Motivation
Cluster-point assignment:

\[ p(z_n = k) \]

Cluster parameters:

\[ \Theta = \{\theta_1, \theta_2, \ldots, \theta_K\} \]
Clusters:

Points:

Cluster-point assignment:
$$p(z_n = k)$$

Cluster component parameters:
$$\Theta = \{\theta_1, \theta_2, \ldots, \theta_K\}$$

The usual scenario:

Loop until convergence:

For $n = 1, \ldots, N$, and $k = 1, \ldots, K$
$$p(z_n = k) \leftarrow f(\Theta, k, n)$$

For $k = 1, \ldots, K$
$$\theta_k \leftarrow g(p(z), k)$$
Loop until convergence:

\[ \text{For } n = 1, \ldots, N, \text{ and } k = 1, \ldots, K \]
\[ p(z_n = k) \leftarrow f(\Theta, k, n) \]

\[ \text{For } k = 1, \ldots, K \]
\[ \theta_k \leftarrow g(p(z), k) \]

- How to keep track of convergence?
  - A simple rule for \textit{k-means}
    - When the assignments don’t change.
  - Alternatively keep track of the \textit{k-means} global objective:
    \[
    L(\Theta) = \sum_n \sum_k \left| x_n - \theta_{z_n} \right|^2
    \]
- Dirichlet Process Mixture with Variational Inference
  - Lower bound on the marginal likelihood
    \[
    \mathcal{L} = h(\Theta, p(z))
    \]
Loop until convergence:

For $n = 1, \ldots, N$, and $k = 1, \ldots, K$
\[ p(z_n = k) \leftarrow f(\Theta, k, n) \]

For $k = 1, \ldots, K$
\[ \theta_k \leftarrow g(p(z), k) \]

- What if the data doesn’t fit in the disk?
- What if we want to accelerate this?

Divide the data into $B$ batches

- Assumption:
  - Independently sampled assignment into batches
  - Enough samples inside each data batch
    - For latent components

$B \ll N$
Loop until $\mathcal{L}$ convergence:

For $n = 1, \ldots, N$, and $k = 1, \ldots, K$

$p(z_n = k) \leftarrow f(\Theta, k, n)$

For $k = 1, \ldots, K$

$\theta_k \leftarrow g(p(z), k)$

Divide the data into $B$ batches

- Clusters are shared between data batches!

Define global / local cluster parameters

Global component parameters:

$\Theta^0 = [\theta_1^0 \ \theta_2^0 \ \cdots \ \theta_K^0]$  

Local component parameter:

$\Theta^1 = [\theta_1^1 \ \theta_2^1 \ \cdots \ \theta_K^1]$  

$\Theta^2 = [\theta_1^2 \ \theta_2^2 \ \cdots \ \theta_K^2]$  

$\vdots$  

$\Theta^B = [\theta_1^B \ \theta_2^B \ \cdots \ \theta_K^B]$
Loop until $L$ convergence:

For $n = 1, \ldots, N$, and $k = 1, \ldots, K$

$$p(z_n = k) \leftarrow f(\Theta, k, n)$$

For $k = 1, \ldots, K$

$$\theta_k \leftarrow g(p(z), k)$$

- How to aggregate the parameters?

**K-means example:**
The global cluster center, is weighted average of the local cluster centers.

- Similar rules holds in DPM:
  - For each component: $k$
    $$\theta^0_k = \sum_b \theta^b_k$$
  - For all components:
    $$\Theta^0 = \sum_b \Theta^b$$
Loop until $\mathcal{L}$ convergence:

For $n = 1, \ldots, N$, and $k = 1, \ldots, K$
\[ p(z_n = k) \leftarrow f(\Theta, k, n) \]

For $k = 1, \ldots, K$
\[ \theta_k \leftarrow g(p(z), k) \]

- How does the algorithm look like?

Loop until $\mathcal{L}$ convergence:
  Randomly choose: $b \in \{1, 2, 3, \ldots, B\}$
  For $n \in B_b$, and $k = 1, \ldots, K$
  \[ p(z_n = k) \leftarrow f(\Theta^0, k, n) \]
  For cluster $k = 1, 2, 3, \ldots, K$
  \[
  \theta_k^{(new)} \leftarrow g(p(z), k, b) \\
  \theta_k^0 \leftarrow \theta_k^0 - \theta_k^{(old)} + \theta_k^{(new)} \\
  \theta_k^{(old)} \leftarrow \theta_k^{(new)}
  \]

- Models and analysis for K-means:

Loop until \( \mathcal{L} \) convergence:

For \( n = 1, \ldots, N \), and \( k = 1, \ldots, K \)
\[
p(z_n = k) \leftarrow f(\Theta, k, n)
\]

For \( k = 1, \ldots, K \)
\[
\theta_k \leftarrow g(p(z), k)
\]

• Compare these two:

  (this work)

Loop until \( \mathcal{L}(q) \) convergence:

Randomly choose: \( b \in \{1, 2, 3, \ldots, B\} \)

For \( n \in B_b \), and \( k = 1, \ldots, K \)
\[
p(z_n = k) \leftarrow f(\Theta^0, k, n)
\]

For cluster \( k = 1, 2, 3, \ldots, K \)
\[
\theta^b_{k(new)} \leftarrow g(p(z), k, b)
\]
\[
\theta^0_0 \leftarrow \theta^0_0 - \theta^b_{k(old)} + \theta^b_{k(new)}
\]
\[
\theta^b_{k(old)} \leftarrow \theta^b_{k(new)}
\]

(Stochastic Optimization for DPM, Hoffman et al., JMLR, 2013)

Loop until \( \mathcal{L}(q) \) convergence:

Randomly choose: \( b \in \{1, 2, 3, \ldots, B\} \)

For \( n \in B_b \), and \( k = 1, \ldots, K \)
\[
p(z_n = k) \leftarrow f(\Theta^0, k, n)
\]

For cluster \( k = 1, 2, 3, \ldots, K \)
\[
\theta^b_k \leftarrow g(p(z), k, b)
\]
\[
\theta^0_0 \leftarrow (1 - \rho_i)\theta^0_0 + \rho_i \cdot \frac{n}{|B_b|}
\]

\[\sum_i \rho_i \rightarrow +\infty, \sum_i \rho_i^2 < +\infty\]
Loop until $\mathcal{L}$ convergence:

For $n = 1, ..., N$, and $k = 1, ..., K$
$$p(z_n = k) \leftarrow f(\Theta, k, n)$$

For $k = 1, ..., K$
$$\theta_k \leftarrow g(p(z), k)$$

- Note:
  - They use a nonparametric model!
  - But ....
    - the inference uses maximum-clusters
  - How to get adaptive number of maximum-clusters?
    - Heuristics to **add** new clusters, or **remove** them.

**Dirichlet Process Mixture (DPM)**
Birth moves

• The strategy in this work:
  • **Collection:**
    • Choose a random target component $k'$
    • Collect all the data points that $p(x_n = k') > \tau_{\text{threshold}}$ ($p(x_n = k') > \tau_{\text{threshold}}$)
  • **Creation:** run a DPM on the subsampled data ($K' = 10$)
  • **Adoption:** Update parameters with $K' + K$
Other birth moves?

- Past: split-merge schema for single-batch learning
  - E.g. EM (Ueda et al., 2000), Variational-HDP (Bryant and Sudderth, 2012), etc.
    - Split a new component
    - Fix everything
    - Run restricted updates.
    - Decide whether to keep it or not
  - Many similar Algorithms for k-means
    - (Hamerly & Elkan, NIPS, 2004), (Feng & Hammerly, NIPS, 2007), etc.
- This strategy unlikely to work in the batch mode:
  - Each batch might not contain enough examples of the missing component
Merge clusters

New cluster $k_m$ takes over all responsibility of old clusters $k_a$ and $k_b$:

$$
\theta_{k_m}^0 \leftarrow \theta_{k_a}^0 + \theta_{k_b}^0
$$

$$
p(z_n = k_m) \leftarrow p(z_n = k_a) + p(z_n = k_b)
$$

Accept or reject:

$$
\mathcal{L}(q_{\text{mrege}}) > \mathcal{L}(q)\
$$

How to choose pair?

• Randomly select $k_a$
• Randomly select $k_b$ proportional to the relative marginal likelihood:

$$
p(k_b | k_a) \propto \frac{\mathcal{L}_{k_a + k_b}}{\mathcal{L}_{k_b}}
$$
Results: toy data

- Data (N=100000) synthetic image patches
- Generated by a zero mean GMM with 8 equally common components
- Each component has $25 \times 25$ covariance matrix producing $5 \times 5$ patches
- Goal: recovering these patches, and their size ($K=8$)
- $B = 100$ (1000 examples per batch)
- MO-BM starts with $K = 1$,
- Truncation-fixed start with $K = 25$ with 10 random initialization
Results: Clustering tiny images

- 108,754 images of size $32 \times 32$
- Projected in 50 dimension using PCA
- MO-BM starting at $K = 1$, others have $K = 100$
- full-mean DP-GMM
Summary

- A distributed algorithm for Dirichlet Process Mixture model
- Split-merge schema
- Interesting improvement over the similar methods for DPM.
- Theoretical convergence guarantees?
- Theoretical justification for choosing batches B, or experiments investigating it?
- Previous “almost” similar algorithms, specially on $k$-means?
Bayesian Inference

\[ p(\theta | y) = \frac{p(y|\theta)p(\theta)}{p(y)} \]

- Goal:
  \[ \theta^* = \arg \max_{\theta} p(\theta | y) \]
- But posterior hard to calculate:
  \[ p(\theta | y) = \frac{p(y|\theta)p(\theta)}{\int p(y|\theta)p(\theta)d\theta} \]
Lower-bounding marginal likelihood

\[ p(\theta|x) \sim q(\theta) \]

\[
\log p(x) \geq \log p(x) - KL(q(\theta)||p(\theta|x)) \\
= \int q(\theta) \log \frac{p(x|\theta)p(\theta)}{q(\theta)} d\theta = \mathcal{L}(q)
\]

Given that,

\[
KL(q(\theta)||p(\theta|x)) = \int q(\theta) \log \frac{q(\theta)}{p(x|\theta)} d\theta
\]

**Advantage**
- Turn Bayesian inference into optimization
- Gives lower bound on the marginal likelihood

**Disadvantage**
- Add more non-convexity to the objective
- Cannot easily applied when non-conjugate family

\[
g(\theta) = p(x|\theta)p(\theta)
\]
Variational Bayes for Conjugate families

- Given the joint distribution:
  \[ p(x, \theta) \]
- And by making following decomposition assumption:
  \[ \theta = [\theta_1, ..., \theta_m], \quad q(\theta_1, ..., \theta_m) = \prod_{i=1}^{m} q(\theta_j) \]
- Optimal updates have the following form:
  \[ q(\theta_k) \propto \exp \{-\mathbb{E}_{q_k} [\log p(x, \theta)]\} \]
Dirichlet Process (Stick Breaking)

For each cluster $k = 1, 2, 3, \ldots$

- Cluster shape: $\phi_k \sim H(\lambda_0)$
- Stick proportion: $v_k \sim \text{Beta}(1, \alpha)$
- Cluster coefficient: $\pi_k = v_k \prod_{l=1}^{k} (1 - v_l)$

$\pi \sim \text{Stick}(\alpha)$

$\pi_1 = v_1$

$\pi_2 = v_2 (1 - v_1)$

$\pi_3 = v_3 (1 - v_2) (1 - v_1)$

$1 - \sum_{k=1}^{K} \pi_k = \prod_{k=1}^{K} (1 - v_k)$

Stick-breaking (Sethuraman, 1994)
Dirichlet Process Mixture model

For each cluster $k = 1, 2, 3, ...$
- Cluster shape: $\phi_k \sim H(\lambda_0)$
- Stick proportion: $v_k \sim \text{Beta}(1, \alpha)$
- Cluster coefficient: $\pi_k = v_k \prod_{l=1}^{k}(1 - v_l)$

For each data point $n = 1, 2, 3, ...$
- Cluster assignment: $z_n \sim \text{Cat}(\pi)$
- Observation: $x_n \sim \phi_{z_n}$

Posterior variables: $\Theta = \{z_n, v_k, \phi_k\}$
Approximation: $q(z_n, v_k, \phi_k)$
Dirichlet Process Mixture model

For each data point \( n \) and clusters \( k \)
- \( q(z_n = k) = r_{nk} \propto \exp\{\mathbb{E}_q[\log \pi_k(v) + \log p(x_n | \phi_k)]\} \)

For cluster \( k = 1, 2, 3, \ldots, K \)
- \( N^0_k \leftarrow \sum_n r_{nk} \)
- \( s^0_k \leftarrow \sum_{n=1}^N r_{nk} t(x_n) \)
- \( \lambda_k \leftarrow \lambda_0 + s^0_k \)

For cluster \( k = 1, 2, 3, \ldots, K \)
- \( \alpha^0_k \leftarrow 1 + N^0_k \)
- \( \alpha^0_k \leftarrow \alpha + \sum_{l>k} N^0_l \)
Stochastic Variational Bayes

Hoffman et al., JMLR, 2013

Stochastically divide data into $B$ batches:

$$\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_B$$

- For each batch: $b = 1, 2, 3, \ldots, B$
  - $r \leftarrow EStep(\mathcal{B}_b, \alpha, \lambda)$
  - For each cluster $k = 1, 2, 3, \ldots, K$
    - $s_k^b \leftarrow \sum_{n \in \mathcal{B}_b} r_{nk} \ t(x_n)$
    - $\lambda_k^b \leftarrow \lambda_0 + \frac{N}{|\mathcal{B}_b|} s_k^b$
    - $\lambda_k \leftarrow \rho_t \lambda_k^b + (1 - \rho_t) \lambda_k$
    - Similarly for stick weights

Convergence condition on $\rho_t$

$$\sum_t \rho_t \rightarrow \infty, \quad \sum_t \rho_t^2 < \infty$$
Memoized Variational Bayes

Hughes & Sudderth, NIPS 2013

Stochastically divide data into $B$ batches:

$\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_B$

- For each batch: $b = 1, 2, 3, \ldots, B$
  - $r \leftarrow EStep(\mathcal{B}_b, \alpha, \lambda)$
  - For data item $k = 1, 2, 3, \ldots, K$
    - $s^b_k \leftarrow s^0_k - s^b_k$
    - $s^b_k \leftarrow \sum_{n \in \mathcal{B}_b} r_{nk} t(x_n)$
    - $s^0_k \leftarrow s^0_k + s^b_k$
    - $\lambda_k \leftarrow \lambda_0 + s^0_k$

Global variables:

$s^0_1 \ s^0_2 \ \ldots \ s^0_K$

$s^0_k = \sum_b s^b_k$

Local variables:

$s^1_1 \ s^1_2 \ \ldots \ s^1_K$
$s^2_1 \ s^2_2 \ \ldots \ s^2_K$

\vdots \ \vdots \ \vdots
$s^B_1 \ s^B_2 \ \ldots \ s^B_K$
Birth moves

- Conventional variational approximation:
  - Truncation on the number of components
- Need to have an adaptive way to add new components
- Past: split-merge schema for single-batch learning
  - E.g. EM (Ueda et al., 2000), Variational-HDP (Bryant and Sudderth, 2012), etc.
  - Split a new component
  - Fix everything
  - Run restricted updates.
  - Decide whether to keep it or not
- This strategy unlikely to work in the batch mode:
  - Each batch might not contain enough examples of the missing component
Birth moves

- The strategy in this work:
  - **Collection**: subsample data in the targeted component $k'$
  - **Creation**: run a DPM on the subsampled data ($K' = 10$)
  - **Adoption**: Update parameters with $K' + K$
Merge clusters

New cluster $k_m$ takes over all responsibility of old clusters $k_a$ and $k_b$: 

$$r_{nk_m} \leftarrow r_{nk_a} + r_{nk_b}$$

$$N_{km}^0 \leftarrow N_{ka}^0 + N_{kb}^0$$

$$s_{km}^0 \leftarrow s_{ka}^0 + s_{kb}^0$$

Accept or reject:

$$\mathcal{L}(q_{merge}) > \mathcal{L}(q)?$$

How to choose pair?

Randomly sample proportional to the relative marginal likelihood:

$$\frac{M(S_{ka} + S_{kb})}{M(S_{ka}) + M(S_{kb})}$$
Results: Clustering Handwritten digits

- Clustering $N = 60000$ MNIST images of handwritten digits 0-9.
- As preprocessing, all images projected to $D = 50$ via PCA.

References