Page No.
Date

Part a

1.1 $MSE(\hat{w}) = \mathbb{E}_T\left[\|\hat{w} - w^*\|_2^2\right]$

$= \mathbb{E}_T\left[(\hat{w} - w^*)^T(\hat{w} - w^*)\right]$

$= \mathbb{E}_T\left\{\left[\hat{w} - \mu\hat{w} + \mu_{\hat{w}} - w^*\right]^T\left[\hat{w} - \mu_{\hat{w}} + \mu_{\hat{w}} - w^*\right]\right\}$

$= \mathbb{E}_T\left[(\hat{w} - \mu_{\hat{w}})^T(\hat{w} - \mu\hat{w}) + 2(\hat{w} - \mu\hat{w})^T(\mu_{\hat{w}} - w^*) + (\mu_{\hat{w}} - w^*)^T(\mu_{\hat{w}} - w^*)\right]$

$= \mathbb{E}_T\left[\|\hat{w} - \mu_{\hat{w}}\|_2^2\right] + 2\mathbb{E}_T\left[(\hat{w} - \mu_{\hat{w}})^T(\mu_{\hat{w}} - w^*)\right] + \|\mu_{\hat{w}} - w^*\|_2^2$

$= \mathbb{E}_T\left[\|\hat{w} - \mu_{\hat{w}}\|_2^2\right] + 2(\mu_{\hat{w}} - w^*)^T\mathbb{E}_T(\hat{w} - \mu_{\hat{w}}) + \|\mu_{\hat{w}} - w^*\|_2^2$

$= \mathbb{E}_T\left[\|\hat{w} - \mu_{\hat{w}}\|_2^2\right] + 0 + \|\mu_{\hat{w}} - w^*\|_2^2$

$= Var(\hat{w}) + \|\mu_{\hat{w}} - w^*\|_2^2$

$= tr(Cov(\hat{w})) + \|\mu_{\hat{w}} - w^*\|_2^2$

Part b

1.2

$$\hat{w}_{OLS} = (X^T X)^{-1} X^T y$$

$$E_T[\hat{w}_{OLS}] = E_T[(X^T X)^{-1} X^T y]$$

$X, y \rightarrow$ random variables

Given model $y = X w^* + \varepsilon$

$\varepsilon \rightarrow$ normal distribution

$$E_T[y] = X w^*$$

So,

$$E_T[\hat{w}_{OLS}] = (X^T X)^{-1} X^T E_T(y)$$

$$= (X^T X)^{-1} X^T X w^* = w^*$$

Hence,

OLS estimator $\rightarrow$ unbiased of w^*

2. $\hat{w}_{ridge} = (X^T X + \alpha I_p)^{-1} X^T y$

$$E_T[\hat{w}_{ridge}] = (X^T X + \alpha I_p)^{-1} X^T E_T[y]$$

$$= (X^T X + \alpha I_p)^{-1} X^T X w^*$$

$\Rightarrow$ $\hat{w}_{ridge}$ is biased of w^*

Page No.
Date

Part - C

1.3

$$\hat{w}_{ridge} = (X^T X + \alpha I_p)^{-1} X^T y$$

$$Cov(\hat{w}_{ridge}) = \sigma^2 (X^T X + \alpha I_p)^{-1}$$

trace of matrix = Sum of eigen values. Lets $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \dots \geq \lambda_p$ are eigenvalues of matrix $X^T X$. Then eigenvalues of $X^T X + \alpha I_p$ are $\lambda_i + \alpha$ for $i = 1, 2 \dots p$

So,

$$tr[Cov(\hat{w}_{ridge})] = \sigma^2 \sum_{i=1}^{p} \frac{1}{\lambda_i + \alpha}$$

Similarly,

$$\hat{w}_{OLS} = (X^T X)^{-1} X^T y$$

$$Cov(\hat{w}_{OLS}) = \sigma^2 (X^T X)^{-1}$$

trace of this Cov matrix = Sum of reciprocals of eigenvalues of $X^T X$

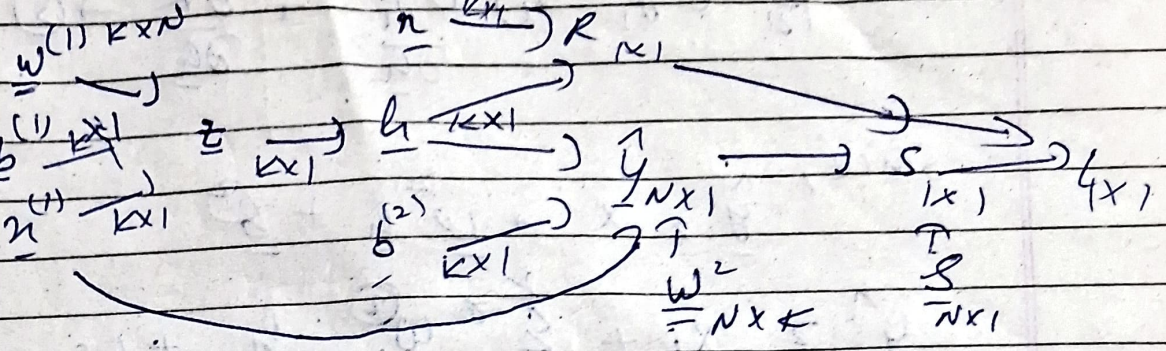
$$tr[Cov(\hat{w}_{OLS})] = \sigma^2 \sum_{i=1}^{p} \frac{1}{\lambda_i}$$

$\lambda_i + \alpha > \lambda_i$ so

$$tr[Cov(\hat{w}_{ridge})] < tr[Cov(\hat{w}_{OLS})]$$

Q3

(a)



(b) $\bar{V} = \frac{\partial L}{\partial V}$ → notation

→ Backward pass:

$$\bar{L} = \frac{\partial L}{\partial L} = 1 \quad \rightarrow ①$$

$$\bar{S} = \frac{\partial L}{\partial S} = 1 \quad \rightarrow ②$$

$$\{ \because A_s \; L = S + R \}$$

$$\bar{\hat{y}} = \frac{\partial L}{\partial \hat{y}} = \frac{\partial L}{\partial S} \frac{\partial S}{\partial \hat{y}} = 1 \cdot \frac{\partial S}{\partial \hat{y}} \quad \rightarrow ③$$

$$S = \frac{1}{2} \| \hat{y} - \underline{s} \|_2^2 = \frac{1}{2} (\hat{y} - \underline{s})^T (\hat{y} - \underline{s})$$

$$\frac{\partial S}{\partial \hat{y}} = \hat{y} - \underline{s}$$

So, $\bar{\hat{y}} = \hat{y} - \underline{s} \quad \rightarrow ④$

$$\bar{h} = \frac{\partial L}{\partial h} = \sum_{i=1}^{N} \frac{\partial L}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial h} + \frac{\partial L}{\partial R} \frac{\partial R}{\partial h} \quad \left\{ \because \frac{\partial L}{\partial \hat{y}_i} = \hat{y}_i - s_i \right\}$$

$$= \sum_{i=1}^{N} (\hat{y}_i - s_i) \frac{\partial \hat{y}_i}{\partial h} + \frac{\partial R}{\partial h} \quad \text{--- (5)}$$

Since,

$$\hat{y} = x + W^{(2)} h + b$$

$$\hat{y}_i = x_i + \sum_{i=1}^{K} w_{ik}^{(2)} h_k + b_i$$

$$\frac{\partial \hat{y}_i}{\partial h} = \begin{bmatrix} w_{i1}^{(2)} \\ w_{i2}^{(2)} \\ \vdots \\ w_{iK}^{(2)} \end{bmatrix} = \sum_{i=1}^{N} (\hat{y}_i - s_i) \cdot \frac{\partial \hat{y}_i}{\partial h}$$

$$= W^{T(2)} [\hat{y} - s]$$

Since, $R = \sum \varepsilon_i h_i$

$$\Rightarrow \frac{\partial L}{\partial h} = \varepsilon$$

So,

$$\varepsilon = W^{T(2)} (\hat{y} - s) + \varepsilon' \quad \text{--- (6)}$$

$$\varepsilon' = \frac{\partial L}{\partial z} = \frac{\partial L}{\partial h} \cdot \frac{\partial h}{\partial z}, \quad \{ \because h = \sigma(z) \}$$

$$= \left(W^{T(2)} (\hat{y} - s) + \varepsilon \right) \cdot \sigma'(z)$$

--- (7)

$$\frac{\partial L}{\partial x} = \sum_{i=1}^{K} \frac{\partial L}{\partial z_i} \frac{\partial z_i}{\partial x} + \sum_{i=1}^{N} \frac{\partial L}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial x}$$

1st term :-

$$z = w^{(1)} x + b^{(1)}$$

$$z_i = \sum_{j=1}^{N} w_{ij} x_j + b_i$$

$$\frac{\partial z_i}{\partial x} = \begin{bmatrix} w_{i1}^{(1)} \\ \vdots \\ w_{iN}^{(1)} \end{bmatrix} \Rightarrow \sum \frac{\partial L}{\partial z_i} \frac{\partial z_i}{\partial x}$$

$$= w^{(1)T} \frac{\partial L}{\partial z}$$

2nd Term :-

$$\hat{y} = x + w^{(2)} h + b^{(2)}$$

$$\hat{y}_i = x_i + \sum_{i=1}^{K} w_{ij} h_j + b_i$$

$$\frac{\partial \hat{y}_i}{\partial x} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \Rightarrow \sum_{i=1}^{N} \frac{\partial L}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial x}$$

$$= 2 \sum_{i=1}^{N} (\hat{y}_i - y_i) \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix}$$

$$= 2 (\hat{\underline{y}} - \underline{y})$$

$$\underline{n} = \underline{w}^{(0)T} \left[(w^{(2)T} (\hat{\underline{y}} - \underline{y}) + z) \cdot \sigma'(z) \right] + (\hat{\underline{y}} - \underline{y})$$

Q4.

(i) Because of mini-batch gradient descent. The update of parameters happen based on only any one batch ~~on~~ only in each iteration. So, cost data suffers a drop in accuracy

(ii) 91.59%

(iii) input layer → 784 + 1 = 785
(input) (bias)

output layer → 10

total parameter = 784 × 10
= 7840

(iv) Epoch → one pass over training data
No. of iteration in each pass

= training data size / each batch size = 60000 / 100

= 600

→ Loop runs 2000 times
So, total epochs = 2000 / 600

= 3.33

Hence,
No. of epochs = 3
We completed

(v) Accuracy 94.78%.
→ Yes, it is improved over single-layer neural network.

(vi) No. of hidden unit = 10
Test accuracy = 91.53%

→ No. of hidden unit = 400
Test accuracy = 94.78%

No. of hidden unit = 600
Test accuracy = 95.05%

→ With increase in hidden units in 1st layer, increase in test accuracy.

(vii) Test accuracy drops to 75.62%.

(Plot)

(viii) For six layers network:
Test accuracy = 88.37%
which is less than 2 layer network with random property

(Plot)

Page No.
Date

(IX) Test accuracy is 97.02%. which is more than sigmoid.

Plot

(X) Test accuracy is 11.35%

Plot

(XI) There is no NaN values within 2000 iterations. But more than 7000 iterations, NaN values start appearing in training set. Because of cross-entropy function, in which directly log of $\hat{y}_i$ is taken. So, when $\hat{y}_i$ is 0 it goes to undefined which might happen in ReLU activation function with negative logit values.

→ The F cross-entropy function adds a similar epsilon before taking log. So, there is no effect on changing F.softmax & returning just the logit has any effect on this.

Therefore,

By changing cross-entropy function to F.cross-entropy function NaN values stopped to appearing.

Plot

(xii) Input layer → 784 + 1 = 785
Hidden layer 1 → 200 + 1 = 201
2 → 100 + 1 = 101
3 → 60 + 1 = 61
4 → 30 + 1 = 31

Output layer = 10

total parameter = 785 × 200 + 201 × 100 + 101 × 60 + 61 × 30 + 31 × 10
= 185300

(xiii) With optim Adam, learning rate = 0.03, ReLU activations, random initialization of weights & 0.1 initialization of biases.
Test accuracy = 97.45%.
Which is more ~~fast~~ than optim. SGS with ReLU activations, with 0.1 learning rate.

Plot

There are 2 changes
We use lower learning rate with use of Adam. Which adjust the size of jump we need to make. So, it will fasten it up if we are moving in some direction. This leads to higher accuracy.

(xiv) Test accuracy = 94.06%.
There is no improvement over 2000 iteration in accuracy; so, increase in more iterations will not improve accuracy.

Page No.
Date

Q.5.

(i) Hidden layer in 3rd layer = 12 x 7 x 7
= 588

↳ flat → total no. of hidden units.

(ii) With 5 layer network & size of last 2 layers being 200 and 100.
Test accuracy = 98.51 %.

Plot

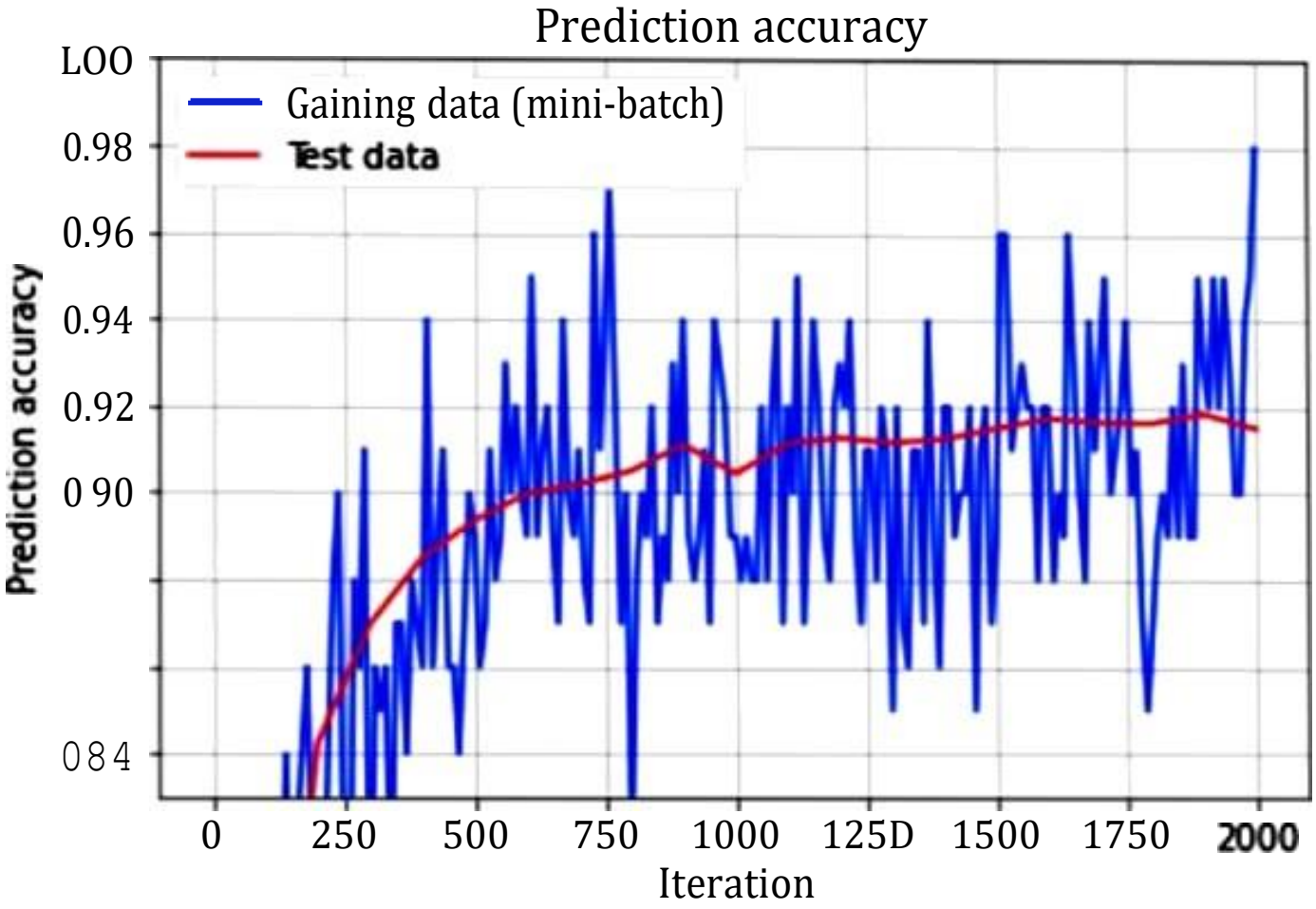
(iii) Test accuracy = 99.10 %.

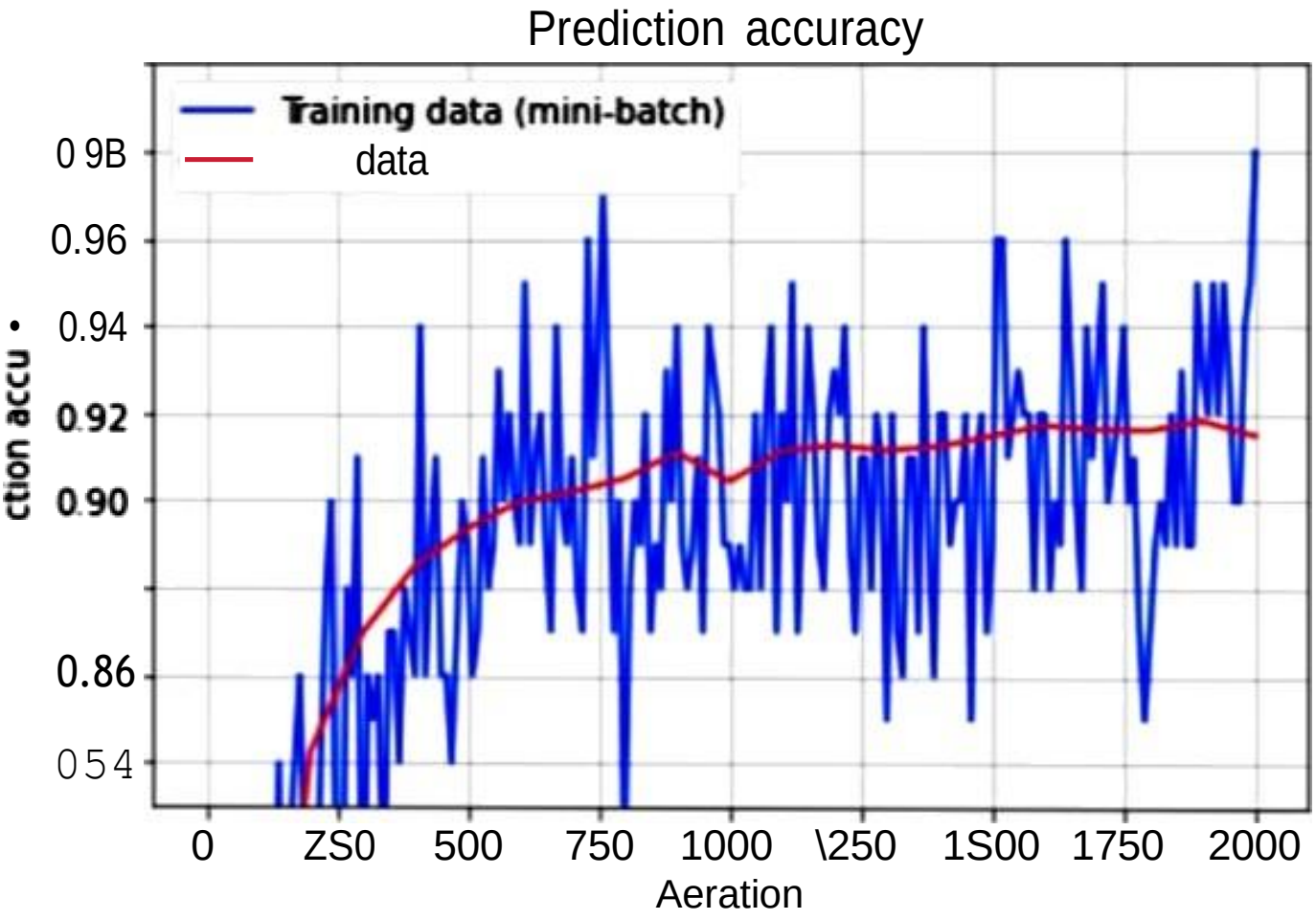
Plot

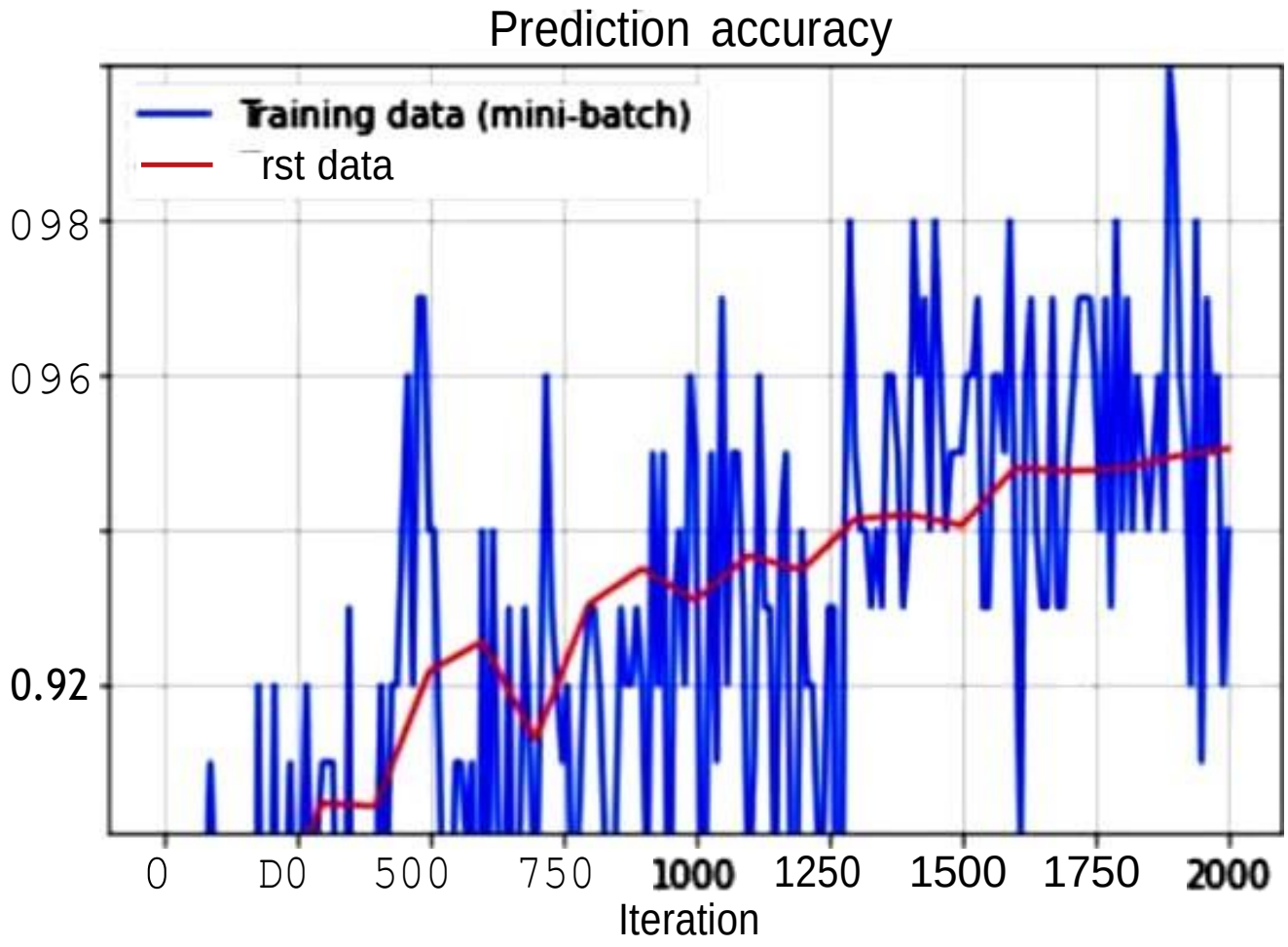
(iv) At last, test accuracy is 100 %. & saturated. Hence, now model is starting to overfit the data. The training accuracy was not tending to 100 %. In earlier neural network w/o CNN. In CNN due to increased in complexity of convolutional & dense layers, the neural network has started overfitting.

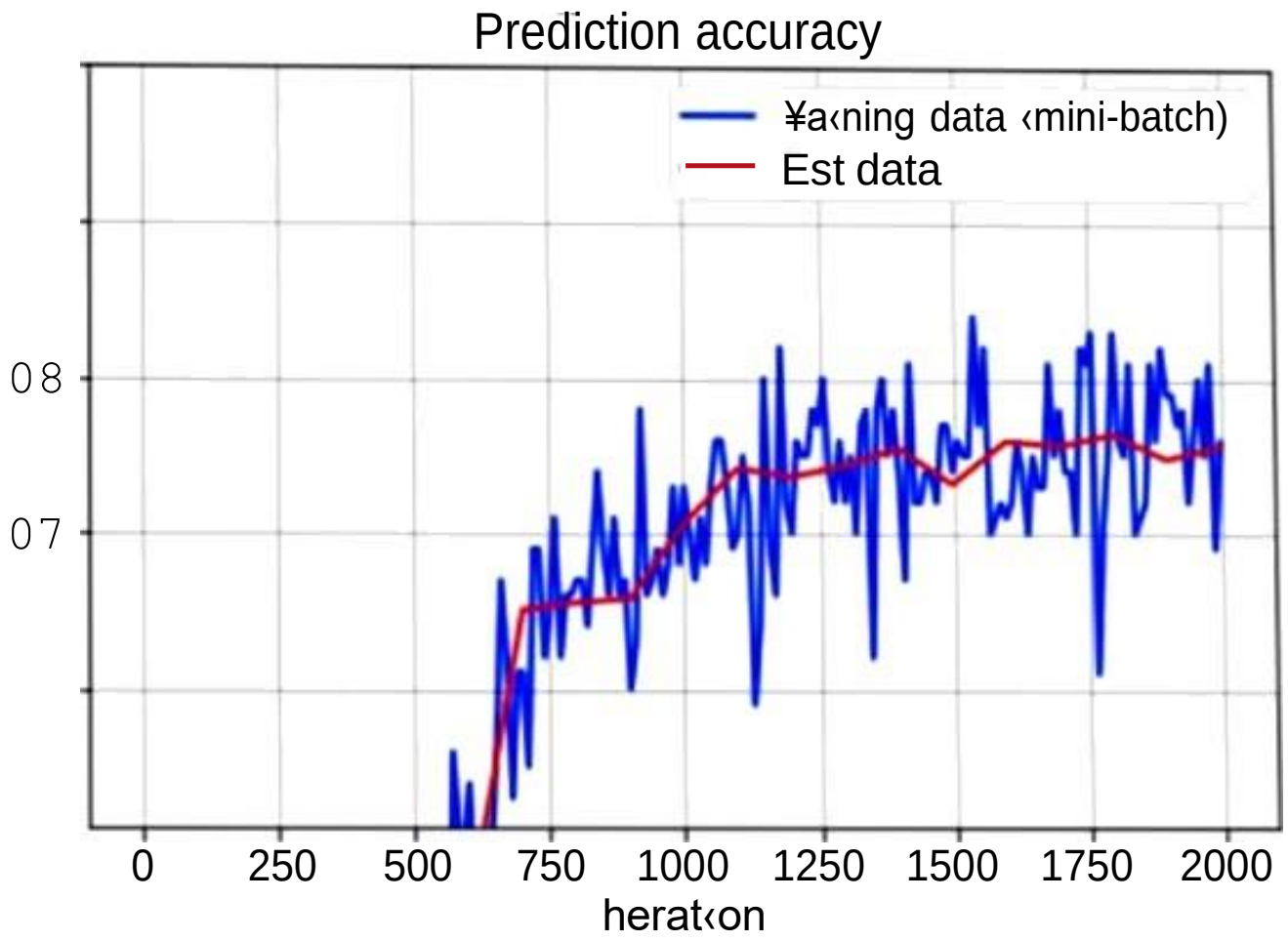
Plot

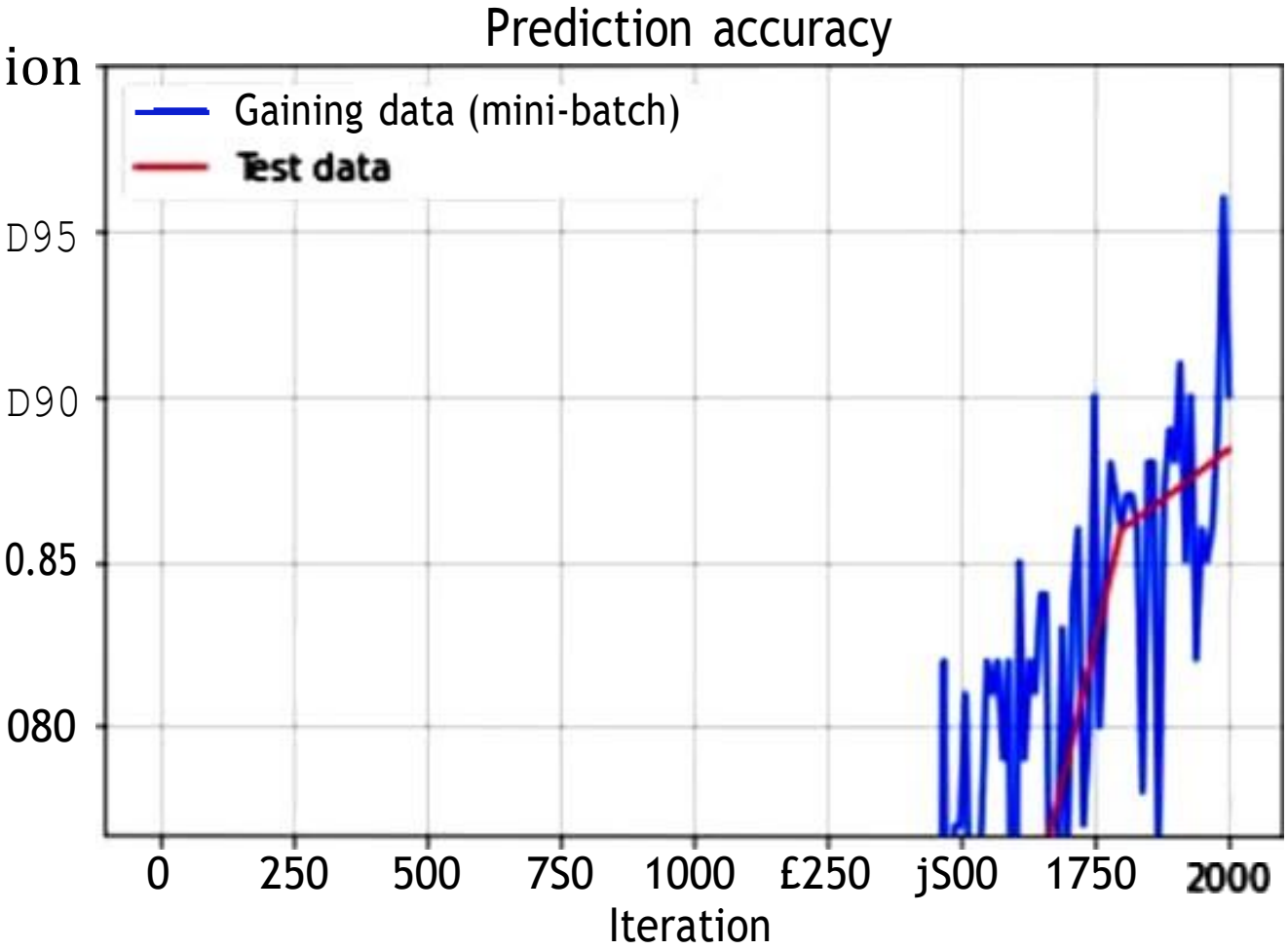
(v) Yes, using adjusted learning rate & with 200 & 100 layer. accuracy is above 99 %.

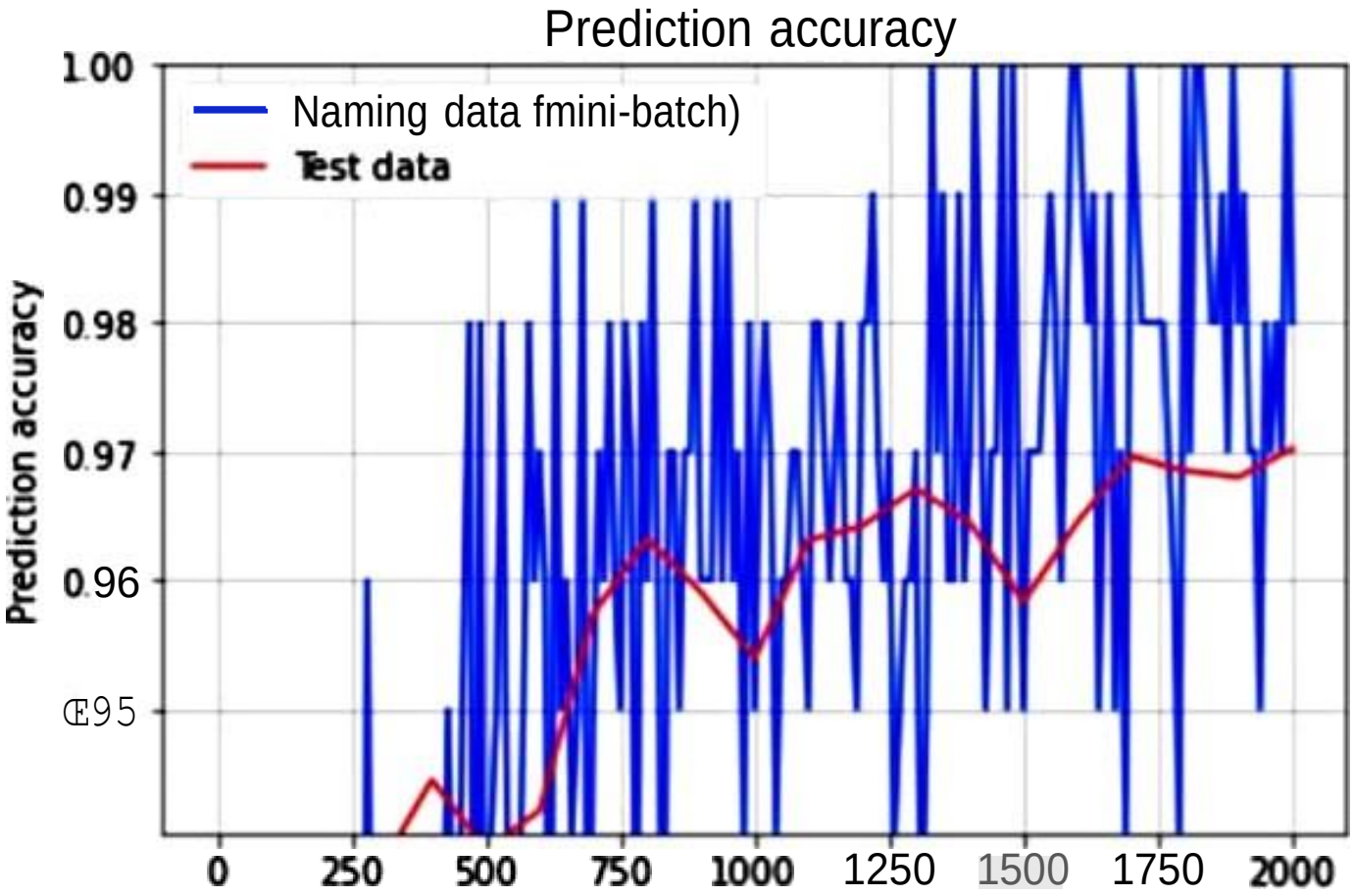

Prediction accuracy
LOO
0.98
0.96
0.94
0.92
0 90
084
Prediction accuracy
Gaining data (mini-batch)
Test data
0
250
500
750
1000
125D
1500
1750
2000
Iteration

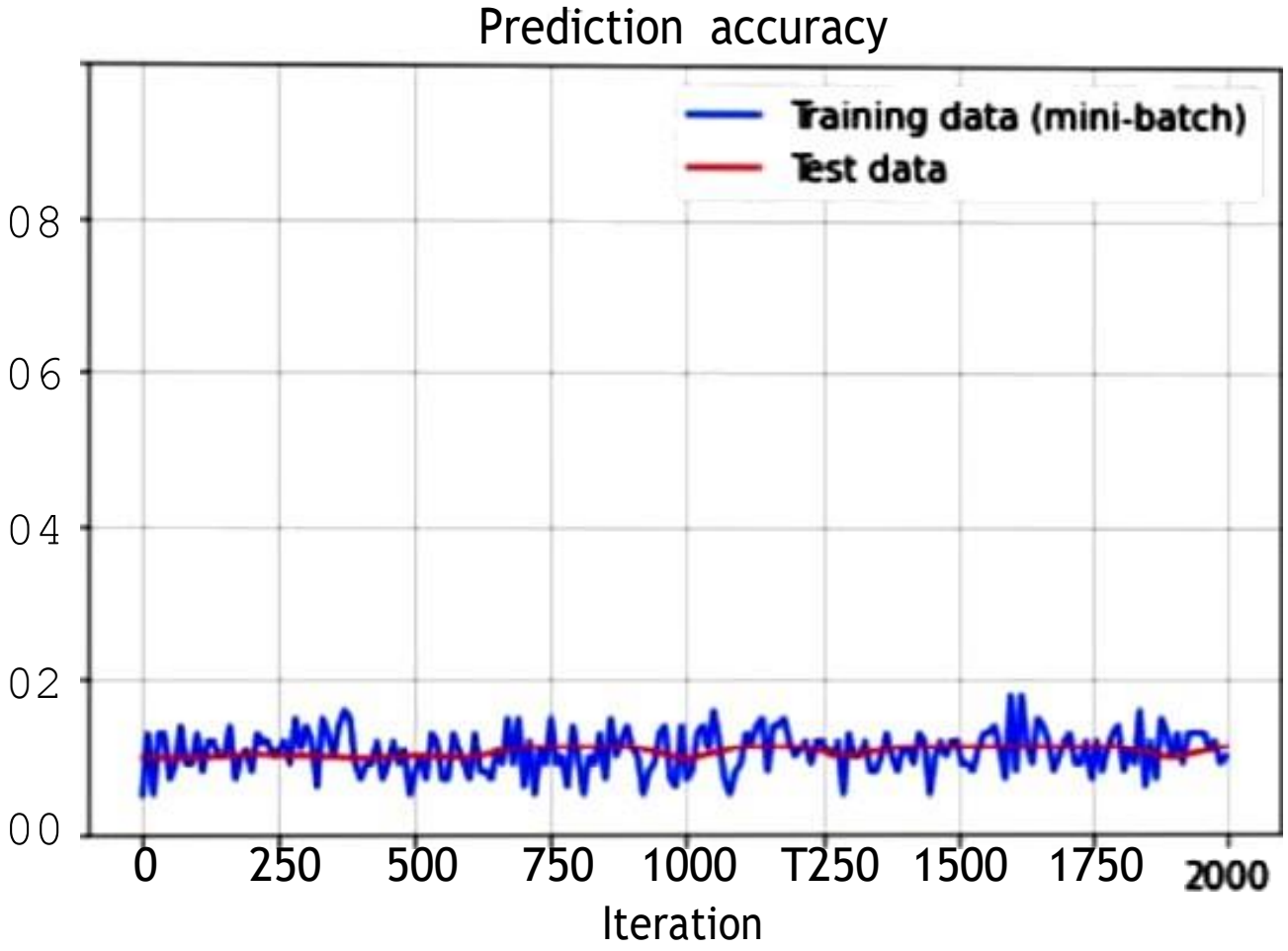

Prediction accuracy
Training data (mini-batch)
data
0 9B
0.96
0.94
0.92
0.90
0.86
054
ction accu •
0
ZS0
500
750
1000
\250
1S00
1750
2000
Aeration

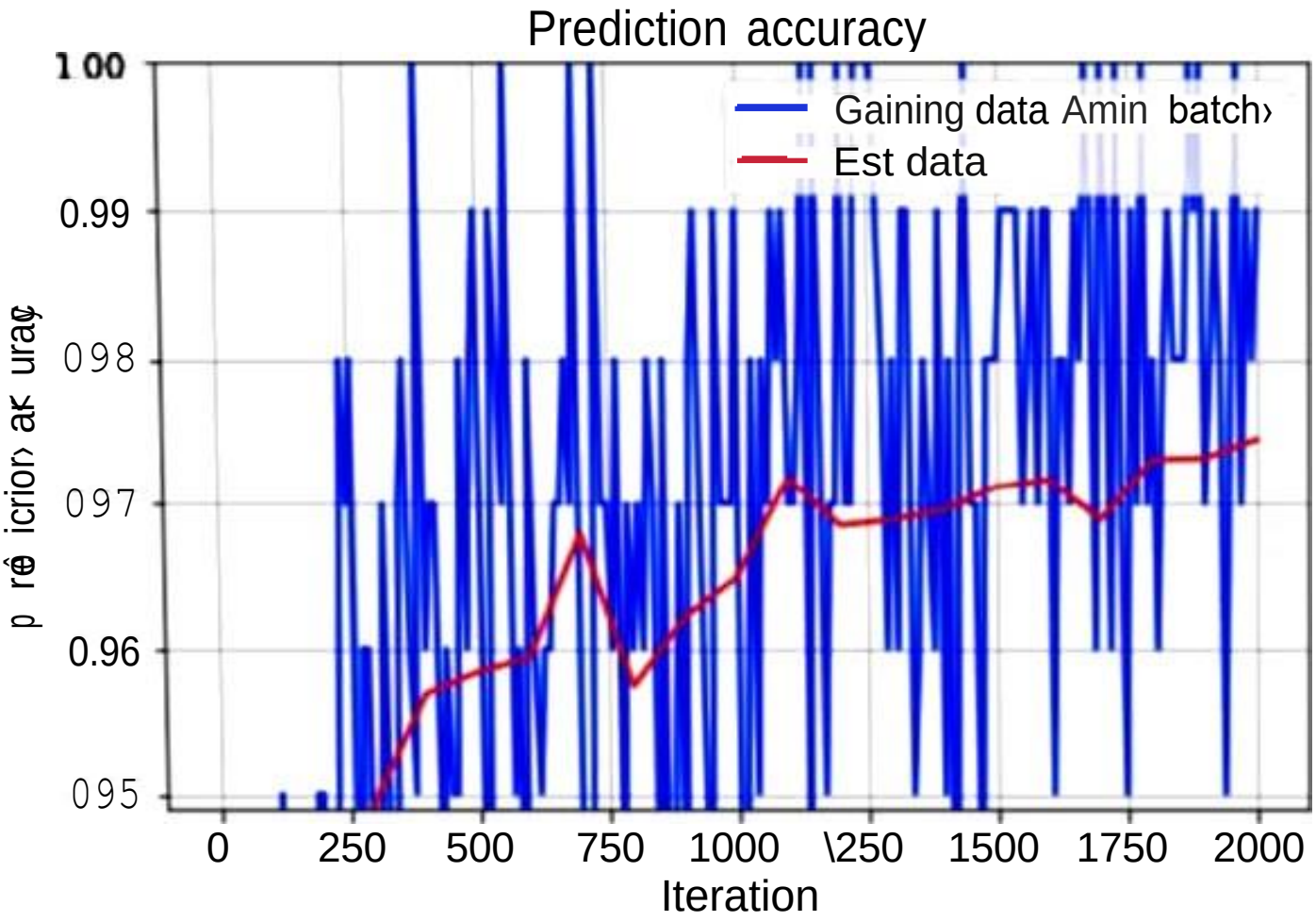

Prediction accuracy
Training data (mini-batch)
rst data
098
096
0.92
0
D0
500
750
1000
1250
1500
1750
2000
Iteration

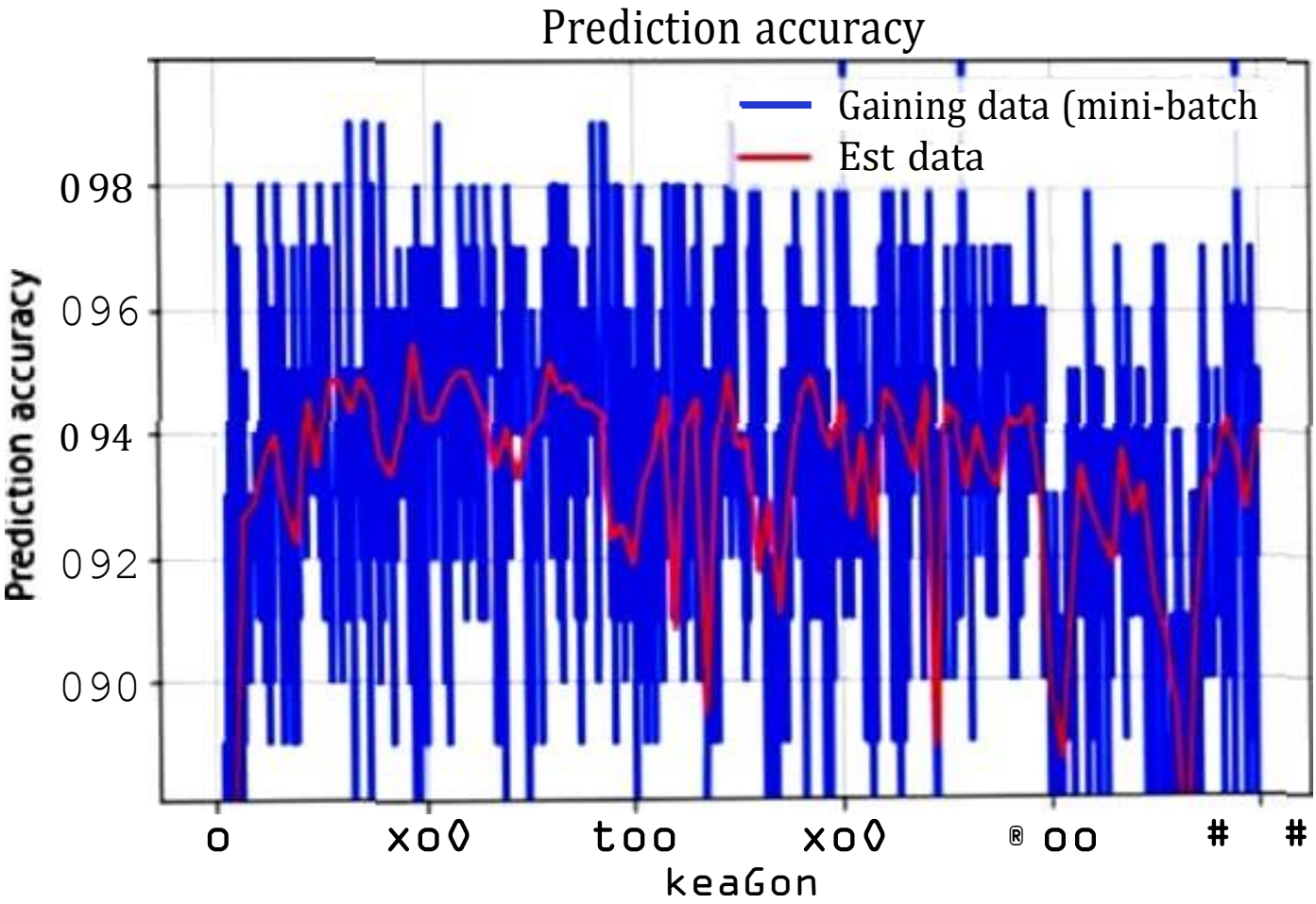

Prediction accuracy
¥a‹ning data ‹mini-batch)
Est data
08
07
0
250
500
750
1000
1250
1500
1750
2000
herat‹on

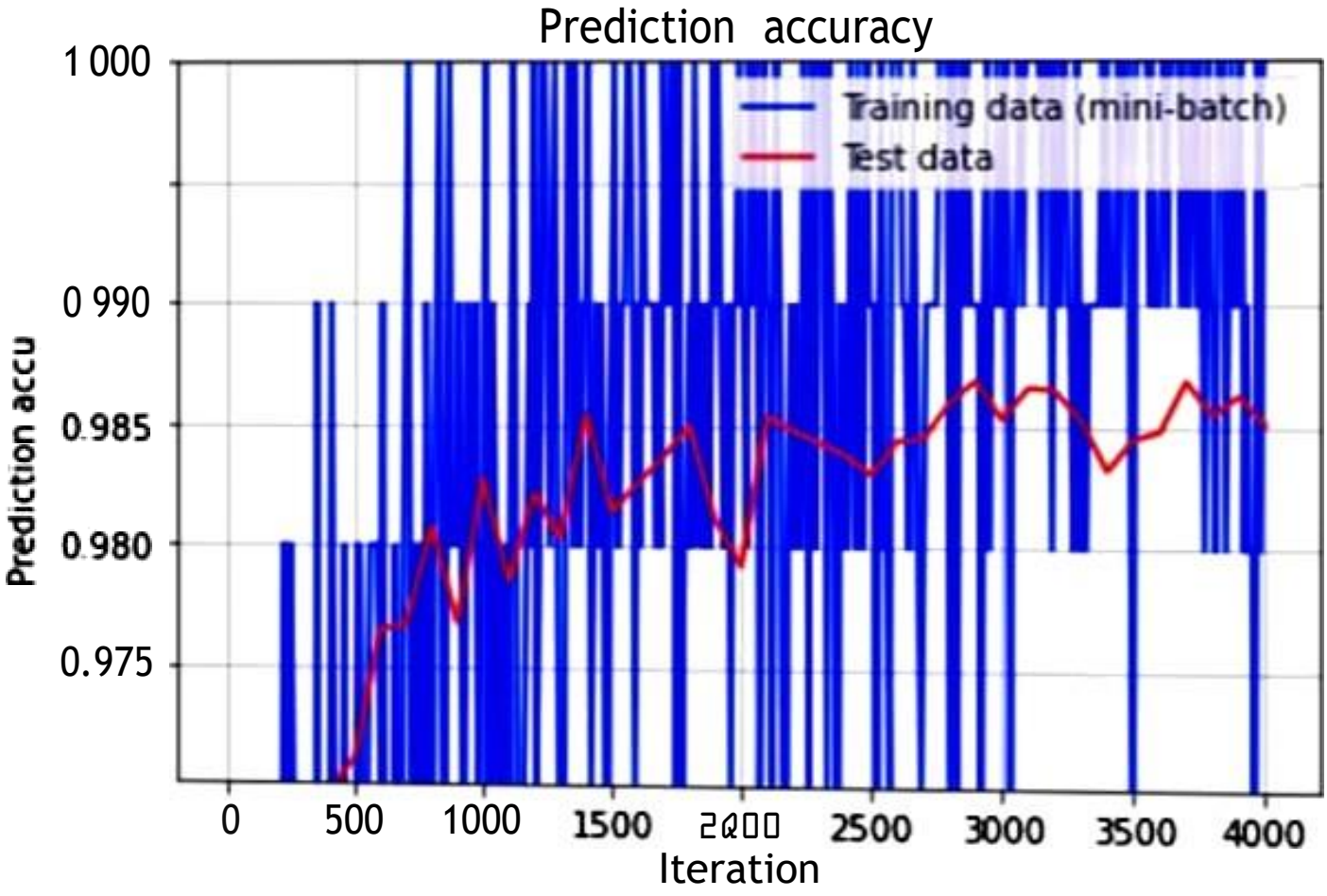

Prediction accuracy
ion
Gaining data (mini-batch)
Test data
D95
D90
0.85
080
0
250
500
7S0
1000
£250
jS00
1750
2000
Iteration

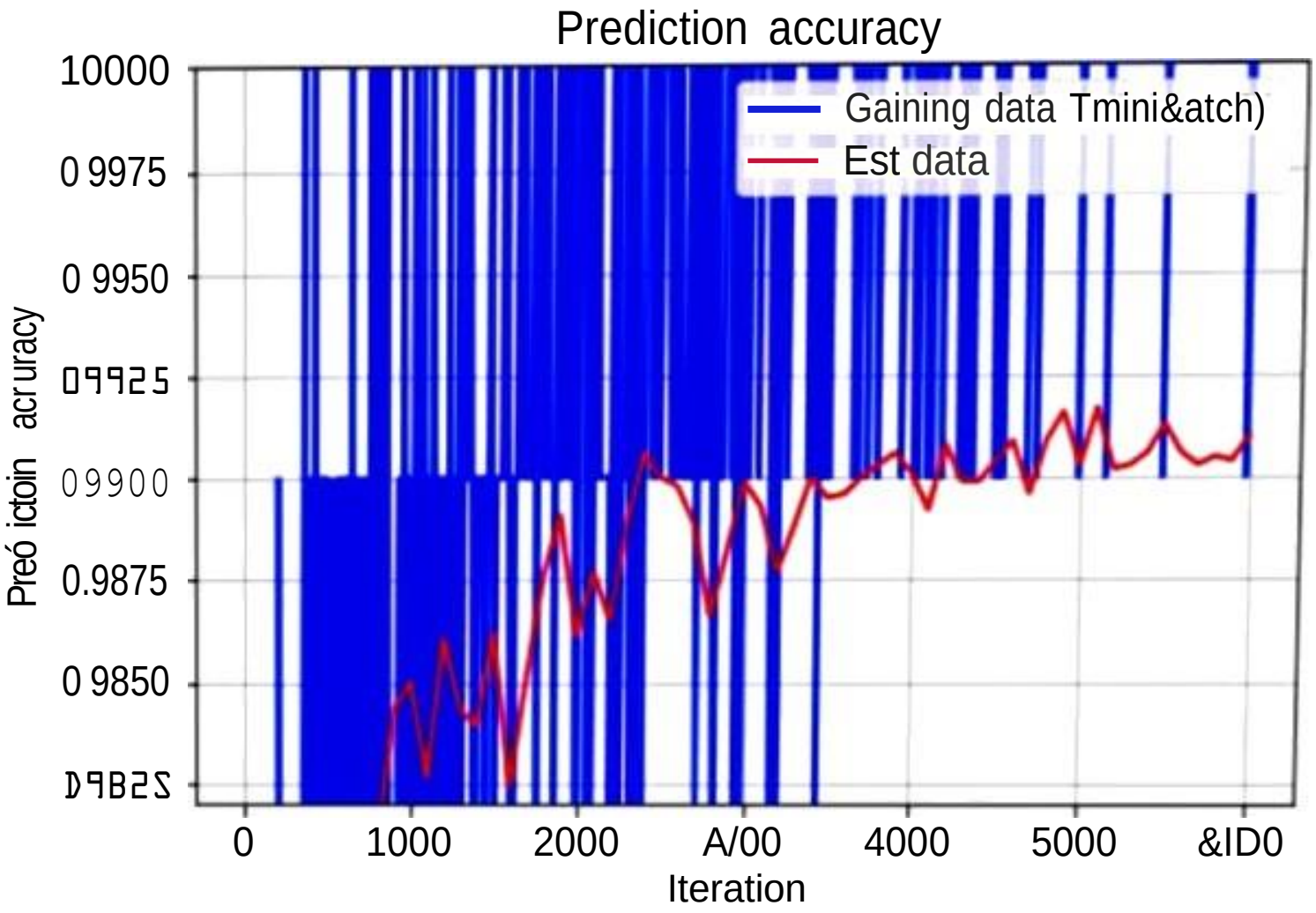

Prediction accuracy
Naming data fmini-batch)
Test data
Prediction accuracy
1.00
0.99
0.98
0.97
0.96
0
250
500
750
1000
1250
1500
1750
2000

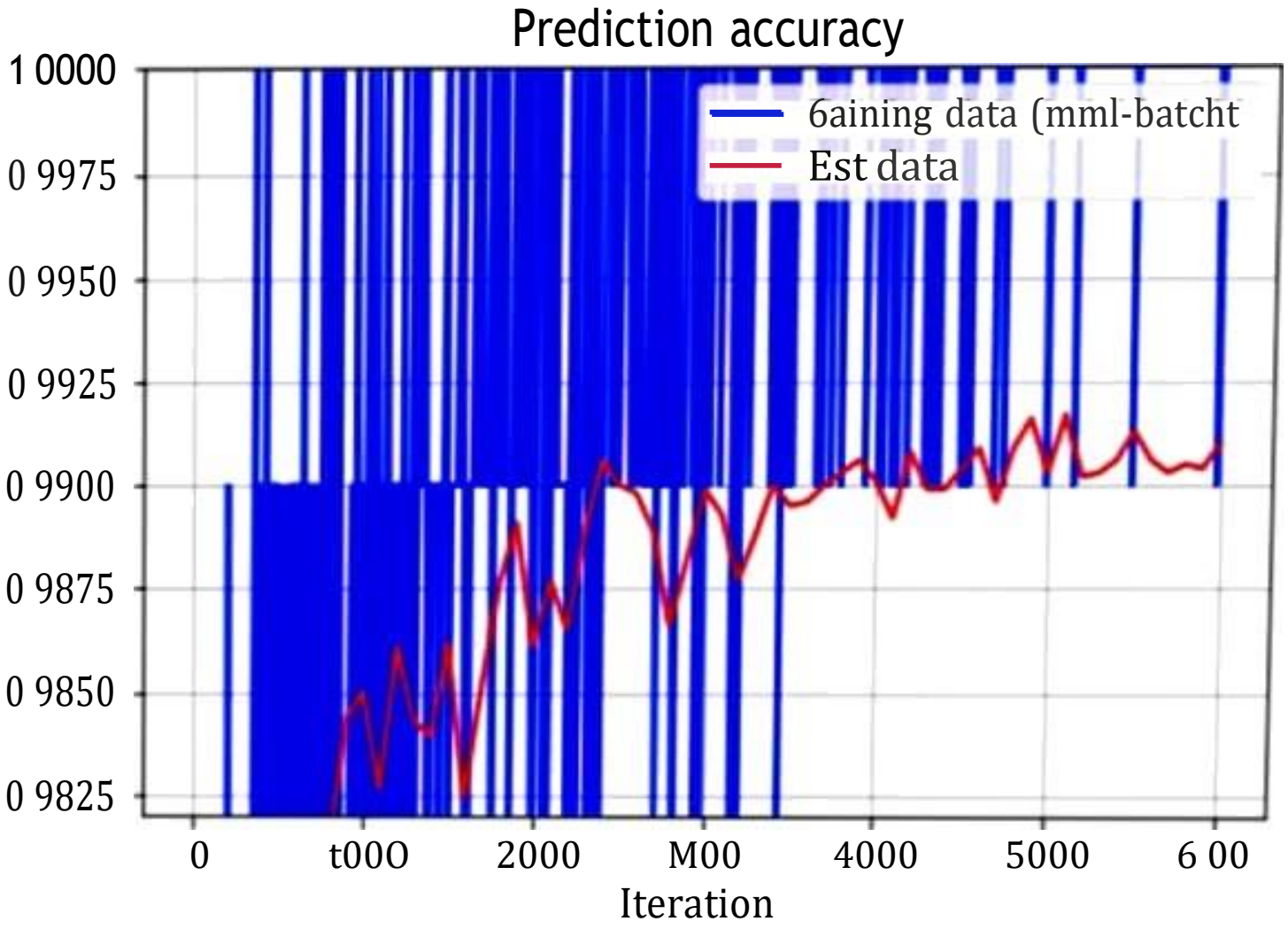

Prediction accuracy
Training data (mini-batch)
Test data
08
06
04
02
00
0
250
500
750
1000
T250
1500
1750
2000
Iteration


Prediction accuracy
1 00
0.99
098
097
0.96
095
Gaining data Amin batch›
Est data
0
250
500
750
1000
\250
1500
1750
2000
Iteration


Prediction accuracy
Gaining data (mini-batch
Est data
098
096
094
092
090
Prediction accuracy
o
xo◊
too
xo◊
® oo
# #
keaGon


Prediction accuracy
1 000
0 990
0.985
0.980
0.975
Prediction accu
Training data (mini-batch)
Test data
0
500
1000
1500
2000
2500
3000
3500
4000
Iteration


Prediction accuracy
10000
0 9975
0 9950
0.9900
0.9875
0 9850
0
1000
2000
4000
5000
Iteration
Gaining data Tmini&atch)
Est data


Prediction accuracy
1 0000
0 9975
0 9950
0 9925
0 9900
0 9875
0 9850
0 9825
6aining data (mml-batcht
Est data
0
t00O
2000
M00
4000
5000
6 00
Iteration