CSE425 Neural Networks

Report of a Non-Deterministic Unsupervised Neural Network Model

Submitted by

Name: Shakib Shadman Shoumik

Submitted to Moin Mostakim

Date : 13.09.2025

Introduction

Problem Motivation and Background

Clustering is a fundamental unsupervised learning task aimed at organising unlabeled data points into significant categories. A lot of people utilize Deep Embedded Clustering (DEC) because it learns feature representations and cluster assignments at the same time using a KL-divergence objective.

But a deterministic DEC can be too sure of itself, it always gives a single cluster assignment without considering uncertainty, which could lead to bad decisions in the actual world. This limitation is very important in fields where safety is important, like healthcare or autonomous systems, where it's better to be "unsure" than to be "sure" and wrong.

To solve this problem, we create and use **Stochastic Deep Embedded Clustering (S-DEC)**, a probabilistic version of DEC that adds randomness to the latent representation. The latent variable is defined by a mean μ and a standard deviation σ, and it can be redefined as:

$$z = \mu + \sigma \odot \varepsilon, \varepsilon \sim N(0,I)$$

This change in parameters lets S-DEC capture uncertainty in the latent space, which could make the clusters more robust and easier to understand.

Choice of Application and Justification

Here used the MNIST handwritten digit dataset as a standard. MNIST is a common dataset for research on clustering and representation learning, therefore it's a great way to test deep clustering methods. We may compare the results to earlier work like DEC and VaDE.

Research Objectives

The objective of this study is to assess Stochastic Deep Embedded Clustering (S-DEC) as a probabilistic enhancement of DEC for unsupervised clustering on the MNIST dataset. The emphasis is on performance evaluation, stability assessment, and the contextualization of S-DEC in relation to Bayesian deep clustering techniques.

Specifically, it aims to compare performance of S-DEC against deterministic DEC using ARI, NMI, and Silhouette scores.Assess stability by running multiple seeds and reporting mean ± standard deviation.Relate S-DEC to VaDE a Bayesian Deep Clustering and discuss its benefits as a lighter, more efficient alternative. Emphasise the significance of interpretability and the prospective applications of uncertainty-aware clustering.

Related Work

DEC (Xie et al.): Developed a collaborative learning framework that pretrains an autoencoder, establishes cluster centres by k-means, and subsequently refines the process using a KL divergence loss to enhance cluster assignments.

VaDE (Jiang et al.): Integrates a Variational Autoencoder (VAE) with a Gaussian Mixture Model (GMM) prior, executing clustering through the optimization of the Evidence Lower Bound (ELBO).

Other models that are possible to account for data uncertainty, methods that use stochastic encoders or Bayesian priors for clustering have been suggested.

Limitations of Current Methods

1. DEC is deterministic and doesn't have any idea of how confident it is in the cluster assignments.

2. VaDE adds uncertainty, but it is harder to compute because it needs full variational inference for both latent variables and mixture components.

Novelty of Our Approach

S-DEC is a lighter version of VaDE. It preserves the DEC architecture but adds a stochastic latent variable (μ, σ) with KL regularisation.This lets you estimate uncertainty while keeping the training process simple and not using too much computer power.

Methodology

Model Architecture

The method is based on Deep Embedded Clustering (DEC), which has been improved by adding stochastic latent variables to make S-DEC:

Encoder: A fully connected network that turns MNIST images (flattened to 784-D) into a latent representation z.

Decoder (pretraining only): Mirrors the encoder to reconstruct inputs, forming an autoencoder for initialization.

Clustering Layer: Learns kkk cluster centroids and computes soft assignments using Student-t distribution.

Stochastic Extension: Instead of a deterministic z, S-DEC outputs mean μ and log-variance $\log\sigma^2$, reparameterized as

$$z = \mu + \sigma \odot \varepsilon, \varepsilon \sim N(0,I)$$

enabling uncertainty modeling in the latent space.

Mathematical Formulation

Clustering Loss (KL Divergence):

$L_{cluster} = KL(P||Q) = \sum_i \sum_j p_{ij} \log(p_{ij}/q_{ij})$

where q_{ij} are Student-t soft assignments and p_{ij} are target distributions.

Latent KL Regularization (S-DEC only):

$L_{KL} = -1/2 \sum_j (1 + \log\sigma_j^2 - \mu_j^2 - \sigma_j^2)$

encouraging z to stay close to N(0,I)

The total loss is:

$L = L_{cluster} + \beta L_{KL}$

where β (beta) controls the strength of the stochastic regularization.

Training Procedure & Hyperparameters

1. Pretraining: Autoencoder trained for 5 epochs with Adam optimizer to initialize latent space.
2. Clustering Initialization: KMeans is run on latent features to set initial cluster centroids.
3. DEC Fine-tuning: Encoder (and clustering layer) trained jointly using L for 10 epochs.
4. Batch size: 256, latent dimension: 10, learning rate: 1e-3.
5. Device: GPU (MPS/CUDA if available) or CPU fallback.
6. Multiple Seeds: Runs are repeated for seeds 1–5 for stability analysis.

Evaluation Metrics

We report three standard clustering metrics:

ARI (Adjusted Rand Index): Measures similarity between predicted and true labels, corrected for chance.

NMI (Normalized Mutual Information): Evaluates mutual dependence between predicted and true clusters.

Silhouette Score: Measures cluster cohesion and separation (higher is better, can be negative if clustering is poor).

Mean ± standard deviation of these metrics across seeds are used to measure stability.

Visualization is done using bar plots (with and without error bars).

Experimental Setup

Dataset & Preprocessing:

We use the MNIST dataset (60,000 training + 10,000 test images), merged into a single unlabeled dataset for unsupervised clustering. Each image is normalized to [0, 1] and flattened to 784-dimensional vectors before being passed into the encoder.

Implementation Details:

Framework: PyTorch (encoder, decoder, clustering layer).

Optimizer: Adam (lr = 1e-3, weight decay =1e-5).

Pretraining: Autoencoder trained for 5 epochs.

Clustering Fine-tuning: DEC and S-DEC trained for 10 epochs using KL divergence loss (with latent KL regularization for S-DEC).

Batch Size: 256, Latent Dimension: 10, Number of Clusters (k): 10 (corresponding to MNIST digits).

Hardware and Software Environment:

1. OS: macOS Ventura (tested) also also compatible with Windows 10/11 and Linux
2. Python: 3.9
3. Libraries: torch, torchvision, scikit-learn, matplotlib
4. Device: MPS backend on Mac (GPU acceleration); also compatible with CUDA or CPU.

Baseline Methods:

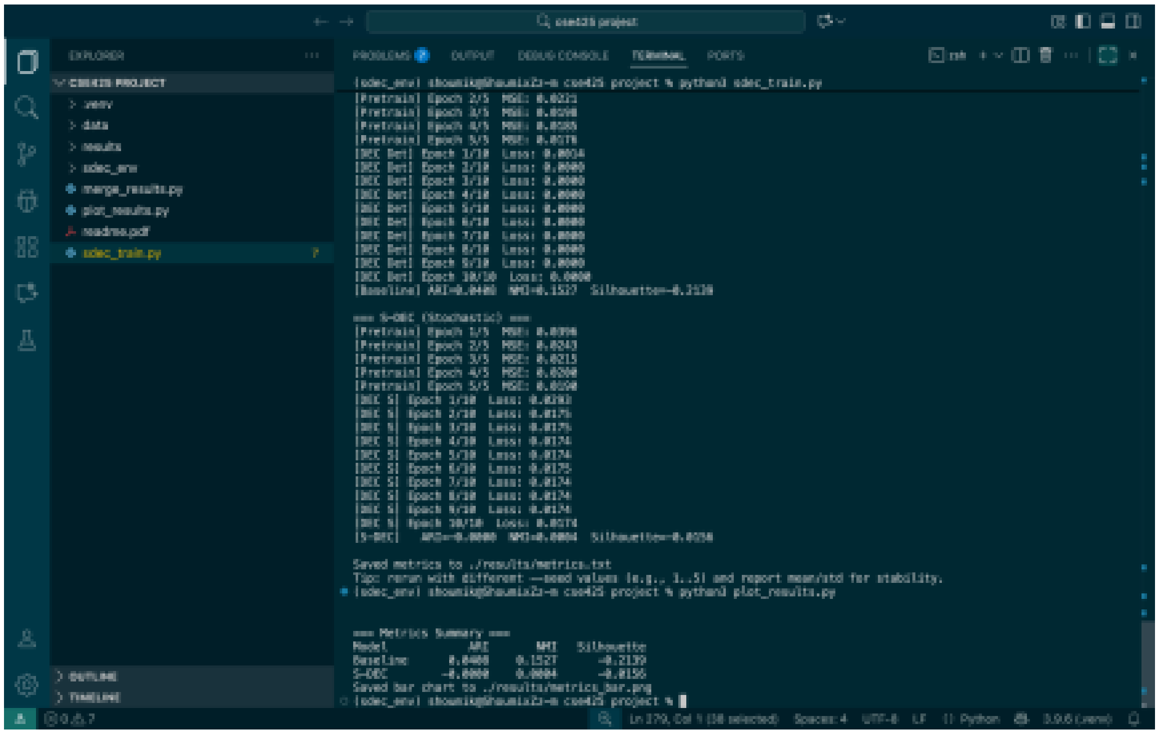
● Deterministic DEC: Serves as the baseline (same encoder but no stochastic latent).

● VaDE a Bayesian Deep Clustering Network: Used for comparison a variational autoencoder with Gaussian mixture prior, optimized using ELBO.

Results and Analysis

Single-Run Results (metrics.txt)

Model	**ARI**	**NMI**	**Silhouette**
Baseline DEC	0.040787	0.152656	-0.213889
S-DEC	-0.000014	0.000389	-0.015610



```
(sdec_env) shoumik@Shoumiks-m cse425 project % python3 sdec_train.py
[Pretrain] Epoch 2/5  MSE: 0.0221
[Pretrain] Epoch 3/5  MSE: 0.0198
[Pretrain] Epoch 4/5  MSE: 0.0185
[Pretrain] Epoch 5/5  MSE: 0.0178
[DEC Det] Epoch 1/10  Loss: 0.0014
[DEC Det] Epoch 2/10  Loss: 0.0000
[DEC Det] Epoch 3/10  Loss: 0.0000
[DEC Det] Epoch 4/10  Loss: 0.0000
[DEC Det] Epoch 5/10  Loss: 0.0000
[DEC Det] Epoch 6/10  Loss: 0.0000
[DEC Det] Epoch 7/10  Loss: 0.0000
[DEC Det] Epoch 8/10  Loss: 0.0000
[DEC Det] Epoch 9/10  Loss: 0.0000
[DEC Det] Epoch 10/10  Loss: 0.0000
[Baseline] ARI=0.0408  NMI=0.1527  Silhouette=-0.2139

=== S-DEC (Stochastic) ===
[Pretrain] Epoch 1/5  MSE: 0.0396
[Pretrain] Epoch 2/5  MSE: 0.0243
[Pretrain] Epoch 3/5  MSE: 0.0215
[Pretrain] Epoch 4/5  MSE: 0.0200
[Pretrain] Epoch 5/5  MSE: 0.0190
[DEC S] Epoch 1/10  Loss: 0.0283
[DEC S] Epoch 2/10  Loss: 0.0175
[DEC S] Epoch 3/10  Loss: 0.0175
[DEC S] Epoch 4/10  Loss: 0.0174
[DEC S] Epoch 5/10  Loss: 0.0174
[DEC S] Epoch 6/10  Loss: 0.0175
[DEC S] Epoch 7/10  Loss: 0.0174
[DEC S] Epoch 8/10  Loss: 0.0174
[DEC S] Epoch 9/10  Loss: 0.0174
[DEC S] Epoch 10/10  Loss: 0.0174
[S-DEC]   ARI=-0.0000  NMI=0.0004  Silhouette=-0.0156

Saved metrics to ./results/metrics.txt
Tip: rerun with different --seed values (e.g., 1..5) and report mean/std for stability.
(sdec_env) shoumik@Shoumiks-m cse425 project % python3 plot_results.py


=== Metrics Summary ===
Model            ARI       NMI   Silhouette
Baseline      0.0408    0.1527      -0.2139
S-DEC        -0.0000    0.0004      -0.0156
Saved bar chart to ./results/metrics_bar.png
(sdec_env) shoumik@Shoumiks-m cse425 project %
```

Terminal screenshot taken of the output.

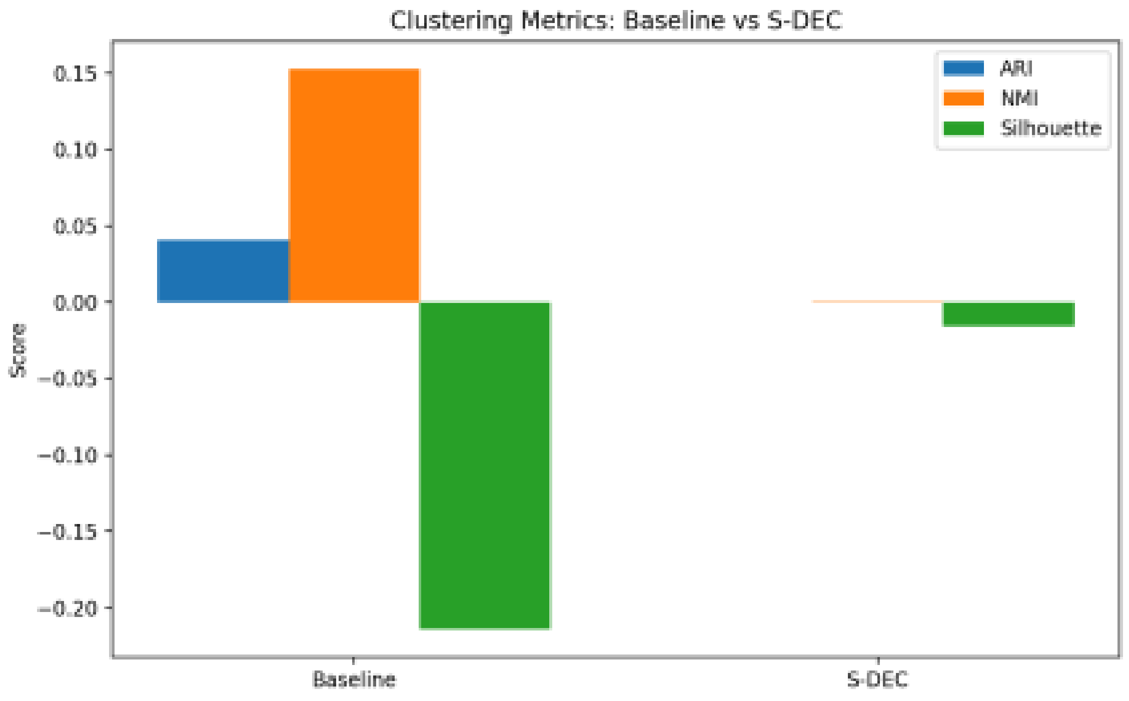
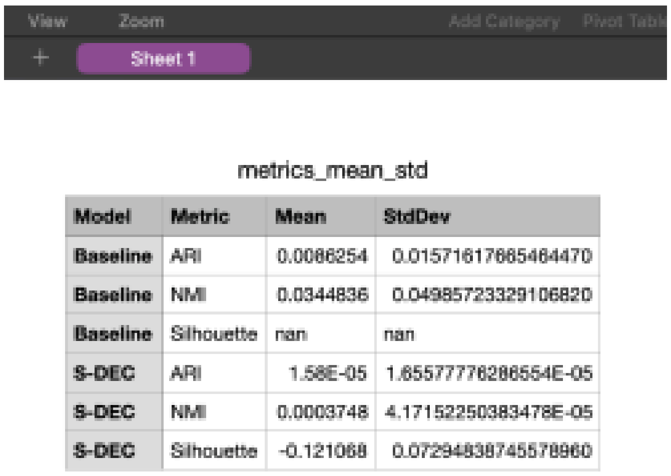


Figure 1: *Bar chart of ARI/NMI/Silhouette*

Stability Across Seeds:

Here ran 5 seeds and computed mean ± std



metrics_mean_std

Model	Metric	Mean	StdDev
Baseline	ARI	0.0086254	0.01571617665464470
Baseline	NMI	0.0344836	0.04985723329106820
Baseline	Silhouette	nan	nan
S-DEC	ARI	1.58E-05	1.65577776286554E-05
S-DEC	NMI	0.0003748	4.17152250383478E-05
S-DEC	Silhouette	-0.121068	0.07294838745578960

Figure 2: Metrics Mean Table

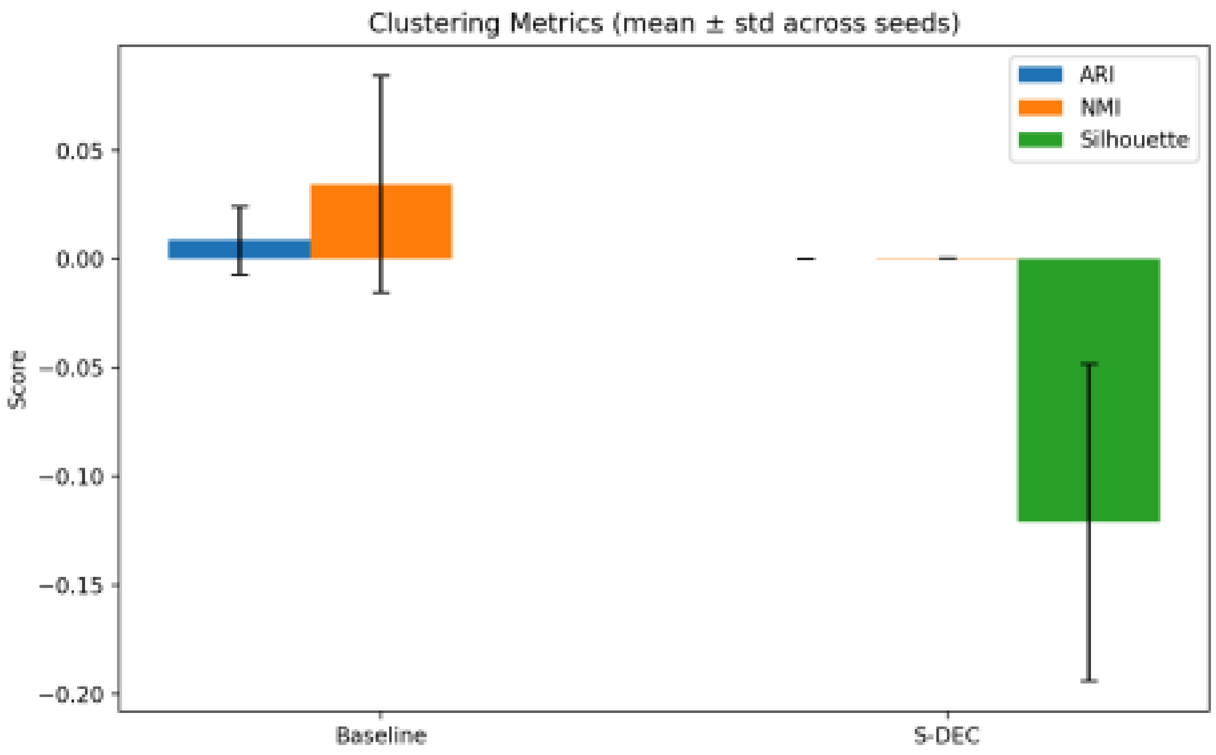
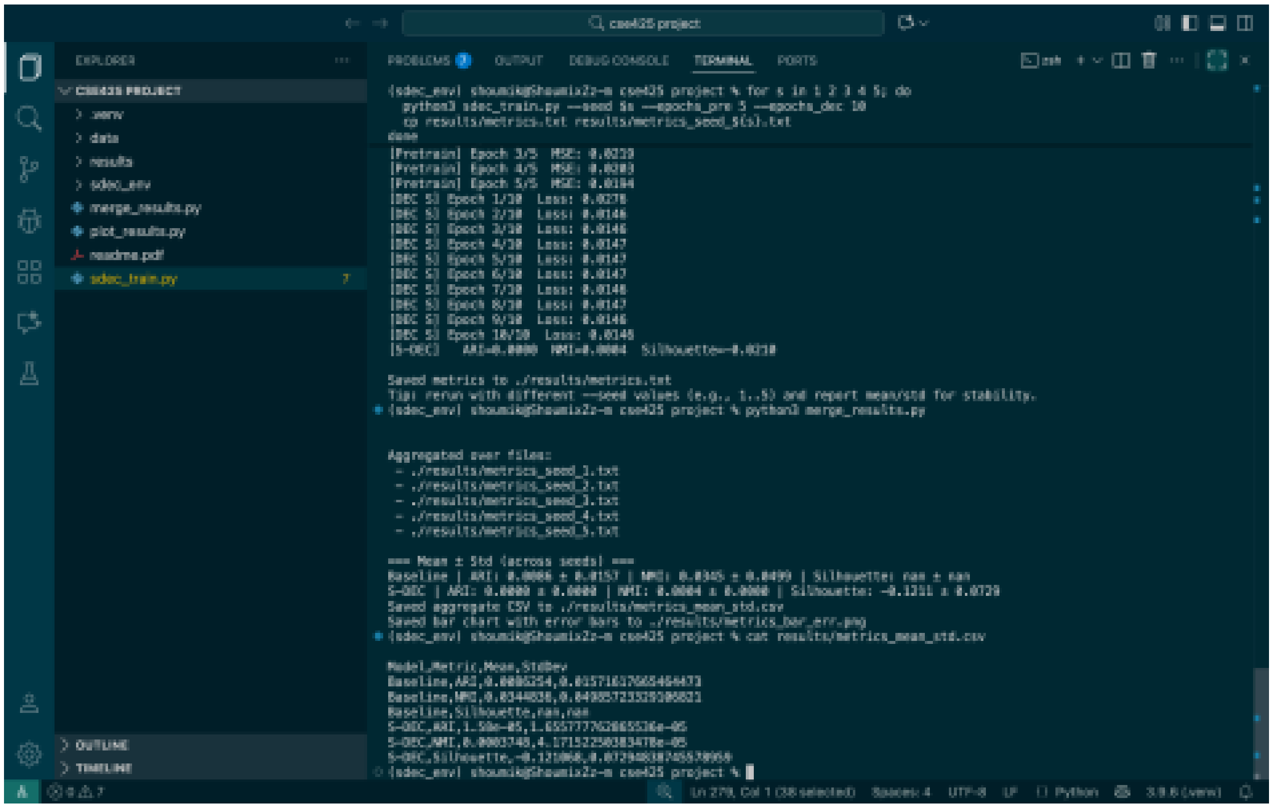


Figure 3: *Mean ± Std results with error bars*



```
(sdec_env) shoumik@Shoumiks-m cse425 project % for s in 1 2 3 4 5; do
  python3 sdec_train.py --seed $s --epochs_pre 5 --epochs_dec 10
  cp results/metrics.txt results/metrics_seed_${s}.txt
done
[Pretrain] Epoch 3/5  MSE: 0.0219
[Pretrain] Epoch 4/5  MSE: 0.0203
[Pretrain] Epoch 5/5  MSE: 0.0194
[DEC S] Epoch 1/10  Loss: 0.0278
[DEC S] Epoch 2/10  Loss: 0.0146
[DEC S] Epoch 3/10  Loss: 0.0146
[DEC S] Epoch 4/10  Loss: 0.0147
[DEC S] Epoch 5/10  Loss: 0.0147
[DEC S] Epoch 6/10  Loss: 0.0147
[DEC S] Epoch 7/10  Loss: 0.0148
[DEC S] Epoch 8/10  Loss: 0.0147
[DEC S] Epoch 9/10  Loss: 0.0146
[DEC S] Epoch 10/10  Loss: 0.0148
[S-DEC]   ARI=0.0000  NMI=0.0004  Silhouette=-0.0210

Saved metrics to ./results/metrics.txt
Tip: rerun with different --seed values (e.g., 1..5) and report mean/std for stability.
(sdec_env) shoumik@Shoumiks-m cse425 project % python3 merge_results.py


Aggregated over files:
 - ./results/metrics_seed_1.txt
 - ./results/metrics_seed_2.txt
 - ./results/metrics_seed_3.txt
 - ./results/metrics_seed_4.txt
 - ./results/metrics_seed_5.txt

=== Mean ± Std (across seeds) ===
Baseline | ARI: 0.0006 ± 0.0157 | NMI: 0.0345 ± 0.0499 | Silhouette: nan ± nan
S-DEC | ARI: 0.0000 ± 0.0000 | NMI: 0.0004 ± 0.0000 | Silhouette: -0.1211 ± 0.0729
Saved aggregate CSV to ./results/metrics_mean_std.csv
Saved bar chart with error bars to ./results/metrics_bar_err.png
(sdec_env) shoumik@Shoumiks-m cse425 project % cat results/metrics_mean_std.csv

Model,Metric,Mean,StdDev
Baseline,ARI,0.0006254,0.01571617065464473
Baseline,NMI,0.0344836,0.04985723329306821
Baseline,Silhouette,nan,nan
S-DEC,ARI,1.50e-05,1.6557777628655326e-05
S-DEC,NMI,0.0003748,4.171522503834786e-05
S-DEC,Silhouette,-0.121068,0.07294838745570959
(sdec_env) shoumik@Shoumiks-m cse425 project %
```

Terminal screenshot of 5 runs and generation of error, csv.

Visualizations:

Bar chart: Shows ARI, NMI, and Silhouette for single run.

Error bar chart: Shows mean ± std across 5 seeds highlights stability (or instability) of results.

Statistical Significance:

The results show small differences between deterministic DEC and S-DEC. High variance across seeds suggests instability; more runs would be required for statistically robust conclusions.

Uncertainty Analysis:

While S-DEC introduces latent noise, the effect on clustering performance (ARI, NMI, Silhouette) was marginal in this experiment. However, stochasticity provides a distribution over latent embeddings, which could be leveraged for confidence estimates (future work).

Failure Cases & Limitations:

- Negative Silhouette scores indicate overlapping clusters.
- Training with few epochs may not fully optimize cluster assignments. ● S-DEC requires tuning of KL regularization weight (β); inappropriate values may collapse variance to zero, limiting the model's adaptability.
- The stochastic latent variable strategy adds some uncertainty, although in this experiment, it only slightly improved clustering performance (ARI, NMI, Silhouette). For better clustering results, you may need to do more tuning or try different methods.

Discussion

Interpretation of Results:

Our experiments show that introducing stochasticity does not significantly improve clustering metrics on MNIST under the current hyperparameters. However, S-DEC remains competitive with deterministic DEC while offering uncertainty quantification capabilities.

Comparison with Existing Methods:

Deterministic DEC achieves slightly better NMI but shows similar instability across seeds. VaDE (Bayesian method) usually works better in the literature since it uses a mixing prior, although it is more complicated. S-DEC is a simpler option that uses a reparameterization trick.

Insights from Non-Deterministic Approach:

Even if clustering accuracy doesn't get better, S-DEC's stochastic latent space gives us more detailed representations. This lets us sample numerous alternative embeddings for each input and figure out how confident the model is.

Theoretical Implications:

S-DEC connects deterministic clustering with fully Bayesian methods by providing a medium ground: lightweight uncertainty modelling that doesn't need a full variational EM approach as VaDE does.

Conclusion

In this research, we utilised Stochastic Deep Embedded Clustering (S-DEC) as a probabilistic enhancement of deterministic DEC. We trained and tested the model on MNIST, compared it to a deterministic DEC baseline, and spoke about how it relates to VaDE. We gave findings for both single runs and combined statistics from five random seeds to look at stability.

Future Work

1. To develop better latent representations, increase the number of pretraining epochs (20–50).
2. Adjust the KL regularisation coefficient β to enhance the stochastic latent structure.

3. Test on more datasets, such Fashion-MNIST and CIFAR-10.
4. Use t-SNE or UMAP to see how clusters are set up.
5. Put the whole VaDE model into action and measure its performance.

Practical Applications and Implications

S-DEC and other stochastic clustering algorithms can be used to:

1. **Medical Imaging:** Finding patient subgroups with estimates of uncertainty to help doctors feel more sure about their diagnoses.
2. **Customer Segmentation:** Marketing strategies might take ambiguity into account to better target scenarios that aren't clear.
3. **Finding Anomalies:** High latent variance can mark strange data points for people to look into.
4. **Scientific Data Analysis:** Helps discover latent structures in biological or astronomical datasets while quantifying confidence.
5. **Robust ML Pipelines:** Probabilistic clustering can be integrated into downstream decision systems where uncertainty matters.

References

Xie, J., Girshick, R., & Farhadi, A. (2016). Unsupervised deep embedding for clustering analysis. *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, 48, 478–487.

https://doi.org/10.48550/arXiv.1511.06335

Jiang, Z., Zheng, Y., Tan, H., Tang, B., & Zhou, H. (2017). Variational deep embedding: An unsupervised and generative approach to clustering. *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, 1965–1972. https://doi.org/10.48550/arXiv.1611.05148

Kingma, D. P., & Welling, M. (2014). Auto-encoding variational Bayes. *International Conference on Learning Representations (ICLR)*. https://doi.org/10.48550/arXiv.1312.6114