CS 181 Exam 2 Notesheet

Elvin Lo

Clustering, Mixture Models, PCA

K-Means: Initialize centroids by selecting data points either randomly or proportionally to the squared distance from the closest cluster center (K-Means++). Then iteratively assign points to clusters and update the centroids until convergence.

- **Lloyd's:** For L2 distance, define the loss as the sum of squared distances to the cluster centers, denoted using one-hot responsibility vectors $\mathbf{r}_n$. Then the optimal centroids are the average of their data points:
$$\mathcal{L}\left(\mathbf{X}, \{\boldsymbol{\mu}\}_{c=1}^{C}, \{\mathbf{r}\}_{n=1}^{N}\right) = \sum_{n=1}^{N}\sum_{c=1}^{C} r_{nc}\left\|\mathbf{x}_n - \boldsymbol{\mu}_c\right\|_2^2 \implies \frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}_c} = -2\sum_{n=1}^{N} r_{nc}\left(\mathbf{x}_n - \boldsymbol{\mu}_c\right) \implies \boldsymbol{\mu}_c = \frac{\sum_{n=1}^{N} r_{nc}\mathbf{x}_n}{\sum_{n=1}^{N} r_{nc}}$$
- **K-Medioids:** For categorical data, we update centroids to the median since the mean may not make sense.
- **Number of clusters:** Loss strictly decreases with number of clusters C; we choose C to be the "elbow."

HAC: Beginning with N clusters for each data point, merge clusters via an intercluster distance metric (linkage criterion).

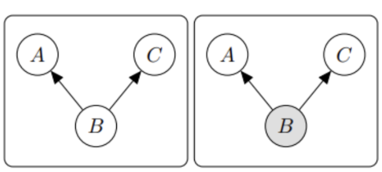
Mixture models: We wish to fit $\boldsymbol{\theta}$, the parameter for our categorical prior, and $\{\mathbf{w}_k\}_{k=1}^{K}$, the parameters for our class-conditional distributions. The complete-data log-likelihood is
$$\log L\left(\boldsymbol{\theta}, \{\beta_n\}_{k=1}^{K}\right) = \sum_{n=1}^{N} \log\left(\prod_{k=1}^{K} p\left(x_n \mid \mathbf{z}_n = C_k\right)^{I(\mathbf{z}_n = C_k)} \theta_k^{I(\mathbf{z}_n = C_k)}\right) = \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk}\big(\log\theta_k + \log p\left(x_n \mid \mathbf{z}_n = C_k\right)\big),$$
and we optimize our parameters iteratively with EM. In the E-step, we calculate for each data point $\mathbf{q_n} = \mathbb{E}\left[z_n \mid x_n\right]$, with $q_{nk} \propto p\left(\mathbf{x}_n \mid \mathbf{z}_n = C_k; \mathbf{w}\right) p\left(\mathbf{z}_n = C_k; \boldsymbol{\theta}\right)$. We then update our parameters to maximize the expected likelihood.

ELBO: For q some other distribution on $\mathbf{z}$, we have
$$\sum_{n=1}^{N} \log\left[\sum_{k=1}^{K} p\left(\mathbf{x}_n \mid \mathbf{z}_n = C_k; \boldsymbol{\theta}, \mathbf{w}\right) p\left(\mathbf{z}_n = C_k \mid \boldsymbol{\theta}, \mathbf{w}\right)\right] = \sum_{n=1}^{N} \log \mathrm{E}_{\mathbf{z}_n \sim q(\mathbf{z}_n)}\left[\frac{p\left(\mathbf{x}_n \mid \mathbf{z}_n; \boldsymbol{\theta}, \mathbf{w}\right) p\left(\mathbf{z}_n \mid \boldsymbol{\theta}, \mathbf{w}\right)}{q\left(\mathbf{z}_n\right)}\right],$$
which by Jensen's is greater than the ELBO (in which we pass the log above into the expectation). So in EM, we choose q maximizing ELBO$(\mathbf{w}, q \mid \mathbf{x})$, keeping the bound tight, and then choose parameters maximizing the ELBO.

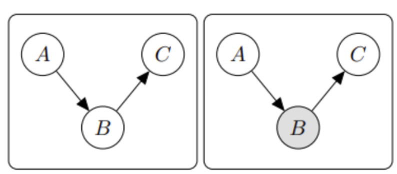
PCA: For mean-centered $\mathbf{X}$, the PCs are the eigenvectors of the empirical covariance matrix $\mathbf{S} = \frac{1}{N}\mathbf{X}\mathbf{X}^\top$. To calculate these, the SVD of a matrix is $\mathbf{X} = \mathbf{U}\mathbf{Z}\mathbf{V}^\top$, where $\mathbf{Z}$ is diagonal and $\mathbf{U}, \mathbf{V}$ are orthogonal (i.e., orthonormal columns). Then eigenvalues of $\mathbf{S}$ are the entries of $\frac{1}{N}\mathbf{Z}^2$, and the columns of $\mathbf{U}$ are the corresponding eigenvectors.

Bayesian networks & HMMs

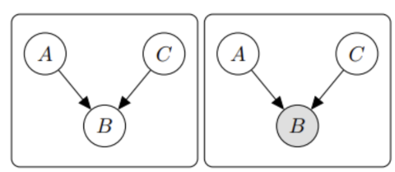
$A \not\perp C \to B$ observed $\to A \perp C$



$A \not\perp C \to B$ observed $\to A \perp C$



$A \perp C \to B$ observed $\to A \not\perp C$



HMM setup: We have N sequences of one-hot emissions, each of form $\mathbf{x}_1, \ldots, \mathbf{x}_n$, with corresponding one-hot latent states $\mathbf{s}_1, \ldots, \mathbf{s}_n$. There are K possible states and M possible observations. We model the joint distribution
$$p\left(\mathbf{s}_1, \ldots, \mathbf{s}_n, \mathbf{x}_1, \ldots, \mathbf{x}_n\right) = p\left(\mathbf{s}_1, \ldots, \mathbf{s}_n\right) p\left(\mathbf{x}_1, \ldots, \mathbf{x}_n \mid \mathbf{s}_1, \ldots, \mathbf{s}_n\right) = p\left(\mathbf{s}_1; \boldsymbol{\theta}\right) \prod_{t=1}^{n-1} p\left(\mathbf{s}_{t+1} \mid \mathbf{s}_t; \mathbf{T}\right) \prod_{t=1}^{n} p\left(\mathbf{x}_t \mid \mathbf{s}_t; \boldsymbol{\pi}\right).$$
We parameterize the categorical prior for $\mathbf{s}_1$ with $\boldsymbol{\theta} \in [0,1]^K$, the transition matrix with $\mathbf{T} \in [0,1]^{K \times K}$ where $T_{i,j}$ is the transition from i to j, and the state-conditional distribution of observations with $\boldsymbol{\pi} \in [0,1]^{K \times M}$.

Forward-backward algorithm: Factor $p\left(\mathbf{x}_1, \ldots, \mathbf{x}_n, \mathbf{s}_t\right) = p\left(\mathbf{x}_1, \ldots, \mathbf{x}_t, \mathbf{s}_t\right) p\left(\mathbf{x}_{t+1}, \ldots, \mathbf{x}_n \mid s_t\right) = \alpha_t(\mathbf{s}_t)\beta_t(\mathbf{s}_t)$, with
$$\alpha_t\left(\mathbf{s}_t\right) = \begin{cases} p\left(\mathbf{x}_1 \mid \mathbf{s}_1\right) p\left(\mathbf{s}_1\right) & \text{if } t = 1, \text{ else} \\ p\left(\mathbf{x}_t \mid \mathbf{s}_t\right) \sum_{\mathbf{s}_{t-1}} p\left(\mathbf{s}_t \mid \mathbf{s}_{t-1}\right) \alpha_{t-1}\left(\mathbf{s}_{t-1}\right). \end{cases} \qquad \beta_t\left(\mathbf{s}_t\right) = \begin{cases} 1 \text{ if } t = n, \text{ else} \\ \sum_{\mathbf{s}_{t+1}} p\left(\mathbf{s}_{t+1} \mid \mathbf{s}_t\right) p\left(\mathbf{x}_{t+1} \mid \mathbf{s}_{t+1}\right) \beta_{t+1}\left(\mathbf{s}_{t+1}\right). \end{cases}$$
After calculating all α_t and β_t, we can perform inference tasks by marginalizing (summing) over possible $\mathbf{s}_t$ or $\mathbf{s}_{t-1}$.

Viterbi: Let γ_t the likelihood of the observations if the current state is $\mathbf{s}_t$ and we already maximized over $\mathbf{s}_1, \ldots, \mathbf{s}_{t-1}$,
$$\gamma_t\left(\mathbf{s}_t\right) = \max_{\mathbf{s}_1, \ldots, \mathbf{s}_{t-1}} p\left(\mathbf{s}_1, \ldots, \mathbf{s}_t, \mathbf{x}_1, \ldots, \mathbf{x}_t\right) = \begin{cases} p\left(\mathbf{x}_1 \mid \mathbf{s}_1\right) p\left(\mathbf{s}_1\right) \text{ if } t = 1, \text{ else} \\ p\left(\mathbf{x}_t \mid \mathbf{s}_t\right) \max_{\mathbf{s}_{t-1}} p\left(\mathbf{s}_t \mid \mathbf{s}_{t-1}\right) \gamma_{t-1}\left(\mathbf{s}_{t-1}\right). \end{cases}$$

To find the optimal path, we then choose $\mathbf{s_t}$ maximizing γ_t, and $\mathbf{s}_{t-1}$ maximizing $p\left(\mathbf{s}_t \mid \mathbf{s}_{t-1}\right)\gamma_{t-1}\left(\mathbf{s}_{t-1}\right)$, and so on.

Fitting HMM parameters with EM: We randomly initialize $\boldsymbol{\theta}, \mathbf{T}, \boldsymbol{\pi}$. For our current values, we compute α- and β-values using Forward-Backward. Then we compute $\{\mathbf{q}_i\}_{i=1}^n$ with $q_{t,k}^i = p\left(\mathbf{s}_t^i = k \mid \mathbf{x}_1^i, \ldots, \mathbf{x}_n^i\right)$, and $\{\mathbf{Q}_{t,t+1}^i\}_{i=1}^n$ with $Q_{t,t+1,k,\ell}^i = p\left(\mathbf{s}_t^i = k, \mathbf{s}_{t+1}^i = \ell \mid \mathbf{x}_1^i, \ldots, \mathbf{x}_n^i\right)$. This yields

$$\mathbb{E}_{\mathbf{s}^i}\left[\ln\left(p\left(\mathbf{x}^i, \mathbf{s}^i\right)\right)\right] = \sum_{k=1}^K q_{1k}^i \ln\theta_k + \sum_{t=1}^{n-1}\sum_{k=1}^K\sum_{\ell=1}^K Q_{t,t+1,k,\ell}^i \ln T_{k,l} + \sum_{t=1}^n\sum_{k=1}^K q_{t,k}^i \sum_{m=1}^M x_{t,m}^i \ln\pi_{k,m}$$

$$\implies \theta_k = \frac{\sum_{i=1}^N q_{1,k}^i}{N} \qquad T_{k,l} = \frac{\sum_{i=1}^N \sum_{t=1}^{n-1} Q_{t,t+1,k,l}^i}{\sum_{i=1}^N \sum_{t=1}^{n-1} q_{t,k}^i} \qquad \pi_{k,m} = \frac{\sum_{i=1}^N \sum_{t=1}^n q_{t,k}^i x_{t,m}^i}{\sum_{i=1}^N \sum_{t=1}^n q_{t,k}^i}$$

MDPs & Model-Free RL

Our environment has states S, actions A, reward function $r: S \times A \to [0,1]$, and transition model $p(s' \mid s, a)$. If known, we have an MDP and can both evaluate policies and learn the optimal policy π^*. If unknown, we must find π^* with RL.

Finite time horizon MDP: Let $V_T^\pi(s)$ be the expected total reward in T timesteps following π. Then for any π,

$$V_T^\pi(s) = \mathbb{E}_{s_1,\ldots,s_T}\left[\sum_{t=0}^T r\left(s_t, \pi_{(T-t)}\left(s_t\right)\right)\right] = \begin{cases} r\left(s, \pi_{(1)}(s)\right) & \text{if } t = 1 \\ r\left(s, \pi_{(t)}(s)\right) + \sum_{s' \in S} p\left(s' \mid s, \pi_{(t)}(s)\right) V_{(t-1)}^\pi\left(s'\right) & \text{otherwise} \end{cases}$$

where $\pi_{(t)}$ is the policy with t timesteps left. In $O\left(|S|^2|A|T\right)$, value iteration finds the optimal policy by taking at each timestep the action maximizing the expected sum of our immediate reward and expected future reward:

$$\pi_{(1)}^*(s) = \arg\max_a [r(s,a)] \qquad \pi_{(t+1)}^*(s) = \arg\max_a \left[r(s,a) + \sum_{s' \in \mathcal{S}} p\left(s' \mid s, a\right) V_{(t)}^*\left(s'\right)\right]$$

Infinite horizon MDP: With $T \to \infty$, we evaluate a policy π by solving a system of Bellman consistency equations,

$$V^\pi(s) \coloneqq \mathbb{E}_{s_1,s_2,\ldots}\left[\sum_{t=0}^\infty \gamma^t r\left(s_t, \pi\left(s_t\right)\right)\right] \qquad V^\pi(s) = r(s,\pi(s)) + \gamma\sum_{s' \in S} p\left(s' \mid s, \pi(s)\right) V^\pi\left(s'\right).$$

Alternatively, we may find V^π by initializing $V(s) = 0, \forall s$, and then iteratively updating $V \leftarrow V'$ with

$$V'(s) = r(s,\pi(s)) + \gamma\sum_{s' \in S} p\left(s' \mid s, \pi(s)\right) V\left(s'\right), \forall s \qquad \text{until} \qquad \Delta = \max\left(|V'(s) - V(s)|\right).$$

- **Value iteration:** To find π^*, we first calculate V^* by initializing $V(s) = 0, \forall s$, and then iteratively updating

  $$V \leftarrow V' \qquad \text{where} \qquad V'(s) = \max_{a \in A}\left[r(s,a) + \gamma\sum_{s' \in S} p\left(s' \mid s, a\right) V\left(s'\right)\right], \forall s$$

  until convergence. Then note that $V^* \triangleq V^{\pi^*}$ and that V^* satisfies Bellman optimality, so we find π^* to be

  $$V^*(s) = \max_{a \in A}\left[r(s,a) + \gamma\sum_{s' \in S} p\left(s' \mid s, a\right) V^*\left(s'\right)\right] \implies \pi^*(s) = \arg\max_{a \in A}\left[r(s,a) + \gamma\sum_{s' \in S} p\left(s' \mid s, a\right) V^*\left(s'\right)\right]$$

- **Policy iteration:** Beginning with some π, we iteratively evaluate V^π(E-step) and improve π (I-step) by the update

  $$\pi'(s) \leftarrow \arg\max_{a \in A}\left[r(s,a) + \gamma\sum_{s' \in S} p\left(s' \mid s, a\right) V^\pi\left(s'\right)\right], \quad \forall s \qquad \text{until } \pi \text{ converges}$$

  Policy iteration takes more computation per iteration, but tends to converge faster in practice.

Value-based methods: We try to learn optimal Q-values, which determine our optimal policy.

$$Q^*(s,a) \coloneqq r(s,a) + \gamma\sum_{s' \in S} p\left(s' \mid s, a\right) V^*\left(s'\right), \forall s, a \qquad \implies \qquad \pi^*(s) = \arg\max_a\ Q^*(s,a),$$

- **SARSA (on-policy):** Given an experience (s, a, r, s', a'), where a' is chosen by an ϵ-greedy method, we update

  $$\pi(s) = \begin{cases} \operatorname{argmax}_a Q(s,a) & \text{with probability } 1-\epsilon \\ \text{random} & \text{with probability } \epsilon \end{cases} \qquad Q(s,a) \leftarrow Q(s,a) + \alpha_t\left[r + \gamma Q\left(s', a'\right) - Q(s,a)\right],\ \alpha_t \in [0,1]$$

- **Q-learning (off-policy):** We update using observations (s, a, r, s'), intuitively performing SGD to bring our Q-values closer to satisfying the Bellman condition,

  $$Q^*(s,a) = r(s,a) + \gamma\sum_{s' \in S} p\left(s' \mid s, a\right)\max_{a' \in A}\left[Q^*\left(s', a'\right)\right], \forall s, a \qquad Q(s,a) \leftarrow Q(s,a) + \alpha_t\left[r + \gamma\max_{a'} Q\left(s', a'\right) - Q(s,a)\right]$$

Policy learning (on-policy): We parameterize the policy space with a finite-dimensional parameter space, such that π_θ is the policy associated to parameter θ. Then we may update θ with SGD to find the optimal policy.