

---

# CVQVAE: A representation learning based method for multi-omics single cell data integration

---

Tianyu Liu<sup>1,2</sup>, Grant Greenberg<sup>1</sup>, Ilan Shomorony<sup>1</sup>

<sup>1</sup> Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign

<sup>2</sup>Department of Biostatistics, Yale University

## Abstract

The rapid development of second-generation sequencing has brought about a significant increase in the amount of omics data. Integrating and analyzing these single-cell datasets is a challenging problem. In this paper, we propose a new model, called as CVQVAE, based on a cross-trained VAE, and strengthened by the Vector Quantization technique for multi-omics data integration. CVQVAE projects data vectors from different omics onto a common latent space in such a way that (1) similar cells are close in the latent space and (2) the original biological information present in each of the omics (including cell cycle and trajectory) are preserved. Our model is trained and optimized solely based on the multi-omics data and requires no additional information such as cell-type labels. We empirically demonstrate the stability and efficiency of our method in data integration (alignment) on datasets from a recent competition on Open Problems in Single Cell Analysis.

## 1 Introduction

Recent advances in single-cell high-throughput sequencing technologies have made it possible to obtain cell-specific information about thousands of cells in parallel. Single-cell multi-omics assays, in particular, can provide a multi-layered picture, magnifying the resolution at which cell heterogeneity can be studied. Typical single-cell data modalities (or “omics”) may include DNA accessibility (7), transcriptomics (RNA-seq) (5), cell surface protein measurements (31; 20). These different “views” allow multiple steps in the process of gene expression to be simultaneously probed, as illustrated in Figure 1.

While joint profiling of multiple modalities in the same single cell is now possible (6; 31), the simultaneous analysis of data from different modalities presents several challenges. First, data from different modalities will typically lie in different ambient dimensions, and the scale of the different modalities may differ significantly. Batch effects, due to differences in sample preparation and sequencing techniques, can also contribute to this difference in the data scales and distribution (19). Second, properly integrating the information from different modalities is not a straightforward task, and simple approaches such as mean removal and normalization are not able to fully “align” the different data modalities. As the cost of joint multi-modal sequencing is still very high (11), it is important to develop new computational methods that can take full advantage of the informational power of multi-modal data.

One natural approach to integrate different data modalities is to jointly map them into a single, common latent space. In the context of single-cell multi-omics data, each cell has multiple data vectors (from each of the different modalities), and our goal is to map those vectors onto a common latent space, as illustrated in Figure 2. Intuitively, these latent representations should achieve two conflicting goals: (1) the vectors representing the same cell across different modalities should be close in the latent space, and (2) the original structure of each of the data modalities, which may carry biological meaning such as cell type and cell trajectories, should be maintained (or improved

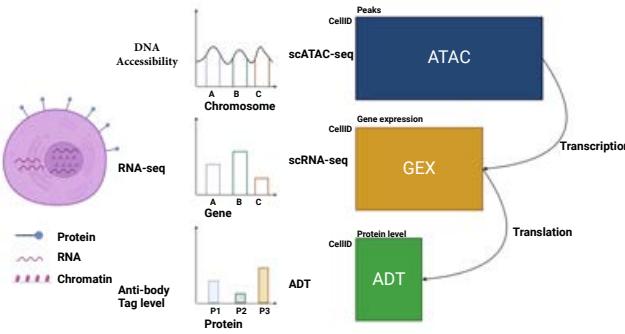


Figure 1: By combining three different single-cell data modalities – chromatin accessibility (ATAC), transcription profiling (GEX), and surface protein quantification via antibody tagging (ADT) – we obtain a comprehensive view of the gene expression process (DNA is transcribed into RNA, which is translated into protein).

by the integration). Given a common representation space, we can implement the joint analysis of multi-omics data at once, including the visualization of different type of biological data in the same space (32; 21; 14; 40), trajectory inference (36), and perturbation analysis (33). Moreover, we can also utilize the data or distribution in the integrated representation to generate multi-omics data (2) or complete the missing relationships in the gene regulatory network (4). Therefore, the integrated representation can help us understand complex multi-omics data in a more comprehensive approach.

Recently, several works have developed deep-learning-based frameworks to build an integrated representation of multi-omics single-cell data. For example, scJoint (21) utilizes neural networks with self-supervised learning, transfer learning, and a self-designed loss function to integrate data from different omics into a common domain. However, running scJoint requires cell type label information. In many cases, particularly when the goal of the multi-omics analysis is to study the cell diversity in a tissue, cell type label information will not be available, limiting the applicability of this method. GLUE (4) is another method for multi-omics data integration based on Graph Variational Autoencoders and an adversarial training strategy. The original version of GLUE requires prior information in the form of a gene regulatory network (8), which is the graph needed to perform data integration. However, most available multi-omics datasets do not include information related to gene regulatory networks, and the accuracy of this information cannot be guaranteed in general. Another method that will be very relevant to our discussion is CLUE (36), which won the NeurIPS 2021 competition on Open Problems in Single Cell Analysis (35) (17). CLUE is based on cross-training a VAE. As we will show, the method we propose has similar performance to CLUE, but with faster training and a simpler architecture. Finally, one proposed model that does not require much prior knowledge is scDART (42). This model is constructed based on a gene activity neural network module and a projection neural network module. However, the pre-processing step of scDART is relatively difficult because we need to manually compute a matrix containing regions-to-gene-activity relations, and this matrix is required as the input to scDART. The main contribution of this paper

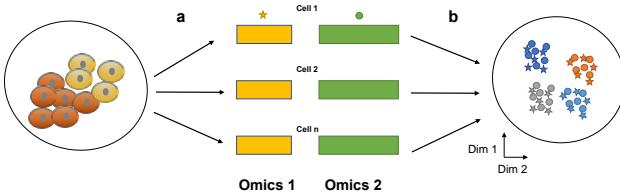


Figure 2: Integration of two data omics. For each cell and each omic, the data vector is mapped onto a common latent space in such a way that points corresponding to the same cell are nearby (circles and stars) and biological information in the original omics (such as cell type clusters) are preserved.

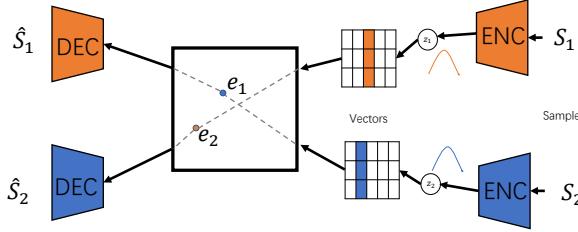


Figure 3: The high-level structure of CVQVAE. We use cross-training of the two autoencoders (i.e., the encoder of one modality is matched with the decoder of the other modality) to create the integrated latent space. In addition, vector quantization is used to improve the overall performance.

is to propose a simple and easily trainable model that uses only necessary information to integrate multi-omics data efficiently.

## 2 Methods

Our method, which we call **Cross Vector-Quantized Variational Auto Encoder (CVQVAE)**, is based on the ideas from deep learning (9; 18; 13; 37) and information theory (16). The model architecture is shown in Figure 3. The key idea is to have two auto-encoders (15) cross trained: the encoder corresponding to one modality is matched to the decoder of the other modality. This encourages the integration of the two data modalities in the same latent space. In addition, we use Vector Quantization (VQ) applied to the output of each encoder. This operation improves the overall model performance for several potential reasons: in single-cell sequencing technology, expression data of cells is measured in discrete counts, and it may be natural to model the embedding space as discrete; due to the operation of the perceptrons, continuous variables will be internally discretized, and it may be reasonable to construct the hidden space via quantization; VQ-VAE performs better than VAE by addressing the posterior collapse problem. We will verify this improvement empirically in the results section by comparing CVQVAE with cross-trained VAE (CVAE). Moreover, in the Discussion section, we will illustrate the difference between these two models from the perspective of model training.

To evaluate the quality of the resulting integrated space, we used scIB packages (24) to load benchmark functions and analyze our final results. In the data pre-processing step, we first load the datasets using Scanpy (39), and then normalize the separate modalities/omics. For scRNA-seq data and ADT data, we first extract the top principal components (PCs) to perform further analysis (28). For scATAC-seq data, we obtain the top Latent Semantic Indexing (LSI) components (32). We use these PCs and LSI components to train our model.

In the model training step, we assume that there are two datasets,  $S_1$  and  $S_2$ , whose entries are matched as in the case of two omics from the same set of single cells. In addition, we allow the presence of batch information (for both modalities  $S_1$  and  $S_2$ ) which can be viewed as another feature  $B$ , that encodes that batch number using the one-hot encoding. Based on the pair information (which vectors from  $S_1$  are matched to which vectors in  $S_2$ ), we can construct the training dataset directly and input our training set into the model shown in Figure 3. Our overall model has four neural networks in total. Two networks are encoders, known as  $E_1$  and  $E_2$ . The other two networks are decoders, known as  $D_1$  and  $D_2$ . In our training process, the output of our encoder model generates a latent vector  $z$ . We also define a discrete space  $K$  based on the embedding function in the Pytorch package (27). Here we do not use  $z$  as the input of our decoder model but instead utilize the nearest neighbor  $e_z$  in the space  $K$  to represent the input of the decoder model given  $z$ . The gradient of the vector  $z$  can be directly inherited from  $e_z$  based on the straight-estimator estimator process (3). To update the embedding space  $K$ , we utilize exponential moving averages (EMA, see Appendix A) (37) based on batch training. Finally, given an encoder  $E$ , decoder  $D$  and discrete space  $K$ , we write down the loss function of this model (VQ-VAE Loss), including reconstruction loss, VQ loss, and

commitment loss as

$$L(E, D, K) = \underbrace{\log p(x | e_z(x))}_{\text{reconstruction loss}} + \underbrace{\|\text{sg}[z(x)] - e_z\|_2^2}_{\text{VQ loss}} + \underbrace{\beta \|z(x) - \text{sg}[e_z]\|_2^2}_{\text{commitment loss}} \quad (1)$$

where  $x$  is an input sample,  $\beta$  is a hyperparameter, and  $\text{sg}(\cdot)$  is the stop-gradient operator (29), which truncates the gradient of this vector. Since we have two encoders, which generate posterior vectors  $z_1$  and  $z_2$ , ideally we would like the embedding of vectors from different modalities to be the same if they correspond to the same cell. Therefore, we would like to include a loss term  $\text{Loss}(z_1, z_2)$  that seeks to minimize the distance between  $z_1$  and  $z_2$  corresponding to the same cell. Moreover, we utilize cross-training in the decoder phase, which means that we choose to compute the reconstruction loss of  $S_1$  by decoding  $e_{z_1}$  via  $D_2$  and the reconstruction loss of  $S_2$  by decoding  $e_{z_2}$  via  $D_1$ , both based on the VQ-VAE Loss. Finally, we combine all the separate loss functions to generate the final loss function of CVQVAE:

$$\begin{aligned} L_{CVQVAE} &= \lambda_1 * L1 + \lambda_2 * L2 + \lambda_3 * L3 \\ L1 &= L(E_1, D_2, K_1), L2 = L(E_2, D_1, K_2), L3 = \text{Loss}(z_1, z_2) \end{aligned} \quad (2)$$

where  $\lambda_1, \lambda_2, \lambda_3$  are the weights of the different loss functions. Loss means a general loss function, and we choose the mean squared error loss here, which is applied to describe the difference between the embeddings from different omics data. We describe the details related to neural networks' parameters and the platform used to run our code in the 'Reproducibility' section. We also utilized GPU acceleration to increase the efficiency of our training process. To optimize our model, we utilized Adam (12), which is a well-known self-adaptive optimizer with theoretical guarantees.

Moreover, to evaluate the performance of our model on the data integration task, we utilize five metrics provided by scIB (24): Batch Average Silhouette Width (ASW) score, Graph connectivity score, Celltype ASW score, Cell-cycle conservation (CC) score, Normalized mutual information (NMI) score (38), and Trajectory conservation (TC) score. The first two benchmark metrics are used to evaluate the omics integration performance, while the last four metrics are used to evaluate the biological information conservation performance. The metrics are described as follows:

- Batch/Cell-type Average Silhouette Width (ASW): ASW is computed based on the average intra-cluster and inter-cluster distance. The inter-cluster distance of a sample is calculated between a point and points in the different clusters, while the intra-cluster distance of a sample is calculated between a point and points in the same cluster. Silhouette width (SW) is defined as:

$$SW = \frac{(d_{\text{inter}} - d_{\text{intra}})}{\max(d_{\text{inter}}, d_{\text{intra}})}. \quad (3)$$

Average silhouette width means we calculate the SW for each point and take the average as our final result. If the clusters are determined based on cell type labels, we expect that the higher Cell-type ASW value means better preservation of biological information. However, the opposite is true for Batch ASW: the lower value, the better is the batch integration. To maintain the consistency, we use  $1 - \text{ASW}$  to represent Batch ASW.

- Graph connectivity: GC evaluates whether cells with the same cell type labels are directly connected in the latent space. Let  $L$  represent the set of cell type labels. GC is given by

$$GC = \frac{1}{|L|} \sum_{c \in L} \frac{|LCC(G(N_c, E_c))|}{|N_c|}, \quad (4)$$

where  $G(N_c, E_c)$  represents a kNN subgraph containing nodes from cell type  $c$ ,  $|LCC(G(N_c, E_c))|$  indicates the number of nodes in the largest connected component (LCC) of the given graph, and  $|N_c|$  is the number of nodes with cell type label  $c$ . A higher graph connectivity value indicates better performance in omics integration.

- Cell-cycle conservation: CC score evaluates the preservation of cell-cycle variations to the expression profile before and after data integration. Cell cycle is a key concept in cell division, and contains a G1 phase, an S phase and an M phase, each of which has unique marker genes (30). The principle is that the variance contribution of the S and G2/M phase scores used to perform the principal component regression towards the raw data should

be retained in the embedding space after data integration. The phase scores were already provided in our experiment dataset, and the formula to calculate the CC score is

$$\text{CC score} = 1 - \frac{|\text{Var}_{\text{before}} - \text{Var}_{\text{after}}|}{\text{Var}_{\text{before}}}. \quad (5)$$

A higher CC score indicates better preservation of biological information.

- Normalized mutual information: NMI quantifies the shared information between two clusterings. We utilize this method to evaluate the performance of cell-type label preservation by comparing Louvain clusters ( $L$ ) computed on the embedding after data integration and the original cell type labels ( $O$ ) from the experiment dataset. The NMI score can be expressed as

$$NMI(L, O) = \frac{MI(L, O)}{GM(H(L), H(O))} \quad (6)$$

where  $MI()$  represents the mutual information of two distributions generated based on  $L$  and  $O$ ,  $H()$  is the entropy function and  $GM()$  represents generalized mean. A higher NMI score indicates better preservation of biological information.

- Trajectory conservation: TC evaluates the preservation of cell state information after data integration. We compute the trajectories based on Scanpy (39) again on our embeddings and compare the new results with trajectory information in the original data. We calculate Spearman’s rank correlation coefficient  $s$  based on the two trajectories and normalize it to compute the final TC score, defined as

$$TC(s) = \frac{s + 1}{2}. \quad (7)$$

A higher TC indicates better preservation of biological information.

### 3 Results

We used the datasets provided by the organizers of the Open Problems competition to test our model (23). These datasets contain two types of multi-omics data. The first type combines single-cell RNA sequencing (scRNA-seq) with Assay for Transposase-Accessible Chromatin using sequencing (scATAC-seq) data, obtained via the 10X Multiome assay. The second type combines scRNA-seq data with Anti-body Tag (ADT) data, obtained via the CITE-seq assay (31). All of the data are from human cells. We also used one multi-omics dataset (25) (Ma dataset), which is from Mouse cells. In these datasets, scATAC-seq data represent DNA accessibility information, scRNA-seq data represent gene expression information, and ADT data represent surface protein information. The combination ‘scRNA-seq+scATAC-seq’ means we have scRNA-seq data and scATAC-seq data from the same cell with the cell pairing. In this paper, we focus on the first type of dataset (10X Multiome) from the Open Problems competition. Our model for multi-omics data integration is quite flexible, and we only need to change the input dimensions of our model to apply to the other datasets. For the integration results on the CITE-seq dataset and Ma dataset, we refer to Appendix B.

#### 3.1 Results on 10X Multiome Data

Using the framework provided by the Open Problems competition (23), we ran our model on ‘scRNA-seq+scATAC-seq’ dataset and generated the new representation for this high-dimensional multi-omics dataset. We then fed the embeddings generated on the low-dimensional space into our benchmark evaluation metrics. We evaluated the model by comparing the results with raw data (PCA+LSI) and some models focusing on data integration. Our benchmark models include scJoint (21), CLUE (GLUE) (36), scDART (42), and CVAE. To demonstrate our results visually, we chose UMAP (26) as a dimensionality reduction tool to help us display the results. The results, including raw data embeddings and integrated embeddings, are shown in Figure 4.

From Figure 4, we verify that our method can successfully integrate scATAC-seq and scRNA-seq data. For the diversity of cell type labels, our method ensures that cells from different omics with same cell types are integrated together without obvious batch effects. For example, T cells are integrated with T cells, while B cells are integrated with B cells.

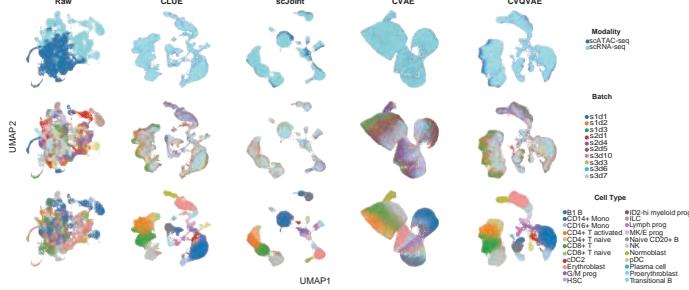


Figure 4: Embeddings from raw data and different models

We evaluate our results based on the metrics discussed in the "Methods" section. We divide the score of different metrics into Batch group and Cell-type group. Considering the importance of biological information, we assign different weights to the two sets of metrics. The final score in Tables 1 and 2 is computed based on the following rule from: (24):

$$\text{Score} = 0.4 \cdot \text{Score}_{\text{Batch}} + 0.6 \cdot \text{Score}_{\text{Cell type}} \quad (8)$$

where  $\text{Score}_{\text{Batch}}$  represents the average of batch related scores and  $\text{Score}_{\text{Cell type}}$  represents the average of cell-type related scores. We run each method four times and take the average as final score. The score of all different models can be found in Table 1. We use OOM to indicate when we ran out of memory while running the model. From Table 1, we see that our model, CVQVAE, achieves the highest score for NMI, CC score, graph connectivity score, and the overall score, which are metrics used for evaluating biological information preservation and omics integration. Moreover, although the final score of CVQVAE and CLUE are close, the running time of our model is about 600 seconds for a dataset containing over 40,000 cells, while it takes over an hour (4431 seconds) for CLUE to finish. For Ma dataset, CVQVAE also reaches the highest final score comparing with other benchmark methods (Appendix B, Table 2). These results provide evidence that CVQVAE can retain more biologically meaningful data than other methods while integrating the data from different omics efficiently. We also tested the variations of CVQVAE, and results can be found in Appendix C.

Table 1: Overall score of different models

Methods	NMI	Celltype ASW	CC	TC	Batch ASW	Graph connectivity	Score
Raw	0.415	0.446	0.430	0.284	0.888	0.597	0.533
scJoint	0.767	0.548	0.344	0.819	<b>0.952</b>	0.920	0.746
CLUE	<b>0.780</b>	<b>0.591</b>	0.847	<b>0.887</b>	0.870	0.963	0.832
scDART	OOM	OOM	OOM	OOM	OOM	OOM	OOM
CVAE	0.474	0.488	0.625	0.823	0.788	0.855	0.690
CVQVAE	<b>0.780</b>	0.569	<b>0.883</b>	0.880	0.883	<b>0.982</b>	<b>0.840</b>

### 3.2 Biological information analysis

It is important to verify that the model output can retain biological information, so in this part, we evaluate the preservation of biological information in gene expression data and pseudotime analysis.

For the gene expression analysis, we know that each cell type has marker genes based on protein-encoding genes with high or low expression. For example, for B cells, we have BANK1 as one marker gene (10), while CD4 T cells have the INPP4B as one marker gene (41). In Appendix D, we present results based on differential gene expression analysis.

The marker genes can be utilized to perform cell type annotation for each cell. The dataset we used has different batches, which were previously annotated separately. Our results suggest that performing annotation on the data after data integration is advantageous. In Figure 5, we see that, for the BANK1 gene, cells with high expression are clustered together, and the cell type label of this cluster is B cell. Similarly, for gene INNP4B, the highly expressed cells were not perfectly clustered together due to the diversity of T cells differentiation, but two major cell groups could still be observed. Therefore, the prior knowledge is preserved when we perform the multi-omics data

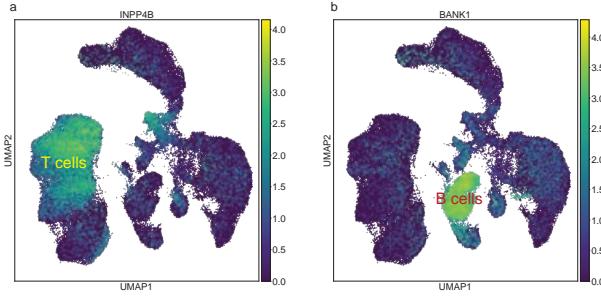


Figure 5: Differential expression visualization. a) Cells in the integrated space are colored by INPP4B expression distribution. b) Cells in the integrated space are colored by BANK expression.

integration, and the cell type annotation process can be applied directly to the clusters generated by the embedding from CVQVAE.

For the pseudotime analysis, or equivalently cell state analysis, we are interested in assessing developmental cell trajectories in the embedding space. These trajectories could be observed on both data sets before multi-omics data integration. Intuitively, we would hope that the trajectory information could still be detected after integration. To verify this, we visually assess the trajectories before and after integration. In Figures 7a and 7b, we see that, before the multi-omics data integration, the trajectories are somewhat dispersed due to the batch effects. After multi-omics data integration, the differentiation trend of cells can be observed in the upper right of Figure 6c. We see that after completing multi-omics data integration, our model retains the trajectory information of cell differentiation and reduces batch effects, making the distribution of related cells more concentrated and making it straightforward to determine the path of cell differentiation. This provides evidence that our model can facilitate trajectory analysis of multi-omics data.

## 4 Discussion

We proposed a quantized VAE model using the cross training strategy to integrate multi-omics single-cell data. Through experiments, we showed that the embeddings generated by our model can eliminate the batch effect among different omics and retain the biological information of data to the maximum extent. Next we provide additional discussion on three points: the selection of the auto-encoder structure, the model parameter selection, and benchmark selection.

### 4.1 Choice of auto-encoder structure

We firstly considered a variational auto-encoder (VAE) for this task. The VAE encoder provides a more robust generalization performance compared with auto-encoder. The encoding generated by a VAE is technically a distribution, and we obtain embedding from the distribution by sampling. However, the original form of VAE suffers from the problem caused by posterior collapse, which means that if the latent space representation (the output of the encoder model) used as decoder

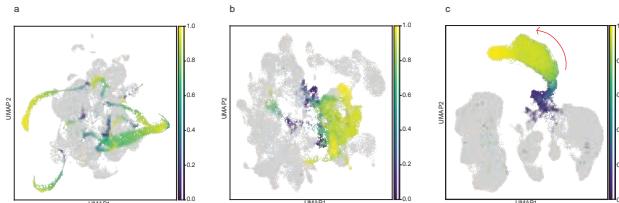


Figure 6: Pseudotime analysis: a) Cell state information on the original (pre-integration) scATAC-seq data. b) Cell state information on the original (pre-integration) scRNA-seq data. c) Cell state information on integrated space. The red arrow indicates the direction of cellular growth.

input are not high-quality (too weak or too noisy), the decoder will prefer to ignore the latent space representation and generate output based on the autoregression property (1). If this happens, the embeddings will lose their representative information.

In addition, we cannot guarantee an effective training process for the VAE model (Appendix E). Since the source of such collapse is the Gaussian distribution assumption of the VAE model and the loss function of this model (34), it is a natural choice to consider modifying the representation of latent space and the loss function. We do that by considering adding a vector quantization operation. In Appendix E, we give a theoretical justification that VQVAE can solve posterior collapse. VQ-VAE utilizes a quantized latent space to increase its flexibility, and VQ-VAE also modifies the loss function to make the training process more manageable. Moreover, for VQ-VAE-based cross-trained model, we can train it more smoothly (Appendix E). Therefore, we decided to utilize the VQ-VAE-based cross model to integrate the multi-omics data.

## 4.2 Parameter combination

Parameter tuning is an essential step in deep learning methods. The right combination of parameters can significantly improve performance. For our CVQVAE, the hyper-parameters to be adjusted are training epoch (default: 200), batch size (default: 1024), learning rate (default: 1e-4), the dimensions of the embedding space (default: 2048), and the set of weighted parameters used to control the loss:  $\lambda_{c1}$  (default: 1),  $\lambda_{c2}$  (default: 1),  $\lambda_z$  (default: 10).

## 4.3 Benchmark selection

In this paper, we focused on developing a novel tool for multi-omics data integration. The integration step of different omics data is essential, but preserving the biological characteristics that are specific to each type of data is crucial. Therefore, we chose more metrics and larger weights focusing on biological information conservation in our evaluation process, and we assigned a larger weight for these scores when computing the final score.

## 5 Conclusion

In this paper, we designed a deep learning based multi-omics data integration method, known as CVQVAE. We have demonstrated the efficiency and good performance of our design through experiments and have justified some model choices theoretically. The experimental results show that our model can preserve certain types of biological information better than other state-of-the-art benchmark methods based on different metric scores. Moreover, based on gene expression and pseudotime analysis, we demonstrated potential benefits of utilizing CVQVAE as a first step before annotation pipelines and cell trajectory analysis. We believed that this model can also integrate other data with paired information. As a future research direction, we plan to further explore the advantages of performing downstream biological analysis on the integrated space created from multi-omics datasets as opposed to working with single modality datasets. Moreover, we wish to investigate the potential of integrated multi-omics datasets to uncover new research insights for disease detection and prevention.

## 6 Reproducibility

All analyses were performed on High Performance Computing (HPC) clusters. The CPU used is an Intel(R) Xeon(R) Gold 5222, 2.6 GHz, the GPU is NVIDIA GeForce RTX3090, and the RAM is 30GB. The codes of this paper can be found in <https://github.com>HelloWorldLY/CVQVAE>.

## 7 Funding

The work of Grant Greenberg and Ilan Shomorony was supported in part by the National Science Foundation CAREER Award under Grant CCF-2046991.

## References

- [1] Anirudh Goyal ALIAS PARTH GOYAL, Alessandro Sordoni, Marc-Alexandre Côté, Nan Rosemary Ke, and Yoshua Bengio. Z-forcing: Training stochastic recurrent networks. *Advances in neural information processing systems*, 30, 2017.
- [2] Tal Ashuach, Mariano I Gabitto, Michael I Jordan, and Nir Yosef. Multivi: deep generative model for the integration of multi-modal data. *bioRxiv*, 2021.
- [3] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.
- [4] Zhi-Jie Cao and Ge Gao. Multi-omics integration and regulatory inference for unpaired single-cell data with a graph-linked unified embedding framework. *bioRxiv*, 2021.
- [5] Geng Chen, Baitang Ning, and Tieliu Shi. Single-cell rna-seq technologies and related computational data analysis. *Frontiers in genetics*, 10:317, 2019.
- [6] Song Chen, Blue B Lake, and Kun Zhang. High-throughput sequencing of the transcriptome and chromatin accessibility in the same cell. *Nature biotechnology*, 37(12):1452–1457, 2019.
- [7] M Ryan Corces, Alejandro E Trevino, Emily G Hamilton, Peyton G Greenside, Nicholas A Sinnott-Armstrong, Sam Vesuna, Ansuman T Satpathy, Adam J Rubin, Kathleen S Montine, Beijing Wu, et al. An improved atac-seq protocol reduces background and enables interrogation of frozen tissues. *Nature methods*, 14(10):959–962, 2017.
- [8] Eric Davidson and Michael Levin. Gene regulatory networks. *Proceedings of the National Academy of Sciences*, 102(14):4935–4935, 2005.
- [9] Gökçen Eraslan, Lukas M Simon, Maria Mircea, Nikola S Mueller, and Fabian J Theis. Single-cell rna-seq denoising using a deep count autoencoder. *Nature communications*, 10(1):1–14, 2019.
- [10] Gonzalo Gómez Hernández, María Morell, and Marta E Alarcón-Riquelme. The role of bank1 in b cell signaling and disease. *Cells*, 10(5):1184, 2021.
- [11] Hani Jieun Kim, Yingxin Lin, Thomas A Geddes, Jean Yee Hwa Yang, and Pengyi Yang. Citefuse enables multi-modal analysis of cite-seq data. *Bioinformatics*, 36(14):4137–4143, 2020.
- [12] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015. URL: <http://arxiv.org/abs/1412.6980>.
- [13] Diederik P Kingma and Max Welling. An introduction to variational autoencoders. *arXiv preprint arXiv:1906.02691*, 2019.
- [14] Anna Klimovskaia, David Lopez-Paz, Léon Bottou, and Maximilian Nickel. Poincaré maps for analyzing complex hierarchies in single-cell data. *Nature communications*, 11(1):1–9, 2020.
- [15] Mark A Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AICHE journal*, 37(2):233–243, 1991.
- [16] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical review E*, 69(6):066138, 2004.
- [17] Christopher Lance, Malte D. Luecken, Daniel B. Burkhardt, Robrecht Cannoodt, Pia Rautenstrauch, Anna Laddach, Aidyn Ubingazhibov, Zhi-Jie Cao, Kaiwen Deng, Sumeer Khan, Qiao Liu, Nikolay Russkikh, Gleb Ryazantsev, Uwe Ohler, NeurIPS 2021 Multimodal data integration competition participants, Angela Oliveira Pisco, Jonathan Bloom, Smita Krishnaswamy, and Fabian J. Theis. Multimodal single cell data integration challenge: Results and lessons learned. In Douwe Kiela, Marco Ciccone, and Barbara Caputo, editors, *Proceedings of the NeurIPS 2021 Competitions and Demonstrations Track*, volume 176 of *Proceedings of Machine Learning Research*, pages 162–176. PMLR, 06–14 Dec 2022. URL: <https://proceedings.mlr.press/v176/lance22a.html>.

- [18] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.
- [19] Jeffrey T Leek, Robert B Scharpf, Héctor Corrada Bravo, David Simcha, Benjamin Langmead, W Evan Johnson, Donald Geman, Keith Baggerly, and Rafael A Irizarry. Tackling the widespread and critical impact of batch effects in high-throughput data. *Nature Reviews Genetics*, 11(10):733–739, 2010.
- [20] Lei Li, Haley L. Dugan, Christopher T. Stamper, Linda Yu-Ling Lan, Nicholas W. Asby, Matthew Knight, Olivia Stoicek, Nai-Ying Zheng, Maria Lucia Madariaga, Kumaran Shanmugarajah, Maud O. Jansen, Siriruk Changrob, Henry A. Utset, Carole Henry, Christopher Nelson, Robert P. Jedrzejczak, Daved H. Fremont, Andrzej Joachimiak, Florian Krammer, Jun Huang, Aly A. Khan, and Patrick C. Wilson. Improved integration of single-cell transcriptome and surface protein expression by linq-view. *Cell Reports Methods*, 1(4):100056, 2021. URL: <https://www.sciencedirect.com/science/article/pii/S2667237521001041>, doi: <https://doi.org/10.1016/j.crmeth.2021.100056>.
- [21] Yingxin Lin, Tung-Yu Wu, Sheng Wan, Jean YH Yang, Wing H Wong, and YX Wang. scjoint integrates atlas-scale single-cell rna-seq and atac-seq data with transfer learning. *Nature Biotechnology*, pages 1–8, 2022.
- [22] Qiao Liu, Shengquan Chen, Rui Jiang, and Wing Hung Wong. Simultaneous deep generative modelling and clustering of single-cell genomic data. *Nature machine intelligence*, 3(6):536–544, 2021.
- [23] Malte D Luecken, Daniel Bernard Burkhardt, Robrecht Cannoodt, Christopher Lance, Aditi Agrawal, Hananeh Aliee, Ann T Chen, Louise Deconinck, Angela M Detweiler, Alejandro A Granados, et al. A sandbox for prediction and integration of dna, rna, and proteins in single cells. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.
- [24] Malte D Luecken, Maren Büttner, Krissadakorn Chaichoompu, Anna Danese, Marta Interlandi, Michaela F Müller, Daniel C Strobl, Luke Zappia, Martin Dugas, Maria Colomé-Tatché, et al. Benchmarking atlas-level data integration in single-cell genomics. *Nature methods*, 19(1):41–50, 2022.
- [25] Sai Ma, Bing Zhang, Lindsay M LaFave, Andrew S Earl, Zachary Chiang, Yan Hu, Jiarui Ding, Alison Brack, Vinay K Kartha, Tristan Tay, et al. Chromatin potential identified by shared single-cell profiling of rna and chromatin. *Cell*, 183(4):1103–1116, 2020.
- [26] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.
- [27] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.
- [28] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.
- [29] Chengjie Qin and Florin Rusu. Speculative approximations for terascale distributed gradient descent optimization. In *Proceedings of the Fourth Workshop on Data analytics in the Cloud*, pages 1–10, 2015.
- [30] Jonathan M Scholey, Ingrid Brust-Mascher, and Alex Mogilner. Cell division. *Nature*, 422(6933):746–752, 2003.
- [31] Marlon Stoeckius, Christoph Hafemeister, William Stephenson, Brian Houck-Loomis, Pratip K Chattopadhyay, Harold Swerdlow, Rahul Satija, and Peter Smibert. Simultaneous epitope and transcriptome measurement in single cells. *Nature methods*, 14(9):865–868, 2017.

- [32] Tim Stuart, Andrew Butler, Paul Hoffman, Christoph Hafemeister, Efthymia Papalexis, William M Mauck III, Yuhan Hao, Marlon Stoeckius, Peter Smibert, and Rahul Satija. Comprehensive integration of single-cell data. *Cell*, 177(7):1888–1902, 2019.
- [33] Indhupriya Subramanian, Srikant Verma, Shiva Kumar, Abhay Jere, and Krishanpal Anamika. Multi-omics data integration, interpretation, and its application. *Bioinformatics and biology insights*, 14:1177932219899051, 2020.
- [34] Yuhta Takida, Wei-Hsiang Liao, Toshimitsu Uesaka, Shusuke Takahashi, and Yuki Mitsufuji. Preventing posterior collapse induced by oversmoothing in gaussian vae. *arXiv preprint arXiv:2102.08663*, 2021.
- [35] Competition Organizing Team. Competition for multimodal single-cell data integration. [https://openproblems.bio/neurips\\_2021/](https://openproblems.bio/neurips_2021/), 2021. Accessed: 2022-9-23.
- [36] Xinming Tu, Zhijie Cao, Chenrui Xia, Sara Mostafavi, and Ge Gao. Cross-linked unified embedding for cross-modality representation learning. In *Thirty-sixth Conference on Neural Information Processing Systems*, 2022.
- [37] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.
- [38] Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *Proceedings of the 26th annual international conference on machine learning*, pages 1073–1080, 2009.
- [39] F Alexander Wolf, Philipp Angerer, and Fabian J Theis. Scanpy: large-scale single-cell gene expression data analysis. *Genome biology*, 19(1):1–5, 2018.
- [40] Karren Dai Yang, Anastasiya Belyaeva, Saradha Venkatachalam, Karthik Damodaran, Abigail Katcoff, Adityanarayanan Radhakrishnan, GV Shivashankar, and Caroline Uhler. Multi-domain translation between single-cell imaging and sequencing data using autoencoders. *Nature communications*, 12(1):1–10, 2021.
- [41] Feng Zhang, Kai Zhu, Lin Liu, Junfeng Zhu, Jiajia Li, Pingping Zhang, Zhongli Hu, and Yuan Yuan. Irf2-inpp4b axis inhibits apoptosis of acute myeloid leukaemia cells via regulating t helper 1/2 cell differentiation. *Cell biochemistry and function*, 38(5):582–590, 2020.
- [42] Ziqi Zhang, Chengkai Yang, and Xiuwei Zhang. scdарт: integrating unmatched scRNA-seq and scATAC-seq data and learning cross-modality relationship simultaneously. *Genome biology*, 23(1):1–28, 2022.

## A Update the discrete dependent variable space via EMA

We can use this algorithm to update  $e$  in the space  $K$ . Now we consider the set  $z_{i,1}, z_{i,2}, \dots, z_{i,k_i}$  as the set of  $k_i$  samples from the encoder which are closed to  $e_i$ , and to update  $e_i$ , we can minimize the loss:

$$L = \sum_j^{k_i} \|z_{i,j} - e_i\|^2$$

Therefore, if we consider the cumulant of  $k_i$  at time  $t$  as  $K_i^{(t)}$ , and the cumulant of  $z_i$  at time  $t$  as  $Z_i^{(t)}$ , we can update  $K_i^{(t)}, Z_i^{(t)}, e_i^{(t)}$  using:

$$\begin{aligned} K_i^{(t)} &:= K_i^{(t-1)} * \gamma + k_i^{(t)}(1 - \gamma) \\ Z_i^{(t)} &:= Z_i^{(t-1)} * \gamma + \sum_j z_{i,j}^{(t)}(1 - \gamma) \\ e_i^{(t)} &:= \frac{Z_i^{(t)}}{K_i^{(t)}}, \end{aligned}$$

where  $\gamma \in (0, 1)$  and we choose  $\gamma = 0.99$  in our model.

## B Multi-omics data integration results

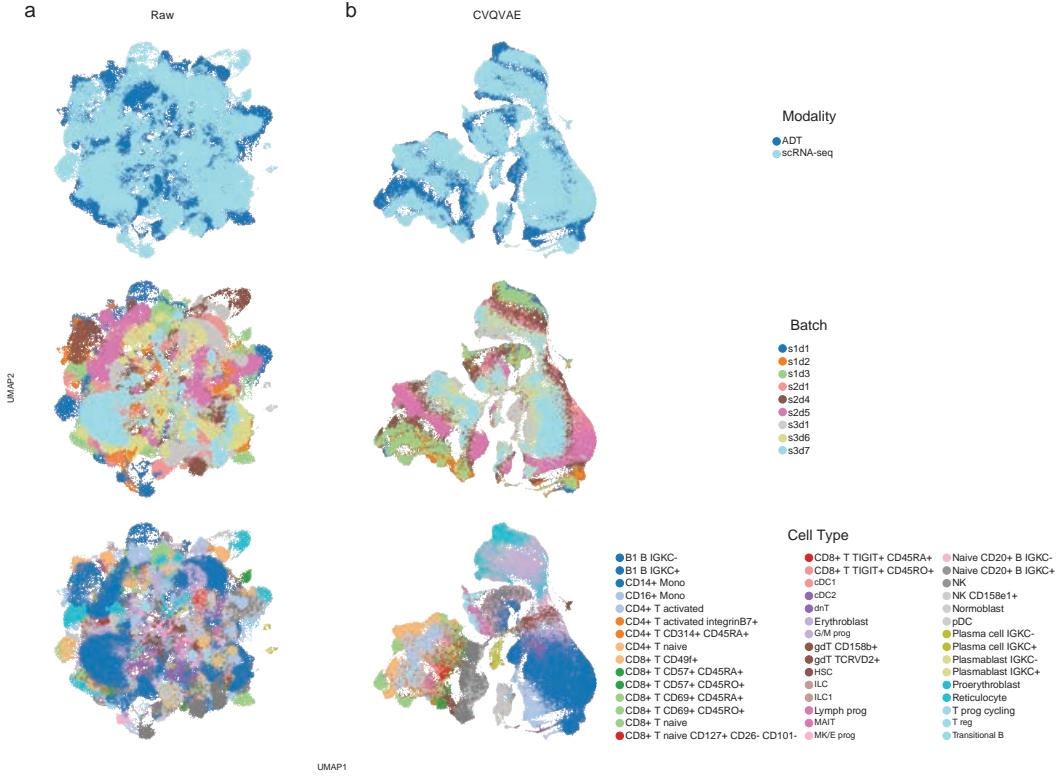


Figure 7: Protein embeddings data from raw data and CVQVAE

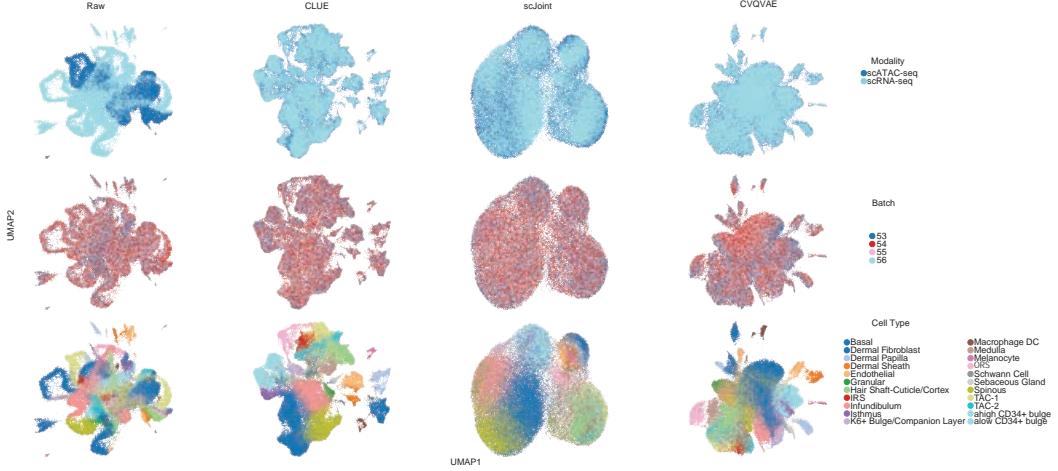


Figure 8: Embeddings data from raw data and different models of Ma dataset

Table 2: Overall score of different models on Ma dataset

Methods	NMI	Celltype ASW	CC	TC	Batch ASW	Graph connectivity	Score
Raw	0.582	0.491	0.661	0.484	0.964	0.730	0.671
scJoint	0.473	0.501	0.916	0.240	0.969	0.852	0.683
CLUE	<b>0.689</b>	<b>0.562</b>	<b>0.943</b>	0.477	<b>0.975</b>	<b>0.958</b>	0.787
CVQ-VAE	0.528	0.501	<b>0.943</b>	<b>0.846</b>	0.959	0.939	<b>0.802</b>

## C Exploring variations of CVQVAE

When we designed our model, we considered several variants, including the removal of the vector quantization, and two ways to combine the latent representations  $z_1$  and  $z_2$  of the same cell and produce a single common representation  $z$ . The four variations we compare in this section are:

1. CVQVAE-ori: This is the main model described in previous sections.
2. CVAE: This is the original design of our model without the vector quantization. In this model, we do not combine the latent representation of the two encoders and use VAE.
3. CVQVAE-avg: This is the version of our model that averages the latent representations. Inspired by multiVI (2), in this version, we set the latent representation for the two modalities to be the average of the two encoder outputs and shared by both decoders.
4. CVQVAE-lin: This is the version of our model where we take a linear combination of the latent representations. Inspired by JAE (22), in this version, after we combine the outputs of the two encoders, we input them into a linear layer of the neural network to obtain the new embedding and use it as input to the decoder.

The comparison between these different variations can be found in Table 3. From Table 3, we see that

Table 3: Overall score for ablation test models

Methods	NMI	Celltype ASW	CC	TC	Batch ASW	Graph connectivity	Score
CVAE	0.474	0.488	0.625	0.823	0.788	0.855	0.690
CVQVAE-ori	<b>0.780</b>	<b>0.569</b>	<b>0.883</b>	<b>0.880</b>	0.883	<b>0.982</b>	<b>0.840</b>
CVQVAE-avg	0.778	0.554	0.842	0.578	0.901	0.976	0.788
CVQVAE-lin	0.767	0.555	0.848	0.879	<b>0.904</b>	0.970	0.832

the method considered in the previous sections, CVQVAE-ori, has the best overall results, followed by CVQVAE-lin. Therefore, we choose CVQVAE-ori as the final version of our model.

## D Differential expression genes

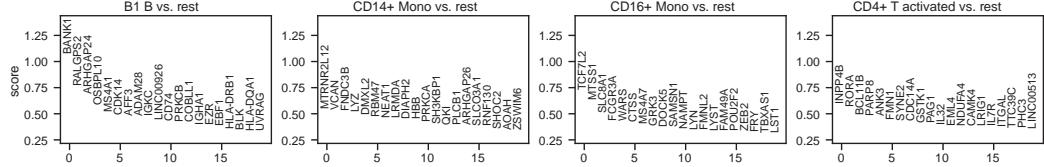


Figure 9: Differential expression test. For four different cell-type labels (B1 B, CD14+ Mono, CD16+ Mono, CD4+ T activated), we identify the genes that are most differentially expressed in the corresponding subset of cells.

## E The advantages of VQ-VAE compared with VAE

In part 1, we will give a proof about how VQ-VAE can address the posterior collapse problem. Now we define the posterior collapse as:

**Definition 1** (Posterior collapse). Given a probability model  $p(\mathbf{x}, \mathbf{z}; \theta)$ , a parameter value  $\theta = \hat{\theta}$ , and a dataset  $\mathbf{x} = (x_1, \dots, x_n)$ , the posterior of the latent variables  $\mathbf{z}$  collapses if

$$p(\mathbf{z} \mid \mathbf{x}; \hat{\theta}) \simeq p(\mathbf{z})$$

**Proof** If we intend to avoid this condition, we need to ensure that  $\mathbf{x}, \hat{\theta}$  should always involve in the posterior calculation. Therefore, we consider such optimization process for the embedding:

$$z_{new}(f_e(x, \hat{\theta})) = argmin_i(||z_i^K - f_e(x, \hat{\theta})||^2)$$

where  $z_{new}$  represent the quantized vector,  $z_i^K$  represent a vector from discrete dependent variable space  $K$ , and  $f_e(x, \hat{\theta})$  represent a function which can map the input data into the latent space, and we use neural networks here to fit this function. Here  $z_{new}$  will become the new input into the decoder model, which means that the decoder can be represented as:

$$D(z_{new}) = f_d(argmin_i(||z_i^K - f_e(x, \hat{\theta})||^2)) = D(z_K, x, \hat{\theta})$$

Now consider the embedding generated by VAE model, which can be represented as:

$$\mathbf{z} = f_\mu(x, \hat{\theta}) + s f_\sigma(x, \hat{\theta}) \text{ where } s \sim Normal(0, 1)$$

Therefore, in the case of using VQ-VAE, because  $z_i$  is an estimator of  $f_e(x, \hat{\theta})$  under the squared error loss, the decoder can always decode the distribution constructed based on the input data. However, for the original VAE design,  $z$  is not an estimator. Based on this property, VQ-VAE is not affected by posterior collapse. That is why our model can address this problem.

In part 2, we will compare CVAE vs CVQVAE based on the loss curve analysis (Figure 10). In the loss curve of CVAE, the initial optimizing process is very fast till the blue star part. This indicates that the model was only optimized in the first few epochs, and the optimization of subsequent epochs was ignored due to the influence of posterior collapse, so the curve appeared very stable, which means that in the following training step the total loss will not change anymore. The results of latent space generated by CVAE are also more mixed.

To address this problem, we utilize VQ-VAE to replace the original VAE model. VQ-VAE discretized the embedding in the optimization process, and allowed us to select the most appropriate posterior vector according to the embedding database after obtaining the output of encoder. Therefore, randomness was added in the selection process. The increment of the complexity and flexibility of the embedding space allows it to contain more information. Our new loss function ensures that the representation of embedding space is optimized while the decoder reconstructs the input data. From the loss curve of CVQVAE, we can observe that the total loss keeps on reducing, which means that the optimizing process can work properly comparing with the VAE version.

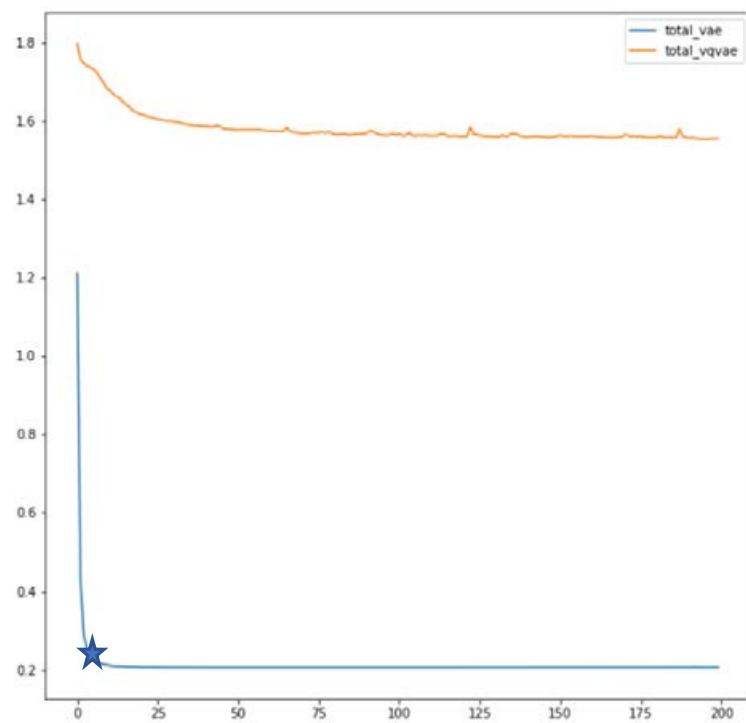


Figure 10: Training loss curves of CVAE (total\_vae) & CVQVAE (total\_vqvae)