Gaussian Mixture Model and Applications with Expectation-Maximization

KK Thuwajit and Khalid Al-Raisi

May 5, 2025

Contents

0 Prerequisites **3**
- 0.1 Motivation for KL divergence in Variational Autoencoders . . . 4
- 0.2 Asymmetry of KL Divergence . . . 5
- 0.3 Forward vs Reverse KL divergence . . . 5

1 Motivation: Gaussian Mixture Model **7**
- 1.1 Defining Gaussian Mixture . . . 7
- 1.2 Potential Applications of Gaussian Mixture . . . 7
- 1.3 Gaussian Mixture as MLE Problem . . . 7

2 Expectation-Maximization Algorithms **8**
- 2.1 The Evidence Lower Bound (ELBO) . . . 8
- 2.2 Defining EM Algorithms . . . 9
- 2.3 EM Algorithm for Gaussian Mixture Models . . . 9
- 2.4 Extending the EM Algorithm . . . 10

3 Applications of EM: Unsupervised Classification **12**
- 3.1 K-means Clustering . . . 12
- 3.2 Gaussian Mixture Classification . . . 13
- 3.3 Comparison . . . 13

4 Applications of EM: Conditional Generation **15**
- 4.1 Pure Gaussian Mixture Models . . . 15
- 4.2 Conditional Variational Autoencoders . . . 15
  - 4.2.1 The M2 Model . . . 15
  - 4.2.2 The GMVAE Model . . . 16
- 4.3 Comparison . . . 16

Preface

This report aims to establish the mathematical derivation of the Gaussian Mixture Model and the Expectation-Maximization (EM) algorithm, two closely-related topic that we have not taught in class, assuming the reader has prerequisites equivalent to CS541. We first referenced the derivation of the EM algorithm from Andrew Ng's lecture notes of CS229 from Stanford University [5] into our language and notation. Then, we proved that several tasks, like clustering and generation, can be solved with the EM algorithm. We finally present related research in generalizing the VAE, which is in and of itself, a generalization of the EM algorithm.

Overall, the report aims to address the first project type, a lecture on a topic we did not (and will not) cover, while being a reproducibility challenge of the cited papers. The implementations can be found in my github repository (`kkuroma/gaussian-mixture`).

0. Prerequisites

We aim to establish the mathematical derivations that will be used throughout the document here.

Theorem 1 (MLE of Categorical Sampling)**.** *Given $y_1, y_2, \ldots, y_n$ are sampled IID from $\{1, 2, \ldots, k\}$ such that $\mathbb{P}(y_i = j) = \phi_j$, $\sum_{j=1}^{k} = \phi_j$, the MLE for ϕ_i is given by*

$$\hat{\phi}_j = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}\{y_i = j\}$$

Proof. The likelihood of the parameters $\phi_1, \phi_2, \ldots, \phi_k$ given observations $y_1, y_2, \ldots, y_n$ can be expressed as

$$f(\phi_1, \phi_2, \ldots, \phi_k) = \prod_{i=1}^{n} \phi_{y_i} = \prod_{j=1}^{k} \phi_j^{\frac{1}{n} \sum_{i=1}^{n} \mathbb{I}\{y_i = j\}}$$

and consequently the log-likelihood

$$g(\phi_1, \phi_2, \ldots, \phi_k) = \frac{1}{n} \sum_{j=1}^{k} \log(\phi_j) \left(\sum_{i=1}^{n} \mathbb{I}\{y_i = j\} \right)$$

we are able to solve for the MLE using Lagrangian multipliers. Consider the objective function

$$\mathcal{L}(\phi_1, \phi_2, \ldots, \phi_k, \lambda) = \sum_{j=1}^{k} \log(\phi_j) \left(\sum_{i=1}^{n} \mathbb{I}\{y_i = j\} \right) + \lambda \left(1 - \sum_{j=1}^{k} \phi_j \right)$$

Taking the partial derivative for the gradient

$$\frac{\partial}{\partial \phi_l} \mathcal{L}(\phi_1, \phi_2, \ldots, \phi_k, \lambda) = \frac{\sum_{i=1}^{n} \mathbb{I}\{y_i = l\}}{\phi_l} - \lambda$$

and the second partial derivative for the hessian

$$\frac{\partial^2}{\partial \phi_l^2} \mathcal{L}(\phi_1, \phi_2, \ldots, \phi_k, \lambda) = -\frac{\sum_{i=1}^{n} \mathbb{I}\{y_i = l\}}{\phi_l^2} \text{ and } \frac{\partial}{\partial \phi_l} \frac{\partial}{\partial \phi_m} \mathcal{L}(\phi_1, \phi_2, \ldots, \phi_k, \lambda) = 0$$

We can see that the hessian is negative semi-definite, meaning setting the gradient as $\mathbf{0}$ or equivalently $\phi_l = \frac{\sum_{i=1}^{n} \mathbb{I}\{y_i = l\}}{\lambda}$ yields the maximum. Since

$$1 = \sum_{j=1}^{k} \phi_j = \sum_{j=1}^{k} \frac{\sum_{i=1}^{n} \mathbb{I}\{y_i = j\}}{\lambda} = \frac{n}{\lambda}$$

therefore, $\lambda = n$. Put together, the empirical prediction $\hat{\phi}_j = \frac{\sum_{i=1}^{n} \mathbb{I}\{y_i = j\}}{n}$ is the MLE, completing the proof □

Theorem 2 (MLE of Multivariate Gaussian)**.** *Given $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ are sampled IID from $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$, the MLE for $\boldsymbol{\mu}$ and Σ are given by*

$$\boldsymbol{\mu} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i \text{ and } \Sigma = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top$$

Proof. The likelihood of the parameters $\boldsymbol{\mu}$ and Σ given observations $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ can be expressed as

$$f(\boldsymbol{\mu}, \Sigma) = \prod_{i=1}^{n} \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2} (\mathbf{x}_i - \boldsymbol{\mu})^\top \Sigma^{-1} (\mathbf{x}_i - \boldsymbol{\mu}) \right)$$

and consequently the log-likelihood

$$g(\boldsymbol{\mu}, \Sigma) = -\frac{n}{2} \log |\Sigma| - \frac{nd}{2} \log(2\pi) - \frac{1}{2} \sum_{i=1}^{n} (\mathbf{x}_i - \boldsymbol{\mu})^\top \Sigma^{-1} (\mathbf{x}_i - \boldsymbol{\mu})$$

To find the MLE, we maximize $g(\boldsymbol{\mu}, \Sigma)$ with respect to $\boldsymbol{\mu}$ and Σ. First, optimizing with respect to $\boldsymbol{\mu}$, we take the gradient and hessian

$$\frac{\partial}{\partial \boldsymbol{\mu}} g(\boldsymbol{\mu}, \Sigma) = \Sigma^{-1} \sum_{i=1}^{n} (\mathbf{x}_i - \boldsymbol{\mu}) \text{ and } \frac{\partial^2}{\partial \boldsymbol{\mu}^2} g(\boldsymbol{\mu}, \Sigma) = -\Sigma^{-1}$$

Since Σ has to be positive semi-definite to be a covariance, so does Σ^{-1}, which implies the gradient to zero or

$$\Sigma^{-1} \sum_{i=1}^{n} (\mathbf{x}_i - \boldsymbol{\mu}) = 0 \implies \sum_{i=1}^{n} (\mathbf{x}_i - \boldsymbol{\mu}) = 0 \implies n\boldsymbol{\mu} = \sum_{i=1}^{n} \mathbf{x}_i \implies \boldsymbol{\mu} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i$$

yields the maximum. Next, optimizing with respect to Σ, plugging $\hat{\boldsymbol{\mu}}$ back in, the log-likelihood simplifies to

$$g(\boldsymbol{\mu}, \Sigma) = -\frac{n}{2} \log |\Sigma| - \frac{1}{2} \sum_{i=1}^{n} (\mathbf{x}_i - \hat{\boldsymbol{\mu}})^\top \Sigma^{-1} (\mathbf{x}_i - \hat{\boldsymbol{\mu}})$$

Here, we use the trace trick:

$$-\frac{n}{2} \log |\Sigma| - \frac{1}{2} \sum_{i=1}^{n} \operatorname{Tr}\left((\mathbf{x}_i - \hat{\boldsymbol{\mu}})^\top \Sigma^{-1} (\mathbf{x}_i - \hat{\boldsymbol{\mu}}) \right)$$

Taking the derivative with respect to Σ^{-1} using the these identities

$$\frac{\partial}{\partial \Sigma^{-1}} \log |\Sigma| = \Sigma \quad \text{and} \quad \frac{\partial}{\partial \Sigma^{-1}} \operatorname{tr}(A\Sigma^{-1}) = -A$$

we obtain

$$\frac{\partial}{\partial \Sigma^{-1}} g(\boldsymbol{\mu}, \Sigma) = \frac{n}{2} \Sigma - \frac{1}{2} \sum_{i=1}^{n} (\mathbf{x}_i - \hat{\boldsymbol{\mu}})(\mathbf{x}_i - \hat{\boldsymbol{\mu}})^\top$$

Setting the gradient to zero, we get

$$n\Sigma = \sum_{i=1}^{n} (\mathbf{x}_i - \hat{\boldsymbol{\mu}})(\mathbf{x}_i - \hat{\boldsymbol{\mu}})^\top \implies \Sigma = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i - \hat{\boldsymbol{\mu}})(\mathbf{x}_i - \hat{\boldsymbol{\mu}})^\top$$

Put together, the empirical predictions are indeed the MLE, thus completing the proof. □

0.1 Motivation for KL divergence in Variational Autoencoders

The Variational Autoencoder (VAE), explained in more details later in this report, involves two loss functions, the reconstruction loss and the Kullback-Leibler (KL) divergence. The reconstruction loss measures how accurate is the generated data compared to the true data. We also require a measure of the statistical distance between the true distribution and the approximating distribution, to ensure a better more accurate latent space. For that, we use KL divergence. KL divergence is preferred over other statistical distance measures in this case, as KL divergence helps in making optimization simpler by allowing us to eliminate an intractable term.

Definition 1 (Kullback-Leibler Divergence)**.** Kullback-Leibler (KL) divergence is a measure of how much different is the approximating probability distribution Q from the true distribution P. Mathematically, we write: For discrete random variables; it is a measure of how much information is lost when a distribution is used to approximate another distribution.

$$D_{KL}(P||Q) = \sum_{x \in X} P(x) \log \frac{P(x)}{Q(x)} \tag{1}$$

where P and Q are two probability distributions over the same variable x.
For continuous random variables,

$$D_{KL}(P||Q) = \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx \tag{2}$$

where $p(x)$ and $q(x)$ are the probability density functions of the distributions P and Q, respectively.

0.2 Asymmetry of KL Divergence

- KL divergence is not symmetric. Mathematically, we have,

$$D_{KL}(P||Q) \neq D_{KL}(Q||P) \tag{3}$$

- Within the context of VAE, the KL divergence terms in both EM and ELBO, we encounter an intractable posterior, as a way around that, we use the reverse KL divergence, despite this asymmetry.

- We write the reverse KL divergence as:

$$D_{KL}(Q||P) = \sum_{x \in X} Q(x) \log \frac{Q(x)}{P(x)} \tag{4}$$

0.3 Forward vs Reverse KL divergence

In general, the forward KL divergence is mode-averaging, while the reverse KL divergence is mode-fitting. Meaning the latter is less sensitive to tails of our distributions, and instead focuses on the mode of our data, so we see a greater likelihood of generating samples similar to the mode instead of a larger number of samples generated that are less likely in our true distribution. This makes our model more robust to outliers, and potentially more accurate. Also justifying the use of reverse KL divergence in our approach despite the asymmetry of KL divergence. Further, this assymetry is also useful since if we consider $D_{KL}(P||Q)$ and $D_{KL}(Q||P)$, since the true posterior is generally more complex than the approximating distribution, so by encoding the approximating distribution using the true posterior we can achieve a lower KL divergence, and thus better optimizating performance. As we will develop next, we would like to minimize the KL divergence when optimizing.

Theorem 3 (KL Divergence of Multivariate Gaussians)**.** *Given two multivariate Gaussian distributions $p = \mathcal{N}(\boldsymbol{\mu}_p, \Sigma_p)$ and $q = \mathcal{N}(\boldsymbol{\mu}_q, \Sigma_q)$, the (reverse) KL divergence between them is*

$$\mathrm{KL}(p\|q) = \frac{1}{2}\left(\log\frac{|\Sigma_q|}{|\Sigma_p|} - d + \mathrm{Tr}(\Sigma_q^{-1}\Sigma_p) + (\boldsymbol{\mu}_q - \boldsymbol{\mu}_p)^\top \Sigma_q^{-1}(\boldsymbol{\mu}_q - \boldsymbol{\mu}_p)\right)$$

Proof. By definition, the KL divergence is

$$\mathrm{KL}(p\|q) = \mathbb{E}_{\mathbf{x}\sim p}\left[\log p(\mathbf{x}) - \log q(\mathbf{x})\right]$$

Using the density of a multivariate Gaussian, we have

$$\log p(\mathbf{x}) = -\frac{d}{2}\log(2\pi) - \frac{1}{2}\log|\Sigma_p| - \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_p)^\top\Sigma_p^{-1}(\mathbf{x}-\boldsymbol{\mu}_p)$$

and

$$\log q(\mathbf{x}) = -\frac{d}{2}\log(2\pi) - \frac{1}{2}\log|\Sigma_q| - \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_q)^\top\Sigma_q^{-1}(\mathbf{x}-\boldsymbol{\mu}_q)$$

Substituting into the KL formula, the constant $-\frac{d}{2}\log(2\pi)$ cancels out, giving

$$\mathrm{KL}(p\|q) = \mathbb{E}_{\mathbf{x}\sim p}\left[-\frac{1}{2}\log|\Sigma_p| + \frac{1}{2}\log|\Sigma_q| - \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_p)^\top\Sigma_p^{-1}(\mathbf{x}-\boldsymbol{\mu}_p) + \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_q)^\top\Sigma_q^{-1}(\mathbf{x}-\boldsymbol{\mu}_q)\right]$$

Grouping constants:

$$\mathrm{KL}(p\|q) = \frac{1}{2}\left(\log\frac{|\Sigma_q|}{|\Sigma_p|} + \mathbb{E}_{\mathbf{x}\sim p}\left[(\mathbf{x}-\boldsymbol{\mu}_q)^\top\Sigma_q^{-1}(\mathbf{x}-\boldsymbol{\mu}_q) - (\mathbf{x}-\boldsymbol{\mu}_p)^\top\Sigma_p^{-1}(\mathbf{x}-\boldsymbol{\mu}_p)\right]\right)$$

Now, expand each quadratic form separately. First,

$$(\mathbf{x}-\boldsymbol{\mu}_q)^\top\Sigma_q^{-1}(\mathbf{x}-\boldsymbol{\mu}_q) = (\mathbf{x}-\boldsymbol{\mu}_p+\boldsymbol{\mu}_p-\boldsymbol{\mu}_q)^\top\Sigma_q^{-1}(\mathbf{x}-\boldsymbol{\mu}_p+\boldsymbol{\mu}_p-\boldsymbol{\mu}_q)$$

Expanding the square,

$$= (\mathbf{x}-\boldsymbol{\mu}_p)^\top\Sigma_q^{-1}(\mathbf{x}-\boldsymbol{\mu}_p) + 2(\mathbf{x}-\boldsymbol{\mu}_p)^\top\Sigma_q^{-1}(\boldsymbol{\mu}_p-\boldsymbol{\mu}_q) + (\boldsymbol{\mu}_p-\boldsymbol{\mu}_q)^\top\Sigma_q^{-1}(\boldsymbol{\mu}_p-\boldsymbol{\mu}_q)$$

Taking expectation under $\mathbf{x} \sim p$, and using that $\mathbb{E}[\mathbf{x}] = \boldsymbol{\mu}_p$ and $\mathbb{E}[(\mathbf{x} - \boldsymbol{\mu}_p)(\mathbf{x} - \boldsymbol{\mu}_p)^\top] = \Sigma_p$, we get

$$\mathbb{E}\left[(\mathbf{x} - \boldsymbol{\mu}_p)^\top \Sigma_q^{-1} (\mathbf{x} - \boldsymbol{\mu}_p)\right] = \mathrm{Tr}(\Sigma_q^{-1}\Sigma_p)$$

and

$$\mathbb{E}\left[2(\mathbf{x} - \boldsymbol{\mu}_p)^\top \Sigma_q^{-1} (\boldsymbol{\mu}_p - \boldsymbol{\mu}_q)\right] = 0$$

because the expectation of $\mathbf{x} - \boldsymbol{\mu}_p$ is zero. The third term is constant and remains

$$(\boldsymbol{\mu}_p - \boldsymbol{\mu}_q)^\top \Sigma_q^{-1} (\boldsymbol{\mu}_p - \boldsymbol{\mu}_q)$$

Similarly, for the second expectation

$$\mathbb{E}\left[(\mathbf{x} - \boldsymbol{\mu}_p)^\top \Sigma_p^{-1} (\mathbf{x} - \boldsymbol{\mu}_p)\right] = \mathrm{Tr}(\Sigma_p^{-1}\Sigma_p) = d$$

Putting everything together:

$$\mathrm{KL}(p\|q) = \frac{1}{2}\left(\log\frac{|\Sigma_q|}{|\Sigma_p|} - d + \mathrm{Tr}(\Sigma_q^{-1}\Sigma_p) + (\boldsymbol{\mu}_q - \boldsymbol{\mu}_p)^\top \Sigma_q^{-1}(\boldsymbol{\mu}_q - \boldsymbol{\mu}_p)\right)$$

thus completing the proof. □

1. Motivation: Gaussian Mixture Model

1.1 Defining Gaussian Mixture

Definition 2 (Gaussian Mixture Problem)**.** For $j = 1, 2, \ldots, k$ suppose there exists some fixed, hidden parameters $\phi_j \in \mathbb{R}, \boldsymbol{\mu}_j \in \mathbb{R}^d, \Sigma_j \in \mathbb{R}^{d\times d}$ such that $\sum_{j=1}^{k} \phi_j = 1$ and Σ_j is a valid covariance matrix, we consider the following procedure

- For $i = 1, 2, \ldots, n$ sample IID z_i from $\{1, 2, \ldots, k\}$ such that $\mathbb{P}(z_i) = \phi_j$
- For $i = 1, 2, \ldots, n$ sample IID $\mathbf{x}_i$ from $\mathcal{N}(\boldsymbol{\mu}_{z_i}, \Sigma_{z_i})$

There are two versions of the problems:

- **Supervised**: for $i = 1, 2, \ldots, n$ we are given $\mathbf{x}_i, z_i$. Find the estimate for $\phi_j, \boldsymbol{\mu}_j, \Sigma_j$
- **Unsupervised**: for $i = 1, 2, \ldots, n$ we are given only $\mathbf{x}_i$. Find the estimate for $\phi_j, \boldsymbol{\mu}_j, \Sigma_j$

This is assuming that there exists k different Gaussian clusters, and $\mathbf{x}_i$ is sampled from one of them with probability specified by ϕ_j. Notice that the supervised version of the problem essentially boils down to running the regular Multivariate Gaussian problem for each class z_i can take. We are more interested in the unsupervised problem, as the MLE could not be solved in a closed form.

1.2 Potential Applications of Gaussian Mixture

We present two applications of the unsupervised version of the Gaussian Mixture defined above. The exact implementations will be explained in later sessions.

- **Unsupervised classification**: the resulting estimate for the Gaussian clusters' mean and covariance allows estimates for the unknown label z_i to be retrieved from $\mathbf{x}_i$.
- **Conditional generation**: the Gaussian Mixture model allows data outside of the given set to be sampled from the resulting Gaussian distributions, conditioned on one of the possible classes.

1.3 Gaussian Mixture as MLE Problem

Given access to only the data vectors $\mathbf{x}_i$ for $i = 1, 2, \ldots, n$, one may model the likelihood of the parameters $\{\phi_j, \boldsymbol{\mu}_j, \Sigma_j\}_{j=1}^{k}$ as follows:

$$\begin{aligned} f(\{\phi_j, \boldsymbol{\mu}_j, \Sigma_j\}_{j=1}^{k}) &= \prod_{i=1}^{n} \sum_{z=1}^{k} \mathbb{P}(Z = k) p(\mathbf{x}_n; \boldsymbol{\mu}_k, \Sigma_k) \\ &= \prod_{i=1}^{n} \sum_{z=1}^{k} \frac{\phi_z}{(2\pi)^{d/2}|\Sigma_z|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}_i - \boldsymbol{\mu}_z)^\top \Sigma_z^{-1}(\mathbf{x}_i - \boldsymbol{\mu}_z)\right) \end{aligned}$$

This results in a double summation once a logarithm is taken, particularly with the term inside logarithm being a summation. While a closed form MLE solution doesn't exist for this expression, the use of the EM algorithm explained over the next sections will help us iteratively solve for a good approximate.

2. Expectation-Maximization Algorithms

This section aims to define a class of algorithms known as the Expectation-Maximization (EM) algorithm, based off Andrew Ng's lecture notes of CS229 from Stanford University [5]. These algorithms are designed to provide an approximate solution to MLE problems where, because of an unknown set of "latent" random variables, don't lead to a closed form solution.

Definition 3 (Distribution Learning with Hidden Latent Variables)**.** Suppose there exists

- A distribution $\mathcal{P}$ from $\mathbb{R}^d$ with a joint pdf p and an unknown parameter θ_*
- A distribution $\mathcal{Q}_*$ from $\mathbb{R}^s$ with a pdf q_*.

Note that $\mathcal{P}$ and $\mathcal{Q}_*$ can either be explicitly given or not. Hidden behind the scenes, we sample IID:

- For $i = 1, 2, \ldots, n$, $\mathbf{z}_i \sim \mathcal{Q}_*$ (that is, following the pdf of $q_*(\mathbf{z}_i)$)
- For $i = 1, 2, \ldots, n$, $\mathbf{x}_i|\mathbf{z}_i \sim \mathcal{P}_{\theta_*}$ (that is, following the pdf of $p(\mathbf{x}_i|\mathbf{z}_i; \theta_*)$)

Given access to only $\mathbf{x}_i$ for $i = 1, 2, \ldots, n$, find an estimate for θ_* and q_* (the latter if not explicitly given).

2.1 The Evidence Lower Bound (ELBO)

Like the expression for Gaussian Mixture Models, we may model the likelihood of θ and q as follows:

$$\begin{aligned}
f(\theta, q) &= \prod_{i=1}^{n} \int p(\mathbf{x}_i, \mathbf{z}; \theta) d\mathbf{z} \\
&= \prod_{i=1}^{n} \int q(\mathbf{z}) p(\mathbf{x}_i|\mathbf{z}; \theta) d\mathbf{z} \\
&= \prod_{i=1}^{n} \mathbb{E}_{\mathbf{z}} \left[p(\mathbf{x}_i|\mathbf{z}; \theta) \right] \\
g(\theta, q) &= \log(f(\theta, q)) \\
&= \sum_{i=1}^{n} \log \left(\mathbb{E}_{\mathbf{z}} \left[p(\mathbf{x}_i|\mathbf{z}; \theta) \right] \right) \quad (*) \\
&\geq \sum_{i=1}^{n} \mathbb{E}_{\mathbf{z}} \left[\log \left(p(\mathbf{x}_i|\mathbf{z}; \theta) \right) \right] \quad (**) \\
&= \sum_{i=1}^{n} \int q(\mathbf{z}) \log(p(\mathbf{x}_i|\mathbf{z}; \theta)) d\mathbf{z}
\end{aligned}$$

We're able to go from $(*)$ to $(**)$ using Jensen's inequality, as log is a convex function. This allows us to minimize the lower bound of $g(\theta, q)$ which indirectly optimizes the log-likelihood itself. This lower bound term is therefore dubbed the **evidence lower bound** (ELBO):

$$\text{ELBO}(\mathbf{x}; \theta, q) := \int q(\mathbf{z}) \log(p(\mathbf{x}|\mathbf{z}; \theta)) d\mathbf{z}$$

Finding the optimal prior: For the tightest bound with Jensen inequality to hold, the expectation must be taken over a constant, meaning $p(\mathbf{x}|\mathbf{z}; \theta) = c$ over some constant c. By Bayes' law,

$$\begin{aligned}
p(\mathbf{x}|\mathbf{z}; \theta) &= c \\
\frac{p(\mathbf{x}, \mathbf{z}; \theta)}{q(\mathbf{z})} &= c \\
q(\mathbf{z}) &\propto p(\mathbf{x}, \mathbf{z}; \theta)
\end{aligned}$$

And since $\int_{\mathbf{z}} q(\mathbf{z})d\mathbf{z} = 1$,

$$\begin{aligned} q(\mathbf{z}) &= \frac{q(\mathbf{z})}{\int_{\mathbf{z}} q(\mathbf{z})d\mathbf{z}} \\ &= \frac{p(\mathbf{x}, \mathbf{z}; \theta)}{\int_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}; \theta)} \\ &= \frac{p(\mathbf{x}, \mathbf{z}; \theta)}{p(\mathbf{x}, \theta)} \\ &= p(\mathbf{z}|\mathbf{x}; \theta) \end{aligned}$$

Therefore, $p(\mathbf{z}|\mathbf{x}; \theta)$ is the optimal choice for $q(\mathbf{z})$. Notice that this choice is dependent on $\mathbf{x}$. As such, for each $\mathbf{x}_i$, a choice q_i is made as $p(\mathbf{z}|\mathbf{x}_i; \theta)$.

Definition 4 (EM Learning Objective)**.** We may isolate θ from the expression using the optimal q_i as follows:

$$\ell(\theta) = \sum_{i=1}^{n} g(\theta, q_i) \geq \sum_{i=1}^{n} \text{ELBO}(\mathbf{x}_i; \theta, q_i) = \sum_{i=1}^{n} \int q_i(\mathbf{z}) \log(p(\mathbf{x}|\mathbf{z}; \theta))d\mathbf{z} \quad \text{when} \quad q_i(\mathbf{z}) = p(\mathbf{z}|\mathbf{x}_i; \theta)$$

2.2 Defining EM Algorithms

The EM algorithm aims to minimize the learning objective defined above, which, if we recall, is the lower bound of the log-likelihood of a parameter θ, given q_i are chosen optimally.

Definition 5 (EM Algorithm)**.** The algorithm takes the scenario defined in the "Distribution Learning with Hidden Latent Variables" problem and carries the following steps

- **Expectation (E) Step:** compute $q_i^{(t)}(\mathbf{z}) = p(\mathbf{z}|\mathbf{x}_i; \theta^{(t)})$ for $i = 1, 2, \ldots, n$
- **Maximization (M) Step:** compute

$$\theta^{(t+1)} = \arg\max_{\theta} \sum_{i=1}^{n} \text{ELBO}(\mathbf{x}_i; \theta, q_i^{(t)}) = \arg\max_{\theta} \sum_{i=1}^{n} \int q_i^{(t)}(\mathbf{z}) \log(p(\mathbf{x}|\mathbf{z}; \theta))d\mathbf{z}$$

With an arbitrary initial value for $\theta^{(0)}$, the algorithm repeats the E and M steps until convergence.

A primary concern with such an algorithm is the convergence, which will promptly be proven

Theorem 4 (EM Algorithm Convergence)**.** *The EM Algorithm defined above updates $\ell(\theta^{(t)})$ monotonically, that is, $\ell(\theta^{(t+1)}) \geq \ell(\theta^{(t)})$.*

Proof. First, recall the the ELBO is defined such that $\ell(\theta) \geq \sum_{i=1}^{n} \text{ELBO}(\mathbf{x}_i; \theta, q_i)$ for any prior q_i. Furthermore, the choice for $q_i^{(t)}$ was made to ensure that Jensen's inequality holds with equality, meaning $\ell(\theta^{(t)}) = \sum_{i=1}^{n} \text{ELBO}(\mathbf{x}_i; \theta^{(t)}, q_i^{(t)})$. Therefore,

$$\begin{aligned} \ell(\theta^{(t+1)}) &\geq \sum_{i=1}^{n} \text{ELBO}(\mathbf{x}_i; \theta^{(t+1)}, q_i^{(t)}) && \text{ELBO's definition} \\ &\geq \sum_{i=1}^{n} \text{ELBO}(\mathbf{x}_i; \theta^{(t)}, q_i^{(t)}) && \theta^{(t+1)}\text{'s definition} \\ &= \ell(\theta^{(t)}) && \text{Jensen's Inequality Equality Case} \end{aligned}$$

As desired. □

2.3 EM Algorithm for Gaussian Mixture Models

We revisit the problem of Gaussian Mixture Models, now armed with the knowledge of the EM algorithm. We first note that the Gaussian Mixture Model problem is a specific case of the "Distribution Learning with Hidden Latent Variables" problem defined earlier. In particular, $\theta = \{\boldsymbol{\mu}_j, \Sigma_j\}_{j=1}^{k}$, p is the Gaussian Mixture pdf, and q is the proportion of each possible class of z showing up. The EM Algorithm becomes

- **Expectation (E) Step:** $q_i^{(t)}(z) = p(z|\mathbf{x}_i;\theta^{(t)})$ for $i = 1, 2, \ldots, n$
- **Maximization (M) Step:** We first use Bayes's rule to re-write the terms

$$\begin{aligned}
\theta^{(t+1)} &= \arg\max_\theta \sum_{i=1}^{n}\sum_{z=1}^{k} q_i^{(t)}(z)\log(p(\mathbf{x}|z;\boldsymbol{\mu}_z,\Sigma_z,\phi_z)) \\
&= \arg\max_\theta \sum_{i=1}^{n}\sum_{z=1}^{k} q_i^{(t)}(z)\log\left(\frac{p(\mathbf{x},z;\boldsymbol{\mu}_z,\Sigma_z,\phi_z)}{q_i^{(t)}(z)}\right) \\
&= \arg\max_\theta \sum_{i=1}^{n}\sum_{z=1}^{k} q_i^{(t)}(z)\log\left(\frac{p(\mathbf{x}|z;\boldsymbol{\mu}_z,\Sigma_z)p(z,\phi_z)}{q_i^{(t)}(z)}\right) \\
&= \arg\max_\theta \sum_{i=1}^{n}\sum_{z=1}^{k} q_i^{(t)}(z)\log\left(\frac{p(\mathbf{x}|z;\boldsymbol{\mu}_z,\Sigma_z)\phi_z}{q_i^{(t)}(z)}\right)
\end{aligned}$$

  therefore, taking the log and removing constant terms

$$\theta^{(t+1)} = \arg\max_\theta \sum_{i=1}^{n}\sum_{z=1}^{k} q_i^{(t)}(z)\left(-\frac{n}{2}\log|\Sigma_z| - \frac{1}{2}\sum_{i=1}^{n}(\mathbf{x}_i-\boldsymbol{\mu}_z)^\top\Sigma_z^{-1}(\mathbf{x}_i-\boldsymbol{\mu}_z) + \log(\phi_z)\right)$$

Notice that the optimization terms for $\boldsymbol{\mu}_z$ and Σ_z is a linear combination of several log-likelihood terms of regular Gaussian fitting. Indeed, we are able to use the results from the prerequisites section to conclude that for $z = 1, 2, \ldots, k$

$$\boldsymbol{\mu}_z^{(t)} = \frac{\sum_{i=1}^n q_i^{(t)}(z)\mathbf{x}_i}{\sum_{i=1}^n q_i^{(t)}(z)} \text{ and } \Sigma_z^{(t)} = \frac{\sum_{i=1}^n q_i^{(t)}(z)(\mathbf{x}_i-\boldsymbol{\mu}_z)(\mathbf{x}_i-\boldsymbol{\mu}_z)^\top}{\sum_{i=1}^n q_i^{(t)}(z)}$$

Furthermore, the optimization terms ϕ_z are a linear combination of several log-likelihood terms of the categorical sampling. Similarly, we can use the results the prerequisites section to conclude that for $z = 1, 2, \ldots, k$

$$\phi_z^{(t)} = \frac{1}{n}\sum_{i=1}^{n} q_i^{(t)}(z)$$

Conclusion: The Gaussian Mixture problem can be seen as a special case for the previously defined "Distribution Learning with Hidden Latent Variables" problem. Using the EM algorithm, we're able to derive an iterative procedure that approximates $\boldsymbol{\mu}_z, \Sigma_z, \phi_z$ in a closed form.

2.4 Extending the EM Algorithm

The Variational Autoencoder (VAE) [3] is a popular extension of the EM Algorithm. Particularly, the **E-step** of most EM algorithm involves computing $q_i^{(t)}(\mathbf{z}) = p(\mathbf{z}|\mathbf{x}_i;\theta^{(t)})$ for $i = 1, 2, \ldots, n$, which is often an **intractable problem** for most distributions. The VAE replaces the posterior $p(\mathbf{z}|\mathbf{x}_i;\theta^{(t)})$ by a learnable distribution q parametrized by ϕ. Commonly, the two distributions $p(\mathbf{x}|\mathbf{z};\theta)$ (**Decoder**) and $q(\mathbf{z}|\mathbf{x};\phi)$ (**Encoder**) are learned through a neural network with parameters θ and ϕ respectively. Revisiting the ELBO, now reparametrized with θ and ϕ as q is parametrized on the latter,

$$\begin{aligned}
\text{ELBO}(\mathbf{x},\theta,\phi) &= \int q(\mathbf{z}|\mathbf{x};\phi)\log\left(\frac{p(\mathbf{x},\mathbf{z};\theta)}{q(\mathbf{z}|\mathbf{x};\phi)}\right)d\mathbf{z} \\
&= \int q(\mathbf{z}|\mathbf{x};\phi)\log\left(\frac{p(\mathbf{x}|\mathbf{z};\theta)p(\mathbf{z};\theta)}{q(\mathbf{z}|\mathbf{x};\phi)}\right)d\mathbf{z} \\
&= \int q(\mathbf{z}|\mathbf{x};\phi)\log(p(\mathbf{x}|\mathbf{z};\theta))d\mathbf{z} - \int q(\mathbf{z}|\mathbf{x};\phi)\log\left(\frac{q(\mathbf{z}|\mathbf{x};\phi)}{p(\mathbf{z};\theta)}\right)d\mathbf{z}
\end{aligned}$$

- The first term $\int q(\mathbf{z}|\mathbf{x};\phi)\log(p(\mathbf{x}|\mathbf{z};\theta))d\mathbf{z} = \mathbb{E}_{\mathbf{z}\sim q_\phi}[\log(p(\mathbf{x}|\mathbf{z};\theta))]$ can be derived further under the assumption that $p(\mathbf{x}|\mathbf{z};\theta)$ is Gaussian of some mean $\mu_\theta(\mathbf{z})$ parameterized on θ and a constant diagonal covariance $\sigma^2\mathbf{I}_{d\times d}$

$$\begin{aligned}
\log(p(\mathbf{x}|\mathbf{z};\theta)) &= \log\left(\frac{1}{\sigma^d(2\pi)^{d/2}}\exp\left(-\frac{\|\mathbf{x}-\mu_\theta(\mathbf{z})\|_2^2}{2\sigma^2}\right)\right) \\
&= -\|\mathbf{x}-\mu_\theta(\mathbf{z})\|_2^2 + C
\end{aligned}$$

when the constant terms are grouped to C. As such, optimizing this term is equivalent to minimizing $\mathbb{E}[\|\mathbf{x} - \mu_\theta(\mathbf{z})\|_2^2]$, the euclidean difference between the data point $\mathbf{x}$ and its reconstruction $\mu_\theta(\mathbf{z})$. During training, this is optimized stochastically with monte-carlo samples.

- The second term $\int q(\mathbf{z}|\mathbf{x};\phi) \log\left(\frac{q(\mathbf{z}|\mathbf{x};\phi)}{p(\mathbf{z};\theta)}\right) d\mathbf{z}$ is the KL divergence between the two distributions $q(\mathbf{z}|\mathbf{x};\phi)$ and $p(\mathbf{z};\theta)$. We first assume $p(\mathbf{z};\theta)$ follows a standard $s-$dimensional Gaussian pdf. We then assume that $q(\mathbf{z}|\mathbf{x};\phi)$ follows $\mathcal{N}(\mu_\phi(\mathbf{x}), \text{diag}(\sigma_\phi^2(\mathbf{x})))$ (when $\text{diag}(\mathbf{v})$ takes $\mathbf{v} \in \mathbb{R}^s$ and returns an $s \times s$ diagonal matrix with entries from the basis coefficient of $\mathbf{v}$). Using results from the prerequisites section, this KL divergence can be expressed as follows:

$$\text{KL}(q(\mathbf{z}|\mathbf{x};\phi)\|p(\mathbf{z};\theta)) = \frac{1}{2}\sum_{i=1}^{s}\left(\sigma_{\phi,i}^2(\mathbf{x}) + \mu_{\phi,i}^2(\mathbf{x}) - 1 - \log\sigma_{\phi,i}^2(\mathbf{x})\right)$$

  Remark: this is known as the reparametrization trick; the encoder q produces two vectors $\mu_\phi(\mathbf{x})$ and $\sigma_\phi^2(\mathbf{x})$ to parameterize the Gaussian distribution z is then sampled from. Oftentimes, the log-variance is predicted instead to prevent negative-valued predictions.

Conclusion the VAE provides an alternative to EM algorithm's **E-step**, which involves an often intractable term $p(\mathbf{z}|\mathbf{x}_i;\theta^{(t)})$. Instead, the VAE learns two distributions $p(\mathbf{x}|\mathbf{z};\theta)$ (**Decoder**) and $q(\mathbf{z}|\mathbf{x};\phi)$ (**Encoder**) simultaneously to maximize the ELBO. By several assumptions of $p(\mathbf{x}|\mathbf{z};\theta)$, $q(\mathbf{z}|\mathbf{x};\phi)$, and $p(\mathbf{z};\theta)$ normality, the ELBO simplifies to a sum of the reconstruction loss and KL divergence

$$\hat{\theta}, \hat{\phi} = \arg\min_{\theta,\phi} \mathbb{E}[\|\mathbf{x} - \mu_\theta(\mathbf{z})\|_2^2] + \frac{1}{2}\sum_{i=1}^{s}\left(\sigma_{\phi,i}^2(\mathbf{x}) + \mu_{\phi,i}^2(\mathbf{x}) - 1 - \log\sigma_{\phi,i}^2(\mathbf{x})\right)$$

which could be learned through (stochastic) gradient descent. Unlike the EM algorithm, the VAE is only able to improve the lower bound (ELBO) of the log-likelihood, meaning convergence could not be guaranteed.

3. Applications of EM: Unsupervised Classification

Definition 6 (Unsupervised Classification)**.** Given data vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ and a set of possible classes $\mathcal{C} = \{1, 2, \ldots, k\}$, we attempt to find $z_1, z_2, \ldots, z_n \in \mathcal{C}$ and parameters θ such that $\prod_{i=1}^n p(\mathbf{x}_i, z_i; \theta)$. In other words, the MLE under a known distribution type is maximized.

This definition can seen as a reformulation of the "Distribution Learning with Hidden Latent Variables" where $\mathbf{z}_i$ is limited to a discrete set. By solving for q with EM, we're able to derive the MLE for the classes z_i as follows

$$\begin{aligned}
\hat{z} &= \arg\max_{z \in \mathcal{C}} p(z|\mathbf{x}; \theta) \\
&= \arg\max_{z \in \mathcal{C}} \frac{p(\mathbf{x}|z; \theta)p(z)}{p(\mathbf{x})} && \text{Bayes' Rule} \\
&= \arg\max_{z \in \mathcal{C}} p(\mathbf{x}|z; \theta)p(z) && p(\mathbf{x}) \text{ is constant} \\
&= \arg\max_{z \in \mathcal{C}} p(\mathbf{x}|z; \theta)\hat{q}(z) && \hat{q} \text{ is an estimate of the prior}
\end{aligned}$$

3.1 K-means Clustering

The K-means clustering algorithm [2] finds an estimate to the unsupervised classification problem using the following procedure. First assign centroids $\boldsymbol{\mu}_j^{(0)}$ for $j = \{1, 2, \ldots, k\}$ to random points. Until convergence, run

- $\mathcal{S}_j^{(t)} = \{i \in \{1, 2, \ldots, n\} : j = \arg\max_l \|\boldsymbol{\mu}_l^{(t)} - \mathbf{x}_i\|_2^2\}$
- $\boldsymbol{\mu}_j^{(t+1)} = \frac{1}{|\mathcal{S}_j^{(t)}|} \sum_{i \in \mathcal{S}_j^{(t)}} \mathbf{x}_i$

Finally, assign $\hat{z}_i = \arg\min_j \|\boldsymbol{\mu}_j^{(t)} - \mathbf{x}_i\|_2^2$. We will first argue that the algorithm above can be seen as a special case of the EM algorithm. Consider a simpler case of the Gaussian Mixture problem such that:

- For $j = \{1, 2, \ldots, k\}$, $\phi_j = \frac{1}{k}$ (each cluster has an equal chance of being assigned)
- For $j = \{1, 2, \ldots, k\}$, $\Sigma_j = \sigma^2 \mathbf{I}_{d \times d}$, this value of σ will be clarified later.

Knowing ϕ_j, we are able to simply the E-step as follows:

$$q_i^{(t)}(z) = p(z|\mathbf{x}_i; \theta^{(t)}) = \frac{p(\mathbf{x}_i, z; \theta^{(t)})}{\sum_{j=1}^k p(\mathbf{x}_i, j; \theta^{(t)})} = \frac{\exp(-\|\mathbf{x}_i - \boldsymbol{\mu}_z^{(t)}\|_2^2/2\sigma^2)}{\sum_{j=1}^k \exp(-\|\mathbf{x}_i - \boldsymbol{\mu}_j\|_2^2/2\sigma^2)}$$

the next trick involves taking the limit of $\sigma \to 0$, which implies that each Gaussian cluster converges to an impulse function. This is also called **hard-labeling**. We could simply to the final term as $1/\sigma$ dominates in the term where $\|\mathbf{x}_i - \boldsymbol{\mu}_j^{(t)}\|_2^2$ is smallest.

$$\lim_{\sigma \to 0} q_i^{(t)}(z) = \lim_{\sigma \to 0} \frac{\exp(-\|\mathbf{x}_i - \boldsymbol{\mu}_z^{(t)}\|_2^2/2\sigma^2)}{\sum_{j=1}^k \exp(-\|\mathbf{x}_i - \boldsymbol{\mu}_j^{(t)}\|_2^2/2\sigma^2)} = \begin{cases} 1 & z = \arg\min_j \|\boldsymbol{\mu}_j^{(t)} - \mathbf{x}_i\|_2^2 \\ 0 & \text{otherwise} \end{cases}$$

substituting to the M-step $\boldsymbol{\mu}_z^{(t)} = \frac{\sum_{i=1}^n q_i^{(t)}(z)\mathbf{x}_i}{\sum_{i=1}^n q_i^{(t)}(z)}$: defining $\mathcal{S}_j^{(t)} = \{i \in \{1, 2, \ldots, n\} : j = \arg\max_l \|\boldsymbol{\mu}_l^{(t)} - \mathbf{x}_i\|_2^2\}$ like above,

$$\boldsymbol{\mu}_z^{(t)} = \frac{1}{|\mathcal{S}_j^{(t)}|} \sum_{i \in \mathcal{S}_j^{(t)}} \mathbf{x}_i$$

which agrees with expression presented in the K-means Clustering algorithm. As such, as a direct result of the EM Algorithm Convergence theorem, the K-means Clustering guarantees that the log-likelihood decreases monotonically in time.

3.2 Gaussian Mixture Classification

As expressed in the beginning of this section, we are able to express the MLE for the label as such:

$$\begin{aligned}\hat{z}_i &= \arg\max_z p(\mathbf{x}_i|z;\theta)\hat{q}(z) \\ &= \arg\max_z \frac{\phi_z}{|\Sigma_z|^{1/2}} \exp\left(\frac{\|\mathbf{x}_i - \boldsymbol{\mu}_z\|^\top \Sigma_z^{-1} \|\mathbf{x}_i - \boldsymbol{\mu}_z\|}{2}\right)\end{aligned}$$

3.3 Comparison

The Gaussian Mixture method generalizes K-means Clustering with the following distinction

- **Soft-labeling**: the Gaussian Mixture model considers the probability of each cluster having distinct spread, and that all points have a possibility of being a tail outlier from a faraway cluster.
- **Cluster probability**: as a direct consequently of hard-labeling, the K-means Clustering algorithm is unable to produce a distribution-type posterior probability $p(z|\mathbf{x}_i;\theta^{(t)})$, while the Gaussian Mixture method naturally produces this value.

Still, both model both utilizes the Gaussian Distribution to model the associated cluster and apply the EM Algorithm to solve for an MLE estimate. As such, both models suffer greatly from poorly initialized $\boldsymbol{\mu}_j$ centroids as their convergence only guarantees up to the monotonic decrease of the loss function, leading to local minima situations. To visualize this, we first generate sampled Gaussian Mixture data from $\mathbb{R}^2$ as shown in 1. Implementations can be found in `code/unsupervised-classification.ipynb`.

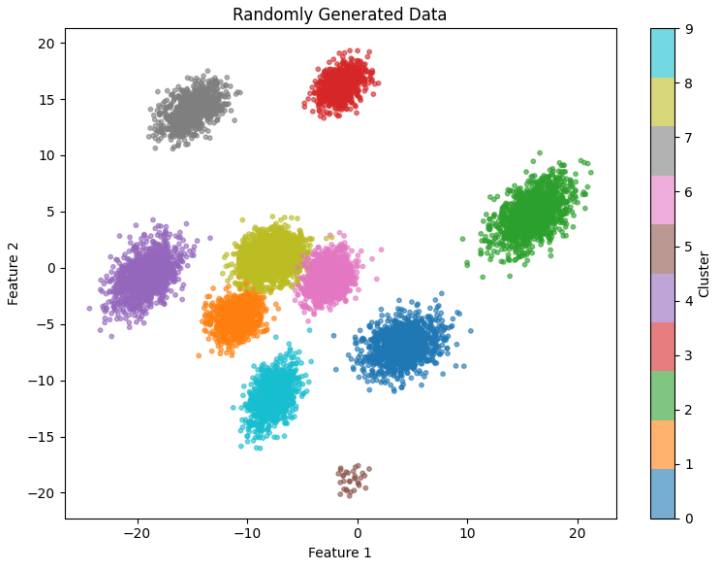

Figure 1: Randomly Generated Cluster Data

We first discuss the results from K-means clustering in figure 2. We notice a poor accuracy in classes 2 and 3 as a result of these two predicted clusters "inhabiting" on of the original clusters. Due to starting points being sampled from the dataset itself, it is possible that the class 2 and 3 centroids are originally picked from this cluster and aren't able to be separated due to local optima. While Gaussian Mixture models (see figure 3) is shown to obtain a higher performance, we notice that Gaussian Mixture fitting also suffers from the same problem (class 9 in the figure). Still, the incorporation of prior terms in Gaussian Mixture models is able to improve its accuracy over K-means (these implementations all use the same seed and starting points).

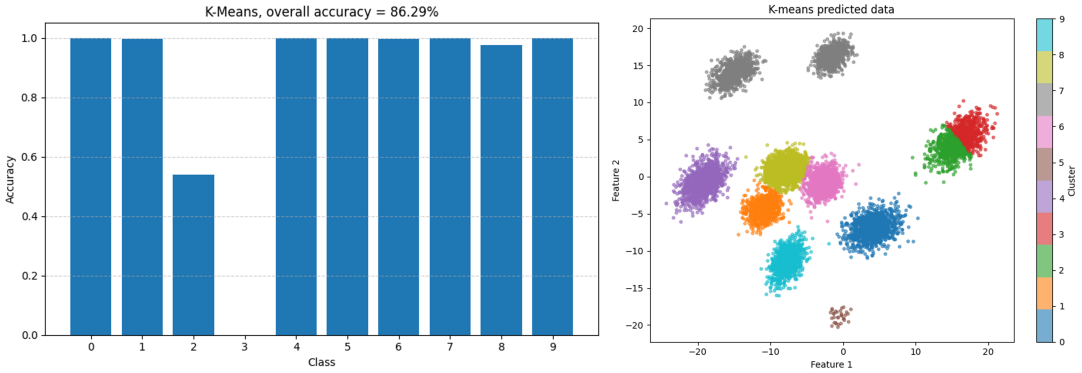

Figure 2: K-Means Clustering Results

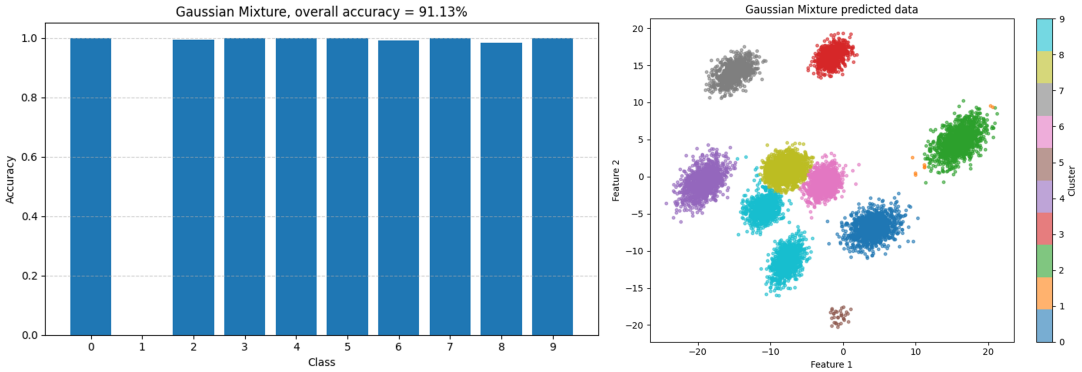

Figure 3: Gaussian Mixture Clustering Results

4. Applications of EM: Conditional Generation

Definition 7 (Generation)**.** Given data vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ sampled IID from a known pdf p with parameters θ, solving for θ allows more IID samples to be generated from $\mathbf{x}$. **Conditional generation** assumes there exists a set of possible classes $\mathcal{C} = \{1, 2, \ldots, k\}$ where $p(x, c; \theta)$ is modeled instead. Knowing the priors $\mathbb{P}(c)$, we are able to model the condition distribution $p(x|c; \theta)$. **Unsupervised conditional generation** further assumes that we are not given access to the c, requiring us to model the clustering of our input for generation. The latter case will be the primary focus of this section.

4.1 Pure Gaussian Mixture Models

Gaussian Mixture fitting with the EM algorithm allows $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ to be modeled with a cluster of Gaussian distribution. During inference time, one could sample from one of the Gaussian cluster conditioned on one of the available class for unsupervised conditional generation. This comes with severe limitations as previously established, as Gaussian Mixture models are prone to local optima as a result of poor starting condition while requiring the input dataset to be virtually noise-free.

4.2 Conditional Variational Autoencoders

Variational Autoencoders allows the $p(x, \mathbf{z}; \theta)$ to be learned and sampled from, fulfilling the generation task. Still, a glaring limitation remains with the Gaussian limitation of the latent space ($p(\mathbf{z}; \theta)$ is $\mathcal{N}(\mathbf{0}, \mathbf{I_s})$). In this section, we aim to survey existing works that generalizes the latent space to a support discrete sampling from a set of classes $\mathcal{C} = \{1, 2, \ldots, k\}$.

4.2.1 The M2 Model

The paper "Semi-supervised Learning with Deep Generative Models" [4] generalizes the regular VAE's latent space with the following assumptions during the sampling process

- Sampled data can take k different classes $\mathcal{C} = \{1, 2, \ldots, k\}$.
- The prior is modeled by two latent components $c \in \mathcal{C}$ and $\mathbf{z} \in \mathbb{R}^s$ instead of just $\mathbf{z}$
- Given $p(c; \theta) \sim \text{Categorical}(\mathcal{C})$ (assumed to be of equal probability) and $p(\mathbf{z}; \theta) \sim \mathcal{N}(\mathbf{0}, \mathbf{I_s})$ (the latter being the exact same as the VAE), we aim to model $p(\mathbf{x}, \mathbf{z}, c; \theta)$ and the corresponding $q(\mathbf{z}, c|\mathbf{x}; \phi)$

This model, dubbed the M2 model in the paper (since M1 refers to the original VAE), assumes that distinct pdf's $p(\mathbf{x}|\mathbf{z}, c; \theta)$ can be modeled conditional on c discretely in addition to a continuous latent $\mathbf{z}$. For the case of image generation, this could be conditioning the generation to a cat via c, and the breed of said cat via $\mathbf{z}$. To model the training objective, the authors first consider a supervised setting, one where c is given:

$$\begin{aligned}
\text{ELBO}(\mathbf{x}, c, \theta, \phi) &= \mathbb{E}_{z \sim q(\mathbf{z}|\mathbf{x}, c; \phi)} \left[\log \left(\frac{p(\mathbf{x}, \mathbf{z}, c; \theta)}{q(\mathbf{z}|\mathbf{x}, c; \phi)} \right) \right] \\
&= \mathbb{E}_{z \sim q(\mathbf{z}|\mathbf{x}, c; \phi)} [\log(p(\mathbf{x}|c, \mathbf{z}; \theta)) + \log(p(c; \theta)) + \log(p(\mathbf{z}; \theta)) - \log(q(\mathbf{z}|\mathbf{x}, c))] \\
&= -\mathcal{L}(\mathbf{x}, c)
\end{aligned}$$

This is the regular reconstruction + KL divergence formulation of the VAE while additionally conditioning the decoder on c. The authors then further generalizes the term in the case where c is not given, resulting in the expression

$$\text{ELBO}(\mathbf{x}, \theta, \phi) = \sum_{c \in \mathcal{C}} q(c|\mathbf{x}; \phi)(-\mathcal{L}(\mathbf{x}, c) + \mathcal{H}(q(c|\mathbf{x}; \phi)))$$

When $\mathcal{H}(q(c|\mathbf{x}; \phi))$ denotes the conditional entropy of c given $\mathbf{x}$ under the classification of q, ϕ. This term optimizes when $q(c|\mathbf{x}; \phi)$ is uniform, which refers to how an unlabeled sample is as likely to be of any class. In a semi-supervised setting, we are given labeled samples to learn $q(c|\mathbf{x}; \phi)$, hence $-\mathcal{L}(\mathbf{x}, c)$ is updated most in the it's most likely class. However, in an unsupervised setting, $q(c|\mathbf{x}; \phi)$ is learned during the optimization of $\text{ELBO}(\mathbf{x}, \theta, \phi)$ where the class with the highest ELBO is the most likely to be predicted.

4.2.2 The GMVAE Model

The paper "Deep Unsupervised Clustering with Gaussian Mixture Variational Autoencoders" [1] further generalizes the latent space into a Gaussian Mixture Model as follows:

- Sampled data can take k different classes $\mathcal{C} = \{1, 2, \dots, k\}$.
- The prior is modeled by two latent components $c \in \mathcal{C}$ and $\mathbf{z} \in \mathbb{R}^s$ instead of just $\mathbf{z}$
- Given $p(c; \theta) \sim \text{Categorical}(\mathcal{C})$ (assumed to be of equal probability) and $p(\mathbf{z}_1; \theta) \sim \mathcal{N}(\mathbf{0}, \mathbf{I_s})$, we model $\mathbf{z}_2$ as a Gaussian Mixture Model where the class weights are $p(c; \theta)$ with means $\mu_\theta(\mathbf{z}_1)$ and covariances $\text{diag}(\sigma^2_\theta(\mathbf{z}_1))$ conditioned on $\mathbf{z}_1$, that is, $\mathbf{z}_2|\mathbf{z}_1, c$ is a Gaussian Mixture Model.
- Likewise, we aim to learn $p(\mathbf{x}, \mathbf{z}_1, c, \mathbf{z}_2; \theta)$ and the corresponding $q(\mathbf{z}_1, c, \mathbf{z}_2|\mathbf{x}; \phi)$

This model, dubbed the Gaussian Mixture VAE (GMVAE), implicitly uses a Gaussian Mixture model as its latent space since only the conditional latent follows a Gaussian Mixture model. Likewise, the author derived the ELBO for this model, which can be expressed like below:

$$\begin{aligned}
\text{ELBO}(\mathbf{x}, \theta, \phi) = \; & \mathbb{E}_{c \sim q(c|\mathbf{x};\phi)} \big[\log p(\mathbf{x} \mid c; \theta)\big] \\
& - \mathbb{E}_{\mathbf{z}_1,\, c \sim q(\mathbf{z}_1|\mathbf{x};\phi)\, p(c|\mathbf{z}_2, \mathbf{z}_1;\theta)} \big[D_{\text{KL}}\big(q(c \mid \mathbf{z}_2, \mathbf{x}; \phi),\, p(c \mid \mathbf{z}_2, \mathbf{z}_1; \theta)\big)\big] \\
& - D_{\text{KL}}\big(q(\mathbf{z}_1 \mid \mathbf{x}; \phi),\, p(\mathbf{z}_1)\big) \\
& - \mathbb{E}_{c,\, \mathbf{z}_1 \sim q(c|\mathbf{x};\phi)\, q(\mathbf{z}_1|\mathbf{x};\phi)} \big[D_{\text{KL}}\big(p(c \mid \mathbf{z}_2, \mathbf{z}_1; \theta),\, p(c \mid \mathbf{z}_2; \theta)\big)\big]
\end{aligned}$$

We are left with one reconstruction term $\mathbb{E}_{c \sim q(c|\mathbf{x};\phi)} \big[\log p(\mathbf{x} \mid c; \theta)\big]$ and three KL divergence terms which computes the KL divergence of $\mathbf{z}_1, c$ under different parametrization (note that the author's derivation does not end up with a divergence term for $\mathbf{z}_2$, which would otherwise be intractable as $\mu_\theta(\mathbf{z}_1)$ and $\sigma^2_\theta(\mathbf{z}_1)$ are both unknown). The authors remarked that, like the reconstruction term, the expectation of KL divergence terms could not be directly computed and are instead optimized stochastically with monte-carlo samples.

The authors additionally pointed out the dominance of c-termed KL divergence, leading to an undesirable optima where these terms dominate the reconstruction. A solution proposed in this paper and the one prior is to set a cut-off, where the terms are incorporated if and only if they are above that threshold.

4.3 Comparison

We first verify that VAEs are able to generate data akin to those being sampled from the dataset. `code/categorical-generation/vae.ipynb` details the Implementation of a standard VAE, in which the underlying neural network architecture will be used with other implementations. Figure 4 depicts such generations from the MNIST dataset.

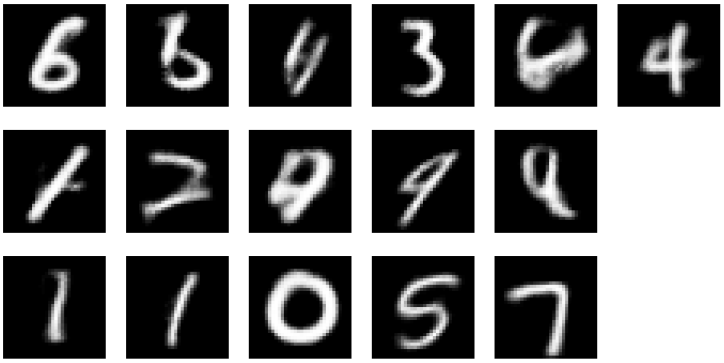

Figure 4: VAE results

Figure 5 shows the categorical generation capabilities of our M2 model implementation. We notice that, due to the dominance of the c KL divergence term, the classes don't separate well and overfit to

select "templates" which are not representative of the digits. However, it is important to note that the M2 model is designed to be trained in a semi-supervised setting where small labeled samples guide the unlabeled ones. Having trained the model unsupervised, the less than desirable accuracy is a result of this poor clustering. Still misclassifications share a general trend with the digit's appearances (i.e. 4 being misclassified as 9).

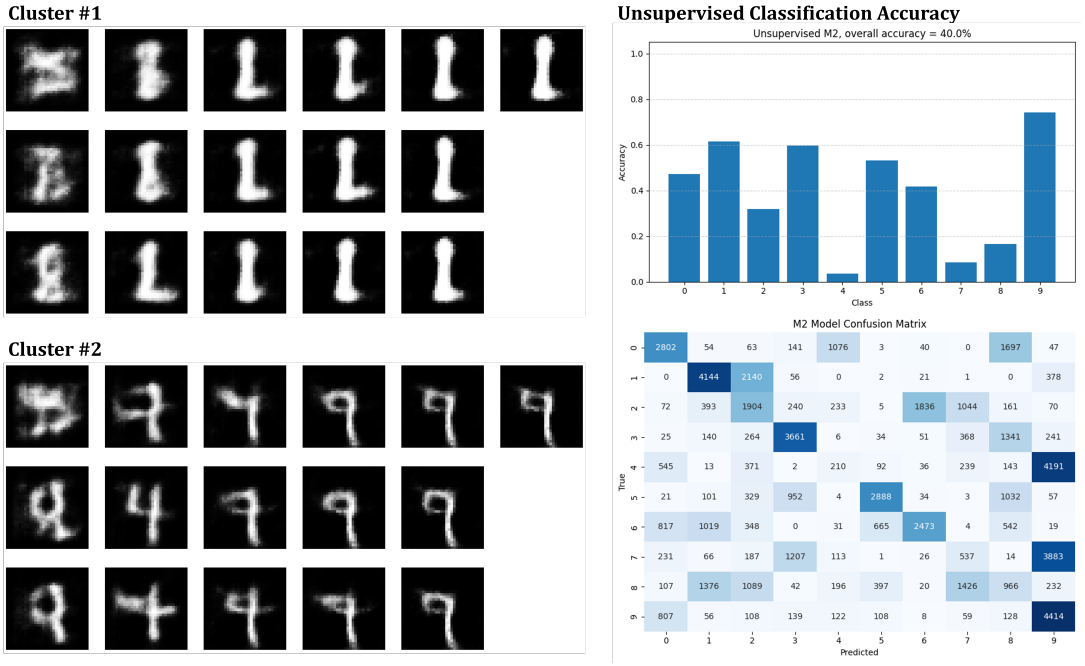

Figure 5: M2 VAE results

Figure 6 shows our attempt at reproducing the results from [1]. While we are able to train a successful VAE, the model does not show impressive clustering capabilities like the original paper. Instead, we notice a subtle pattern between images generated when conditioned with the same c. This is because $q(\mathbf{z}_2|\mathbf{z}_1, c)$ is only optimized indirectly, and more specific training conditions are needed for an exact reproduction. Still, we are able to identify that images from the same c share a common "style" (i.e. blurry lines on one of the clusters, thicker strokes on the other).

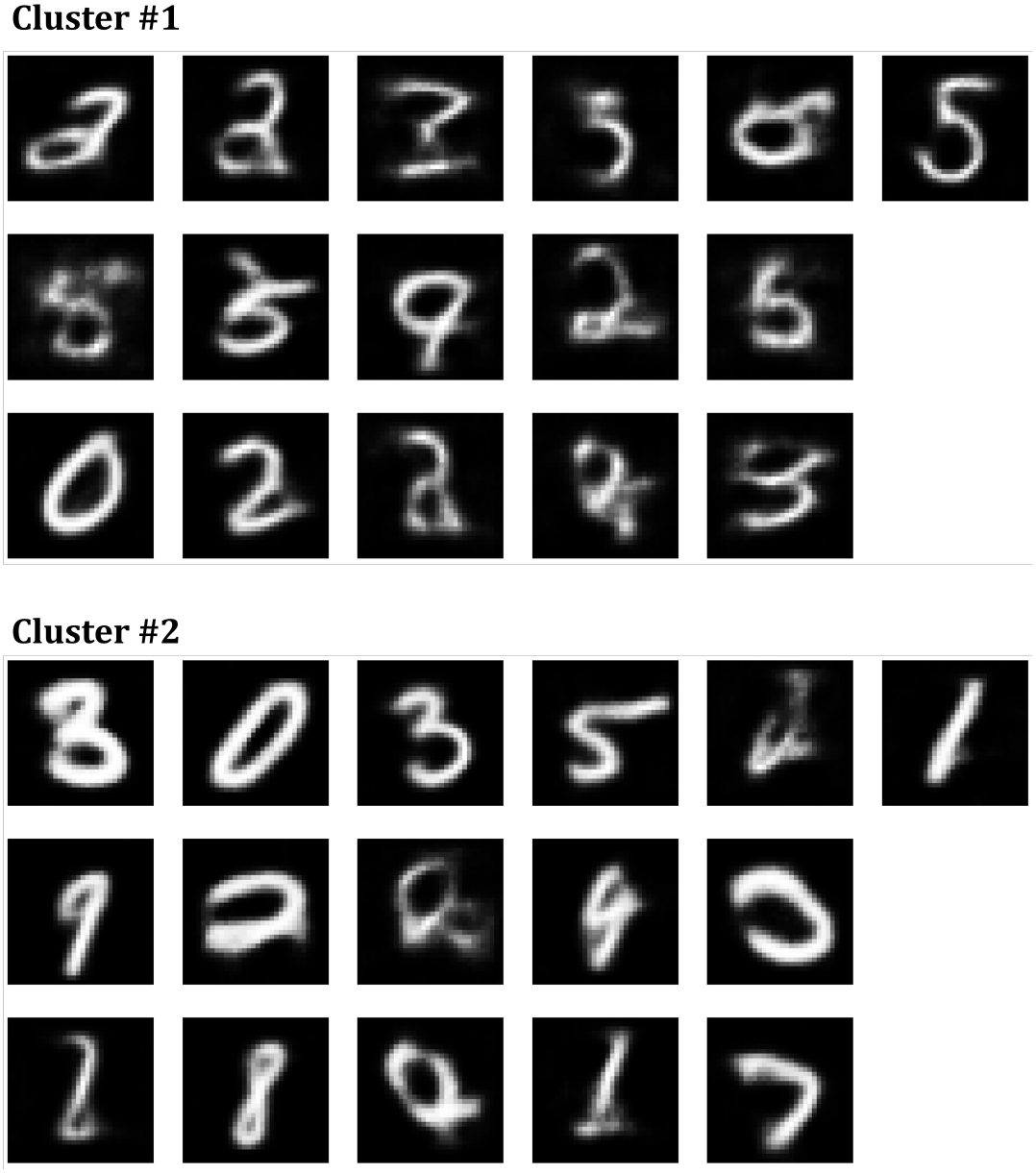

Figure 6: GMVAE results

References

[1] Nat Dilokthanakul, Pedro AM Mediano, Marta Garnelo, Matthew CH Lee, Hugh Salimbeni, Kai Arulkumaran, and Murray Shanahan. Deep unsupervised clustering with gaussian mixture variational autoencoders. *arXiv preprint arXiv:1611.02648*, 2016.

[2] Edward W Forgy. Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *biometrics*, 21:768–769, 1965.

[3] Diederik P Kingma, Max Welling, et al. Auto-encoding variational bayes, 2013.

[4] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. *Advances in neural information processing systems*, 27, 2014.

[5] Andrew Ng and Tengyu Ma. Cs229 lecture notes part ix: The em algorithm. `https://cs229.stanford.edu/summer2023/cs229-notes8.pdf`, 2019. CS229: Machine Learning, Summer 2023, Stanford University.