Implementing scMVP

1 Create and manage a virtualenv

Pipenv is a Python virtualenv management tool that supports many systems and nicely bridges the gaps between pip, python (using system python, pyenv, or asdf), and virtualenv. Therefore, I decided to use `pipenv` to create and manage a virtualenv for my projects.

The packages that have been installed are as follows:

```
# CUDA 11.7
pipenv install torch==2.0.1 torchvision==0.15.2 torchaudio==2.0.2
pipenv install pandas
pipenv install scipy
pipenv install scikit-learn==0.22.2
pipenv install scanpy
```

2 Dataset

2.1 sciCAR dataset

For the sci-CAR dataset, only co-assay cells were used for further analysis, and cells with fewer than 200 peaks or genes and peaks or genes with fewer than 10 cells were removed from further analysis.

2.2 Paired-seq dataset

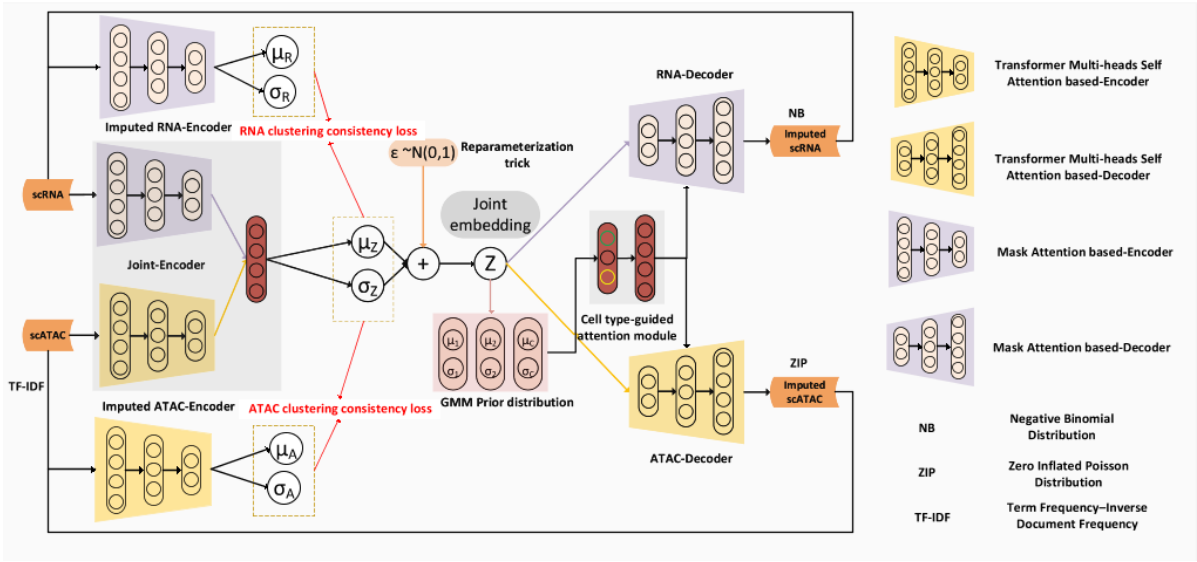
For Paired-seq dataset, cells with fewer than 200 peaks or genes, peaks, or genes with fewer than 10 cells or peaks with more than 336 cells were removed from further analysis.

2.3 SNARE-seq dataset

For SNARE-seq dataset, cells with fewer than 200 peaks or genes and peaks or genes with fewer than 10 cells were removed from further analysis.

3 The Model

scMVP consists of a two-channel encoder network and a two-channel decoder to integrate the information from scRNA-seq and scATAC-seq, and the input dimension of each channel is determined by the gene and peak number.



3.1 The RNA encoder

$$RNA\_Encoder(x) = encoder(x) * px\_decoder\_aux(x) \quad (1)$$

$$encoder(x) = ReLu(BatchNorm(LayerNorm(Linear(x))) \quad (2)$$

$$px\_decoder\_aux(x) = Sigmoid(Linear(Linear(x))) \quad (3)$$

$$mean\_encoder(x) = Linear(x) \quad (4)$$

$$var\_encoder(x) = Linear(x) \quad (5)$$

```
# Parameters for latent distribution
q = self.encoder(x, *cat_list) * self.px_decoder_aux(x)
q_m = self.mean_encoder(q)
q_v = torch.exp(self.var_encoder(q)) + 1e-4
latent = reparameterize_gaussian(q_m, q_v)
return q_m, q_v, latent
```

$$\mu_R = q\_m, \sigma_R = q\_v \quad (6)$$

3.2 The ATAC encoder

$$encoder(x) = ReLu(BatchNorm(LayerNorm(Linear(x))) \quad (7)$$

$$px\_decoder\_aux(x) = Sigmoid(Linear(Linear(x))) \quad (8)$$

$$mean\_encoder(x) = Linear(x) \quad (9)$$

$$var\_encoder(x) = Linear(x) \quad (10)$$

```
# Parameters for latent distribution
q = self.encoder(x, *cat_list)
assert q.shape[1] % self.n_heads == 0, "n_heads cann't be divided by seq length!"
```

```
Q = self.w_q(q).view(q.shape[0], self.n_heads, q.shape[1] // self.n_heads, -1)
K = self.w_k(q).view(q.shape[0], self.n_heads, q.shape[1] // self.n_heads, -1)
V = self.w_v(q).view(q.shape[0], self.n_heads, q.shape[1] // self.n_heads, -1)
energy = torch.matmul(Q, K.permute(0, 1, 3, 2))
attention = self.do(torch.softmax(energy, dim=-1))
q_a = torch.matmul(attention, V).view(q.shape[0], q.shape[1])

q_m = self.mean_encoder(q_a)
q_v = torch.exp(self.var_encoder(q_a)) + 1e-4
latent = reparameterize_gaussian(q_m, q_v)
return q_m, q_v, latent
```

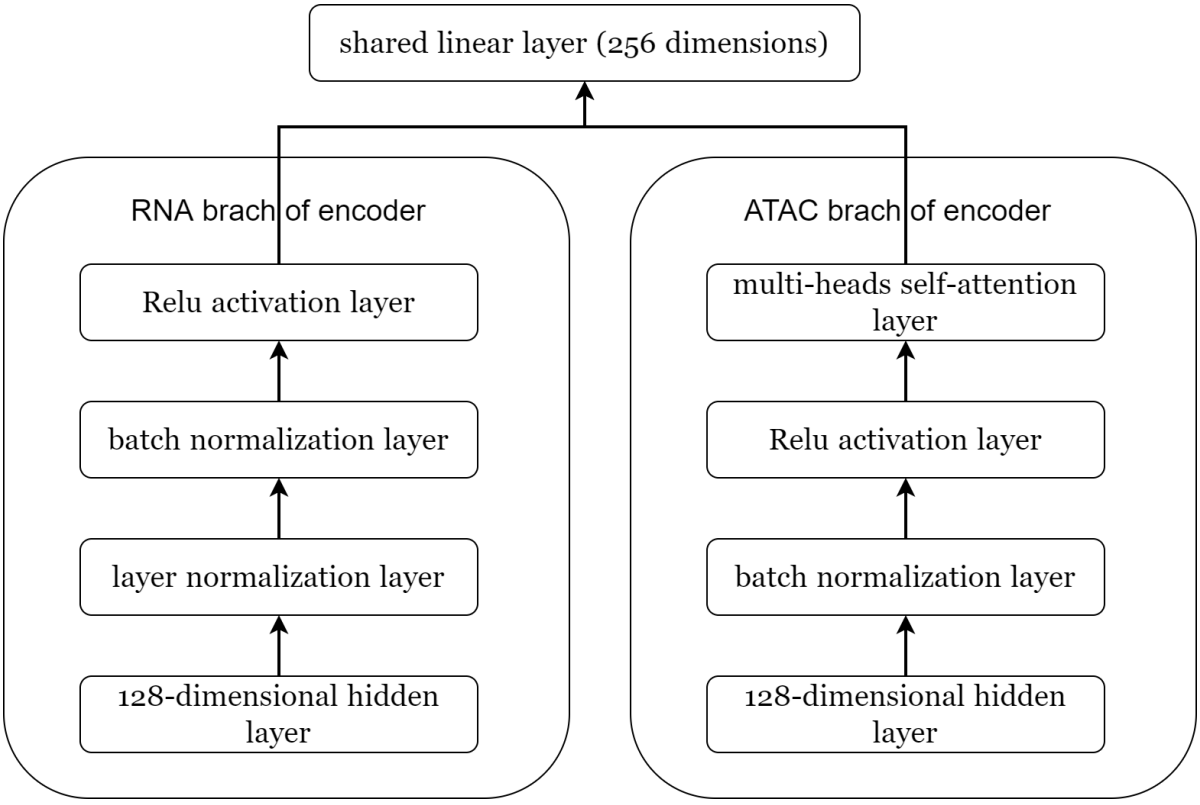
Encodes the data into latent space using the encoder network.

Generates a mean `q_m` and variance `q_v` (clamped to ([-5, 5]))

Samples a new value from an i.i.d. multivariate normal $\sim N(q\_m, \mathbf{I}q\_v))$

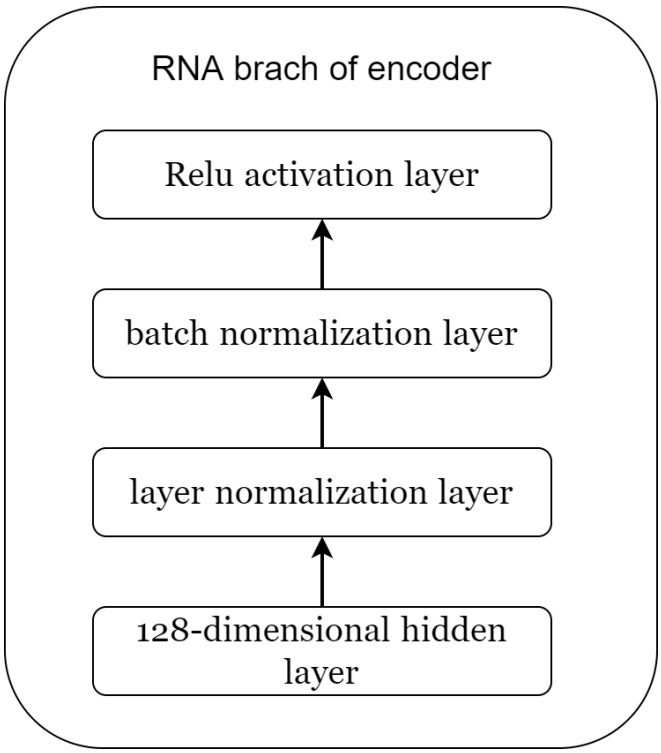
3.3 A two-channel encoder

scMVP uses a **mask attention channel** for the RNA branch and a **self-attention channel** for the ATAC branch to identify the cell type-associated information and capture the intra-omics distal correlation. The output two channels are combined together to form a shared linear layer (256 dimensions).



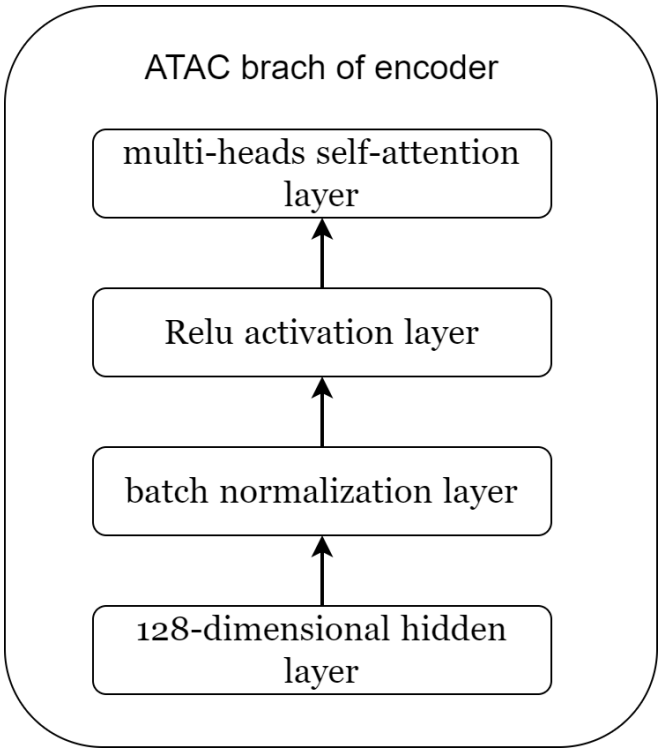
3.3.1 The RNA branch of the encoder

RNA branch of the encoder sequentially concatenates a 128-dimensional hidden layer, a layer normalization layer, a batch normalization layer, and an output Relu activation layer, which is weighted by a mask attention tensor generated from the first 128-dimensional hidden layer.



3.3.2 The ATAC branch of the encoder

The ATAC branch of the encoder sequentially concatenates a 128-dimensional hidden layer, a batch normalization layer, a Relu activation layer, and a multi-heads self-attention layer, which is designed as 8 self-attention heads and each head takes a 16-dimension feature in this study, and a layer normalization.

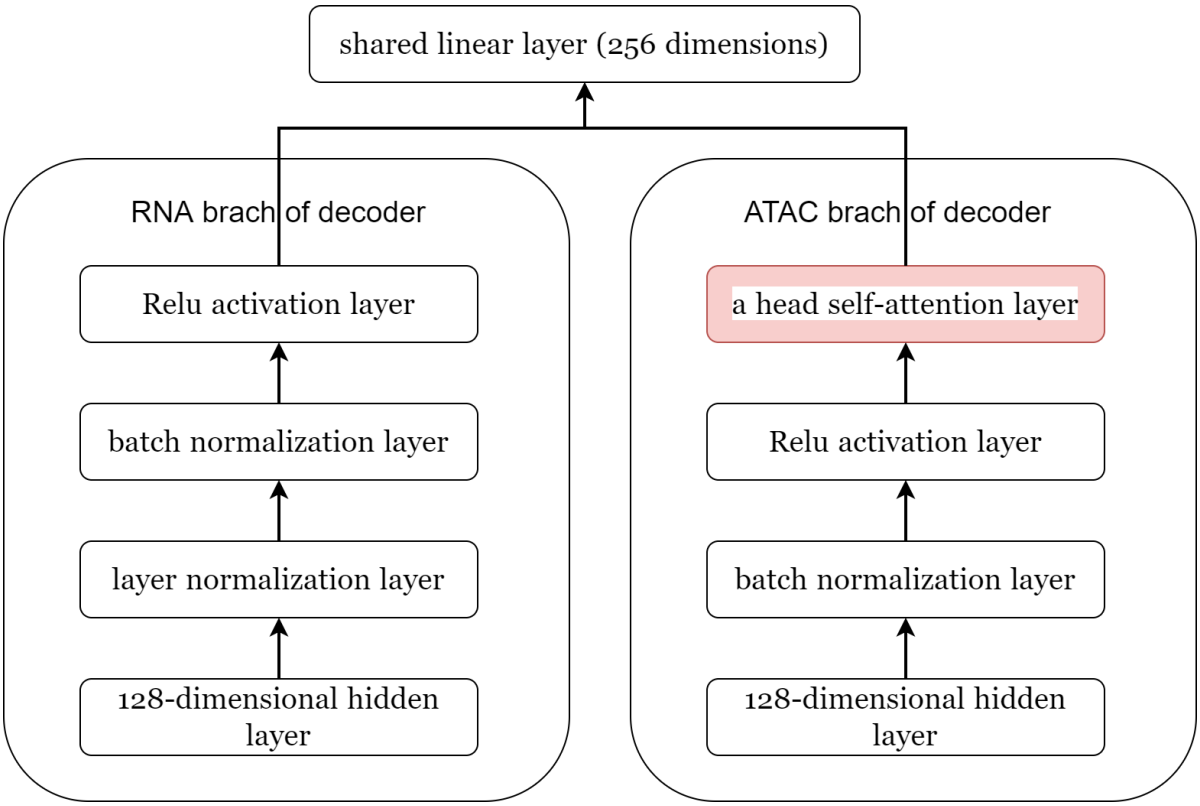


3.4 The attention module

The attention module receives the $p(c|z)$ for all K components as input, by a linear layer (128 dimensions), and then weights the last layer of each decoder channel with a SoftMax activation function.

3.5 A two-channel decoder

A two-channel decoder is employed to determine the distribution parameters of NB and ZIP for the reconstruction of scRNA-seq and scATAC-seq, which utilize a similar network structure with the network except for **an attention module**.



$$p(c) = Cat(\pi) = \prod_{k=1}^{K} \pi_k^{c_k},\ \pi = [\pi_1, \pi_2, \ldots, \pi_K] \tag{11}$$

$$p(z|c) = N(z|\mu_c, \sigma_c I) = \frac{1}{\sqrt{2\pi}\sigma} e^{(-\frac{(z-\mu_c)^2}{2\sigma_c^2})} \tag{12}$$

Here, c represents one of the K components(clusters) of Gaussian mixture distribution, which is extracted from a categorical distribution with probability π_c, then the common embedding latent variable z is derived from the component c with a probability $p(z|c) = N(z|\mu_c, \sigma_c I)$, which means the latent variable z associated cells belongs to a specific cluster (cell type) c.

$$\alpha_x, \beta_x = Decoder_x(z) \tag{13}$$

$$p\left(\mu_x \mid \alpha_x, \beta_x\right) = Gamma\left(\alpha_x, \beta_x\right) = \frac{\beta_x^{\alpha_x} \bar{x}^{\alpha_x - 1} e^{-\beta_x \bar{x}}}{\Gamma\left(\alpha_x\right)} \tag{14}$$

$$p\left(\mathrm{x} \mid \mu_x\right) = Poisson\left(\mu_x\right) = \frac{\mu_x^{\mathrm{x}}}{\mathrm{x}!} e^{-\mu_x} \tag{15}$$

Then, a two-channel decode network is used to generate the parameters of the NB and ZIP distribution to reconstruct the original observed x (RNA) and TF-IDF transformed y (ATAC) from the common latent variable z. As a result, the RNA counts can be imputed with the mean of the Poisson distribution.

$$\mu_y, \tau_y = Decoder_y(z) \tag{16}$$

$$p\left(\bar{y} \mid \mu_y\right) = Poisson\left(\mu_y\right) = \frac{\mu_y^{\bar{y}}}{\bar{y}!} e^{-\mu_y} \tag{17}$$

$$\mathrm{p}\left(\omega_y \mid \tau_y\right) = Bernoulli\left(\tau_y\right) = {\tau_y}^{\omega_y} (1 - \tau_y)^{1 - \omega_y} \tag{18}$$

$$\begin{aligned} \mathrm{p}(\mathrm{y} \mid \bar{y}, \omega_y) = & \left[p\left(\bar{y} \mid \mu_y\right) * \mathrm{p}\left(\omega_y = 1 \mid \tau_y\right)\right]_{y>0} \\ & + \left[\mathrm{p}\left(\omega_y = 0 \mid \tau_y\right) + p\left(\bar{y} \mid \mu_y\right) * \mathrm{p}\left(\omega_y = 1 \mid \tau_y\right)\right]_{y=0} \end{aligned} \tag{19}$$

Similarly, the ZIP distribution is decomposed as a Poisson distribution and a Bernoulli distribution, and the mean μ_y of the Poisson distribution is worked here as the imputation of scATAC-seq data

3.6 cycle-GAN

To further improve the performance of the scMVP model to extreme sparse dataset as joint profiling dataset, a **cycle-GAN** like clustering consistency auxiliary network to coordinate the latent embedding of each scMVP imputed profile with the joint embedding from the raw profile, was introduced.

Duo to the different characteristics of scRNA data and scATAC data,

3.7 The optimizer

scMVP model is optimized by maximizing the log likelihood probability of the generated scRNA and ATAC data according to variational Bayesian inference: