Why do we need generative learning?

Tools & Ingredients

Maximum Likelihood Estimator, consistency

Expectation Maximisation Algorithm
1. Why do we need generative learning?

**Discriminative learning:** $p(x,y)$ unknown

- define a hypothesis class $\mathcal{H}$ of predictors $h: \mathcal{X} \to \mathcal{Y}$,
- given a training set $\mathcal{T}^m$, learn $h_m$ by empirical risk minimisation.

**However:**

- what if we need the uncertainty of the prediction $h_m(x)$?
- how to learn the predictor if only a part of the training data is annotated?
- what if the statistical relation between $x$ and $y$ depends on some latent variables $z$, which we can not observe in principle? I.e. $p(x,y,z)$, but we never see $z$ in the training data.
- what if we want to learn models that can generate realistic data $x$?
1. Why do we need generative learning?

Generative learning:

- Try to model the unknown distribution \( p(x, y) \) and estimate it from training data \( \mathcal{T}^m \Rightarrow p_m(x, y) \).

- Then predict by

\[
    h(x) = \arg\min_{y \in \mathcal{Y}} \sum_{y' \in \mathcal{Y}} p_m(y' | x) \ell(y', y).
\]

- Uncertainty of the prediction can be obtained from \( p_m(y | x) \).

- Data can be generated from \( p_m(x | y) \).

When trying to estimate \( p_m(x, y) \), we need to restrict the search to some finite or infinite set of distributions.

We also need similarity measure(s) for distributions.
2. Tools & ingredients

**Parametrised distribution family:** A set of distributions with common structure, defined up to unknown parameters.

**Example 1.** Set of all multivariate normal distributions $\mathcal{N}(\mu, V)$ on $\mathbb{R}^n$

$$p_{\mu, V}(x) = \frac{1}{(2\pi)^{n/2}|V|^{1/2}} \exp \left[ -\frac{1}{2} (x - \mu) \cdot V^{-1} \cdot (x - \mu) \right]$$

parametrised by $\mu \in \mathbb{R}^n$ and a positive (semi) definite $m \times m$ matrix $V$.

**Example 2.** An **exponential family** with density

$$p_\theta(x) = \exp \left[ \langle \phi(x), \theta \rangle - A(\theta) \right],$$

where

- $\phi(x) \in \mathbb{R}^n$ is the sufficient statistic,
- $\theta \in \mathbb{R}^n$ is the (natural) parameter and
- $A(\theta)$ is the cumulant function defined by

$$A(\theta) = \log \int_X \exp \left[ \langle \phi(x), \theta \rangle \right] dx$$
2. Tools & ingredients

Kullback-Leibler divergence: Similarity of distributions $p(x)$ and $q(x)$:

$$D_{KL}(p(x) \parallel q(x)) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)}$$

$D_{KL}$ is non-negative and is zero if and only if $p(x) = q(x) \ \forall x \in \mathcal{X}$. This follows from strict concavity of the function $\log(x)$

$$-D_{KL}(p \parallel q) = \sum_{x \in \mathcal{X}} p(x) \log \frac{q(x)}{p(x)} \leq \log \sum_{x \in \mathcal{X}} q(x)p(x) = \log 1 = 0$$

- $D_{KL}$ can be generalised for continuous distributions.
- it becomes $\infty$ if the support of $p$ is not contained in the support of $q$. 
3. Maximum Likelihood estimator, consistency

Given: a parametrised family of distributions $p_\theta(x, y)$, $\theta \in \Theta$ and an i.i.d. training set $T^m = \{(x^j, y^j) \in \mathcal{X} \times \mathcal{Y} \mid j = 1, \ldots, m\}$ generated from $p_{\theta_0}(x, y)$ with unknown $\theta_0$.

Task: estimate $\theta_0$

Maximum likelihood estimator: estimate $\theta_0$ by maximising the joint probability (density) of the training set w.r.t. $\theta$

$$\theta_m \in \arg \max_{\theta \in \Theta} \sum_{j=1}^{m} \log p_\theta(x^j, y^j)$$

Notice that $\theta_m$ depends on $T^m$, thus it is a random variable. MLE has following properties

- MLE can be biased, however
- MLE is asymptotically consistent, i.e. the sequence $\theta_m$, $m \to \infty$ converges in probability to $\theta_0$
- MLE has lowest possible variance (MSE) among all consistent estimators.
3. Maximum Likelihood estimator, consistency

Example 3. (Gaussian Discriminative Analysis)

\(x \in \mathbb{R}^n, y \in \{0, 1\}\) with \(y \sim \text{Ber}(\alpha)\) and \(x \mid y \sim \mathcal{N}(\mu_y, V)\), i.e.

\[
p(y) = \alpha^y (1 - \alpha)^{1-y}
\]

\[
p(x \mid y) = \frac{1}{(2\pi)^{n/2}|V|^{1/2}} \exp\left[-\frac{1}{2}(x - \mu_y)^T V^{-1} (x - \mu_y)\right]
\]

MLE for training data \(\mathcal{T}^m = \{(x^j, y^j) \mid j = 1, \ldots, m\}\):

Denote \(I_1 = \{j \mid y^j = 1\}\) and \(I_0\) correspondingly.

\[
\alpha^* = \frac{1}{m} |I_1|
\]

\[
\mu_0^* = \frac{1}{|I_0|} \sum_{j \in I_0} x^j, \quad \mu_1^* = \frac{1}{|I_1|} \sum_{j \in I_1} x^j
\]

\[
V^* = \frac{1}{m} \sum_{j=1}^{m} (x^j - \mu_{y^j}) \otimes (x^j - \mu_{y^j})
\]
3. Maximum Likelihood estimator, consistency

Let \( T^m = \{ x^j \mid j = 1, \ldots, m \} \) be i.i.d. generated from \( p_{\theta_0}(x) \), with \( \theta_0 \in \Theta \) unknown.

Which conditions ensure consistency of the MLE \( \theta_m = \arg\max_{\theta \in \Theta} \log p_{\theta}(T^m) \)?

\[
\mathbb{P}_{\theta_0}(\|\theta_0 - \theta_m(T^m)\| > \epsilon) \xrightarrow{m \to \infty} 0
\]

Denote log-likelihood of training data by \( L(\theta, T^m) = \frac{1}{m} \sum_{i=1}^{m} \log p_{\theta}(x^i) \)

and expected log-likelihood \( L(\theta) = \mathbb{E}_{\theta_0}(L(\theta, T^m)) = \sum_{x \in X} p_{\theta_0}(x) \log p_{\theta}(x) \)

Consider \( L(\theta, T^m) = L(\theta) + \left[ L(\theta, T^m) - L(\theta) \right] \)

- The model should be identifiable, i.e. \( \theta_0 = \arg\max_{\theta \in \Theta} L(\theta) \)
- Ensure that the Uniform Law of Large Numbers (ULLN) holds, i.e.

\[
\mathbb{P}_{\theta_0}(\sup_{\theta \in \Theta} |L(\theta, T^m) - L(\theta)| > \epsilon) \xrightarrow{m \to \infty} 0
\]

for any \( \epsilon > 0 \).
3. Maximum Likelihood estimator, consistency

**Identifiability** of the model $\theta_0$ is easy to prove if $p_{\theta_0}(x) \neq p_\theta(z)$ holds $\forall \theta \neq \theta_0$.

$$L(\theta_0) - L(\theta) = D_{KL}(p_{\theta_0}(x) \parallel p_\theta(x)) \geq 0$$

and becomes zero if and only if $\theta = \theta_0$.

**ULLN** can be ensured e.g. by requiring that

- $L(\theta, T^m)$ is continuous in $\theta$ and $\Theta \subset \mathbb{R}^k$ is compact.
- $L(\theta, T^m)$ can be upper bounded: $\log p_\theta(x) \leq d(x)$ $\forall \theta$ with $\mathbb{E}_{\theta_0}d(x) < \infty$. 
4. The Expectation Maximisation Algorithm

Unsupervised generative learning:

- The joint p.d. $p_{\theta}(x, y)$, $\theta \in \Theta$ is known up to the parameter $\theta \in \Theta$,
- given training data $\mathcal{T}^m = \{x^i \in \mathcal{X} \mid i = 1, 2, \ldots, m\}$ i.i.d. generated from $p_{\theta_0}$.

How shall we implement the MLE

$$
\theta_m(\mathcal{T}^m) = \arg \max_{\theta \in \Theta} \frac{1}{m} \sum_{x \in \mathcal{T}^m} \log p_{\theta}(x) = \arg \max_{\theta \in \Theta} \mathbb{E}_{\mathcal{T}^m} \left[ \log \sum_{y \in \mathcal{Y}} p_{\theta}(x, y) \right]
$$

- If $\theta$ is a single parameter or a vector of homogeneous parameters $\Rightarrow$ maximise the log-likelihood directly.
- If $\theta$ is a collection of heterogeneous parameters $\Rightarrow$ apply the **Expectation Maximisation Algorithm** (Schlesinger, 1968, Sundberg, 1974, Dempster, Laird, and Rubin, 1977)
4. The Expectation Maximisation Algorithm

**EM approach:**

- Introduce auxiliary variables $\alpha_x(y) \geq 0$, for each $x \in T^m$, s.t. $\sum_{y \in Y} \alpha_x(y) = 1$
- Construct a lower bound of the log-likelihood $L(\theta, T^m) \geq L_B(\theta, \alpha, T^m)$
- Maximise this lower bound by block-wise coordinate ascent.

Construct the bound:

$$L(\theta, T^m) = \mathbb{E}_{T^m} \left[ \log \sum_{y \in Y} p_\theta(x, y) \right] = \mathbb{E}_{T^m} \left[ \log \sum_{y \in Y} \frac{\alpha_x(y)}{\alpha_x(y)} p_\theta(x, y) \right] \geq$$

$$L_B(\theta, \alpha, T^m) = \mathbb{E}_{T^m} \sum_{y \in Y} \left[ \alpha_x(y) \log p_\theta(x, y) - \alpha_x(y) \log \alpha_x(y) \right]$$

The following equivalent representation shows the difference between $L(\theta, T^m)$ and $L_B(\theta, \alpha, T^m)$:

$$L_B(\theta, \alpha, T^m) = \mathbb{E}_{T^m} \left[ \log p_\theta(x) \right] - \mathbb{E}_{T^m} \left[ D_{KL}(\alpha_x(y) \| p_\theta(y | x)) \right]$$
4. The Expectation Maximisation Algorithm

Maximise $L_B(\theta, \alpha, T^m)$ by block-coordinate ascent:

Start with some $\theta^{(0)}$ and iterate

E-step Fix the current $\theta^{(t)}$, maximise $L_B(\theta^{(t)}, \alpha, T^m)$ w.r.t. $\alpha$-s. This gives

$$\alpha_x^{(t)}(y) = p_{\theta^{(t)}}(y \mid x).$$

M-step Fix the current $\alpha^{(t)}$ and maximise $L_B(\theta, \alpha^{(t)}, T^m)$ w.r.t. $\theta$.

$$\theta^{(t+1)} = \arg \max_{\theta \in \Theta} \mathbb{E}_{T^m} \left[ \sum_{y \in \mathcal{Y}} \alpha_x^{(t)}(y) \log p_\theta(x, y) \right]$$

This is equivalent to solving the MLE for annotated training data.

Claims:

- The bound is tight if $\alpha_x(y) = p_{\theta}(y \mid x)$,
- The sequence of likelihood values $L(\theta^{(t)}, T^m), t = 1, 2, \ldots$ is increasing, and the sequence $\alpha^{(t)}, t = 1, 2, \ldots$ is convergent (under mild assumptions).
4. The Expectation Maximisation Algorithm

Example: Latent mode model (mixture) for images of digits

- $x = \{x_i \mid i \in D\}$ image on the pixel domain $D \in \mathbb{Z}^2$,
- $x_i \in \{0, 1, 2, \ldots, 255\}$
- $k \in K$ latent variable (mode indicator),
- joint distribution - Naive Bayes model

$$p(x, k) = p(k) \prod_{i \in D} p(x_i \mid k)$$

Learning problem: Given i.i.d. training data $\mathcal{T}^m = \{x^j \mid j = 1, 2, \ldots, m\}$, estimate the mode probabilities $p(k)$ and the conditional probabilities $p(x_i \mid k)$, $\forall x_i \in \mathcal{B}, k \in K$ and $i \in D$. 
4. The Expectation Maximisation Algorithm

Applying the EM algorithm: Start with some model \( p^{(0)}(k), p^{(0)}(x_i | k) \) and iterate the following steps until convergence.

**E-step** Given the current model estimate \( p^{(t)}(k), p^{(t)}(x_i | k) \), compute the posterior mode probabilities for each image \( x \) in the training data \( \mathcal{T}^m \)

\[
\alpha_x^{(t)}(k) = p^{(t)}(k | x) = \frac{p^{(t)}(k) \prod_{i \in D} p^{(t)}(x_i | k)}{\sum_{k'} p^{(t)}(k') \prod_{i \in D} p^{(t)}(x_i | k')}.
\]

**M-step** Re-estimate the model by solving

\[
\mathbb{E}_{\mathcal{T}^m} \left[ \sum_{k \in K} \alpha_x^{(t)}(k) \left[ \log p(k) + \sum_{i \in D} \log p(x_i | k) \right] \right] \rightarrow \max_p
\]

This gives

\[
p^{(t+1)}(k) = \mathbb{E}_{\mathcal{T}^m} \left[ \alpha_x^{(t)}(k) \right]
\]

\[
p^{(t+1)}(x_i = b | k) = \frac{\mathbb{E}_{\mathcal{T}^m} \left[ \alpha_x^{(t)}(k) | x_i = b \right]}{\mathbb{E}_{\mathcal{T}^m} \left[ \alpha_x^{(t)}(k) \right]}
\]
4. The Expectation Maximisation Algorithm

Additional reading:

Schlesinger, Hlavac, Ten Lectures on Statistical and Structural Pattern Recognition, Chapter 6, Kluwer 2002 (also available in Czech)

Thomas P. Minka, Expectation-Maximization as lower bound maximization, 1998 (short tutorial, available on the internet)