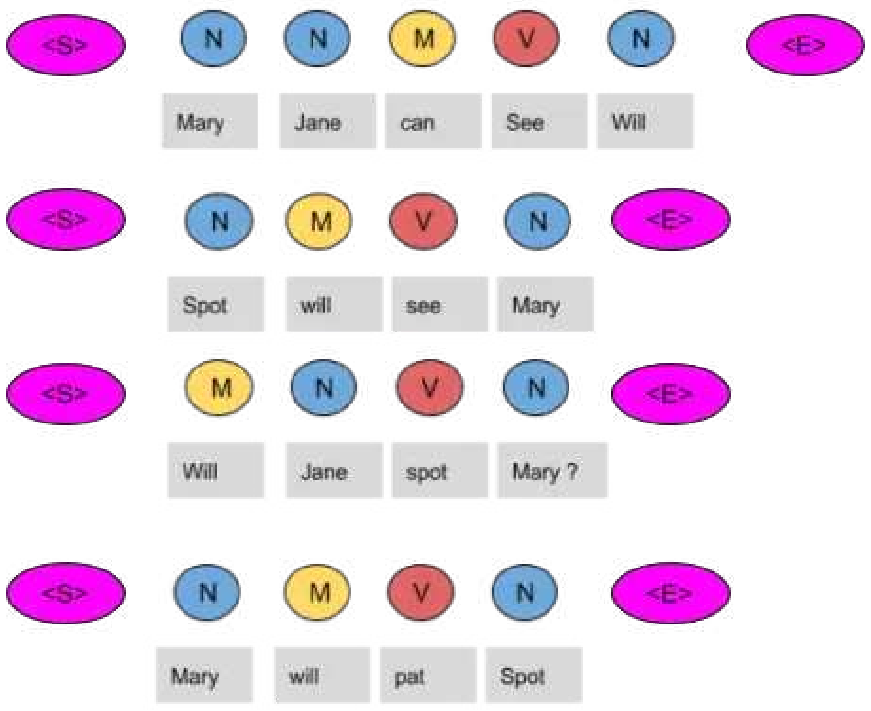

FIGURE 6.3: Example Sentences with POS Tags and Transition Tags $< S >$ and $< E >$ (Great Learning Team (2022)

Let's create another table and populate it with the co-occurrence counts of the tags.

TABLE 6.3: Co-occurrence Counts of the Tags (Great Learning Team (2022))

	N	M	V	$< E >$
$< S >$	3	1	0	0
N	1	3	1	4
M	1	0	3	0
V	4	0	0	0

In the above figure, we see that the $< S >$ tag is followed by the N tag three times, thus the first entry is 3. The model tag follows the $< S >$ just once, so the second entry is 1. The rest of the table is filled in a similar manner.

Next, we divide each term in a row by the total number of co-occurrences of the tag in consideration. For instance, the Model tag is followed by any other tag four times, so we divide each element in the third row by four.

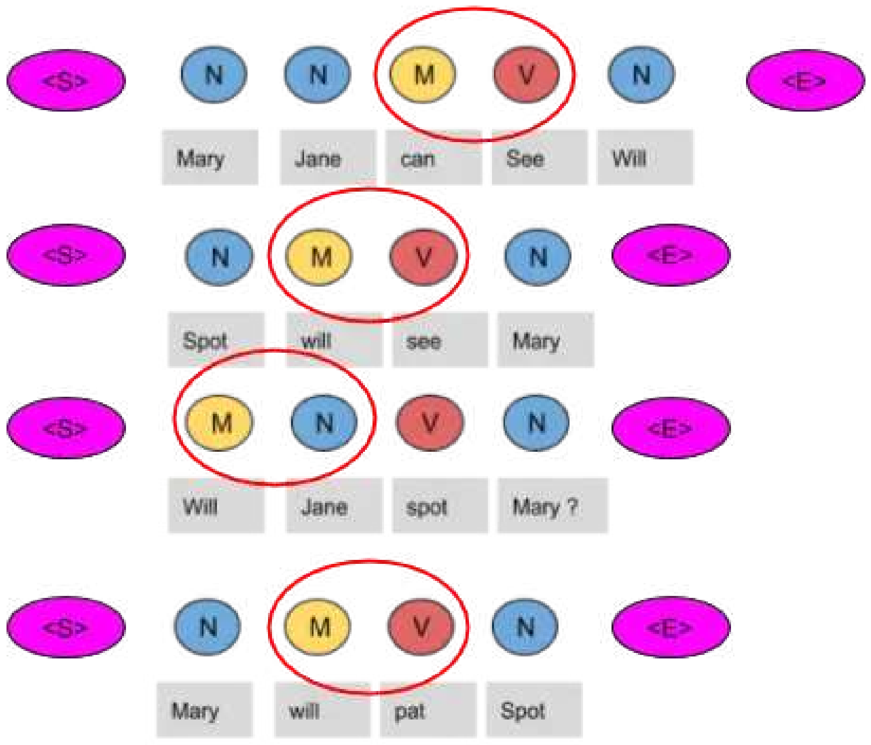

FIGURE 6.4: Example Sentences with POS Tags and Transition Tags $< S >$ and $< E >$ (Great Learning Team (2022))

TABLE 6.4: Normalized Co-occurrence Counts of the Tags (Great Learning Team (2022))

	N	M	V	$< E >$
$< S >$	$\frac{3}{4}$	$\frac{1}{4}$	0	0
N	$\frac{1}{9}$	$\frac{3}{9}$	$\frac{1}{9}$	$\frac{4}{9}$
M	$\frac{1}{4}$	0	$\frac{3}{4}$	0
V	$\frac{4}{4}$	0	0	0

These are the respective transition probabilities for the above four sentences. Now how does the HMM determine the appropriate sequence of tags for a particular sentence from the above tables? Let us find it out.

Take a new sentence and tag them with wrong tags. Let the sentence, ' Will can spot Mary' be tagged as:

- Will as a model
- Can as a verb
- Spot as a noun
- Mary as a noun

Now calculate the probability of this sequence being correct in the following manner.

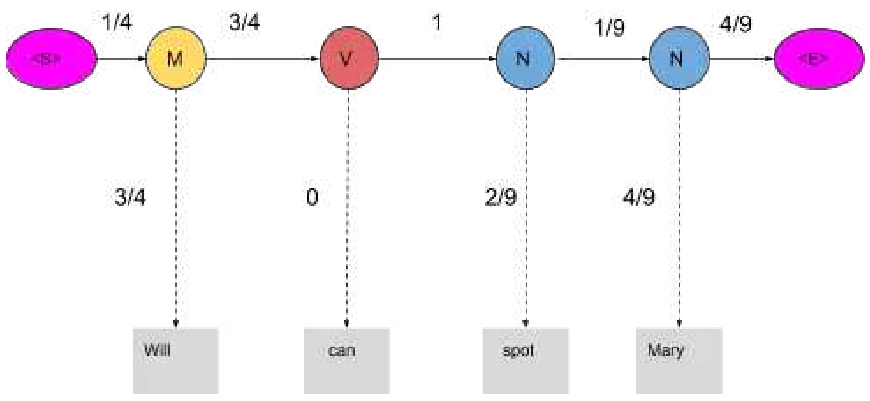

FIGURE 6.5: Sequence Probability Calculation (Great Learning Team (2022))

The probability of the tag Model (M) following the tag is $\frac{1}{4}$ as seen in the table. Additionally, the probability that the word "Will" is a Model is $\frac{3}{4}$. Similarly, we calculate each probability in the graph. The product of these probabilities represents the likelihood that this sequence is correct. Since the tags are not correct, the product is zero:

$$\frac{1}{4} \times \frac{3}{4} \times \frac{3}{4} \times 0 \times 1 \times \frac{2}{9} \times \frac{1}{9} \times \frac{4}{9} \times \frac{4}{9} = 0$$

When these words are correctly tagged, we get a probability greater than zero, as shown below.

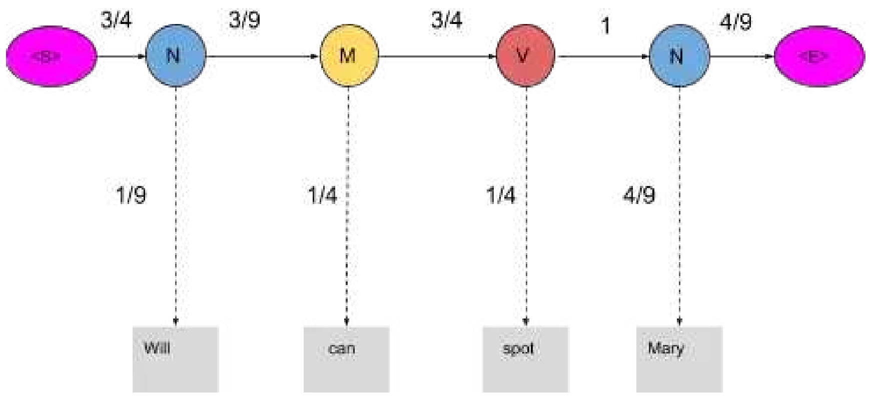

FIGURE 6.6: Sequence Probability Calculation (Great Learning Team (2022))

By calculating the product of these terms, we get:

$$\frac{3}{4} \times \frac{1}{9} \times \frac{3}{9} \times \frac{1}{4} \times \frac{3}{4} \times 1 \times \frac{4}{9} = 0.00025720164$$

For our example, considering just the three POS tags mentioned, 81 different combinations of tags can be formed. Calculating the probabilities for all 81 combinations seems feasible. However, when tagging a larger sentence and including all the POS tags in the Penn Treebank project, the number of possible combinations increases exponentially, making the task seemingly impossible to accomplish.

Let's visualize these 81 combinations as paths, using the transition and emission probability to mark each vertex and edge, as shown below.

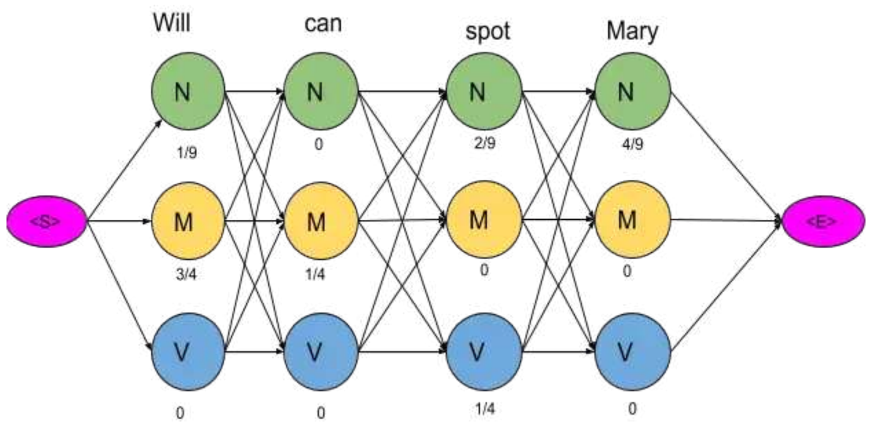

FIGURE 6.7: Full Probability Paths (Great Learning Team (2022)

The next step is to remove all vertices and edges with zero probability, as well as any vertices that do not lead to the endpoint. Also, we will highlight the remaining paths.

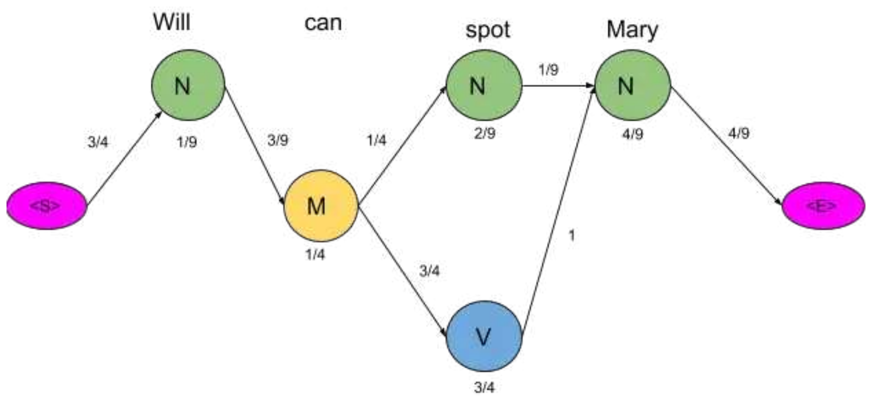

FIGURE 6.8: Pruned Probability Paths (Great Learning Team (2022)

Now, there are only two paths that lead to the endpoint. Let's calculate the probability associated with each path:

For the path $< S > \rightarrow N \rightarrow M \rightarrow N \rightarrow N \rightarrow < E >$:

$$\frac{3}{4} \times \frac{1}{9} \times \frac{3}{9} \times \frac{1}{4} \times \frac{1}{4} \times \frac{2}{9} \times \frac{1}{9} \times \frac{4}{9} \times \frac{4}{9} = 0.00000846754$$

For the path $< S > \rightarrow N \rightarrow M \rightarrow N \rightarrow V \rightarrow < E >$:

$$\frac{3}{4} \times \frac{1}{9} \times \frac{3}{9} \times \frac{1}{4} \times \frac{3}{4} \times \frac{1}{4} \times 1 \times \frac{4}{9} \times \frac{4}{9} = 0.00025720164$$

Clearly, the probability of the second sequence is much higher. Therefore, the HMM will tag each word in the sentence according to this sequence. (Great Learning Team (2022)

6.4 Optimizing HMM with Viterbi Algorithm

The Viterbi algorithm is a dynamic programming method used to identify the most probable sequence of hidden states, known as the Viterbi path, that results in a sequence of observed events. This is particularly useful in the context of Markov information sources and hidden Markov models (HMM).

In the previous section, we optimized the HMM and reduced our calculations from 81 possible sequences to just two. Now, we will further optimize the HMM by applying the Viterbi algorithm. Let's use the same example as before and demonstrate the application of the Viterbi algorithm.

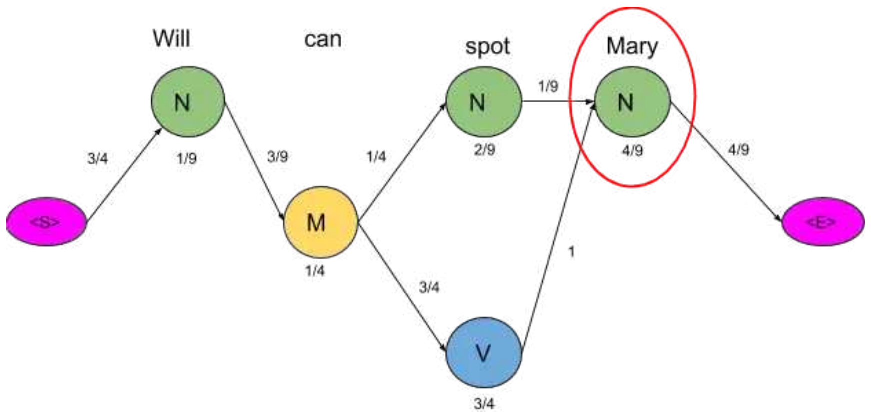

FIGURE 6.9: Paths Leading to the Circled Vertex with Probabilities (Great Learning Team (2022)

Consider the vertex circled in the example above. There are two paths leading to this vertex, as shown below, along with the probabilities of these two mini-paths.

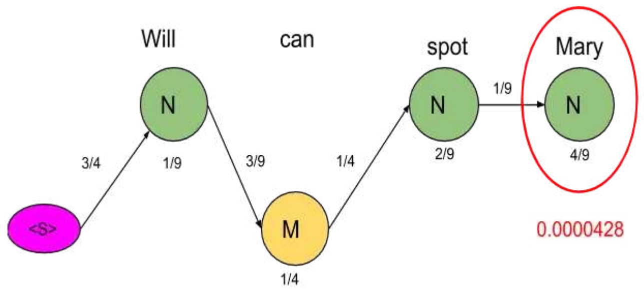

FIGURE 6.10: Paths Leading to the Circled Vertex with Probabilities (Great Learning Team (2022)

Now we are really concerned with the mini path having the lowest probability. The same procedure is done for all the states in the graph as shown in the figure below

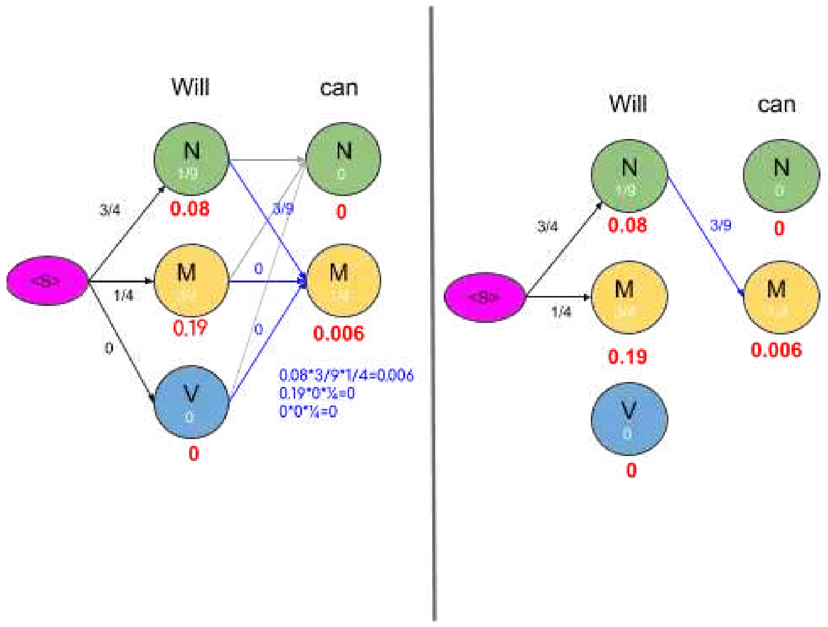

FIGURE 6.11: Paths Leading to the Circled Vertex with Probabilities (Great Learning Team (2022))

As illustrated in the figure above, we calculate the probabilities of all paths leading to a node and then remove the edges or paths with lower probability. Additionally, nodes with a probability of zero have no edges attached to them, as all paths to these nodes have zero probability. The graph obtained after computing the probabilities of all paths leading to each node is shown below.

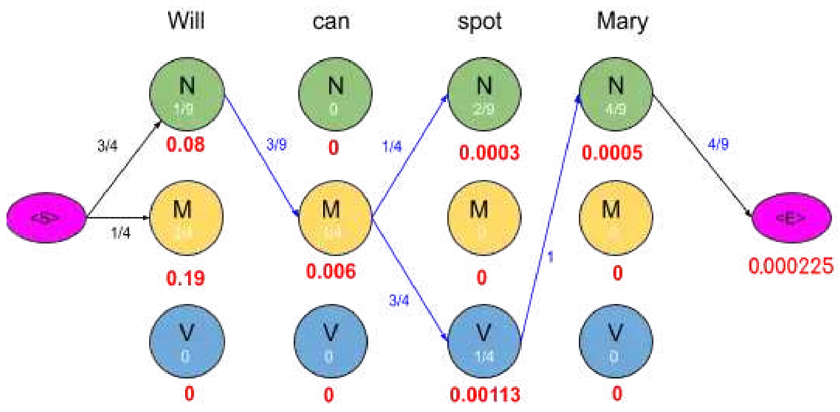

FIGURE 6.12: Computed Probabilities for Each Path (Great Learning Team (2022))

To obtain an optimal path, we start from the end and trace backward, since each state has only one incoming edge. This gives us the path shown below.

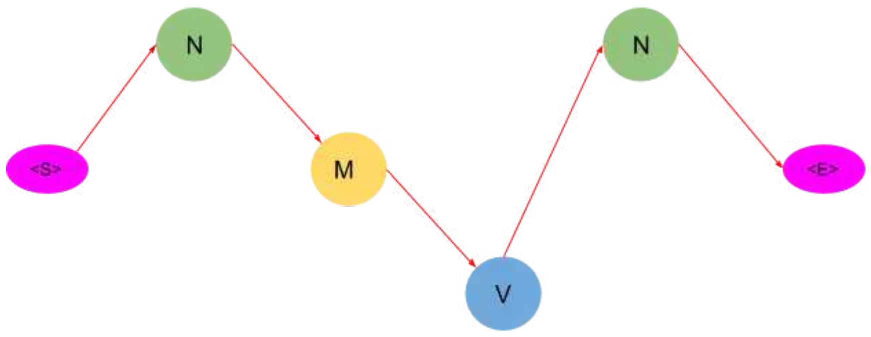

FIGURE 6.13: Optimal Path (Great Learning Team (2022)

As you may have observed, this algorithm returns only one path, whereas the previous method suggested two paths. By using the Viterbi algorithm, we have significantly reduced the number of computations.

After applying the Viterbi algorithm, the model tags the sentence as follows:

- Will as a noun
- Can as a model
- Spot as a verb
- Mary as a noun

These tags are correct, indicating that the model successfully tags the words with their appropriate POS tags. (Great Learning Team (2022)

6.5 Summary

Introduction to POS Tagging

POS tagging involves assigning each word in a sentence its corresponding part of speech, such as noun, verb, adjective, or adverb. Traditionally a manual task, POS tagging has largely been automated in modern applications, with manual annotation reserved for creating training datasets. POS tags provide valuable contextual information for various NLP tasks, such as parsing, information retrieval, text-to-speech systems, and more advanced NLP applications like machine translation.

Techniques for POS Tagging

Several techniques are utilized for POS tagging:

1. Rule-based POS Tagging: Applies manually crafted rules using contextual information to assign tags.

2. Transformation-based Tagging: Combines predefined rules with rules generated during training.

3. Deep Learning Models: Utilizes advanced models like Meta-BiLSTM to achieve high accuracy.

4. Stochastic (Probabilistic) Tagging: Uses statistical methods like Hidden Markov Models (HMMs) to assign tags based on probability.

Hybrid approaches combine these techniques to enhance accuracy, while contextual tagging leverages surrounding words to improve disambiguation.

POS Tagging with Hidden Markov Models (HMMs)

HMMs are widely used in POS tagging due to their ability to model sequential data probabilistically. The process involves calculating transition and emission probabilities to determine the most likely sequence of POS tags for a given sentence. Transition probabilities determine the likelihood of one POS tag following another, while emission probabilities estimate the likelihood of a word corresponding to a specific POS tag.

An example is provided where sentences are tagged, and the emission probabilities are calculated by normalizing the counts of word occurrences as specific tags. Transition probabilities are also calculated and used to determine the most likely tag sequence for a sentence. The example shows how the HMM can correctly identify POS tags based on these calculated probabilities.

Optimizing HMM with the Viterbi Algorithm

The Viterbi algorithm is a dynamic programming technique used to find the most probable sequence of hidden states (POS tags) that produce a given sequence of observed events (words). By applying the Viterbi algorithm, the HMM can be further optimized to reduce the number of possible tag sequences, minimizing computational complexity.

The section demonstrates the application of the Viterbi algorithm to the earlier example, showing how it effectively identifies the correct POS tags for a sentence by tracing the most likely path through the sequence of states. This optimization results in accurate POS tagging with reduced computational effort.

Chapter VII

Conclusion

This thesis has provided an extensive exploration of Hidden Markov Models (HMMs), offering a thorough understanding of their theoretical foundations, key algorithms, and diverse practical applications. Throughout this journey, we have traversed from the core concepts that define HMMs to their sophisticated uses in various domains, thereby underscoring their significance in both academic research and real-world problem-solving.

We began by establishing a solid foundation in the fundamental concepts of HMMs, dissecting their components—states, observations, transition probabilities, and emission probabilities—and the relationships between them. This foundational knowledge set the stage for a deeper dive into the different types of HMMs, including their standard form and the more advanced variants such as Hierarchical Hidden Markov Models (HHMMs) and Coupled Hidden Markov Models (CHMMs). By understanding these variations, we appreciated the flexibility of HMMs in modeling complex systems with multiple layers of dependencies or interacting processes.

Central to the utility of HMMs are the algorithms that enable their application, and this thesis devoted significant attention to these key algorithms. The Forward-Backward algorithm was explored as a means to compute the probability of a sequence of observations, while the Viterbi algorithm was highlighted for its role in decoding the most likely sequence of hidden states. The Baum-Welch algorithm was presented as a method for estimating the parameters of an HMM when they are unknown, ensuring that the model can be trained effectively on data. Together, these algorithms form the backbone of HMM functionality, enabling their use in a wide array of sequential data problems.

In addition to algorithmic insights, this thesis discussed essential estimation techniques that are critical for implementing HMMs effectively. Maximum Likelihood Estimation (MLE) was detailed as a foundational method for parameter estimation, focusing on maximizing the likelihood of the observed data. Meanwhile, Bayesian

Estimation Methods provided a probabilistic framework for incorporating prior knowledge into the model, offering a robust approach to parameter estimation, particularly when dealing with limited data. These techniques contribute to the reliability and accuracy of HMMs in practical applications.

Model selection and evaluation metrics were also a key focus, ensuring that the HMMs developed are not only theoretically sound but also effective in practice. Techniques for selecting the optimal model complexity were discussed, such as choosing the appropriate number of hidden states to avoid overfitting or underfitting. The use of cross-validation, along with metrics like Log-Likelihood, Akaike Information Criterion (AIC), and Bayesian Information Criterion (BIC), provided a comprehensive framework for evaluating model performance and ensuring that the chosen model generalizes well to new data.

A significant portion of this thesis was dedicated to the application of HMMs in Part of Speech (POS) tagging within Natural Language Processing (NLP). This practical case study demonstrated the utility of HMMs in tagging words with their appropriate parts of speech, which is a crucial task in various NLP applications such as machine translation, information retrieval, and text-to-speech systems. Different techniques for POS tagging were explored, including rule-based tagging, transformation-based tagging, deep learning models, and stochastic approaches. The optimization of HMMs with the Viterbi algorithm was particularly emphasized, showcasing how these models can be fine-tuned for enhanced accuracy and efficiency. Through practical examples and visual aids, this section provided a tangible understanding of how HMMs operate in real-world NLP tasks.

The versatility and adaptability of HMMs, as highlighted throughout this thesis, make them an indispensable tool for modeling sequential data and making predictions based on hidden states.Their applications are vast, spanning from speech recognition—where they model the temporal dynamics of speech signals—to bioinformatics, where they are used to analyze biological sequences and infer evolutionary relationships. In finance, HMMs are employed to model and predict time series data, aiding in stock market analysis and risk management. In robotics and control systems, HMMs are used to model dynamic systems, enabling autonomous navigation and fault detection.

This thesis has underscored the importance of HMMs in both theoretical and practical contexts, demonstrating their broad utility and potential for innovation. As data becomes increasingly complex and the demand for intelligent systems grows, the role of HMMs is likely to expand further. The work presented here not only contributes to the understanding of HMMs but also lays the groundwork for continued research and development in this dynamic field. By combining theoretical rigor with practical applications, this thesis has highlighted the enduring relevance of HMMs in modern computational science, opening new avenues for their use in future innovations.

In conclusion, Hidden Markov Models remain a powerful and flexible tool in the arsenal of data scientists, engineers, and researchers. As we continue to explore the vast potential of sequential data modeling, HMMs will undoubtedly play a crucial role in shaping the future of technology and science. The insights gained from this thesis will hopefully inspire further exploration and application of HMMs across an even broader range of disciplines.

Bibliography

(2024) 'Probability of transition and observation in hmm.' Inspiration from here. Available at: https://ristohinno.medium.com (Accessed: 16th August 2024)

Anzhi, Y. (2023) 'Example of a hidden semi-markov model (hsmm).' Accessed: 2024-08-16. Available at: https://www.researchgate.net/figure/Example-of-a-hidden-semi-Markov-model-HSMM_fig5_324387006

Appel, J. (2021) 'Hierarchical hidden markov models: An approximation of neocortex structures, according to ray kurzweil.' Accessed: August 17, 2024. Available at: https://towardsdatascience.com/hierarchical-hidden-markov-models-an-approximation-of-neocortex-structures-according-to-ray-kurzweil-5f4dd554edc1

Atrey, P. K. *et al.* (2010) 'Multimodal fusion for multimedia analysis: A survey.' *Multimedia Systems*, 16(6), pp. 345–379.

Baker, J. K. (1975) 'Stochastic modeling for automatic speech understanding.' *Proceedings of the 1975 IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 193–195.

Baum, L. E. and Petrie, T. (1970) 'Maximization techniques in the statistical analysis of probabilistic functions of markov chains.' *The Annals of Mathematical Statistics*, 41(1), pp. 164–171.

Bishop, C. M. (2006) 'Pattern recognition and machine learning.'

Blackwell, D. and Koopmans, L. (1957) 'On the identifiability problem for functions of finite markov chains.' *The Annals of Mathematical Statistics*, 28(4), pp. 1011–1015. Accessed: 07-08-2024 12:29 UTC. Available at: https://www.jstor.org/stable/2237062

Blei, D. M., Ng, A. Y. and Jordan, M. I. (2003) 'Latent dirichlet allocation.' *Journal of Machine Learning Research*, 3, pp. 993–1022.

Brand, M., Oliver, N. and Pentland, A. (1997) 'Coupled hidden markov models for complex action recognition.' In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 994–999. IEEE.

Bui, H. H., Venkatesh, S. and West, G. (2002) 'Policy recognition in the abstract hidden markov model.' *Journal of Artificial Intelligence Research*, 17, pp. 451–499.

Burnham, K. P. and Anderson, D. R. (2004) *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach*. 2nd edn. Springer Science & Business Media.

Di Masi, G. B. and Finesso, L. (1996) 'Bayesian estimators for finite hmm's.' *Presented at: The 12th International Symposium on the Mathematical Theory of Networks and Systems (MTNS1996)*.

Durbin, R. *et al.* (1998) *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press.

Eddy, S. R. (1998) 'Profile hidden markov models.' *Bioinformatics*, 14(9), pp. 755–763.

Elliott, R. J., Aggoun, L. and Moore, J. B. (2008) *Hidden Markov models: estimation and control*. Springer Science & Business Media.

Fine, S., Singer, Y. and Tishby, N. (1998) 'Hierarchical hidden markov models: Representation, estimation, and inference.' In *Machine Learning*, vol. 32, pp. 41–62. Springer.

Gales, M. J. F. and Young, S. J. (2007) 'Application of hidden markov models in speech recognition.' *Foundations and Trends in Signal Processing*, 1(3), pp. 195–304.

Ghahramani, Z. and Jordan, M. I. (1997) 'Factorial hidden markov models.' *Machine Learning*, 29(2-3), pp. 245–273.

Graves, A., Mohamed, A.-r. and Hinton, G. (2013) 'Speech recognition with deep recurrent neural networks.' *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 6645–6649.

Great Learning Team (2022) 'Part of speech (pos) tagging with hidden markov model.' Accessed: August 17, 2024. Available at: https://www.mygreatlearning.com/blog/pos-tagging/

Hassan, M. R. and Nath, B. (2009) 'A combination of hidden markov model and artificial neural networks for stock market forecasting.' *Neurocomputing*, 72(16–18), pp. 3439–3446.

Hastie, T., Tibshirani, R. and Friedman, J. (2009) 'The elements of statistical learning: Data mining, inference, and prediction.' pp. 222–230.

Jurafsky, D. and Martin, J. H. (2009) *Speech and Language Processing*. Pearson Prentice Hall.

Kumar, A. (2023) 'Hidden markov models: Concepts, examples.' Accessed: August 15, 2024. Available at: https://vitalflux.com/hidden-markov-models-concepts-explained-with-examples/

Li, X. (2024) 'Figure 1.' Uploaded by Xiaolin Li, Content may be subject to copyright. Available at: https://www.researchgate.net (Accessed: 16th August 2024)

Lipton, Z. C. (2018) 'The mythos of model interpretability.' *Communications of the ACM*, 61(10), pp. 36–43.

Manning, C. D. and Schütze, H. (1999) *Foundations of Statistical Natural Language Processing*. MIT Press.

Mariano, L. J. *et al.* (2015) 'Modeling strategic use of human computer interfaces with novel hidden markov models.' *Frontiers in Psychology*, 6, p. 919. doi:10.3389/fpsyg.2015.00919.

Mohamed, A.-r., Dahl, G. and Hinton, G. E. (2011) 'Deep belief networks for phone recognition.' *Neural Information Processing Systems*, 23, pp. 2199–2207.

Murphy, K. P. and Paskin, M. A. (2001) 'Linear time inference in hierarchical hmms.' In *Advances in neural information processing systems*, pp. 833–840.

Oliver, N., Pentland, A. and Tian, F. (2000) 'Graphical models for recognizing human interactions: a comparative study.' In *Proceedings 2000 International Conference on Image Processing (Cat. No. 00CH37101)*, vol. 2, pp. 1001–1004. IEEE.

Rabiner, L. R. (1989) 'A tutorial on hidden markov models and selected applications in speech recognition.' *Proceedings of the IEEE*, 77(2), pp. 257–286. doi:10.1109/5.18626.

Thrun, S., Burgard, W. and Fox, D. (2005) *Probabilistic robotics*. MIT Press.

Vidhya, A. (????) 'Hidden markov models: Concepts, examples.' Accessed: 2024-08-15. Available at: https://vitalflux.com/hidden-markov-models-concepts-explained-with-examples/

Zen, H., Tokuda, K. and Kitamura, T. (2004) 'Experimental results show that the use of hsmm training improves the naturalness of the synthesized speech.' *Interspeech 2004*. Accessed: August 16, 2024. Available at: https://semanticscholar.org/paper/Experimental-results-show-that-the-use-of-HSMM-Zen-Tokuda/220313656

Zhang, Z. and Lessmann, V. H. (2008) 'A coupled hidden markov model for multi-sensor fusion.' In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1489–1494. IEEE.

Zou, Y. *et al.* (2022) 'Multivariate analysis of car-following behavior data using a coupled hidden markov model.' *Transportation Research Part C: Emerging Technologies*, 144, p. 103914. doi:10.1016/j.trc.2022.103914. Accessed: August 17, 2024.

*