.一、研究背景與動機

在探索宇宙知識邊界的過程中，廖教授發現了一系列粒子加速器的實驗數據。這些數據可能藏有某種特殊的粒子特徵，幫助他理解宇宙中還沒被發現的現象。為了找出這些潛藏的訊號，他希望能把資料中的數據依照內在特徵分成不同的類別。

這份任務的目標，就是利用分群（Clustering）的方式，把資料自動分成幾個不同的群組。假如資料是 n 維，我們預期會有 4n - 1 個分群，也就是說這是一個需要根據維度決定群數的問題，而不是事先知道答案。

分群的好壞以 Folkes – Mallows Index（FMI）來評分，這個指標介於 0 到 1 之間，分數越高表示我們分得越準確。

二、理論基礎與相關技術

2.1 分群方法概述

分群（Clustering）是一種機器學習中的無監督式學習方法，目的是將資料依其特徵相似度，自動歸類為數個群組。常見的傳統分群法包含：

- K-Means：根據距離將資料劃分為固定數量的群。
- DBSCAN：基於密度的分群方法，適合非球狀或含雜訊資料。
- Hierarchical Clustering：利用資料間的層級關係進行分類。

這些方法雖然簡單直觀，但在處理高維度資料或非線性結構時，往往面臨效能與準確度的瓶頸。

近年來，隨著深度學習的發展，出現了結合神經網路的分群方法，能夠從資料中學習出更具表現力的潛在特徵。本文分群中有使用的方法為：

- Autoencoder-based Clustering：利用神經網路將資料轉換為低維潛在空間，再進行分群。
- Deep Embedded Clustering（DEC）：將降維與分群同時學習，強化分群品質。

2.2 傳統的分群方法：K-Means 與 K-Means++(改良版)

2.2.1 K-Means

K-Means 是一種常見的中心導向分群（centroid-based clustering）方法，適用於結構明確、距離可量化的資料。其基本概念是將資料劃分為事先指定的 k 個群，並使每個資料點屬於距離某一個中心（centroid）最近的群。

K-Means 的流程如下：

1. 任意選定 k 個初始中心點。
2. 將所有資料點依據歐氏距離歸類到最近的中心點。
3. 重新計算每一群的中心位置。
4. 重複步驟 2 和 3，直到中心點收斂或達到最大迭代次數。

問題：隨機初始化可能導致：

- 不同 run 結果不一致（high variance）。
- 初始中心落在彼此靠近的位置 → 導致局部最小值（non-optimal partition）。
- 對 outliers 敏感，增加計算成本與誤差。

2.2.2 K-Means++(改良版)

目標是讓初始中心**盡量分散**，以提高收斂速度與準確率。
K-Means++ 的流程如下：

1. 隨機選擇一個點作為第一個中心c_1。
2. 對於資料集中每個點x，計算其與目前已選中心中最近中心的距離平方：

$$D(x)^2 = \min_{c \in C} \|x - c\|^2$$

3. 依據每個點的$D(x)^2$分布進行機率抽樣選出下一個中心：

$$P(x) = \frac{D(x)^2}{\sum_{x' \in X} D(x')^2}$$

4. 重複直到選出 k 個中心為止。
5. 然後進行標準 K-Means 的群分與更新程序。

- **優點：**

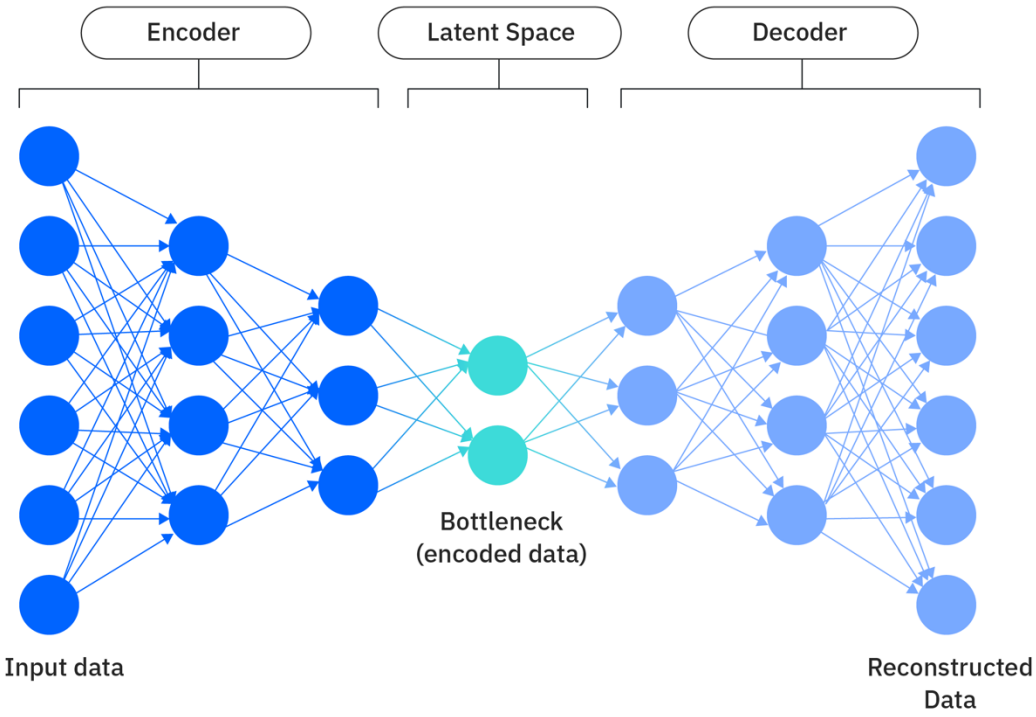
- 初始中心分布更合理，**加速收斂**。
- **結果穩定性更高**，比較不容易卡在差的局部解。
- 常被用作 K-Means 的預設初始化方法（如在 scikit-learn 中）。

2.3 深度學習分群法

2.3.1 Autoencoder（Encoder－Decoder 架構）

Autoencoder 是一種無監督學習的神經網路架構，通常用於資料的非線性降維與重建。其核心由兩部分構成：

- **Encoder**：將高維輸入資料轉換為低維潛在空間（latent space）表示。
- **Decoder**：將潛在表示還原為近似原始輸入資料。



Autoencoder-based Clustering 的基本流程為：

1. 先訓練 Autoencoder，使其學習有效的資料表示。
2. 將 Encoder 輸出的潛在特徵向量作為輸入，使用傳統分群法（如 K-Means）進行分群。

透過這種方式，可以將原始資料中高維、非線性且雜訊多的特徵，轉換為更利於分群的表徵空間，進而提升分群準確性。

Encoder 與 Decoder 的設計概念與用途

Autoencoder 是一種典型的非監督式神經網路架構，用來學習資料的**潛在特徵表示（latent representation）**。其核心設計是將輸入資料壓縮（encoding），再還原（decoding）成原始形式，並透過最小化重建誤差來學習最能代表資料本質的特徵。

- **Encoder**：將輸入資料$x \in R^d$映射到潛在空間$x \in R^k$，其中。這個映射過程可表為$z = f(x)$，其中$f(\cdot)$為多層神經網路。
- **Decoder**：試圖從潛在空間的表示z重建出原始資料$\hat{x} = g(z)$。

整體模型的訓練目標是最小化重建誤差，例如常見的 MSE（Mean Squared Error）：

$$\mathcal{L}_{recon} = \|x - \hat{x}\|^2 = \|x - g(f(x))\|^2$$

Autoencoder 的設計沒有依賴任何標籤資訊，因此非常適合用於無標註的資料處理任務，例如資料壓縮、去雜訊、特徵抽取與降維等。

Autoencoder 如何進行特徵提取與降維

Autoencoder 在 Encoder 的過程中會強迫模型以壓縮形式（潛在向量）表達輸入資料的資訊。在這個過程中：

- 網路會學習哪些資訊是保留重建品質所必需的。
- 無用或冗餘的資訊將被自動過濾掉。
- 最終得到的潛在向量z就是一組具有辨識力的**特徵表示**（representation）。

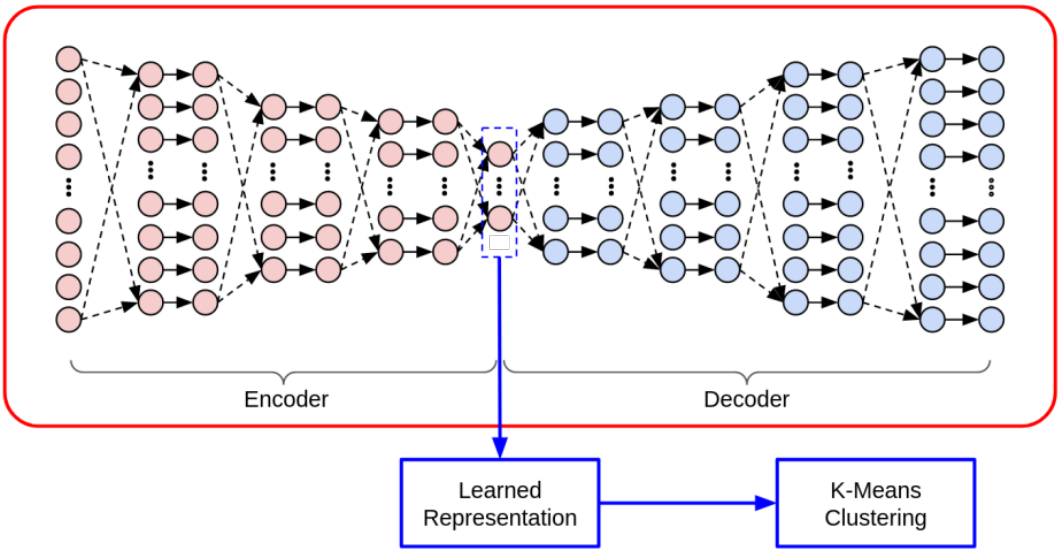
由於中間層的維度比輸入低（$k < d$），這樣的特徵提取等同於**非線性降維**，與 PCA（主成分分析）等線性降維法相比，Autoencoder 能捕捉更複雜的資料結構。

這樣的特徵空間 z 通常維度更小、結構更清晰，因此對後續任務（例如分群）非常有利。

Autoencoder 與 K-Means 的整合應用

Autoencoder 經常與 K-Means 分群演算法搭配使用，形成簡單但有效的分群流程，其步驟如下：

1. **特徵學習**：使用 Autoencoder 對原始高維資料進行壓縮與重建，並在訓練完成後，保留 Encoder 作為特徵轉換器。
2. **潛在空間表示**：將所有資料經由 Encoder 映射至潛在空間，得到$z_1, z_2, \cdots, z_n$。
3. **分群**：對這些潛在向量應用 K-Means 分群演算法，劃分資料群組。



這樣做的好處是：

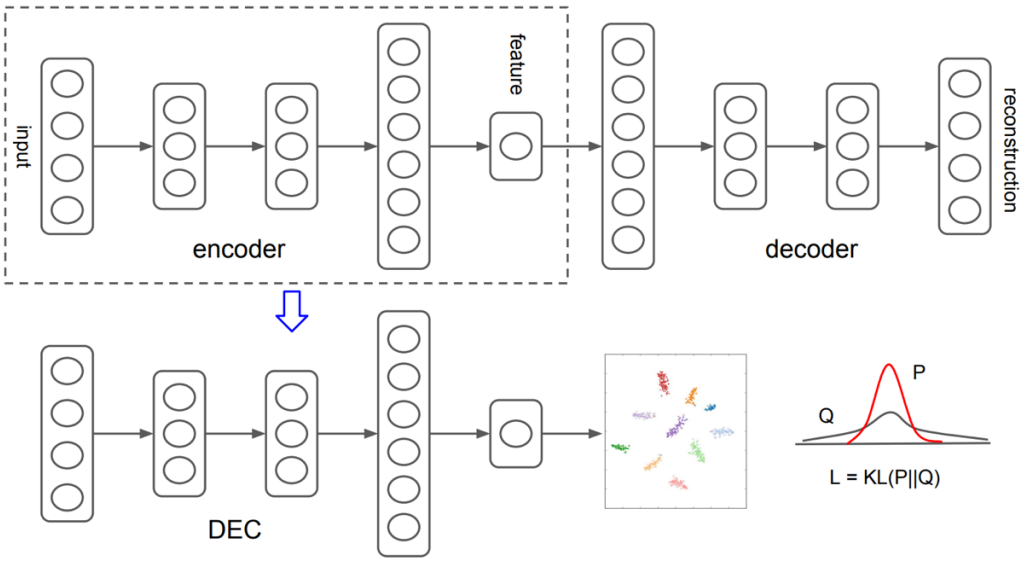
- 資料已經降維、去除雜訊，使 K-Means 不容易陷入局部最佳解。
- 經過非線性轉換後，原本在輸入空間中難以分離的群體，在潛在空間中變得更容易辨識。

2.3.2 Deep Embedded Clustering（DEC）

Deep Embedded Clustering 是一種結合 Autoencoder 與分群目標的深度學習模型，其特點在於**將特徵學習與分群同時進行優化**，克服傳統 Autoencoder 分離式訓練的侷限。

DEC 的訓練流程

1. 使用標準 Autoencoder 預訓練，獲得良好的初始化潛在空間(Latent Space)。
2. 移除 Decoder，僅保留 Encoder 作為後續分群用的特徵抽取器。。
3. 加上分群層（Clustering Layer），並使用 Kullback-Leibler 散度（KL Divergence）作為損失函數，透過自訂的目標分布（Target Distribution）不斷優化分群結果。



在 DEC 中，對每筆資料x_i，先經由 Encoder 得到潛在特徵向量$z_i \in \mathbb{R}^d$。接著以 Student's t-distribution 計算該點屬於每個分群中心μ_j的**相似度機率** q_{ij}：

$$q_{ij} = \frac{\left(1 + \|z_i - \mu_j\|^2/\alpha\right)^{-\frac{\alpha+1}{2}}}{\sum_{j'}\left(1 + \|z_i - \mu_{j'}\|^2/\alpha\right)^{-\frac{\alpha+1}{2}}}$$

，其中：

q_{ij}：第 i 筆資料屬於群集 j 的概率

z_i：第 i 筆資料的潛在表示

μ_j：分群 j 的中心(可訓練參數)

α:自由度(通常設為 1)

為了讓模型聚焦於分群結果更明確的資料，定義一個加權過的**目標分布** p_{ij}，強化高置信度的指派結果：

$$p_{ij} = \frac{q_{ij}^2 f_j}{\sum_{j'} \frac{q_{ij'}^2}{f_{j'}}}$$

，其中：

$$f_j = \sum_i q_{ij}$$

這樣的設計會強化那些較明確屬於某群的點的影響力，進而加速收斂。

DEC 的優點在於，模型會針對分群任務調整特徵空間，使相同群內樣本靠得更近、群與群之間更分離。這種端到端學習方式，有助於提升在高維資料上的聚類表現。

損失函數：Kullback-Leibler Divergence（KL Divergence）

衡量模型分佈 Q 與目標分佈 P 之間的差異

$$\mathcal{L}_{KL} = KL(P||Q) = \sum_i \sum_j p_{ij} \log\left(\frac{p_{ij}}{q_{ij}}\right)$$

透過最小化此損失，DEC 可同時調整分群中心μ_j和 Encoder 權重，使潛在空間中的資料點更聚集於適當的分群中心。

模型優勢與應用

與傳統方法相比，DEC 有以下優勢：

- 透過自動學習特徵，能適應非線性、高維度的資料分布。
- 分群與降維共同優化，可提升整體分群準確率。
- KL 散度強化聚類信心，使分群邊界更清晰。

2.3.3 Fowlkes – Mallows Index(FMI)

Fowlkes – Mallows Index 是一種用來衡量**兩個分群結果之間相似度**的指標。常見於**分群分析（Clustering）中，特別是評估非監督式學習的分類結果與真實標籤（ground truth）的一致性**。公式如下：

$$FMI = \sqrt{\frac{TP}{TP + FP} \times \frac{TP}{TP + FN}}$$

其中：

TP：True Positives（實際同群且分群結果也同群）

FP：False Positives（實際不同群但分群結果錯誤的分為同一群）

FN：False Negatives（實際同群但分群結果錯誤的分為不同群）

FMI 是一個介於 0 到 1 之間的指標，越接近 1 表示分群結果與真實分群越一致。以下是 FMI 評價標準（無絕對門檻，依任務而異）：

FMI 值範圍	解釋
0.9 – 1.0	分群結果與真實分群幾乎完美一致性
0.8 – 0.9	非常好的分群表現
0.6 – 0.8	還不錯，有部分偏差
0.4 – 0.6	中等表現，可能存在分群錯誤
0.2 – 0.4	分群與實際分類有明顯落差
0.0 – 0.2	幾乎無法反映實際標籤

三、研究方法與實作流程

3.1 數據來源與描述

本研究會使用兩筆資料集來進行實驗：

public data：4 維 49,771 筆資料

private data：6 維 200,000 筆資料

透過這兩組不同維度的資料，我們將實際測試幾種常見的分群方法，觀察它們在這個任務中的效果與差異。分群的好壞會 Folkes – Mallows Index（FMI）評分，這個指標介於 0 到 1 之間，分數越高表示我們分得越準確。

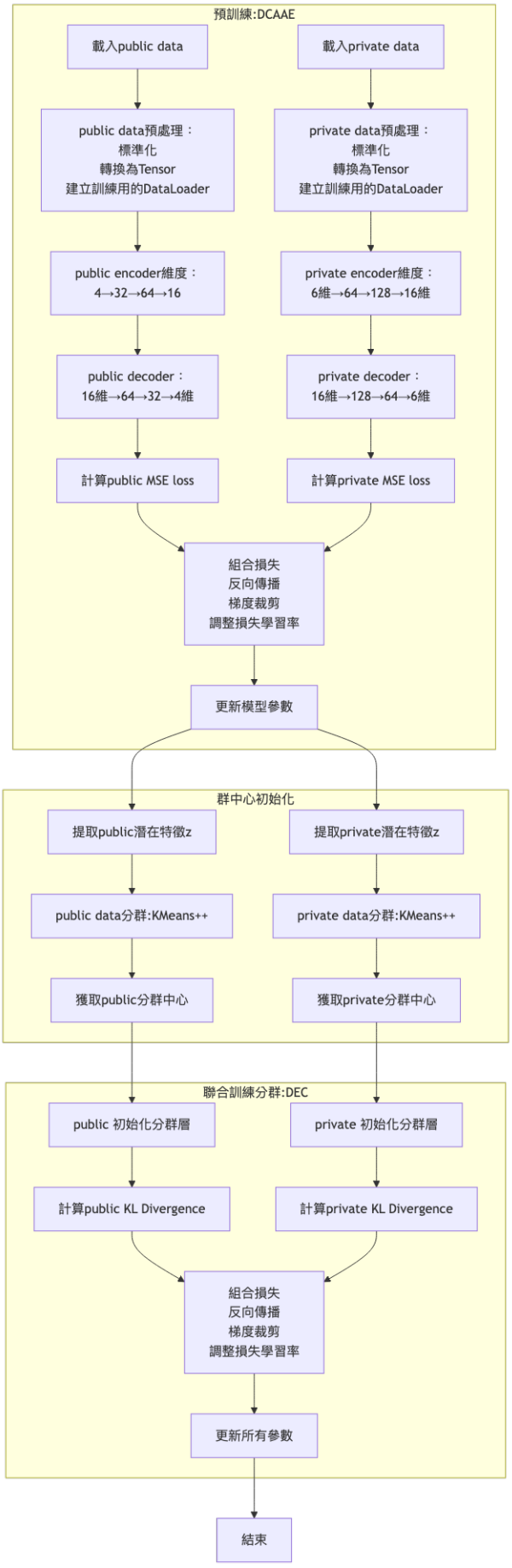
3.2 實驗環境說明

本實驗於 Google Colab 環境中進行，使用預設提供的虛擬 CPU 計算資源，未啟用 GPU。

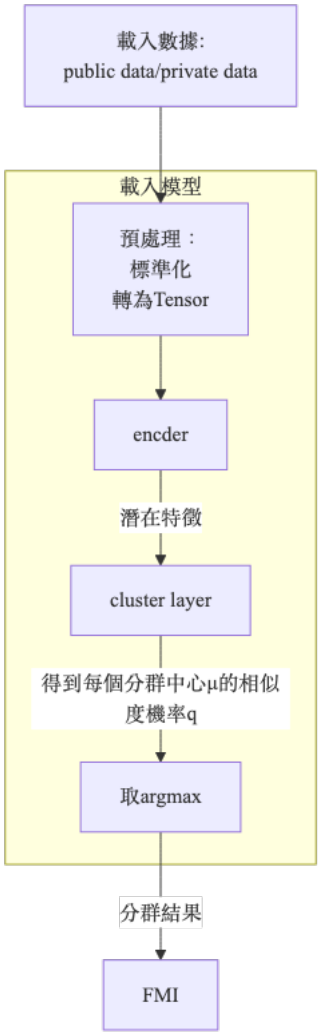
- 實際執行設備如下：
  處理器：AMD EPYC 7B12，雙執行緒單核心（2 線程）
  指令集架構：x86_64，支援 AVX2、SSE4.2 等指令集
  記憶體：12 GB（可用約 10 GB）
  運行平台：KVM 虛擬機，Linux 作業系統
- Python 版本：3.11.3
- 深度學習框架：
  PyTorch:2.6.0+cu124
- 數據處理與機器學習套件：
  NumPy 2.0.2
  pandas 2.2.2
  scikit-learn 1.6.1

3.3 實作流程

3.3.1 模型訓練流程


預訓練:DCAAE
載入public data
載入private data
public data預處理：
標準化
轉換為Tensor
建立訓練用的DataLoader
private data預處理：
標準化
轉換為Tensor
建立訓練用的DataLoader
public encoder維度：
4→32→64→16
private encoder維度：
6維→64→128→16維
public decoder：
16維→64→32→4維
private decoder：
16維→128→64→6維
計算public MSE loss
計算private MSE loss
組合損失
反向傳播
梯度裁剪
調整損失學習率
更新模型參數
群中心初始化
提取public潛在特徵z
提取private潛在特徵z
public data分群:KMeans++
private data分群:KMeans++
獲取public分群中心
獲取private分群中心
聯合訓練分群:DEC
public 初始化分群層
private 初始化分群層
計算public KL Divergence
計算private KL Divergence
組合損失
反向傳播
梯度裁剪
調整損失學習率
更新所有參數
結束

3.3.2 模型分群流程



3.5 實驗結果與分析

Github link: https://github.com/inwater0929/Final-Project-Big-Data

模型			FMI
版本	可匯入資料	可分群數	
1	public data	15	0.6749
1-調整後	public data	15	0.7256
2	public data private data	public data:15 private data:15	0.7293
3	public data private data	public data:15 private data:23	0.8948

3. 5.1 單一資料來源（V1）

初始模型僅使用 public data 進行訓練。在不調整參數的情況下，FMI 為 0.6749。
調整 Encoder 深度與預訓練迭代次數後，FMI 可提升至 0.7256，顯示特徵表徵能力有所改善。

3.5.2 雙資料來源並列訓練（V2）

由於 private data 為六維資料，無法直接與 四維的 public data 對應，因此將兩者資料各自編碼後合併訓練，並設定相同的分群數 15 群。
此方式 FMI 約略穩定在 0.7292 – 0.7293，顯示引入多元資料雖然有助提升準確度，但在維度對齊與分群數固定的限制下，模型表現仍有限。

3.5.3 分開訓練、共享 Loss 的變體架構（V3）

針對 V2 的限制，V3 改採「各自訓練 Encoder 並共享損失函數」的方式，允許 public data 與 private data 分別學習適合自身維度的表示，同時透過共享目標引導兩者同步優化。
此外，根據 4n-1 的分群原則，將 private data 的分群數提升至 23 群。此架構在資料完整度與適配性上取得平衡，FMI 明顯提升至 0.8948，顯示效果顯著。

3.6 小結與觀察

從結果來看，模型表現與三項因素密切相關：

- 資料的維度對應與特徵表達能力
- 預訓練階段對 Encoder 收斂性的影響
- 分群數設定是否符合理論預期（4n-1）

整體而言，第三版模型的設計提供一種有效策略，可應對異質維度資料並提升分群表現，未來亦可考慮應用於多模態或增量式分群任務。

四、結論與未來展望

4.1 結論

本研究以非監督式學習為核心，探討在高維異質資料情境下的分群任務，透過 Autoencoder 結合 Deep Embedded Clustering（DEC）架構，進行特徵學習與聚類分析。實驗過程中，從單一資料訓練（V1）逐步擴展至雙資料整合（V2），最終採用共享損失、分別訓練的變體模型（V3），在不同架構下比較聚類成效。

主要研究成果如下：

- 調整 Encoder 結構與訓練流程，可明顯提升特徵表徵能力與分群準確度。
- 公私資料的異質性（如維度差異）會對模型收斂與分群品質產生影響，須設計額外機制處理對齊與分工。
- 採用共享 Loss 且分別訓練的變體模型能同時解決維度對應與分群數不同的問題，FMI 成效顯著優於其他版本（最高達 0.8948）。

此實驗驗證了深度表示學習搭配分群任務的可行性，特別是在無監督、資料異質、分群數預知的條件下，提出的結構可作為類似任務的參考基準。

4.2 未來工作建議

儘管本研究已初步完成分群任務並達成預期目標，仍有數項潛在方向可供進一步探討與優化：

1. **引入多模態資料（Multimodal Fusion）**：
   若 future dataset 包含不同來源（如影像、文字、數值資料），可探討多通道 Encoder 結構，提升表徵多樣性。
2. **改用 GPU 訓練以加速模型優化流程**：
   目前實驗全程以 CPU 執行，導致訓練時間較長，特別是在預訓練與共享 loss 的架構下，需多次反覆更新 encoder 與 clustering layer。未來可透過 GPU 計算資源，加速向量計算與梯度正向傳播或反向傳播，提升實驗效率並更快速進行架構調參與模型收斂。
3. **導入動態分群數預測機制**：
   現階段模型需事先設定固定的分群數（如 4n-1），未來可考慮整合如 Dirichlet Process、X-means 或 silhouette score 為基礎的方法，自動推估合理群數，使模型更具自適應性。

總結而言，Autoencoder + DEC 架構為非監督式分群提供一套有效工具組，尤其適用於資料高維、異質或分群需求明確但無標籤的任務，具有高度延展性與實務價值。

五、參考文獻

https://arxiv.org/pdf/1511.06335
https://arxiv.org/pdf/2102.07472
https://doi.org/10.1016/j.compbiomed.2023.107570
https://arxiv.org/abs/1511.05644