Big Learning Expectation Maximization

Anonymous submission

Abstract

Mixture models serve as one fundamental tool with versatile applications. However, their training techniques, like the popular Expectation Maximization (EM) algorithm, are notoriously sensitive to parameter initialization and often suffer from bad local optima that could be arbitrarily worse than the optimal. To address the long-lasting bad-local-optima challenge, we draw inspiration from the recent ground-breaking foundation models and propose to leverage their underlying big learning principle to upgrade the EM. Specifically, we present the Big Learning EM (BigLearn-EM), an EM upgrade that simultaneously performs joint, marginal, and orthogonally transformed marginal matchings between data and model distributions. Through simulated experiments, we empirically show that the BigLearn-EM is capable of delivering the optimal with high probability; comparisons on benchmark clustering datasets further demonstrate its effectiveness and advantages over existing techniques.

1 Introduction

As a fundamental and prominent tool in statistical machine learning and data science, mixture models are ubiquitously used in versatile practical applications that are associated with density estimation (Correia et al. 2023), clustering (Chandra, Canale, and Dunson 2023), anomaly detection (Qu et al. 2020; An, Wang, and Zhang 2022), feature extraction (Saire and Rivera 2022; Lin et al. 2023), model explanation (Xie and Chen 2022), flexible multimodal prior (Saseendran et al. 2021; Lee et al. 2021), deblurring (Guerrero-Colón, Mancera, and Portilla 2007; Yu, Sapiro, and Mallat 2011), *etc.* Among many variants of mixture models (Li, Yu, and Mandic 2020; Li et al. 2020), the most popular one is the Gaussian Mixture Model (GMM), thanks both to its simplicity and to its capability in approximating any continuous distribution arbitrarily well (Lindsay 1995; Peel and MacLahlan 2000). In this paper, we focus on the GMM for presentation, but the presented techniques can be readily extended to other mixture models.

Although mixture models are widely utilized in practical applications, most of their training techniques are known to be sensitive to parameter initialization (Bishop 2006; Jin et al. 2016; Kolouri, Rohde, and Hoffmann 2018), which alternatively restricts their actual performance. For example, the representative Expectation Maximization (EM) algorithm has been proven to converge to a bad local optimum that could be arbitrarily worse than the optimal solution with an exponentially high probability, when the number of mixture components exceeds three (Jin et al. 2016).

To address that long-lasting bad-local-optima challenge, we draw inspiration from the recent ground-breaking foundation models, by noticing that they benefit significantly from their massive diverse pretraining tasks, such as mask-and-predict (Devlin et al. 2018; He et al. 2022) and next-word-prediction (Radford et al. 2018, 2019; Brown et al. 2020). Specifically, (Cong and Zhao 2022) reveal that most of those pretraining strategies actually fall under the big learning principle, *i.e.,* leveraging one foundation model to simultaneously and consistently implement many/all joint, conditional, marginal matchings, as well as their transformed matchings, between data and model distributions.

Inspired by that, we propose to leverage the big learning principle to upgrade the conventional EM algorithm to a newly presented Big Learning EM (BigLearn-EM), demonstrating knowledge feedback from cutting-edge foundation models to conventional machine learning. Specifically, the BigLearn-EM exhaustively exploits its training data with a tailored big learning setup, where joint, marginal, and orthogonally transformed marginal matchings between data and model distributions are simultaneously considered. On simulated data, the BigLearn-EM delivers the optimal solution with high profitability, manifested as an encouraging direction to address the bad-local-optima challenge.

Our contributions are summarized as follows.

- We propose the BigLearn-EM, a novel, effective, and easy-to-use algorithm for training mixture models with only EM-type analytical parameter update formulas.
- We reveal that marginal/conditional matching could help joint matching getting out of bad local optima, which serves as one explanation justifying the successes of foundation models and the big learning principle.
- Comprehensive clustering experiments are conducted to demonstrate the superiority of the BigLearn-EM and its robustness to the scarcity of training data.

2 Preliminaries

We briefly review the preliminaries that lay the foundation of the presented technique, *i.e.,* mixture models, the EM algorithm, and the big learning principle.

Mixture Models

Mixture modeling leverages a mixture (*i.e.,* convex combination) of K (often simple) distributions $p_i(\boldsymbol{x}|\boldsymbol{\nu}_i)$ with parameters $\boldsymbol{\nu}_i$ and $i \in \{1, \cdots, K\}$ to construct a (more powerful) mixture model $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ for a random variable $\boldsymbol{x} \in \mathbb{R}^d$, *i.e.,*

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum\nolimits_{i=1}^{K} \pi_i p_i(\boldsymbol{x}|\boldsymbol{\nu}_i), \tag{1}$$

where the mixture weights $\pi_i > 0, \sum_{i=1}^{K} \pi_i = 1$ and $\boldsymbol{\theta} = \{\pi_i, \boldsymbol{\nu}_i\}_{i=1}^{K}$ denotes the model's parameters.

Among various mixture models (Li et al. 2020; Li, Yu, and Mandic 2020), the Gaussian Mixture Model (GMM), also called Mixture of Gaussians (MoG), is the most popular one; its probability density function is

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum\nolimits_{i=1}^{K} \pi_i \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), \tag{2}$$

where $\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i$ are the mean vector and the covariance matrix of the i^{th} Gaussian component, respectively.

The Expectation-Maximization Algorithm

The Expectation-Maximization (EM) algorithm (Dempster, Laird, and Rubin 1977) is the prominent way of estimating a (Gaussian) mixture model $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ from a collection of data sampled from an underlying data distribution $q(\boldsymbol{x})$.

Based on the variational inference framework with latent code $z \in \{1, \cdots, K\}$ and an inference arm $q(z|\boldsymbol{x})$ (Bishop 2006; Dieng and Paisley 2019), the EM algorithm (termed Joint-EM hereafter) maximizes the log-likelihood[1]

$$\begin{aligned}\mathbb{E}_{q(\boldsymbol{x})} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \mathbb{E}_{q(\boldsymbol{x})}\Big[\mathbb{E}_{q(z|\boldsymbol{x})} \log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, z)}{q(z|\boldsymbol{x})} \\ + \mathrm{KL}[q(z|\boldsymbol{x})||p_{\boldsymbol{\theta}}(z|\boldsymbol{x})]\Big]\end{aligned} \tag{3}$$

via alternatively updating $q(z|\boldsymbol{x})$ with an *E-step* and maximizing over $\boldsymbol{\theta}$ with an *M-step*, that is,

$$\begin{aligned}\textbf{E-step:}\quad & q(z|\boldsymbol{x}) = p_{\boldsymbol{\theta}}(z|\boldsymbol{x}) = \frac{\pi_z \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_z, \boldsymbol{\Sigma}_z)}{\sum_{i=1}^{K} \pi_i \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)} \\ \textbf{M-step:}\quad & \boldsymbol{\mu}_z = \frac{\mathbb{E}_{q(\boldsymbol{x})}[q(z|\boldsymbol{x})\boldsymbol{x}]}{\mathbb{E}_{q(\boldsymbol{x})}[q(z|\boldsymbol{x})]} \\ & \boldsymbol{\Sigma}_z = \frac{\mathbb{E}_{q(\boldsymbol{x})}[q(z|\boldsymbol{x})(\boldsymbol{x}-\boldsymbol{\mu}_z)(\boldsymbol{x}-\boldsymbol{\mu}_z)^T]}{\mathbb{E}_{q(\boldsymbol{x})}[q(z|\boldsymbol{x})]} \\ & \pi_z = \mathbb{E}_{q(\boldsymbol{x})}[q(z|\boldsymbol{x})].\end{aligned} \tag{4}$$

Maximizing the log-likelihood in (3) is equivalent to minimizing the Kullback-Leibler (KL) divergence $\mathrm{KL}[q(\boldsymbol{x})||p_{\boldsymbol{\theta}}(\boldsymbol{x})]$, leading to the KL-based *joint matching* in the joint $\boldsymbol{x}$-space, or informally $p_{\boldsymbol{\theta}}(\boldsymbol{x}) \to q(\boldsymbol{x})$.

Big Learning

Foundation models (Stickland and Murray 2019; Brown et al. 2020; He et al. 2021; Ramesh et al. 2022; Bao et al. 2023; OpenAI 2022; Ouyang et al. 2022) have demonstrated ground-breaking successes across diverse domains, thanks mainly to their large-scale pretraining on big data.

Observing that the pretraining strategies of foundation models share the similar underlying principle of comprehensively exploiting the data information from diverse perspectives, (Cong and Zhao 2022) condenses those diverse strategies into a unified big learning principle that contains most of them as special cases. Specifically, the big learning leverages one universal model with parameters $\boldsymbol{\theta}$ to simultaneously match many/all joint, marginal, and conditional data distributions across potentially diverse domains, as defined below.

Definition 1 ((Uni-modal) big learning (Cong and Zhao 2022))**.** *Given data samples $\boldsymbol{x} \in \mathbb{R}^L$ from the underlying data distribution $q(\boldsymbol{x})$, the index set $\mathbb{L} = \{1, \cdots, L\}$, and any two non-overlapping subsets $\mathbb{S} \subset \mathbb{L}$ and $\mathbb{T} \subseteq \mathbb{L}, \mathbb{T} \neq \emptyset$, the (uni-modal) big learning leverages a universal foundation model $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{\mathbb{T}}|\boldsymbol{x}_{\mathbb{S}}), \forall(\mathbb{S}, \mathbb{T})$ to model many/all joint, conditional, and marginal data distributions simultaneously,* i.e.,

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}_{\mathbb{T}}|\boldsymbol{x}_{\mathbb{S}}) \longrightarrow q(\boldsymbol{x}_{\mathbb{T}}|\boldsymbol{x}_{\mathbb{S}}), \forall(\mathbb{S}, \mathbb{T}) \in \boldsymbol{\Omega}, \tag{5}$$

where $\boldsymbol{\Omega}$ is the set that contains the $(\mathbb{S}, \mathbb{T})$ pairs of interest. Given different settings for $(\mathbb{S}, \mathbb{T})$, $q(\boldsymbol{x}_{\mathbb{T}}|\boldsymbol{x}_{\mathbb{S}})$ may represent a joint/marginal/conditional data distribution, whose samples are readily available from the training data. The actual objective measuring the distance/divergence (or encouraging the matching) between both sides of (5) *should be selected base on the application of interest.*

Based on Remark 3.5 of (Cong and Zhao 2022), one may alternatively or additionally do big learning in transformed domains, *e.g.,* via $p_{\boldsymbol{\theta}}(\hat{\boldsymbol{x}}_{\mathbb{T}}|\hat{\boldsymbol{x}}_{\mathbb{S}}) \longrightarrow q(\hat{\boldsymbol{x}}_{\mathbb{T}}|\hat{\boldsymbol{x}}_{\mathbb{S}})$ with transformation $\hat{\boldsymbol{x}} = g(\boldsymbol{x})$.

Below we will combine the above big learning principle in Definition 1 and Remark 3.5 of (Cong and Zhao 2022) to upgrade the Joint-EM in (4) into its big-learning extension, where the universal model $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{\mathbb{T}}|\boldsymbol{x}_{\mathbb{S}})$ has an analytical mixture expression for any $(\mathbb{S}, \mathbb{T})$ pair.

3 Big Learning Expectation Maximization

We first reveal a simple but somewhat counter-intuitive fact that lays the foundation of the proposed Big Learning EM (BigLearn-EM) algorithm. Then, based on that fact and the flexible big learning principle, we design a tailored big-learning task that consists of diverse matchings between data and model distributions. Finally, we summarizes and present the easy-to-use BigLearn-EM with only EM-type analytical parameter update formulas.

Marginal/Conditional Matching Gets Joint Matching Out of Bad Local Optima

It's well-known that the Joint-EM in (4) (*i.e.,* joint matching $p_{\boldsymbol{\theta}}(\boldsymbol{x}) \to q(\boldsymbol{x})$) often converges to a bad local optimum that could be arbitrarily worse than the optimal with an exponentially high probability (Bishop 2006; Jin et al. 2016; Kolouri, Rohde, and Hoffmann 2018), when the number of mixture components exceeds three. Fig. 1a illustrates an example bad local optimum when implementing the Joint-EM

[1] In practice, $\mathbb{E}_{q(\boldsymbol{x})}[\cdot]$ is estimated with data samples from $q(\boldsymbol{x})$.

on simulated data sampled from a GMM with 25 components (abbreviated as 25-GMM hereafter).

Next, with notations $\boldsymbol{x} \in \mathbb{R}^L$ and its index set $\mathbb{L} = \{1, \cdots, L\}$, let's consider the relationships among joint matching with $p_{\boldsymbol{\theta}}(\boldsymbol{x}) \rightarrow q(\boldsymbol{x})$, marginal matching with $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{\mathbb{T}}) \rightarrow q(\boldsymbol{x}_{\mathbb{T}})$, and conditional matching with $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{\mathbb{T}}|\boldsymbol{x}_{\mathbb{S}}) \rightarrow q(\boldsymbol{x}_{\mathbb{T}}|\boldsymbol{x}_{\mathbb{S}})$, where $\mathbb{T} \subseteq \mathbb{L}$, $\mathbb{T} \neq \emptyset$, $\mathbb{S} \subset \mathbb{L}$, $\mathbb{S} \cap \mathbb{T} = \emptyset$, and $\boldsymbol{x}_{\mathbb{T}}$ is the marginal vector of $\boldsymbol{x}$ indexed by $\mathbb{T}$.

Intuitively, one may anticipate that performing joint matching (with *e.g.,* the Joint-EM in (4)) will automatically lead to the convergences of both marginal matching (with *e.g.,* the following Marginal-EM in (6)) and conditional matching (via *e.g.,* maximizing the conditional log-likelihood in (7)).

Marginal Matching $\quad \mathbb{E}_{q(\boldsymbol{x}_{\mathbb{T}})} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_{\mathbb{T}})$

$$\begin{aligned}
&\textbf{E-step: } q(z|\boldsymbol{x}_{\mathbb{T}}) = p_{\boldsymbol{\theta}}(z|\boldsymbol{x}_{\mathbb{T}}) = \frac{\pi_z \mathcal{N}(\boldsymbol{x}_{\mathbb{T}}|\boldsymbol{\mu}_{z\mathbb{T}}, \boldsymbol{\Sigma}_{z\mathbb{T}\mathbb{T}})}{\sum_{i=1}^K \pi_i \mathcal{N}(\boldsymbol{x}_{\mathbb{T}}|\boldsymbol{\mu}_{i\mathbb{T}}, \boldsymbol{\Sigma}_{i\mathbb{T}\mathbb{T}})} \\
&\textbf{M-step: } \boldsymbol{\mu}_{z\mathbb{T}} = \frac{\mathbb{E}_{q(\boldsymbol{x}_{\mathbb{T}})}[q(z|\boldsymbol{x}_{\mathbb{T}})\boldsymbol{x}_{\mathbb{T}}]}{\mathbb{E}_{q(\boldsymbol{x}_{\mathbb{T}})}[q(z|\boldsymbol{x}_{\mathbb{T}})]} \\
&\qquad \boldsymbol{\Sigma}_{z\mathbb{T}\mathbb{T}} = \frac{\mathbb{E}_{q(\boldsymbol{x}_{\mathbb{T}})}[q(z|\boldsymbol{x}_{\mathbb{T}})(\boldsymbol{x}_{\mathbb{T}} - \boldsymbol{\mu}_{z\mathbb{T}})(\boldsymbol{x}_{\mathbb{T}} - \boldsymbol{\mu}_{z\mathbb{T}})^T]}{\mathbb{E}_{q(\boldsymbol{x}_{\mathbb{T}})}[q(z|\boldsymbol{x}_{\mathbb{T}})]} \\
&\qquad \pi_z = \mathbb{E}_{q(\boldsymbol{x}_{\mathbb{T}})}[q(z|\boldsymbol{x}_{\mathbb{T}})],
\end{aligned} \tag{6}$$

where $\boldsymbol{\mu}_{z\mathbb{T}}$ and $\boldsymbol{\Sigma}_{z\mathbb{T}\mathbb{T}}$ represent the marginal vector/matrix of $\boldsymbol{\mu}_z$ and $\boldsymbol{\Sigma}_z$, respectively.

Conditional Matching $\quad \mathbb{E}_{q(\boldsymbol{x}_{\mathbb{S}})q(\boldsymbol{x}_{\mathbb{T}}|\boldsymbol{x}_{\mathbb{S}})} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_{\mathbb{T}}|\boldsymbol{x}_{\mathbb{S}}) \quad (7)$

However, we empirically reveal below that the above intuition will *not* hold true when joint matching (or Joint-EM) gets stuck at a bad local optimum.

Specifically, we conduct two-stage experiments on simulated data from a 25-GMM $q(\boldsymbol{x})$ (see Fig. 1a), where the model $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ is also a 25-GMM with random initialization[2], Stage 1 implements joint matching (with Joint-EM in (4)), and, directly following Stage 1, Stage 2 either implements marginal matching (with Marginal-EM in (6)) or conditional matching (via maximizing the conditional log-likelihood in (7) with gradient accent.

Fig. 1 demonstrates the results. It's clear from Fig. 1a that joint matching gets stuck at a bad local optimum. As shown in Fig. 1b, the convergence of joint matching in Stage 1 does not necessarily result in the convergence of marginal matching, because continually performing Marginal-EM in Stage 2 further improves marginal matching. Similar phenomena are observed in Fig. 1c for conditional matching. That means bad local optima where joint matching gets stuck are not local optima for marginal/conditional matching, as illustrated in the left and right dashed lines of the schematic diagram in Fig. 1d. Alternatively, that inconsistency among joint, marginal, and conditional matchings may be leveraged, *e.g.,* to detect bad local optima of each matching or to help each other get out of bad local optima.

[2]Different from prior methods initializing parameters $\{\boldsymbol{\mu}_i\}_{i=1}^K$ with uniformly sampled training data, we use the more challenging Gaussian random initialization for $\{\boldsymbol{\mu}_i\}_{i=1}^K$ to highlight the power of the proposed BigLearn-EM.

It's worth highlighting that the center dashed line in Fig. 1d is located at a *consistent local optimum* for joint, marginal, and conditional matchings; more importantly, that consistency property is what the optimal solution must satisfy. The above analysis serves as an example justification for simultaneous joint, marginal, and conditional matchings, *i.e.,* the big learning principle in (5) that underlies most successful foundation models.

On Tailoring a Big-Learning Task to Produce an Easy-To-Use BigLearn-EM

Based on what's revealed in the previous section, one may naively follow the vanilla big learning principle in (5) to conduct multitasking joint, marginal, conditional matchings in the original $\boldsymbol{x}$-space, *i.e.,*

$$\max_{\boldsymbol{\theta}} \mathbb{E}_{q(\mathbb{S},\mathbb{T})} \mathbb{E}_{q(\boldsymbol{x}_{\mathbb{S}})q(\boldsymbol{x}_{\mathbb{T}}|\boldsymbol{x}_{\mathbb{S}})} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_{\mathbb{T}}|\boldsymbol{x}_{\mathbb{S}}), \tag{8}$$

where $q(\mathbb{S}, \mathbb{T})$ represents the sampling process of $(\mathbb{S}, \mathbb{T})$. Note $q(\mathbb{S}, \mathbb{T})$ actually determines the relative weightings among joint, marginal, and conditional matchings. However, it's not easy to design EM-type analytical update formulas for conditional matching in (7), even though such formulas are readily available for both joint and marginal matchings, as given in (4) and (6), respectively.

To avoid a hybrid algorithm that contain both EM-type and gradient accent updates and thus may not easy to use, we leverage the flexibility of big learning discussed in Remark 3.5 of (Cong and Zhao 2022) to further combine marginal matchings in randomly transformed $\boldsymbol{y}$ domains with the joint and marginal matchings in the original $\boldsymbol{x}$ domain, to form the tailored big-learning task.

Specifically, we employ orthogonal transformations $\boldsymbol{y} = \mathbf{A}\boldsymbol{x}$, where $\mathbf{A}$ is a randomly sampled orthogonal matrix. Correspondingly, the transformed training data $\boldsymbol{y} \sim \bar{q}_{\mathbf{A}}(\boldsymbol{y})$ are generated via $\boldsymbol{y} = \mathbf{A}\boldsymbol{x}, \boldsymbol{x} \sim q(\boldsymbol{x})$, the model in a transformed domain $\bar{p}_{\boldsymbol{\theta},\mathbf{A}}(\boldsymbol{y})$ is also a GMM with the analytical expression of

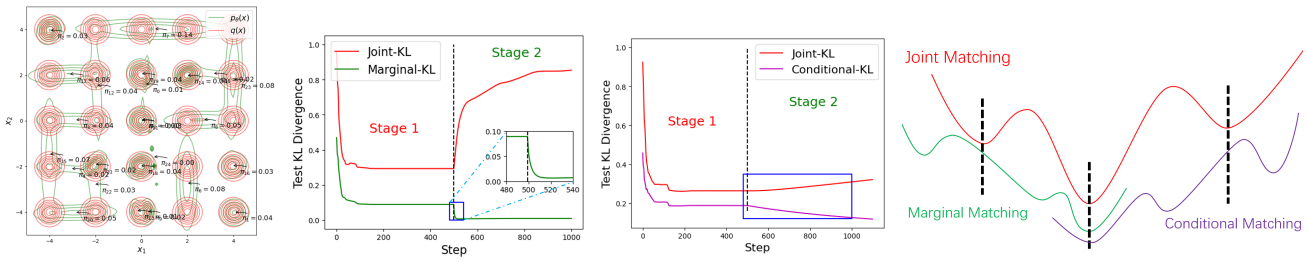
$$\bar{p}_{\boldsymbol{\theta},\mathbf{A}}(\boldsymbol{y}) = p_{\boldsymbol{\theta}}(\boldsymbol{x}) \left|\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{y}}\right| = \sum_{i=1}^K \pi_i \mathcal{N}(\boldsymbol{y}|\bar{\boldsymbol{\mu}}_i, \bar{\boldsymbol{\Sigma}}_i), \tag{9}$$

where $\bar{\boldsymbol{\mu}}_i = \mathbf{A}\boldsymbol{\mu}_i$, $\bar{\boldsymbol{\Sigma}}_i = \mathbf{A}\boldsymbol{\Sigma}_i\mathbf{A}^T$, and the transformed marginal matching has EM-type analytical update formulas

Randomly Transformed Marginal Matching $\quad \mathbb{E}_{\bar{q}_{\mathbf{A}}(\boldsymbol{y}_{\mathbb{T}})} \log \bar{p}_{\boldsymbol{\theta},\mathbf{A}}(\boldsymbol{y}_{\mathbb{T}})$

$$\begin{aligned}
&\textbf{E-step: } \bar{q}_{\mathbf{A}}(z|\boldsymbol{y}_{\mathbb{T}}) = \bar{p}_{\boldsymbol{\theta},\mathbf{A}}(z|\boldsymbol{y}_{\mathbb{T}}) = \frac{\pi_z \mathcal{N}(\boldsymbol{y}_{\mathbb{T}}|\bar{\boldsymbol{\mu}}_{z\mathbb{T}}, \bar{\boldsymbol{\Sigma}}_{z\mathbb{T}\mathbb{T}})}{\sum_{i=1}^K \pi_i \mathcal{N}(\boldsymbol{y}_{\mathbb{T}}|\bar{\boldsymbol{\mu}}_{i\mathbb{T}}, \bar{\boldsymbol{\Sigma}}_{i\mathbb{T}\mathbb{T}})} \\
&\textbf{M-step: } \bar{\boldsymbol{\mu}}_{z\mathbb{T}} = \frac{\mathbb{E}_{\bar{q}_{\mathbf{A}}(\boldsymbol{y}_{\mathbb{T}})}[\bar{q}_{\mathbf{A}}(z|\boldsymbol{y}_{\mathbb{T}})\boldsymbol{y}_{\mathbb{T}}]}{\mathbb{E}_{\bar{q}_{\mathbf{A}}(\boldsymbol{y}_{\mathbb{T}})}[\bar{q}_{\mathbf{A}}(z|\boldsymbol{y}_{\mathbb{T}})]} \\
&\qquad \bar{\boldsymbol{\Sigma}}_{z\mathbb{T}\mathbb{T}} = \frac{\mathbb{E}_{\bar{q}_{\mathbf{A}}(\boldsymbol{y}_{\mathbb{T}})}[\bar{q}_{\mathbf{A}}(z|\boldsymbol{y}_{\mathbb{T}})(\boldsymbol{y}_{\mathbb{T}} - \bar{\boldsymbol{\mu}}_{z\mathbb{T}})(\boldsymbol{y}_{\mathbb{T}} - \bar{\boldsymbol{\mu}}_{z\mathbb{T}})^T]}{\mathbb{E}_{\bar{q}_{\mathbf{A}}(\boldsymbol{y}_{\mathbb{T}})}[\bar{q}_{\mathbf{A}}(z|\boldsymbol{y}_{\mathbb{T}})]} \\
&\qquad \pi_z = \mathbb{E}_{\bar{q}_{\mathbf{A}}(\boldsymbol{y}_{\mathbb{T}})}[\bar{q}_{\mathbf{A}}(z|\boldsymbol{y}_{\mathbb{T}})] \\
&\textbf{Update } \boldsymbol{\theta}\textbf{: } \boldsymbol{\mu}_z = \mathbf{A}^T\bar{\boldsymbol{\mu}}_z^{'}, \quad \boldsymbol{\Sigma}_z = \mathbf{A}^T\bar{\boldsymbol{\Sigma}}_z^{'}\mathbf{A},
\end{aligned} \tag{10}$$

where $\bar{\boldsymbol{\mu}}_z^{'}/\bar{\boldsymbol{\Sigma}}_z^{'}$ is the $\mathbb{T}$-partially updated $\bar{\boldsymbol{\mu}}_z/\bar{\boldsymbol{\Sigma}}_z$ after the M-step. Note any joint matching in the transformed $\boldsymbol{y}$ domain will deliver the same update formulas as in (4).



(a) **(b)** **(c)** **(d)**

Figure 1: Marginal/Conditional matching gets joint matching out of bad local optima. The simulated data distribution $q(\boldsymbol{x})$ is set as a GMM with 25 components (*i.e.,* a 25-GMM); the model $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ is also a 25-GMM with random initialization. (a) The Joint-EM of joint matching converges to a bad local optimum. (b) Continuing joint matching in Stage 1, marginal matching in Stage 2 may be further improved and get joint matching out of that bad local optima. (c) Similar results are observed when Stage 2 implements conditional matching. (d) Schematic diagram of what happens in (b) and (c) from the loss perspective.

To summarize, the tailored big-learning task contains three kinds of matching, that is, joint, marginal, and transformed marginal matchings, each of which has EM-type analytical formulas for parameter updates, *i.e.,* (4), (6), and (10), respectively.

Finalizing the BigLearn-EM

Before finalizing our BigLearn-EM, an issue of the EM-type updates should be addressed. It's easy to verify that, during the EM iterations, once a mixture weight π_z becomes zero, it stays zero thereafter. Empirically, this issue hinders the EM-type updates in (4), (6), and (10) from making full use of the available mixture components, even though the occupied components have no enough modeling capacity.

To address that issue, we leverage the Maximum a posteriori (MAP) estimate in place of the vanilla maximum log-likelihood estimate on the mixture weights $\boldsymbol{\pi}$ following (Bishop 2006). Accordingly, taking Joint-EM in (4) as an example, the update rule for $\boldsymbol{\pi}$ is replaced by

$$\pi_z = \frac{\mathbb{E}_{q(\boldsymbol{x})}[q(z|\boldsymbol{x})] + \eta}{1 + K\eta}, \tag{11}$$

where $\eta > 0$ is a small constant. Similar modifications are also applied to (6) and (10), respectively. Detailed derivations are given in Appendix A.

Based on the aforementioned tailored big-learning task and the MAP modification on $\boldsymbol{\pi}$, we finalize the training objective of the BigLearn-EM as

$$\begin{aligned}\max_{\boldsymbol{\theta}} \ & \mathbb{E}_{q(\mathbb{S},\mathbb{T})q(\mathbf{A})}\mathbb{E}_{\bar{q}_{\mathbf{A}}(\boldsymbol{y}_{\mathbb{S}})\bar{q}_{\mathbf{A}}(\boldsymbol{y}_{\mathbb{T}}|\boldsymbol{y}_{\mathbb{S}})}\log \bar{p}_{\boldsymbol{\theta},\mathbf{A}}(\boldsymbol{y}_{\mathbb{T}}|\boldsymbol{y}_{\mathbb{S}}) \\ & + \gamma \log p_{\alpha}(\boldsymbol{\pi}),\end{aligned} \tag{12}$$

where $q(\mathbb{S},\mathbb{T})$ and $q(\mathbf{A})$ represent the sampling process of $(\mathbb{S},\mathbb{T})$ and the orthogonal matrix $\mathbf{A}$, respectively. $p_{\alpha}(\boldsymbol{\pi})$ is the prior for $\boldsymbol{\pi}$. γ is a hyper-parameter. Joint/Marginal matching may be recovered with $\mathbb{S} = \emptyset$, $\mathbf{A} = \mathbf{I}$.

Algorithm 1 summarizes the presented BigLearn-EM, where only easy-to-use EM-type updates are employed. Besides, it's worth highlighting that the BigLearn-EM can naturally handle incomplete data (via its marginal matchings) thanks to its big learning nature.

Algorithm 1: Big Learning Expectation Maximization

Input: Training data, the number K of mixture components, probabilities $[P_1, P_2]$, and the number W of local updates.

Output: A consistent local optimum $\boldsymbol{\theta}^* = \{\pi_i^*, \boldsymbol{\mu}_i^*, \boldsymbol{\Sigma}_i^*\}_{i=1}^K$.

Randomly initialize $\boldsymbol{\theta} = \{\pi_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^K$
while Not Mixing **do**
  With probability P_1,
    do Joint-EM with (4)/(11) for W iterations
  With probability P_2,
    (i) uniformly sample an index subset $\mathbb{T}$, and
    (ii) do Marginal-EM with (6)/(11) for W iters
  With probability $1 - P_1 - P_2$,
    (i) uniformly sample an orthogonal matrix $\mathbf{A}$
      ▷ `scipy.stats.ortho_group`
    (ii) uniformly sample an index subset $\mathbb{T}$, and
    (iii) do Transformed Marginal-EM with
      (10)/(11) for W iterations
end while

4 Related Work

Analysis and Improvements of the EM Algorithm In general settings, the EM (*i.e.,* Joint-EM) algorithm only have local convergence guarantee, that is, it converges to the optimal only if the parameters are initialized within a close neighborhood of that optimal (Yan, Yin, and Sarkar 2017; Zhao, Li, and Sun 2020; Balakrishnan, Wainwright, and Yu 2017). Although (Xu, Hsu, and Maleki 2016; Daskalakis, Tzamos, and Zampetakis 2017; Qian, Zhang, and Chen 2019) have established the global convergence for Joint-EM on learning GMMs with two components, a global convergence guarantee is not generally possible for GMMs with $K \geq 3$ components (Jin et al. 2016), where Joint-EM converges to a bad local optimum with an exponentially high probability (Jin et al. 2016). To deal with the challenge associated with bad local optima, many efforts have been made to improve Joint-EM, most of which focus on clever parameter initialization, seeking to help Joint-EM bypass bad local

optima before E-M iterates (Bachem et al. 2016b,a; Scrucca et al. 2016; Bachem, Lucic, and Krause 2018; Exarchakis, Oubari, and Lenz 2022; Tobin, Ho, and Zhang 2023). By contrast, the proposed BigLearn-EM, with random initialization, directly tackle the bad-local-optima challenge with diverse joint, marginal, transformed marginal EM updates, empirically delivering boosted performance than the Joint-EM (see the experiments).

Other Methods for Learning Mixture Models Besides the popular EM algorithm, many other methods for learning GMMs have also been developed based on, *e.g.,* Markov chain Monte Carlo (MCMC) (Rasmussen 1999; Favaro and Teh 2013; Das 2014), moments (Ge, Huang, and Kakade 2015; Kane 2021; Pereira, Kileel, and Kolda 2022), adversarial learning (Lin et al. 2018; Farnia et al. 2023), and optimal transport (Kolouri, Rohde, and Hoffmann 2018; Li et al. 2020; Yan, Wang, and Rigollet 2023). Specifically, the SW-GMM (Kolouri, Rohde, and Hoffmann 2018) leverages the Radon transform to randomly project the high-dimensional GMM learning task into *one-dimensional* sliced subspace, where the sliced Wasserstein distance between the projected data and model distributions is minimized *w.r.t.* GMM parameters. However, the computational complexity of the SW-GMM grows exponentially as the number of dimensions, rendering it unsuitable for modeling high-dimensional data (Li et al. 2020; Deshpande et al. 2019; Kolouri et al. 2019). Different from the aforementioned methods resorting to unstable adversarial learning or complicated Wasserstein distances, the presented BigLearn-EM is both easy-to-understand and easy-to-use, since it's a direct big-learning upgrade of the EM algorithm with only EM-type analytical formulas for parameter updates (and thus the same computational complexity as that of the EM).

5 Experiments

We first present the detailed ablation study that produces the BigLearn-EM from the vanilla Joint-EM. Then, we demonstrate the effectiveness of the BigLearn-EM in comprehensive real-world clustering applications. Finally, modified clustering experiments are conducted to reveal its robustness to data scarcity.

Ablation Study That Produces the BigLearn-EM

Based on the 25-GMM simulation setup shown in Fig. 1, we first present the detailed ablation study that produces the BigLearn-EM in Algorithm 1. Specifically, we start with the Joint-EM in (4) and test the performance when gradually introducing additional MAP estimate for $\boldsymbol{\pi}$ (marked "+Pr"), Marginal Matching in (6) (marked "+MM"), Conditional Matching in (7) (marked "+CM"), and Randomly Transformed Marginal Matching in (10) (marked "+RTMM") with different number W of local updates in Algorithm 1 (marked "+W").

The results from 10 different runs (with different random seeds) are summarized in Table 1, where it's clear that introducing prior for $\boldsymbol{\pi}$ (*i.e.,* "+Pr") improves the test joint KL divergence by 14.4% on average, despite with a doubly worsened standard deviation. By additionally employing marginal/conditional matching (*i.e.,* "+MM/+CM"), both the mean and standard deviation improve steadily, highlighting the benefits of the implicit diverse inter-regularization among various learning objectives of big learning. Further, boosted performance emerges from employing the Randomly Transformed Marginal Matching (*i.e.,* "+RTMM"), thanks to its significantly expanded diversity of matching, highlighting the effectiveness of the big learning principle as well as the importance of the diversity of big-learning tasks.

Table 1: Ablation study on the 25-GMM simulated datasets. "+Pr" means employing the MAP estimate for $\boldsymbol{\pi}$ with (11). "+MM/+CM/+RTMM" means introducing additional Marginal Matching, Conditional Matching, and Randomly Transformed Marginal Matching, respectively. "+W5" indicates employing $W = 5$ local updates in Algorithm 1.

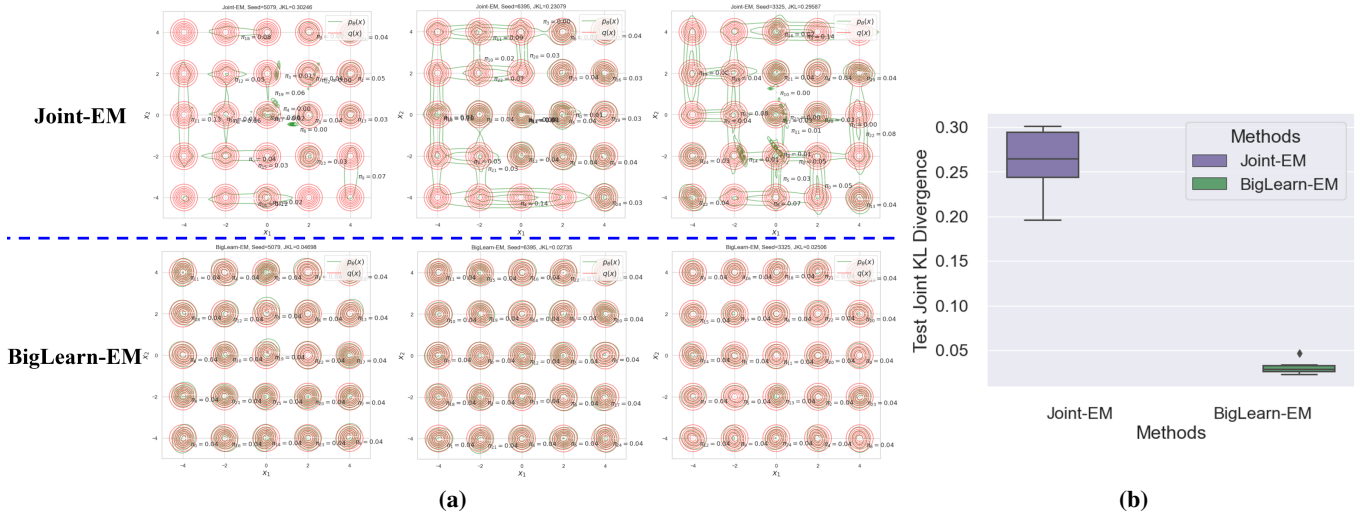
Method	Test Joint KL Divergence	
	Mean	Standard Deviation
Joint-EM	0.263	0.035
+ Pr	0.225	0.073
+ Pr + MM	0.141	0.054
+ Pr + MM + CM	0.124	0.044
+ Pr + MM + RTMM + W1	0.077	0.034
+ Pr + MM + RTMM + W5 *(BigLearn-EM)*	$\mathbf{0.030}$	$\mathbf{0.006}$
+ Pr + MM + RTMM + W10	0.031	0.007

For explicit comparisons between the Joint-EM and the BigLearn-EM, Fig. 2a demonstrates the local optima where both methods converge. It's clear that Joint-EM fails to make full use of the available 25 mixture components, suffering from bad local optima that could be arbitrarily worse than the optimal solution (Jin et al. 2016). By contrast, the presented BigLearn-EM, thanks to its big-learning nature, manages to fully exploit the 25 mixture components by placing each component to one data mode, delivering global optima with high probability on this simulation (refer to Fig. 2b). By considering that the BigLearn-EM merely uses the Gaussian random initialization for $\{\boldsymbol{\mu}_i\}_{i=1}^K$, it's therefore interesting to theoretically verify whether big learning could contribute to a global convergence guarantee for GMMs with $K \geq 3$ components; we leave that as future research.

BigLearn-EM for Real-World Data Clustering

Clustering stands as a representative application of GMM, addressing the task of categorizing unlabeled data into coherent and distinct clusters.

To validate the effectiveness of the BigLearn-EM in real-world clustering applications, we conduct comprehensive experiments on diverse clustering datasets, including Connect-4, Covtype, Glass, Letter, Pendigits, Satimage, Seismic, Svmguide2, and Vehicle (see Appendix B for details). The presented BigLearn-EM is systematically benchmarked against representative established clustering techniques, *i.e.,* the K-Means (Bottou and Bengio 1994), the SW-GMM (Kolouri, Rohde, and Hoffmann 2018), and the WM-

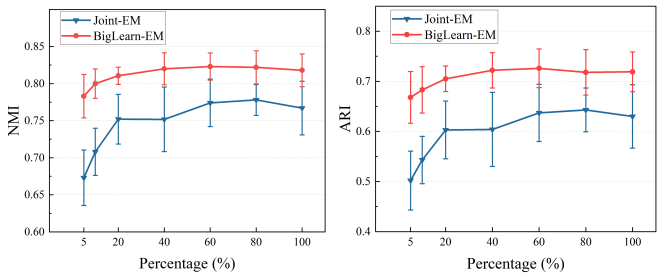


(a) (b)

Figure 2: Comparisons between the Joint-EM and the BigLearn-EM. (a) Explicit demonstrations of the local optima from both techniques *w.r.t.* different random seeds of 5079, 6395, and 3325, respectively. (b) Boxplot of the test joint KL divergences from 10 runs of both methods with 10 different random seeds.

Table 2: Clustering performance on real-world benchmark datasets. All the compared methods share the same settings for the GMM model $p_{\boldsymbol{\theta}}(\boldsymbol{x})$. The results are calculated based on 10 runs with different random seeds. Higher is better for all the metrics.

Dataset	Metric	K-Means	WM-GMM	SW-GMM	Joint-EM	BigLearn-EM
Connect-4	NMI	0.0027 ± 0.0005	0.0017 ± 0.0079	0.0015 ± 0.0086	$\mathbf{0.0031 \pm 0.0018}$	0.0021 ± 0.0011
	ARI	0.0005 ± 0.0003	0.0004 ± 0.0018	0.0003 ± 0.0013	$\mathbf{0.0028 \pm 0.0034}$	0.0019 ± 0.0015
	Joint-LL	—	89.006 ± 4.33	84.165 ± 8.0472	91.145 ± 7.680	$\mathbf{95.320 \pm 5.7727}$
Covtype	NMI	0.153 ± 0.0355	0.119 ± 0.0098	0.159 ± 0.0219	0.131 ± 0.0194	$\mathbf{0.181 \pm 0.0107}$
	ARI	0.035 ± 0.0089	0.068 ± 0.00562	0.0468 ± 0.0168	0.065 ± 0.0140	$\mathbf{0.072 \pm 0.0110}$
	Joint-LL	—	72.556 ± 0.029	72.773 ± 0.6811	72.967 ± 0.7598	$\mathbf{74.311 \pm 0.4227}$
Glass	NMI	0.434 ± 0.0503	0.445 ± 0.0794	0.450 ± 0.0319	0.434 ± 0.0487	$\mathbf{0.461 \pm 0.0298}$
	ARI	0.170 ± 0.0508	0.213 ± 0.0587	0.192 ± 0.0382	0.202 ± 0.0367	$\mathbf{0.203 \pm 0.0420}$
	Joint-LL	—	7.255 ± 0.8608	$\mathbf{7.387 \pm 1.2482}$	7.206 ± 2.0089	7.239 ± 2.1341
Letter	NMI	0.368 ± 0.0067	0.276 ± 0.0037	0.478 ± 0.0253	0.502 ± 0.0189	$\mathbf{0.526 \pm 0.0141}$
	ARI	0.128 ± 0.0045	0.010 ± 0.0021	0.188 ± 0.0200	0.203 ± 0.0222	$\mathbf{0.234 \pm 0.0126}$
	Joint-LL	—	11.750 ± 0.0758	19.03 ± 0.1498	19.402 ± 0.0245	$\mathbf{19.63 \pm 0.0145}$
Pendigits	NMI	0.714 ± 0.0056	0.794 ± 0.0168	0.756 ± 0.0315	0.767 ± 0.0362	$\mathbf{0.818 \pm 0.0221}$
	ARI	0.587 ± 0.0214	0.695 ± 0.0314	0.622 ± 0.0500	0.630 ± 0.0632	$\mathbf{0.719 \pm 0.0397}$
	Joint-LL	—	10.161 ± 0.0583	10.008 ± 0.2073	9.984 ± 0.0435	$\mathbf{10.283 \pm 0.0064}$
Satimage	NMI	0.608 ± 0.0008	0.598 ± 0.0114	0.596 ± 0.0492	0.592 ± 0.0294	$\mathbf{0.622 \pm 0.0151}$
	ARI	0.506 ± 0.0004	0.518 ± 0.0331	0.503 ± 0.1001	0.472 ± 0.0624	$\mathbf{0.523 \pm 0.0306}$
	Joint-LL	—	39.479 ± 0.0020	39.453 ± 0.070	39.445 ± 0.0055	$\mathbf{39.492 \pm 0.0004}$
Seismic	NMI	0.121 ± 0.0004	0.161 ± 0.0008	0.200 ± 0.071	0.193 ± 0.0023	$\mathbf{0.211 \pm 0.0089}$
	ARI	0.105 ± 0.0003	0.104 ± 0.3292	0.113 ± 0.0311	0.045 ± 0.0124	$\mathbf{0.127 \pm 0.0191}$
	Joint-LL	—	41.525 ± 0.027	42.317 ± 0.0612	42.202 ± 0.0940	$\mathbf{42.430 \pm 0.0329}$
Svmguide2	NMI	0.102 ± 0.0291	0.099 ± 0.0366	0.092 ± 0.0573	0.068 ± 0.0531	$\mathbf{0.204 \pm 0.0662}$
	ARI	0.076 ± 0.0254	0.056 ± 0.0369	0.083 ± 0.0721	0.029 ± 0.0560	$\mathbf{0.212 \pm 0.0932}$
	Joint-LL	—	10.270 ± 0.5096	10.483 ± 0.4825	$\mathbf{10.491 \pm 0.2925}$	10.425 ± 0.2210
Vehicle	NMI	0.166 ± 0.0267	0.243 ± 0.0198	0.189 ± 0.0579	0.222 ± 0.0860	$\mathbf{0.278 \pm 0.0452}$
	ARI	0.089 ± 0.0278	0.129 ± 0.0082	0.089 ± 0.0474	0.115 ± 0.0597	$\mathbf{0.152 \pm 0.0361}$
	Joint-LL	—	23.008 ± 1.2318	22.232 ± 1.8622	22.623 ± 2.2000	$\mathbf{23.480 \pm 1.3002}$



(a) NMI **(b)** ARI

Figure 3: Demonstration of the BigLearn-EM's robustness to the scarcity of its training data.

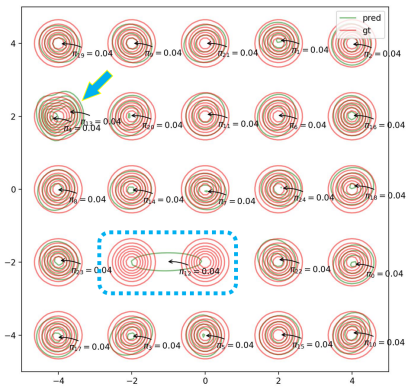


Figure 4: An example state where BigLearn-EM wanders around for many iterations.

GMM (Li et al. 2020), and the vanilla EM (Joint-EM) algorithm. Three testing metrics are adopted for performance evaluation, including

1. the normalized mutual information (NMI) (Strehl and Ghosh 2002), which quantifies how much the predicted clustering is informative about the true labels;
2. the adjusted rand index (ARI) (Hubert and Arabie 1985; Steinley 2004), which measures the degree of agreement between an estimated clustering and a reference clustering; and
3. the test joint log-likelihood (Joint-LL), which reflects how well the learned model describes the testing data from the joint KL divergence perspective.

Table 2 summarizes the results on the tested real-world clustering datasets. It's clear that the BigLearn-EM delivers overall boosted performance over the compared techniques, especially on the NMI and ARI values. That is expected because, as demonstrated in Fig. 2a, the presented BigLearn-EM is capable of making full use of the available mixture components to deliver precise and accurate local matching, which alternatively contributes to better NMI/ARI values. When compared to the Joint-EM, the BigLearn-EM demonstrates significantly improved performance, even though both of them are based on E-M iterations; that further substantiates the effectiveness of the big learning principle in addressing the bad-local-optima challenge inherent in the vanilla Joint-EM algorithm; more importantly, the BigLearn-EM also delivers smaller standard deviations across the majority of tested datasets, demonstrating the potential of the big learning to bring better learning stability and consistency. When compared to the WM-GMM and SW-GMM techniques that are developed based on complicated Wasserstein distances, the BigLearn-EM, which yields better performance, is clearly much easier to understand in theory and, simultaneously, easier to use in practice.

BigLearn-EM Is More Robust to the Scarcity of Its Training Data

Noticing that, in scenarios with limited training data, like the Svmguide2 dataset with 391 data samples in Table 2, the BigLearn-EM exhibits remarkably superior NMI/ARI performance than other clustering techniques. We posit that the BigLearn-EM is more robust to the scarcity of its training data, because its big-learning operation is expected to significantly increase the utilization rate of the information within each sample. We then design modified real-world clustering experiments to verify that hypothesis.

Specifically, based on the Pendigits dataset, we randomly select its 80%, 60%, 40%, 20%, 10%, and 5% training data to form a series of modified clustering datasets with gradually increased scarcity, where the BigLearn-EM is compared to the Joint-EM to highlight the influence of the big learning principle.

Fig. 3 demonstrates the experimental results, where it's clear that the BigLearn-EM is more robust to the scarcity of its training data than the Joint-EM, even though both of them utilize similar EM-type parameter update formulas, highlighting the effectiveness of the big learning.

6 Conclusions

By leveraging the big learning principle that underlies recent groundbreaking foundation models, we upgrade the vanilla EM algorithm to its big-learning extension that is termed the BigLearn-EM. The BigLearn-EM simultaneously performs joint, marginal, and orthogonal-transformed marginal matchings between data and model distributions, empirically demonstrating great potential in addressing the long-lasting bad-local-optima challenge of the EM algorithm. Comprehensive experiments on real-world clustering datasets demonstrate its boosted performance and its robustness to data scarcity.

Although the BigLearn-EM perform better than existing techniques in the tested scenarios, some issues remain unsolved. For example, (i) whether the BigLearn-EM *theoretically* addresses the bad-local-optima challenge of the EM is unanswered, (ii) a suitable stopping criteria, mimicking that of a MCMC, is still missing in Algorithm 1, and (iii) the exploration power of the BigLearn-EM may need further strengthening, as we find it may wander around a state like the one in Fig. 4 for many iterations.

References

An, P.; Wang, Z.; and Zhang, C. 2022. Ensemble unsupervised autoencoders and Gaussian mixture model for cyberattack detection. *Information Processing & Management*, 59(2): 102844.

Bachem, O.; Lucic, M.; Hassani, H.; and Krause, A. 2016a. Fast and provably good seedings for k-means. *Advances in neural information processing systems*, 29.

Bachem, O.; Lucic, M.; Hassani, S. H.; and Krause, A. 2016b. Approximate k-means++ in sublinear time. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

Bachem, O.; Lucic, M.; and Krause, A. 2018. Scalable k-means clustering via lightweight coresets. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1119–1127.

Balakrishnan, S.; Wainwright, M. J.; and Yu, B. 2017. Statistical guarantees for the EM algorithm: From population to sample-based analysis.

Bao, F.; Nie, S.; Xue, K.; Li, C.; Pu, S.; Wang, Y.; Yue, G.; Cao, Y.; Su, H.; and Zhu, J. 2023. One Transformer Fits All Distributions in Multi-Modal Diffusion at Scale. *arXiv preprint arXiv:2303.06555*.

Bishop, C. 2006. *Pattern Recognition and Machine Learning*. Springer.

Bottou, L.; and Bengio, Y. 1994. Convergence properties of the k-means algorithms. *Advances in neural information processing systems*, 7.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *NeurIPS*, 33: 1877–1901.

Chandra, N. K.; Canale, A.; and Dunson, D. B. 2023. Escaping The Curse of Dimensionality in Bayesian Model-Based Clustering. *J. Mach. Learn. Res.*, 24: 144–1.

Cong, Y.; and Zhao, M. 2022. Big Learning. *arXiv preprint arXiv:2207.03899*.

Correia, A. H.; Gala, G.; Quaeghebeur, E.; de Campos, C.; and Peharz, R. 2023. Continuous mixtures of tractable probabilistic models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 7244–7252.

Das, R. 2014. Collapsed Gibbs sampler for dirichlet process Gaussian mixture models (DPGMM). *Technical report, Carnegie Mellon University, United States*.

Daskalakis, C.; Tzamos, C.; and Zampetakis, M. 2017. Ten steps of EM suffice for mixtures of two Gaussians. In *Conference on Learning Theory*, 704–710. PMLR.

Dempster, A. P.; Laird, N. M.; and Rubin, D. B. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society: series B (methodological)*, 39(1): 1–22.

Deshpande, I.; Hu, Y.-T.; Sun, R.; Pyrros, A.; Siddiqui, N.; Koyejo, S.; Zhao, Z.; Forsyth, D.; and Schwing, A. G. 2019. Max-sliced wasserstein distance and its use for gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10648–10656.

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dieng, A. B.; and Paisley, J. 2019. Reweighted expectation maximization. *arXiv preprint arXiv:1906.05850*.

Exarchakis, G.; Oubari, O.; and Lenz, G. 2022. A sampling-based approach for efficient clustering in large datasets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12403–12412.

Farnia, F.; Wang, W. W.; Das, S.; and Jadbabaie, A. 2023. GAT–GMM: Generative Adversarial Training for Gaussian Mixture Models. *SIAM Journal on Mathematics of Data Science*, 5(1): 122–146.

Favaro, S.; and Teh, Y. W. 2013. MCMC for normalized random measure mixture models. *Statist. Sci.*, 28(3): 335–359.

Ge, R.; Huang, Q.; and Kakade, S. M. 2015. Learning mixtures of gaussians in high dimensions. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, 761–770.

Guerrero-Colón, J. A.; Mancera, L.; and Portilla, J. 2007. Image restoration using space-variant Gaussian scale mixtures in overcomplete pyramids. *IEEE Transactions on Image Processing*, 17(1): 27–41.

He, K.; Chen, X.; Xie, S.; Li, Y.; Dollár, P.; and Girshick, R. 2021. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*.

He, K.; Chen, X.; Xie, S.; Li, Y.; Dollár, P.; and Girshick, R. 2022. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 16000–16009.

Hubert, L.; and Arabie, P. 1985. Comparing partitions. *Journal of classification*, 2: 193–218.

Jin, C.; Zhang, Y.; Balakrishnan, S.; Wainwright, M. J.; and Jordan, M. I. 2016. Local maxima in the likelihood of gaussian mixture models: Structural results and algorithmic consequences. *Advances in neural information processing systems*, 29.

Kane, D. M. 2021. Robust learning of mixtures of gaussians. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1246–1258. SIAM.

Kolouri, S.; Nadjahi, K.; Simsekli, U.; Badeau, R.; and Rohde, G. 2019. Generalized sliced wasserstein distances. *Advances in neural information processing systems*, 32.

Kolouri, S.; Rohde, G. K.; and Hoffmann, H. 2018. Sliced wasserstein distance for learning gaussian mixture models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3427–3436.

Lee, D. B.; Min, D.; Lee, S.; and Hwang, S. J. 2021. Meta-gmvae: Mixture of gaussian vae for unsupervised meta-learning. In *International Conference on Learning Representations*.

Li, S.; Yu, Z.; and Mandic, D. 2020. A universal framework for learning the elliptical mixture model. *IEEE Transactions on Neural Networks and Learning Systems*, 32(7): 3181–3195.

Li, S.; Yu, Z.; Xiang, M.; and Mandic, D. 2020. Solving general elliptical mixture models through an approximate Wasserstein manifold. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 4658–4666.

Lin, D. J.; Gazi, A. H.; Kimball, J.; Nikbakht, M.; and Inan, O. T. 2023. Real-Time Seismocardiogram Feature Extraction Using Adaptive Gaussian Mixture Models. *IEEE Journal of Biomedical and Health Informatics*.

Lin, Z.; Khetan, A.; Fanti, G.; and Oh, S. 2018. Pacgan: The power of two samples in generative adversarial networks. *Advances in neural information processing systems*, 31.

Lindsay, B. G. 1995. Mixture models: theory, geometry, and applications. Ims.

OpenAI. 2022. ChatGPT: Optimizing Language Models for Dialogue.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C. L.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*.

Peel, D.; and MacLahlan, G. 2000. Finite mixture models. *John & Sons*.

Pereira, J. M.; Kileel, J.; and Kolda, T. G. 2022. Tensor moments of gaussian mixture models: Theory and applications. *arXiv preprint arXiv:2202.06930*.

Qian, W.; Zhang, Y.; and Chen, Y. 2019. Global convergence of least squares EM for demixing two log-concave densities. *Advances in Neural Information Processing Systems*, 32.

Qu, J.; Du, Q.; Li, Y.; Tian, L.; and Xia, H. 2020. Anomaly detection in hyperspectral imagery based on Gaussian mixture model. *IEEE Transactions on Geoscience and Remote Sensing*, 59(11): 9504–9517.

Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I.; et al. 2018. Improving language understanding by generative pre-training.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8): 9.

Ramesh, A.; Dhariwal, P.; Nichol, A.; Chu, C.; and Chen, M. 2022. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*.

Rasmussen, C. 1999. The infinite Gaussian mixture model. *Advances in neural information processing systems*, 12.

Saire, D.; and Rivera, A. R. 2022. Global and Local Features Through Gaussian Mixture Models on Image Semantic Segmentation. *IEEE Access*, 10: 77323–77336.

Saseendran, A.; Skubch, K.; Falkner, S.; and Keuper, M. 2021. Shape your space: A gaussian mixture regularization approach to deterministic autoencoders. *Advances in Neural Information Processing Systems*, 34: 7319–7332.

Scrucca, L.; Fop, M.; Murphy, T. B.; and Raftery, A. E. 2016. mclust 5: clustering, classification and density estimation using Gaussian finite mixture models. *The R journal*, 8(1): 289.

Steinley, D. 2004. Properties of the hubert-arable adjusted rand index. *Psychological methods*, 9(3): 386.

Stickland, A.; and Murray, I. 2019. BERT and PALs: Projected attention layers for efficient adaptation in multi-task learning. *arXiv preprint arXiv:1902.02671*.

Strehl, A.; and Ghosh, J. 2002. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, 3(Dec): 583–617.

Tobin, J.; Ho, C. P.; and Zhang, M. 2023. Reinforced EM Algorithm for Clustering with Gaussian Mixture Models.

Xie, Z.; and Chen, D. 2022. Joint Gaussian Mixture Model for Versatile Deep Visual Model Explanation.

Xu, J.; Hsu, D. J.; and Maleki, A. 2016. Global analysis of expectation maximization for mixtures of two gaussians. *Advances in Neural Information Processing Systems*, 29.

Yan, B.; Yin, M.; and Sarkar, P. 2017. Convergence of gradient EM on multi-component mixture of Gaussians. *Advances in Neural Information Processing Systems*, 30.

Yan, Y.; Wang, K.; and Rigollet, P. 2023. Learning Gaussian Mixtures Using the Wasserstein-Fisher-Rao Gradient Flow. *arXiv preprint arXiv:2301.01766*.

Yu, G.; Sapiro, G.; and Mallat, S. 2011. Solving inverse problems with piecewise linear estimators: From Gaussian mixture models to structured sparsity. *IEEE Transactions on Image Processing*, 21(5): 2481–2499.

Zhao, R.; Li, Y.; and Sun, Y. 2020. Statistical convergence of the EM algorithm on Gaussian mixture models.

Appendix of Big Learning Expectation Maximization

Anonymous Authors

A On Introducing the MAP Estimate of $\boldsymbol{\pi}$

As the mixture weights $\boldsymbol{\pi}$ is located in a simplex, *i.e.,* $\pi_i > 0, \sum_{i=1}^K \pi_i = 1$, a commonly used prior for $\boldsymbol{\pi}$ is a Dirichlet distribution

$$p_{\boldsymbol{\alpha}}(\boldsymbol{\pi}) = \mathrm{Dir}(\boldsymbol{\pi}; \boldsymbol{\alpha}) = \frac{\Gamma(\sum_{i=1}^K \alpha_i)}{\prod_{i=1}^K \Gamma(\alpha_i)} \prod_{i=1}^K \pi_i^{\alpha_i - 1}, \tag{13}$$

where the concentration parameters $\boldsymbol{\alpha} = (\alpha_1, \cdots, \alpha_K)$. Often, to encourage a full utilization of mixture components, one would prefer setting $\alpha_i > 1$. We set $\alpha_i = \alpha_j = \alpha > 1$ in this paper.

Taking the Joint-EM in (4) of the main manuscript as a demonstration example, we next elaborate on the the MAP Estimate of $\boldsymbol{\pi}$, where the objective for $\boldsymbol{\pi}$ is

$$\gamma \log p_{\boldsymbol{\alpha}}(\boldsymbol{\pi}) + \mathbb{E}_{q(\boldsymbol{x})} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}), \tag{14}$$

where $\gamma > 0$ is a hyper-parameter that balances between the prior and the likelihood. Note the first prior term is independent to $\{\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^K$; therefore, the update rules for $q(z|\boldsymbol{x})$ and $\{\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^K$ are the same as those of the Joint-EM in (4). One need only focus on the estimate of $\boldsymbol{\pi}$.

The objective in (14) can be simplified *w.r.t.* $\boldsymbol{\pi}$ as

$$\begin{aligned} \gamma \mathrm{log} p_{\boldsymbol{\alpha}}(\boldsymbol{\pi}) + \mathbb{E}_{q(\boldsymbol{x})} \mathrm{log} p_{\boldsymbol{\theta}}(\boldsymbol{x}) = C + \sum\nolimits_{i=1}^K \gamma(\alpha - 1) \mathrm{log} \pi_i + \sum\nolimits_{z=1}^K \mathbb{E}_{q(\boldsymbol{x})}[q(z|\boldsymbol{x})] \mathrm{log} \pi_z \\ \textit{s.t.} \ \sum\nolimits_{z=1}^K \pi_z = 1 \\ \pi_z \geq 0, \end{aligned} \tag{15}$$

which is constrained optimization problem that can be readily solved by the method of Lagrange multipliers. Accordingly, we have the MAP estimate of $\boldsymbol{\pi}$ as

$$\pi_z^* = \frac{\mathbb{E}_{q(\boldsymbol{x})}[q(z|\boldsymbol{x})] + \gamma(\alpha - 1)}{1 + \sum_{z=1}^K \gamma(\alpha - 1)} = \frac{\mathbb{E}_{q(\boldsymbol{x})}[q(z|\boldsymbol{x})] + \eta}{1 + K\eta} \tag{16}$$

where $\eta = \gamma(\alpha - 1) > 0$ and we conclude the derivation of (11) of the main manuscript.

B Settings of the Real-World Clustering Experiments

Clustering Datasets We adopt nine representative datasets[3] that are extensively employed in the context of clustering, with their statistics summarized in Table 3. We follow () to normalize the data feature-wisely to the interval $[0, 1]$, using the min-max scaling. For performance evaluation, we use the official testing data set if it's available; otherwise, we randomly select 20% data samples to form a testing set.

Table 3: Statistics of the adopted real-world clustering datasets.

Dataset	Dimension	Number	Class
Connect-4	126	67557	3
Covtype	54	581012	7
Glass	9	214	6
Letter	16	20000	26
Pendigits	16	10992	10
Satimage	36	6435	6
Seismic	50	98528	3
Svmguide2	20	391	3
Vehicle	18	846	4

Performance Evaluation Metrics We adopt three metrics for testing, that is,

[3] https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multiclass.html

- Normalized Mutual Information (NMI) (Strehl and Ghosh 2002). The NMI score rescales mutual information scores using a generalized mean of the entropy of the true label set Ω and the cluster label set C. This process can be mathematically formulated as

$$\mathrm{NMI}(\Omega, C) = \frac{I(\Omega; C)}{(H(\Omega) + H(C))/2}, \tag{17}$$

  where $I(\Omega; C) = H(\Omega) + H(C) - H(\Omega, C)$ denotes the mutual information between Ω and C, and H is the information entropy.

- Adjusted Rand Index (ARI) (Hubert and Arabie 1985; Steinley 2004). The ARI score represents an adjusted version of the Rand Index (RI) that accounts for chance. The RI itself serves as a measure of similarity, evaluating all possible pairs of samples and quantifying the instances where pairs are assigned to the same or distinct clusters in both predicted and true label assignments. The formalization of ARI is articulated as

$$\mathrm{ARI} = \frac{(\mathrm{RI} - \text{ Expected RI })}{(\max(\mathrm{RI}) - \text{ Expected RI})}. \tag{18}$$

- Test Joint Log-Likelihood (Joint-LL). The Joint-LL measures the extent to which the acquired model characterizes the testing dataset based on the joint KL divergence. It's calculated based on the testing data $\boldsymbol{x}$ as

$$\log p(\boldsymbol{x}) = \log \sum\nolimits_{i=1}^{K} \pi_i \mathcal{N}(\boldsymbol{x} | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i). \tag{19}$$

Parameter Settings The primary model parameters, *i.e.,* the number K of mixture components, adopted for each datasets are listed in Table 4. For all the compared methods, we maintain the numerical stability by preventing the covariance matrices from being singular; specifically, we restrict the eigenvalues of each covariance matrix to be larger than hyperparameter $\epsilon > 0$. For the BigLearn-EM in Algorithm 1 of the main manuscript, to sample an index subset $\mathbb{T}$ from the full index set $\mathbb{L}$, we first sample a ratio $r_{\mathbb{T}} \sim \mathrm{Beta}(\beta_1, \beta_2)$ from the Beta distribution $\mathrm{Beta}(\beta_1, \beta_2)$ with hyperparameters (β_1, β_2); then, we uniformly sample $r_{\mathbb{T}}$-ratio indices from $\mathbb{L}$ to yield $\mathbb{T}$; we run the BigLearn-EM for 10000 iterations on all dataset and report the mean NMI, ARI, and Joint-LL values that are calculated based on the last 200 iterations.

Table 4: Parameter configurations.

Dataset	Connect-4	Covtype	Glass	Letter	Pendigits	Satimage	Seismic	Svmguide2	Vehicle
K	6	10	6	26	12	8	4	3	6
ϵ	1×10^{-2}	5×10^{-3}	1×10^{-2}	1×10^{-3}	1×10^{-2}	1×10^{-2}	1×10^{-2}	2×10^{-2}	1×10^{-3}
(β_1, β_2)	$(5, 1)$	$(5, 1)$	$(5, 1)$	$(5, 1)$	$(5, 1)$	$(5, 1)$	$(5, 1)$	$(5, 1)$	$(5, 1)$