A REPROT FOR DEEP LEARNING LAB EXERCISE 1

Ziyi Guo, MSc Data Science

zg2u21@sonton.ac.uk

1 GRADIENT-BASED MATRIX FACTORISATION

In the first part of the experiment, a gradient-based factorisation is implemented in the code block as below:

```
def sgd_factorise(A: torch.Tensor, rank: int, num_epochs=1000, lr = 0.01) ->Tuple[torch.Tensor, torch.tensor]:
    N = num_epochs
    m = A.size()[0]
    n = A.size()[1]
    r = rank
    U = torch.randn((m,r))
    V = torch.randn((n,r))
    for epoch in range(N):
        for r in range(m):
            for c in range(n):
                e = A[r][c] - U[r,:]@V[c,:].t()
                U[r,:] += lr*e*V[c,:]
                V[c,:] += lr*e*U[r,:]
    return U, V
```

Illustrating the target matrix $\boldsymbol{A} = \begin{bmatrix} 0.3374 & 0.6005 & 0.1735 \\ 3.3359 & 0.0492 & 1.8374 \\ 2.9407 & 0.5301 & 2.2620 \end{bmatrix}$ and rank-2 factorisation, a number of experiments are carried out to test the algorithm. In the abundant experiments, a series of reconstruction losses are acquired in a range of [0.1219, 0.4822], while most losses are less than 0.2, indicating the effectiveness but randomness of the algorithm. In the case of the minimum loss of 0.1219, the algorithm outputs the factorisation results of $\boldsymbol{U} = \begin{bmatrix} 0.5913 & 1.5092 & 1.9775 \\ -0.4994 & 0.8254 & 0.1241 \end{bmatrix}$, and $\boldsymbol{V} = \begin{bmatrix} 1.4551 & 0.3416 & 1.0341 \\ 1.2777 & -0.6369 & 0.5031 \end{bmatrix}$, based on which the best reconstruction matrix was computed as $\boldsymbol{A}^* = \begin{bmatrix} 0.2222 & 0.5201 & 0.3602 \\ 3.2506 & -0.0101 & 1.9759 \\ 3.0360 & 0.5966 & 2.1074 \end{bmatrix}$.

2 MATRIX RECONSTRUCTION BASED ON TRUNCATED SVD

In the second part of the experiment, a truncated-SVD-based matrix reconstruction is implemented in the code block as below:

```
def svd_factorise(A: torch.Tensor) ->Tuple[torch.Tensor]:
    U,S,V_T = torch.svd(A)
    S[2] = 0
    SS_ = torch.diag(S)
    A_ = U@SS_@V_T.t()
    return A_
```

Illustrating the same target matrix $\boldsymbol{A}$, the algorithm outputs the reconstruction matrix $\boldsymbol{A}^{\sim} = \begin{bmatrix} 0.3374 & 0.6005 & 0.1735 \\ 3.3359 & 0.0492 & 1.8374 \\ 2.9407 & 0.5301 & 2.2620 \end{bmatrix}$, and the corresponding reconstruction loss is quite approximating the minimum loss in the gradient-based factorisation as 0.1219. Further exploring the reason, the Eckart-Young theorem can be illustrated here to clarify that in the case of the SVD of a matrix $\boldsymbol{A} = \boldsymbol{U} \boldsymbol{\Sigma} \boldsymbol{V}^T$, define a rank of k and truncated matrix $\boldsymbol{A}_k = \sum_{i=1}^{k} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T$, then for any matrix of rank k, the minimal error is achieved with $\boldsymbol{A}_k$. This is exactly explaining the case where the rank-2 truncated matrix of SVD gives the best result of rank-2 factorisation proposed in the part before.

3 MATRIX COMPLETION

In the third part of the experiment, a masked matrix factorisation is implemented in the code block as below:

```
def sgd_factorise_masked(A: torch.Tensor, M:torch.Tensor, rank: int, num_epochs=1000, lr = 0.01):
    N = num_epochs
    m = A.size()[0]
    n = A.size()[1]
    r = rank
    U = torch.randn((m,r))
    V = torch.randn((n,r))
    for epoch in range(N):
        for r in range(m):
            for c in range(n):
                if M[r][c] == 1:
                    e = A[r][c] - U[r,:]@V[c,:].t()
                    U[r,:] += lr*e*V[c,:]
                    V[c,:] += lr*e*U[r,:]
                else:
                    continue
    return U, V
```

Illustrating the same matrix $\boldsymbol{A}$ and a mask matrix $\boldsymbol{M} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$, the rank-2 factorisation of a target matrix $\boldsymbol{T} = \begin{bmatrix} 0.3374 & 0.6005 & 0.1735 \\ & 0.0492 & 1.8374 \\ 2.9407 & & 2.2620 \end{bmatrix}$ is figured out, based on which the reconstruction matrix (estimation of the complete target matrix $\boldsymbol{T}^*$) and error are computed. Similarly abundant experiments are carried out to test the difference between estimation $\boldsymbol{T}^*$ and the original matrix $\boldsymbol{A}$. Among the experiments, a series of losses are acquired in a range of [0.1220, 112.1676] and in the case where the minimum loss appears to be 0.2540, the masked factorisation outputs the results of $\boldsymbol{U} = \begin{bmatrix} -0.0023 & 1.1480 & 1.1147 \\ -0.2080 & 1.0990 & 1.3024 \end{bmatrix}$, and $\boldsymbol{V} = \begin{bmatrix} 1.6743 & -1.4381 & -0.4038 \\ 0.8307 & 1.5634 & 2.0792 \end{bmatrix}$, based on which the estimation $\boldsymbol{T}^*$ is computed as $\begin{bmatrix} 0.1690 & 0.3284 & 0.4334 \\ 2.8350 & 0.0671 & 1.8215 \\ 2.9481 & 0.4330 & 2.2577 \end{bmatrix}$. It is similar to the origin matrix $\boldsymbol{A}$. However, in a number of cases where the loss is extremely high, the estimation can be really biased. It is indicated that the gradient-based matrix completion can be effective based on abundant experiments, while can be unstably performing because of the randomness of gradient initialization.