Representation

CS236, Stanford University

Lecture 2
Overview

- What is a generative model
- Representing probability distributions
  - Curse of dimensionality
  - Crash course on graphical models (Bayesian networks)
  - Generative vs discriminative models
  - Neural models
Learning a generative model

- We are given a training set of examples, e.g., images of dogs

![Image of dogs]

- We want to learn a probability distribution $p(x)$ over images $x$ such that
  - **Generation**: If we sample $x_{new} \sim p(x)$, $x_{new}$ should look like a dog (sampling)
  - **Density estimation**: $p(x)$ should be high if $x$ looks like a dog, and low otherwise (anomaly detection)
  - **Unsupervised representation learning**: We should be able to learn what these images have in common, e.g., ears, tail, etc. (features)

- First question: how to represent $p(x)$
Basic discrete distributions

- Bernoulli distribution: (biased) coin flip
  - $D = \{Heads, Tails\}$
  - Specify $P(X = Heads) = p$. Then $P(X = Tails) = 1 - p$.
  - Write: $X \sim Ber(p)$
  - Sampling: flip a (biased) coin

- Categorical distribution: (biased) $m$-sided dice
  - $D = \{1, \cdots, m\}$
  - Specify $P(Y = i) = p_i$, such that $\sum p_i = 1$
  - Write: $Y \sim Cat(p_1, \cdots, p_m)$
  - Sampling: roll a (biased) die
Example of joint distribution

Modeling a single pixel’s color. Three discrete random variables:

- Red Channel $R$. $\text{Val}(R) = \{0, \cdots, 255\}$
- Green Channel $G$. $\text{Val}(G) = \{0, \cdots, 255\}$
- Blue Channel $B$. $\text{Val}(B) = \{0, \cdots, 255\}$

Sampling from the joint distribution $(r, g, b) \sim p(R, G, B)$ randomly generates a color for the pixel. How many parameters do we need to specify the joint distribution $p(R = r, G = g, B = b)$?

$$256 \times 256 \times 256 - 1$$
Example of joint distribution

Suppose $X_1, \ldots, X_n$ are binary (Bernoulli) random variables, i.e., $\text{Val}(X_i) = \{0, 1\} = \{\text{Black}, \text{White}\}$.

How many possible images (states)?

$$2 \times 2 \times \cdots \times 2 = 2^n$$

Sampling from $p(x_1, \ldots, x_n)$ generates an image.

How many parameters to specify the joint distribution $p(x_1, \ldots, x_n)$ over $n$ binary pixels?

$$2^n - 1$$
Structure through independence

- If $X_1, \ldots, X_n$ are independent, then

$$p(x_1, \ldots, x_n) = p(x_1)p(x_2)\cdots p(x_n)$$

- How many possible states? $2^n$
- How many parameters to specify the joint distribution $p(x_1, \ldots, x_n)$?
  - How many to specify the marginal distribution $p(x_1)$? 1
- $2^n$ entries can be described by just $n$ numbers (if $|\text{Val}(X_i)| = 2$)!
- Independence assumption is too strong. Model not likely to be useful
  - For example, each pixel chosen independently when we sample from it.
Two important rules

1. **Chain rule** Let $S_1, \ldots, S_n$ be events, $p(S_i) > 0$.

   \[ p(S_1 \cap S_2 \cap \cdots \cap S_n) = p(S_1)p(S_2 \mid S_1) \cdots p(S_n \mid S_1 \cap \cdots \cap S_{n-1}) \]

2. **Bayes’ rule** Let $S_1, S_2$ be events, $p(S_1) > 0$ and $p(S_2) > 0$.

   \[ p(S_1 \mid S_2) = \frac{p(S_1 \cap S_2)}{p(S_2)} = \frac{p(S_2 \mid S_1)p(S_1)}{p(S_2)} \]
Structure through conditional independence

- Using Chain Rule

\[ p(x_1, \ldots, x_n) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2) \cdots p(x_n \mid x_1, \cdots, x_{n-1}) \]

- How many parameters? \( 1 + 2 + \cdots + 2^{n-1} = 2^n - 1 \)
  - \( p(x_1) \) requires 1 parameter
  - \( p(x_2 \mid x_1 = 0) \) requires 1 parameter, \( p(x_2 \mid x_1 = 1) \) requires 1 parameter
    Total 2 parameters.
  - \( \ldots \)

- \( 2^n - 1 \) is still exponential, chain rule does not buy us anything.
- Now suppose \( X_{i+1} \perp X_1, \ldots, X_{i-1} \mid X_i \), then

\[
\begin{align*}
p(x_1, \ldots, x_n) &= p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2) \cdots p(x_n \mid x_1, \cdots, x_{n-1}) \\
&= p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_2) \cdots p(x_n \mid x_{n-1})
\end{align*}
\]

- How many parameters? \( 2n - 1 \). Exponential reduction!
Bayes Network: General Idea

- Use conditional parameterization (instead of joint parameterization)
- For each random variable $X_i$, specify $p(x_i|x_{A_i})$ for set $X_{A_i}$ of random variables
- Then get joint parametrization as

$$p(x_1, \ldots, x_n) = \prod_i p(x_i|x_{A_i})$$

- Need to guarantee it is a *legal* probability distribution. It has to correspond to a chain rule factorization, with factors simplified due to assumed conditional independencies
Bayesian networks

- A **Bayesian network** is specified by a *directed acyclic* graph (DAG) $G = (V, E)$ with:
  1. One node $i \in V$ for each random variable $X_i$
  2. One conditional probability distribution (CPD) per node, $p(x_i \mid x_{Pa(i)})$, specifying the variable’s probability conditioned on its parents’ values

- Graph $G = (V, E)$ is called the structure of the Bayesian Network

- Defines a joint distribution:

$$p(x_1, \ldots, x_n) = \prod_{i \in V} p(x_i \mid x_{Pa(i)})$$

- **Claim:** $p(x_1, \ldots, x_n)$ is a valid probability distribution because of ordering implied by DAG

- **Economical representation:** exponential in $|Pa(i)|$, not $|V|$
Example

Directed cycle

DAG stands for Directed Acyclic Graph
Consider the following Bayesian network:

![Bayesian Network Diagram]

What is its joint distribution?

\[ p(x_1, \ldots, x_n) = \prod_{i \in V} p(x_i \mid x_{Pa(i)}) \]

\[ p(d, i, g, s, l) = p(d)p(i)p(g \mid i, d)p(s \mid i)p(l \mid g) \]
Bayesian network structure implies conditional independencies!

- The joint distribution corresponding to the above BN factors as
  \[ p(d, i, g, s, l) = p(d)p(i)p(g \mid i, d)p(s \mid i)p(l \mid g) \]

- However, by the chain rule, any distribution can be written as
  \[ p(d, i, g, s, l) = p(d)p(i \mid d)p(g \mid i, d)p(s \mid i, d, g)p(l \mid g, d, i, s) \]

- Thus, we are assuming the following additional independencies:
  \[ D \perp I, \quad S \perp \{D, G\} \mid I, \quad L \perp \{I, D, S\} \mid G. \]
Bayesian networks given by \((G, P)\) where \(P\) is specified as a set of local conditional probability distributions associated with \(G\)'s nodes.

- Efficient representation using a graph-based data structure.
- Computing the probability of any assignment is obtained by multiplying CPDs.
- Can sample from the joint by sampling from the CPDs according to the DAG ordering.
- Can identify some conditional independence properties by looking at graph properties.
- In this class, graphical models will be simple (e.g., only 2 or 3 random vectors).
- Next: generative vs. discriminative; functional parameterizations.
Naive Bayes for single label prediction

- Classify e-mails as spam ($Y = 1$) or not spam ($Y = 0$)
  - Let $1 : n$ index the words in our vocabulary (e.g., English)
  - $X_i = 1$ if word $i$ appears in an e-mail, and 0 otherwise
  - E-mails are drawn according to some distribution $p(Y, X_1, \ldots, X_n)$
- Words are conditionally independent given $Y$:

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

Then
Example: naive Bayes for classification

- Classify e-mails as spam ($Y = 1$) or not spam ($Y = 0$)
  - Let $1:n$ index the words in our vocabulary (e.g., English)
  - $X_i = 1$ if word $i$ appears in an e-mail, and 0 otherwise
  - E-mails are drawn according to some distribution $p(Y, X_1, \ldots, X_n)$
- Suppose that the words are conditionally independent given $Y$. Then,

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

**Estimate** parameters from training data. **Predict** with Bayes rule:

\[
p(Y = 1 \mid x_1, \ldots, x_n) = \frac{p(Y = 1) \prod_{i=1}^{n} p(x_i \mid Y = 1)}{\sum_{y=\{0,1\}} p(Y = y) \prod_{i=1}^{n} p(x_i \mid Y = y)}
\]

- Are the independence assumptions made here reasonable?
- Philosophy: Nearly all probabilistic models are “wrong”, but many are nonetheless useful
Discriminative versus generative models

- Using chain rule $p(Y, X) = p(X \mid Y)p(Y) = p(Y \mid X)p(X)$. Corresponding Bayesian networks:

  ![Bayesian networks diagram]

- However, suppose all we need for prediction is $p(Y \mid X)$

- In the left model, we need to specify/learn both $p(Y)$ and $p(X \mid Y)$, then compute $p(Y \mid X)$ via Bayes rule

- In the right model, it suffices to estimate just the **conditional distribution** $p(Y \mid X)$
  - We never need to model/learn/use $p(X)$!
  - Called a **discriminative** model because it is only useful for discriminating $Y$’s label when given $X$
Discriminative versus generative models

Since $\mathbf{X}$ is a random vector, chain rules will give

- $p(Y, \mathbf{X}) = p(Y)p(X_1 | Y)p(X_2 | Y, X_1) \cdots p(X_n | Y, X_1, \cdots, X_{n-1})$
- $p(Y, \mathbf{X}) = p(X_1)p(X_2 | X_1)p(X_3 | X_1, X_2) \cdots p(Y | X_1, \cdots, X_{n-1}, X_n)$

We must make the following choices:

1. In the generative model, $p(Y)$ is simple, but how do we parameterize $p(X_i | \mathbf{X}_{pa(i)}, Y)$?
2. In the discriminative model, how do we parameterize $p(Y | \mathbf{X})$? Here we assume we don’t care about modeling $p(\mathbf{X})$ because $\mathbf{X}$ is always given to us in a classification problem.
For the generative model, assume that $X_i \perp X_{-i} \mid Y$ (naive Bayes)
For the discriminative model, assume that

\[ p(Y = 1 \mid x; \alpha) = f(x, \alpha) \]

Not represented as a table anymore. It is a parameterized function of \( x \) (regression)
- Has to be between 0 and 1
- Depend in some simple but reasonable way on \( x_1, \ldots, x_n \)
- Completely specified by a vector \( \alpha \) of \( n + 1 \) parameters (compact representation)

Linear dependence: let \( z(\alpha, x) = \alpha_0 + \sum_{i=1}^{n} \alpha_i x_i \). Then,

\[ p(Y = 1 \mid x; \alpha) = \sigma(z(\alpha, x)), \text{ where } \sigma(z) = 1/(1 + e^{-z}) \] is called the logistic function:
Logistic regression

Linear dependence: let $z(\alpha, x) = \alpha_0 + \sum_{i=1}^{n} \alpha_i x_i$. Then, $p(Y = 1 \mid x; \alpha) = \sigma(z(\alpha, x))$, where $\sigma(z) = 1/(1 + e^{-z})$ is called the logistic function.

1. Decision boundary $p(Y = 1 \mid x; \alpha) > 0.5$ is linear in $x$
2. Equal probability contours are straight lines
3. Probability rate of change has very specific form (third plot)
Discriminative models are powerful

- Logistic model does not assume $X_i \perp X_{-i} \mid Y$, unlike naive Bayes.
- This can make a big difference in many applications.
- For example, in spam classification, let $X_1 = 1[“bank” \text{ in e-mail}]$ and $X_2 = 1[“account” \text{ in e-mail}]$.
- Regardless of whether spam, these always appear together, i.e. $X_1 = X_2$.
- Learning in naive Bayes results in $p(X_1 \mid Y) = p(X_2 \mid Y)$. Thus, naive Bayes double counts the evidence.
- Learning with logistic regression sets $\alpha_1 = 0$ or $\alpha_2 = 0$, in effect ignoring it.
Generative models are still very useful

Using chain rule \( p(Y, X) = p(X | Y)p(Y) = p(Y | X)p(X) \). Corresponding Bayesian networks:

1. Using a conditional model is only possible when \( X \) is always observed
   - When some \( X_i \) variables are unobserved, the generative model allows us to compute \( p(Y | X_{\text{evidence}}) \) by marginalizing over the unseen variables
In discriminative models, we assume that

\[ p(Y = 1 \mid x; \alpha) = f(x, \alpha) \]

1. **Linear** dependence:
   - Let \( z(\alpha, x) = \alpha_0 + \sum_{i=1}^{n} \alpha_i x_i \).
   - \( p(Y = 1 \mid x; \alpha) = \sigma(z(\alpha, x)) \), where \( \sigma(z) = 1/(1 + e^{-z}) \) is the **logistic function**
   - Dependence might be too simple

2. **Non-linear** dependence: let \( h(A, b, x) = f(Ax + b) \) be a non-linear transformation of the inputs (features).
   - \( p_{\text{Neural}}(Y = 1 \mid x; \alpha, A, b) = \sigma(\alpha_0 + \sum_{i=1}^{h} \alpha_i h_i) \)
   - More flexible
   - More parameters: \( A, b, \alpha \)
Neural Models

1. In discriminative models, we assume that
   \[ p(Y = 1 \mid x; \alpha) = f(x, \alpha) \]

2. **Linear** dependence: let \( z(\alpha, x) = \alpha_0 + \sum_{i=1}^{n} \alpha_i x_i \).
   \[ p(Y = 1 \mid x; \alpha) = f(z(\alpha, x)), \text{ where } f(z) = \frac{1}{1 + e^{-z}} \] is the logistic function
   - Dependence might be too simple

3. **Non-linear** dependence: let \( h(A, b, x) = f(Ax + b) \) be a non-linear transformation of the inputs (features).
   \[ p_{\text{Neural}}(Y = 1 \mid x; \alpha, A, b) = f(\alpha_0 + \sum_{i=1}^{h} \alpha_i h_i) \]
   - More flexible
   - More parameters: \( A, b, \alpha \)
   - Can repeat multiple times to get a neural network
Bayesian networks vs neural models

- Using Chain Rule

\[ p(x_1, x_2, x_3, x_4) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2)p(x_4 \mid x_1, x_2, x_3) \]

Fully General

- Bayes Net

\[ p(x_1, x_2, x_3, x_4) \approx p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2)p(x_4 \mid x_1, x_2, x_3) \]

Assumes conditional independencies

- Neural Models

\[ p(x_1, x_2, x_3, x_4) \approx p(x_1)p(x_2 \mid x_1)p_{\text{Neural}}(x_3 \mid x_1, x_2)p_{\text{Neural}}(x_4 \mid x_1, x_2, x_3) \]

Assume specific functional form for the conditionals. A sufficiently deep neural net can approximate any function.
Continuous variables

- If $X$ is a continuous random variable, we can usually represent it using its **probability density function** $p_X : \mathbb{R} \rightarrow \mathbb{R}^+$. However, we cannot represent this function as a table anymore. Typically consider parameterized densities:
  - **Gaussian:** $X \sim \mathcal{N}(\mu, \sigma)$ if $p_X(x) = \frac{1}{\sigma \sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}$
  - **Uniform:** $X \sim \mathcal{U}(a, b)$ if $p_X(x) = \frac{1}{b-a} 1[a \leq x \leq b]
  - Etc.

- If $X$ is a continuous random vector, we can usually represent it using its **joint probability density function**:
  - **Gaussian:** if $p_X(x) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp \left(-\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right)$

- Chain rule, Bayes rule, etc all still apply. For example,

$$p_{X,Y,Z}(x, y, z) = p_X(x)p_{Y|X}(y | x)p_{Z|\{X,Y\}}(z | x, y)$$
Continuous variables

- This means we can still use Bayesian networks with continuous (and discrete) variables. Examples:

**Mixture of 2 Gaussians**: Bayes net $Z \to X$ with factorization $p_{Z,X}(z,x) = p_Z(z)p_{X|Z}(x \mid z)$ and
  - $Z \sim \text{Bernoulli}(p)$
  - $X \mid (Z = 0) \sim \mathcal{N}(\mu_0, \sigma_0)$, $X \mid (Z = 1) \sim \mathcal{N}(\mu_1, \sigma_1)$
  - The parameters are $p, \mu_0, \sigma_0, \mu_1, \sigma_1$

- Bayes net $Z \to X$ with factorization $p_{Z,X}(z,x) = p_Z(z)p_{X|Z}(x \mid z)$
  - $Z \sim \mathcal{U}(a,b)$
  - $X \mid (Z = z) \sim \mathcal{N}(z, \sigma)$
  - The parameters are $a, b, \sigma$

**Variational autoencoder**: Bayes net $Z \to X$ with factorization $p_{Z,X}(z,x) = p_Z(z)p_{X|Z}(x \mid z)$ and
  - $Z \sim \mathcal{N}(0, 1)$
  - $X \mid (Z = z) \sim \mathcal{N}(\mu_\theta(z), e^{\sigma_\phi(z)})$ where $\mu_\theta : \mathbb{R} \to \mathbb{R}$ and $\sigma_\phi$ are neural networks with parameters (weights) $\theta, \phi$ respectively
  - **Note**: Even if $\mu_\theta, \sigma_\phi$ are very deep (flexible), functional form is still Gaussian