Lecture 4

Genetic models $P(I|w), P(w)$.

How to learn $P(w)$?

For simplicity, we will discuss learning a distribution $P(w)$. Replace $w$ by $x$ in this lecture.

**Ideal Method:**

Assume a parameterized model for the distribution of form $P(x|\theta) \sim \text{a model with parameters } \theta$.

**6. Gaussian distribution:**

$$P(x|\mu, \sigma) = \frac{1}{\sqrt{2\pi \sigma}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad \theta = (\mu, \sigma).$$

Assume that data is independent identically distributed (i.i.d.).

$$P(x_1, \ldots, x_n | \lambda) \sim \prod_{i=1}^{n} P(x_i | \lambda) \quad \text{(product for independence)}$$

**Choose:**

$$\hat{\lambda} = \arg\max_{\lambda} P(x_1, \ldots, x_n | \lambda) = \arg\max_{\lambda} \log P(x_1, \ldots, x_n | \lambda)$$

Hence $P(x_1, \ldots, x_n (\hat{\lambda}) \geq P(x_1, \ldots, x_n | \lambda), \text{ for all } \lambda$.
Example: Gaussian

\[
\log P(x_1, \ldots, x_n | \mu, \sigma) = \frac{N}{2} \log \sigma - \frac{1}{2} \sum_{i=1}^{N} \frac{(x_i - \mu)^2}{\sigma^2}
\]

Differentiate w.r.t. \( \mu \) gives

\[
\frac{\partial}{\partial \mu} \log P(x_1, \ldots, x_N | \mu, \sigma) = \frac{1}{\sigma^2} \sum_{i=1}^{N} (x_i - \mu)
\]

Differentiate w.r.t. \( \sigma \) gives

\[
\frac{\partial}{\partial \sigma} \log P(x_1, \ldots, x_N | \mu, \sigma) = -\frac{N}{2} \log \sigma - \frac{1}{2} \sum_{i=1}^{N} \frac{(x_i - \mu)^2}{\sigma^2}
\]

Maxima occurs at

\[
\hat{\mu} = \frac{1}{N} \sum_{i=1}^{N} x_i
\]

\[
\hat{\sigma}^2 = \frac{1}{N} \sum_{i=1}^{N} (x_i - \hat{\mu})^2
\]

Easy to check these are maximum by computing

\[
\frac{\partial^2}{\partial \mu^2} \frac{\partial^2}{\partial \sigma^2} \frac{1}{\sigma^2}
\]

Note: Similar results hold for Gaussian distributions in many variables.

Note: The Gaussian is a special case. It is often impossible to solve \( \frac{\partial}{\partial \sigma} \log P(x_1, \ldots, x_N | \sigma) \) analytically. An algorithm is 

(3) \textbf{Exponential Distributions}

\[ P(x | \lambda) = \frac{1}{Z[\lambda]} e^{\lambda \cdot \phi(x)} \]

- Parameter: \( \lambda \)
- \( \phi(x) \): statistics

\( Z[\lambda] \): normalization factor.

Almost every named distribution can be expressed as an exponential distribution.

For Gaussian in 1-dimensions,

write \( \phi(x) = (x^2) \) \( Z = \gamma_1, \gamma_2 \)

\[ P(x | \lambda) = \frac{1}{Z[\lambda]} e^{\lambda_1 x + \lambda_2 x^2} \]

\[ \frac{1}{Z[\lambda]} \text{ compare to } \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x - \mu)^2}{2\sigma^2}}. \]

\textbf{Translation}

\[ \begin{cases} \lambda_2 = -\frac{1}{2\sigma^2} \\ \lambda_1 = \frac{\mu}{\sigma^2} \\ Z[\lambda] = \sqrt{2\pi\sigma^2} e^{\frac{\mu^2}{2\sigma^2}} \end{cases} \]

Similar translation into exponential distribution can be made for Poisson, Beta, Dirichlet, and most (all) distributions you have been taught.
Learning an Exponential Distribution

You can learn them by Maximum Likelihood, which again can be interpreted in terms of minimizing the KL divergence between the empirical distribution of the data, and the model distribution.

Example: \( (x_1, x_2, \ldots, x_n) \)

\[
P(\langle x_1, x_2, \ldots, x_n \rangle | \lambda) = \prod_{i=1}^{n} \frac{\lambda \cdot \phi(x_i)}{Z[\lambda]}
\]

Maximize w.r.t. \( \lambda \)

This has a very nice form, which occurs because the exponential distribution depends on the data \( x \) only in terms of the function \( \phi(x) - \) the sufficient statistics.

Note:

\[
Z[\lambda] = \sum_{x} \lambda \cdot \phi(x) e^{-\lambda \cdot \phi(x)}
\]

\[
\frac{\partial \log Z[\lambda]}{\partial \lambda} = \sum_{x} \phi(x) e^{-\lambda \cdot \phi(x)}
\]

\[
\frac{\partial^2 \log Z[\lambda]}{\partial \lambda^2} = \sum_{x} \phi(x) P(x | \lambda)
\]
(5) \[ \frac{\text{ML}}{} \text{ maximizes:} \]
\[ \sum_{n=1}^{N} \frac{1}{2} \phi(\mathbf{x}_n) - N \log \mathbb{E}[Z] \]
\[ \frac{\partial \theta}{\partial \theta} \rightarrow \sum_{n=1}^{N} \phi(\mathbf{x}_n) - N \sum_{k=1}^{K} \phi(\mathbf{x}) \mathbb{P}(\mathbf{x}|\theta). \]

Pick the parameters \( \theta \) so that the average of the statistics \( \phi(\mathbf{x}) \) over the distribution \( \mathbb{P}(\mathbf{x}|\theta) \) is equal to the average of the statistics of the sampler.

\[ \text{Solve:} \quad \frac{\partial}{\partial \theta} \sum_{n=1}^{N} \phi(\mathbf{x}_n) \mathbb{P}(\mathbf{x}_n|\theta) = \frac{N}{N} \sum_{n=1}^{N} \phi(\mathbf{x}_n), \]

This is equivalent to minimizing \( \log \mathbb{E}[Z] - 2.71 \).

It can be shown that this function is convex and has a unique solution:

\[ \text{(Because } \frac{\partial^2}{\partial \theta^2} \left( \log \mathbb{E}[Z] - 2.71 \right) \text{ is positive definite).} \]
ML estimation for exponential distribution is a convex optimization function—this means that there are algorithms which are guaranteed to converge to the correct solution.

**Example: Generalized Iterative Scaling (GIS)**

\[
\begin{align*}
\mathbf{\Psi}^{t+1} &= \mathbf{\Psi}^t - \log \mathbf{\Psi}^t + \log \mathbf{\Psi} \\
\text{where} & \quad \mathbf{\Psi}^t = \frac{1}{n} \sum_{x} \Phi(x) \mathbf{P}(x | \mathbf{\Psi}^t),
\end{align*}
\]

**Notation:** \( \log \mathbf{\Psi} \) is a vector with components \( \log \psi_1, \log \psi_2, \ldots, \log \psi_m \)

But this requires computing

\[
\frac{1}{n} \sum_{x} \Phi(x) \mathbf{P}(x | \mathbf{\Psi}^t)
\]

which is often difficult.

(For people who took CS 202, there are often statistical/MCMC methods which can (approximately) compute this rapidly.)
(7) How does this apply to visual?

Consider the weak membrane model of images

Geman & Geman, Blake & Zisserman, Mumford-Shah

In probabilistic terms, this can be formulated

\[ P(I \mid W) = \prod_i P(I_i \mid W_i) \]

\[ \text{with} \quad P(I_i \mid W_i) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(I_i - W_i)^2}{2\sigma^2}} \]

The observed image \( I \) is a corrupted version of a
true image \( W \). The corruption is by additive Gaussian noise

\[ I_i = W_i + \xi_i \quad \xi_i \sim N(0, \sigma^2) \]

\[ \xi_i \]

\[ W \]

Gibbs distribution

\[ P(W) = \frac{1}{Z} e^{-E[W]} \]

\[ E[W] = \sum_{i \in \text{image}} \psi(W_i \mid W_j) \]

\[ \psi \]

\[ \text{German tram}\]
(8) \text{ \textit{Walk(i) is the neighborhood structure}} \\
\text{\textit{e.g. image lattice.}}

\text{What function } \Psi(w_i, w_j) \text{?}

\text{A natural choice is } \Psi(w_i, w_j) = (w_i - w_j)^2

\text{penalize the square of the difference between the intensity of neighboring pixels.}

\text{Advantages:}

(i) this makes it easy to learn the distribution from training data. It is a Gaussian distribution which, as we have seen, can be learnt by analytic methods (i.e. no need for steepest descent or BFGS).

(ii) this makes inference easy. To estimate \( \hat{w} = \arg \max P(w | x) \)

\text{reduces to minimizing:}

\[ E[w] = \frac{1}{2} \sum_{i=1}^{2d} (w_i - x_i)^2 + \frac{1}{2} \sum_{i,j \in N(i)} \Psi(w_i, w_j) \]

\text{if } \Psi(w_i, w_j) \text{ is quadratic - } (w_i - w_j)^2 \text{ then } E[w] \text{ is quadratic, and its minimum can be found by solving linear equations } \frac{\partial E}{\partial w} = 0 \]
But although $(w_i - w_j)^2$ is good for inference and learning, it is not a good prior distribution $q(w_i)$.

It penalizes large $\Delta w = (w_i - w_j)$ too much.

It penalizes small $\Delta w = (w_i - w_j)$ too little.

It prefers unique means and dislikes means like $\overline{\mu}$.

A better prior is $P(w_i, w_j) = |w_i - w_j|$.

This prefers means that are very smooth.

It discourages means like $\overline{\mu}$.

But tolerates them better than the quadratic penalty.

Now, try to learn the prior from data.
\[ H(\Delta w) \] is the histogram.

This is inconsistent with a Gaussian distribution → the distribution is peaked at 0 (i.e. the intensity of most pixels is similar to those of their neighbors), but \( \Delta w \) can also be large (i.e. at edges).

\[ H(x) = \sum_i \sum_j \delta(x, w_i - w_j) \]

Exponential distribution (max-entropy principle): next lecture.

\[ P(w) = \frac{1}{Z} e^{-\frac{1}{\gamma} \sum_i \sum_j \delta(x, w_i - w_j)} \]

\[ \sum_i \sum_j \delta(x, w_i - w_j) = \sum_i \sum_j \delta(x, w_j - w_i) = \sum_i \sum_j \delta(x, w_j - w_i) \]

\[ \sum_i \sum_j \delta(x, w_j - w_i) = \sum_i \sum_j \sum_i \sum_j \delta(x, w_j - w_i) \]

\[ = \sum_i \sum_j \lambda(w_i - w_j). \]
Thus, we obtain a probability distribution with the same form of the potential — i.e., \( \sum_{i \in \text{neighb(w)}} \phi(w_i, w_j) \) — by assuming an exponential form for the distribution and choosing the statistics \( H(\lambda) = \sum_{i \in \text{neighb}(w)} \sum_{j \in \text{neighb}(w)} S(\lambda, w_i - w_j) \). (Zhu & Lampert, 1997)

The form of the learnt potential is

\[
P(w) \rightarrow P(w, L) = \frac{1}{Z} \exp[-E(w, L)]
\]

\[
E(w, L) = \sum_{i \in \text{neighb}(w)} \sum_{j \in \text{neighb}(w)} (w_i - w_j)^2 (1 - L_{ij}) + \kappa \sum_{i \in \text{neighb}(w)} \sum_{j \in \text{neighb}(w)} L_{ij}
\]

\( L_{ij} \in \{0, 1\} \) is a binary-valued variable. \( L_{ij} = 1 \) "cuts" the smoothness between \( w_i \) and \( w_j \), giving lower energy (higher probability) of \( L_{ij} = 1 \) when \( |w_i - w_j| > \kappa \), \( L_{ij} = 0 \) when \( |w_i - w_j| \leq \kappa \).
\[
P(w, l) = e^{-E(w, l) / T}
\]

\[
P(w) = \frac{1}{Z} \sum_{l} P(w, l) = e^{-\frac{E(w)}{T}} \sum_{l} \Phi(w_i, w_j)
\]

It can be shown that \( \Phi(w_i, w_j) \) has a form similar to the least for.

Note: more accurate prior can be obtained by considering higher order filters \( \Phi(w_i, w_j, ..., w_k) \).

It can be shown that the statistics of any filter \( \sum_{i} A_i w_i \), such that \( \sum_{i} A_i = 0 \)
has very similar form on each image (M. Green, Mattes, UCSD).

But this leads to complicated probability distribution.

Hard to do injustice on them.