1

Introduction

Probabilistic graphical modeling is a branch of machine learning that uses probability distributions to describe the world and to make useful predictions about it. Underlying this modeling framework is an elegant body of theory that bridges two mathematical traditions: *probability* and *graph theory* (Figure 1.1). Probabilistic graphical modeling also intersects with philosophy in intriguing ways, especially through questions of *causality*. This tutorial aims to provide a concise introduction to the formalisms, methods, and applications of this powerful framework.

Probabilistic graphical modeling has had far-reaching impacts on theoretical and applied research in computing and beyond. It has been used to solve problems across diverse domains, including medicine, biology, chemistry, physics, electrical engineering, natural language processing, and computer vision. This combination of elegant theory and practical applications has made probabilistic graphical modeling one of the most fascinating topics in modern computer science and artificial intelligence.

Testament to the impact of this research area, the Association for Computing Machinery awarded the 2011 Turing Award[1] to Judea Pearl

[1]The Turing Award is widely regarded as the "Nobel Prize of Computing."

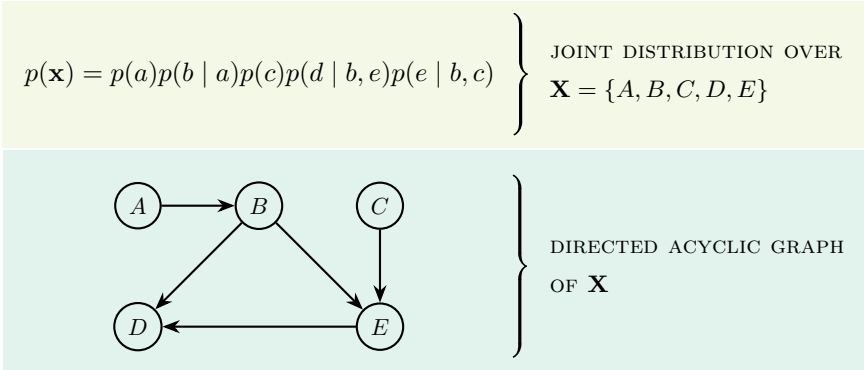

Figure 1.1: This tutorial demonstrates how probability and graph theory provide complementary languages for representing, learning, and querying models of real-world phenomena. The reader will gain intuition for reasoning over complex domains, both in terms of probability distributions and in terms of their graphical representations. For example, this tutorial will explain when and why the factorization of joint distribution $p(\mathbf{x})$ implies the directed graph of $\mathbf{X}$ depicted above.

for establishing the foundations of probabilistic graphical modeling and introducing a calculus for causal reasoning (Gotlieb, 2022). These contributions "revolutionized the field of artificial intelligence" and precipitated vast advancements across engineering, the natural sciences, and the social sciences (ACM, 2011).

In the remainder of this brief introduction, we provide a high-level overview of what to expect from this tutorial.

1.1 A Framework for Reasoning Under Uncertainty

When solving a real-world problem using mathematics, it is very common to define a mathematical model of the world in the form of an equation. Perhaps the simplest model would be a linear equation of the form

$$y = \boldsymbol{\beta}^T \mathbf{x},$$

where y is an outcome variable that we want to predict and $\mathbf{x}$ are observed variables that affect this outcome. For example, y may be the price of a house, while $\mathbf{x}$ denotes a series of features that affect this price (e.g., location, number of bedrooms, age of the house, etc.). We assume that y is a linear function of this input, parameterized by $\boldsymbol{\beta}$.

Often, the real world that we are trying to model is very complicated. In particular, it often involves a significant amount of *uncertainty* (e.g., a house's price y can fluctuate based on the inherent variability in buyers' subjective preferences about its features $\mathbf{x}$). It is therefore very natural to deal with this uncertainty by modeling the world in the form of a probability distribution,

$$p(\mathbf{x}, y).$$

Given such a model, we could ask various questions. For example: *Given that the house costs $100,000, what is the probability that it has three bedrooms?*

The probabilistic aspect of modeling is very important. Typically, we cannot perfectly predict the future. We rarely have sufficient knowledge about the world, and many real-world processes are stochastic in nature. Additionally, we need to assess the confidence of our predictions. Often, predicting a single value is not enough: we need the system to output its *beliefs* about the world, and to what extent these are uncertain.

In this tutorial, we present principled ways of *reasoning about uncertainty*. We draw from both probability and graph theory to derive efficient machine learning algorithms for this task, answering such questions as the following.

- *What are the tradeoffs between computational complexity and the richness of a probabilistic model?*
- *What is the best model for inferring facts about the future, given a fixed dataset and computational budget?*
- *How does one combine prior knowledge with observed evidence in a principled way to make predictions?*

We will contextualize our discussions with applications to real-world problems, such as image and language analysis.

Challenges in Probabilistic Modeling

To get a first taste of the challenges that lie ahead, consider a simple application of probabilistic modeling: spam classification.

Suppose we have a model $p(y, x_1, \ldots, x_n)$ of word occurrences in spam and non-spam mail. The binary random variables $\{X_i\}_{i=1}^n$ encode whether the i^{th} English word is present in the email. The binary random variable Y indicates whether the email is spam. In order to classify a new email, we can look at the probability $p(y = 1 \mid x_1, \ldots, x_n)$.

What is the "size" of the function p that we just defined? Our model defines a probability in $[0, 1]$ for each combination of inputs $\{y, x_1, \ldots, x_n\}$. Specifying each of these probabilities requires the calculation of a staggering 2^{n+1} different values, one for each assignment to our $n + 1$ binary variables. Since n is the size of the English vocabulary, this is clearly impractical from both a computational standpoint (i.e., *how do we store a data object of this size?*) and from a statistical perspective (i.e., *how do we efficiently estimate the parameters from limited data?*). More generally, this example illustrates one of the main challenges that this tutorial will deal with: probability distributions are inherently exponentially-sized objects, and so to manipulate them we must make simplifying assumptions about their structure.

The main simplifying assumption that we will make in this tutorial is that of *conditional independence* among variables. For example, suppose that the English words in our spam example are all conditionally independent given y. That is, the event that a given word occurs in the email is independent of whether a different word occurs, given that the email is spam. This is clearly an oversimplification, as some words are likely to co-occur in English text regardless of y (e.g., *ibuprofen* and *pain*). However, these events will indeed be independent for most words (e.g., *penguin* and *muffin* do not have an obvious relationship to each other in the English language). We can therefore justify that this assumption will not significantly degrade the accuracy of the model.

We refer to this particular choice of independencies as the *Naive Bayes assumption*. Given this assumption, we can write the model probability as a product of *factors*,

$$p(y, x_1, \ldots, x_n) = p(y) \prod_{i=1}^{n} p(x_i \mid y).$$

Each factor $p(x_i \mid y)$ can be completely described by a small number of parameters (four parameters with two degrees of freedom to be exact).

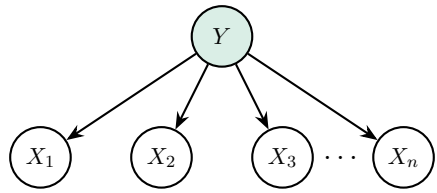

Figure 1.2: Graphical representation of the Naive Bayes spam classification model. We can interpret the directed graph as conveying a story about how the data was generated. First, a spam/non-spam label was chosen at random ($Y = y$). Then, a subset of n possible English words were sampled independently and at random ($\{X_i\}_{i=1}^n = \{x_i\}_{i=1}^n$). Thus, the graphical representation of this data generating process displays directed edges from Y to each X_i (denoting dependence), but no edges among the individual X_i (denoting independence).

The entire distribution is parameterized by $O(n)$ parameters, which we can tractably estimate from data and make predictions over.

Describing Probabilities with Graphs

The previous independence assumption can be conveniently represented in the form of a *directed graph*, where directed edges $Y \to X_i$ visually represent the relationships that are also captured in the joint factorization $p(y) \prod_{i=1}^n p(x_i \mid y)$ (Figure 1.2). This graphical representation has the immediate advantage of being easy to understand. It can be interpreted as telling us a story: an email was generated by first choosing at random whether the email is spam or not (indicated by $Y = y$), and then by sampling words one at a time. Conversely, if we have a story for how our dataset was generated, we can naturally express it as a graph with an associated probability distribution.

More importantly, we want to submit various queries to our model (e.g., what is the probability of spam given that I see the word *congratulations*?). Answering these questions will require specialized inference algorithms that are most naturally defined using graph-theoretic concepts. We will also use graph theory to analyze the speed of learning algorithms and to quantify the computational complexity (e.g., NP-hardness) of different learning tasks. In essence, there is an intimate connection between probability distributions and graphs that will be exploited throughout this tutorial for the purposes of defining, learning,

and querying probabilistic models.

1.2 A Bird's Eye View of This Tutorial

Tutorial Structure

After briefly reviewing the fundamentals of probability and graph theory (Chapter 2), our discussion will be divided into three major themes: representation (Chapter 3), inference (Chapters 4 and 5), and learning (Chapter 6).

1. **Representation** (i.e., *how to specify a model*). How do we express a probability distribution that models some real-world phenomenon? This is not a trivial problem: we have seen that a naive model for classifying spam messages with n possible words generally requires the specification of $O(2^n)$ parameters. We will address this difficulty via general techniques for constructing tractable models. These recipes will make heavy use of graph theory.

2. **Inference** (i.e., *how to ask the model questions*). Given a probabilistic model, how do we obtain answers to relevant questions about the world? Such questions often reduce to querying the marginal or conditional probabilities of certain events of interest. More concretely, we will typically be interested in asking two types of questions:

   (a) *Marginal inference.* What is the probability of a given variable in our model after we sum everything else out? Such queries generally take the form

   $$p(x_1) = \sum_{x_2} \sum_{x_3} \cdots \sum_{x_n} p(x_1, x_2, \ldots, x_n).$$

   (b) *Maximum a posteriori (MAP) inference.* Here, we ask for the most likely assignment of variables. For example, we can determine the most likely spam message under our model. Such queries generally take the form

   $$\operatorname*{argmax}_{x_1,\ldots,x_n} p(x_1, \ldots, x_n, y = 1).$$

Queries often involve *evidence*, in which case we fix the assignment of a subset of the variables (e.g., $y = 1$ above).

It turns out that inference is a very challenging task. For many probabilistic models of interest, it will be NP-hard to answer the kinds of questions that we have described. Crucially, whether inference is tractable will depend on the structure of the graph that describes the underlying probability distribution. We will show that even when *exact inference* is intractable, we can still obtain useful answers via *approximate inference* methods.

3. **Learning** (i.e., *how to fit a model to real-world data*). Our last key task refers to fitting a model to a dataset (e.g., a large number of labeled examples of spam). By inspecting the data, we can infer useful patterns (e.g., which words are found more frequently in spam emails). We can then leverage our knowledge of these patterns to make predictions about the future. However, we will see that learning and inference are also inherently linked in a more subtle way: inference is a key subroutine that is called repeatedly within learning algorithms. The topic of learning also shares important connections with two prominent fields:

    - *Computational learning theory*, which deals with questions such as generalization from limited data and overfitting; and
    - *Bayesian statistics*, which offers an elegant mathematical language for combining prior knowledge and observed evidence in a principled way.

The three themes of representation, inference, and learning are closely linked. As we have alluded, the derivation of efficient inference and learning algorithms requires an adequately represented model. Furthermore, learning will require inference as a subroutine. Thus, it is best to always keep these three tasks in mind rather than focusing on them in isolation. In Chapter 7, we highlight these interconnections with a concluding discussion on a powerful deep probabilistic model: the *variational autoencoder* (Kingma and Welling, 2019), whose widespread influence was recognized with the Test of Time Award at the 2024

International Conference on Learning Representations (ICLR, 2024; Kingma and Welling, 2024).

Relation to Prior Works

This tutorial is intended to be a concise and accessible overview of the state of the field. Throughout, we offer recommendations for exceptional resources that provide additional depth. We encourage the reader to explore the works of Judea Pearl for many authoritative resources on probabilistic and causal inference through a graphical lens (Pearl, 1988; Geffner *et al.*, 2022). Koller and Friedman (2009) provide an extensive textbook that makes a strong accompaniment to this tutorial. Many seminal machine learning texts dedicate chapters to related topics, including Bishop (2006), Murphy (2012), and Murphy (2023). This tutorial discusses several topics that have been previously described in *Foundations and Trends*, including graphical models and variational inference (Wainwright and Jordan, 2008), conditional random fields (Sutton, 2012), Monte Carlo methods (Lindsten, 2013; Naesseth *et al.*, 2019), Bayesian inference (Angelino *et al.*, 2016), and variational autoencoders (Kingma and Welling, 2019). This tutorial provides a compact survey of the core theory and methods in this large literature, highlighting their interrelationships and drawing connections to modern deep generative modeling.