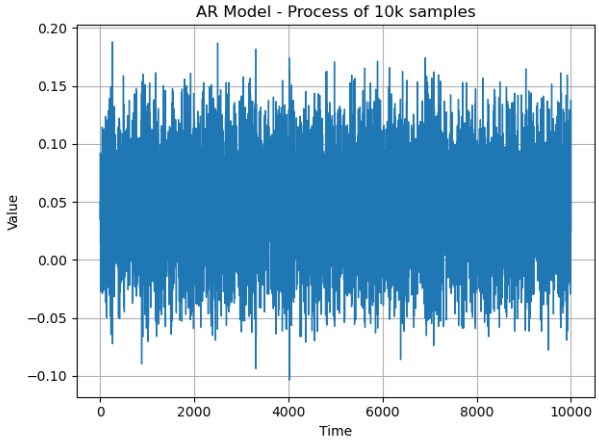
Homework Set - RNN Part 1

AR Model

Generated an AR model and created an RNN that predicts it.

$$X_t = a_1 X_{t-1} + a_2 X_{t-2} + a_3 X_{t-3} + U_t$$

Where $a_1 = 0.6, a_2 = -0.5, a_3 = -0.2$ and U_t is a uniform i.i.d noise w.p. $Uniform(0,0.1)$.

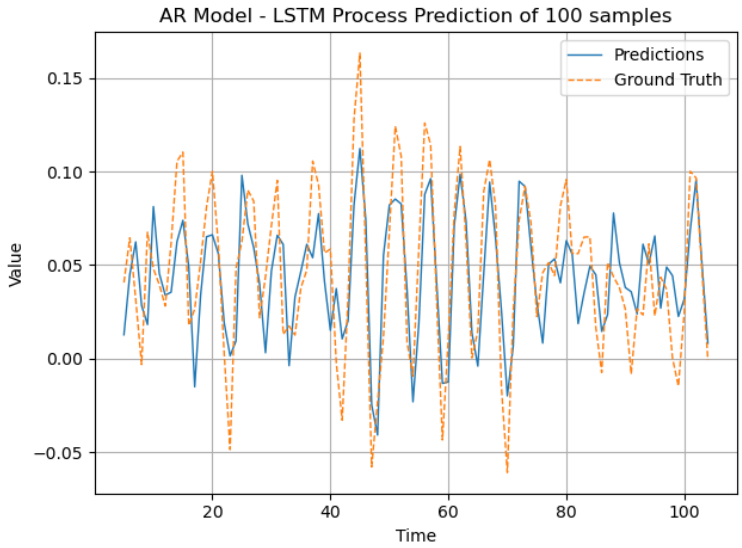


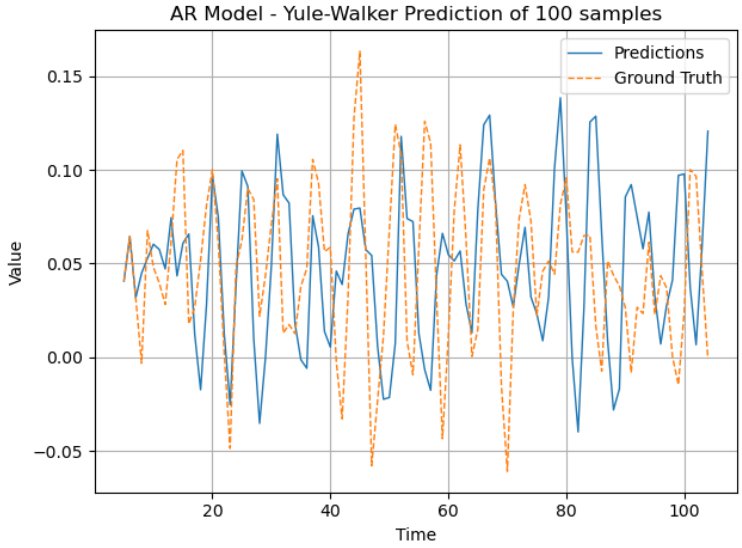
- We will check Yule-Walker estimator to confirm theoretical bounds.
  By using the samples $\{X_i\}_{i=1}^{N}$ we can estimate $\{a_i\}_{i=1}^{M}$.
  Thus, after solving the equations the minimal MSE is σ_U^2 , which is the variance of the noise. $\min(MSE) = \sigma_U^2 = \frac{0.1^2}{12} = 0.00083$.
- When running Yule-Walker estimator we got the exact coefficients that we created the process with, making it mean square error to be the minimum.
- Training on LSTM gave the same results, means that LSTM is at the minimum MSE.

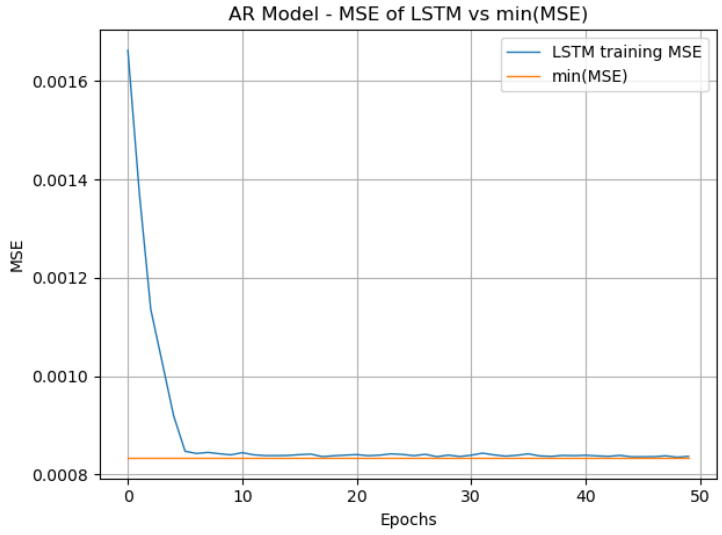
Results:

Used the following LSTM model –

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 5, 16)             1152
_________________________________________________________________
lstm_1 (LSTM)                (None, 16)                2112
_________________________________________________________________
dense (Dense)                (None, 1)                 17
=================================================================
Total params: 3,281
Trainable params: 3,281
Non-trainable params: 0
_________________________________________________________________
```







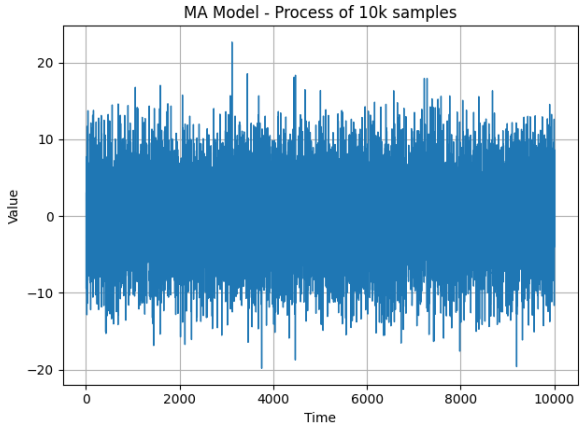
As we can see, the MSE converges with the minimum.

MA Model

Generated an MA model and created RNN to predict it.

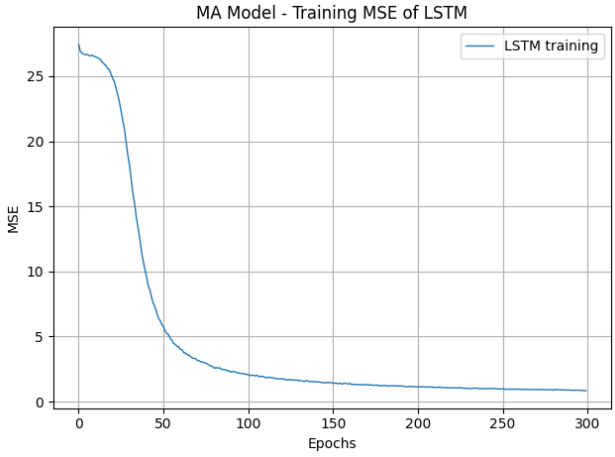
$$X_t = U_t + a_1 U_{t-1} + a_2 U_{t-2} + a_3 U_{t-3} + a_4 U_{t-4} + a_5 U_{t-5}$$

Where $a_1 = 5, a_2 = a_3 = a_4 = a_5 = -1$ and U_t is a normal i.i.d noise w.p. $Normal(0,1)$.

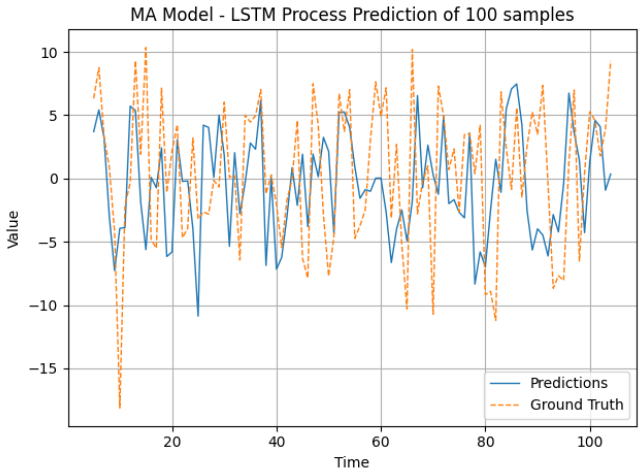


Results:

- After training the LSTM model we got the following results:

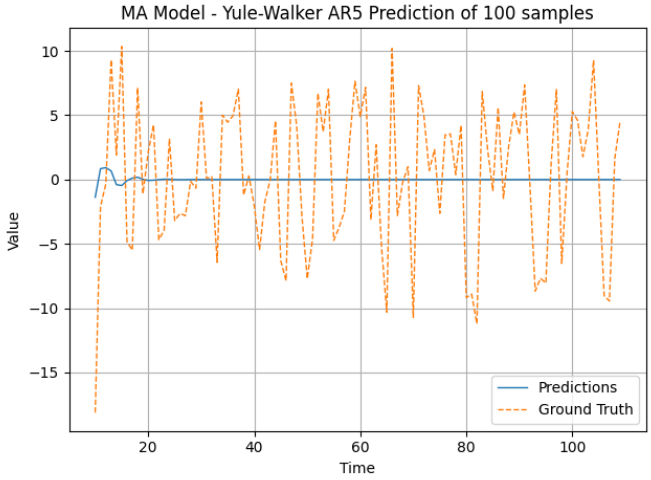


Minimal MSE is 0.8203 on the training dataset. As we can see the results are quite similar to the ground truth.

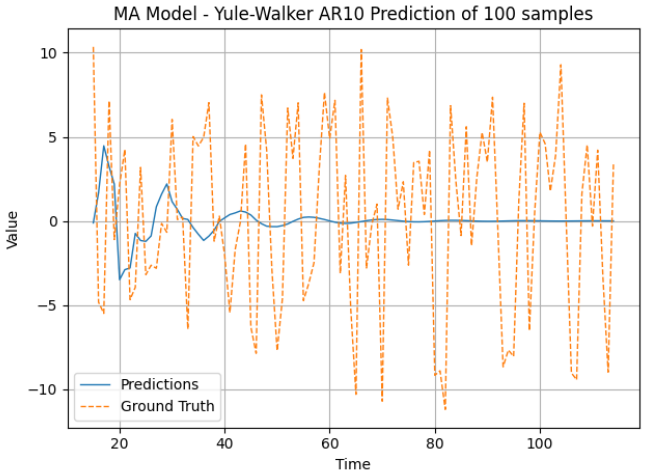


- Now we tried to sample the data from Yule-Walker equations for several numbers of coefficients.

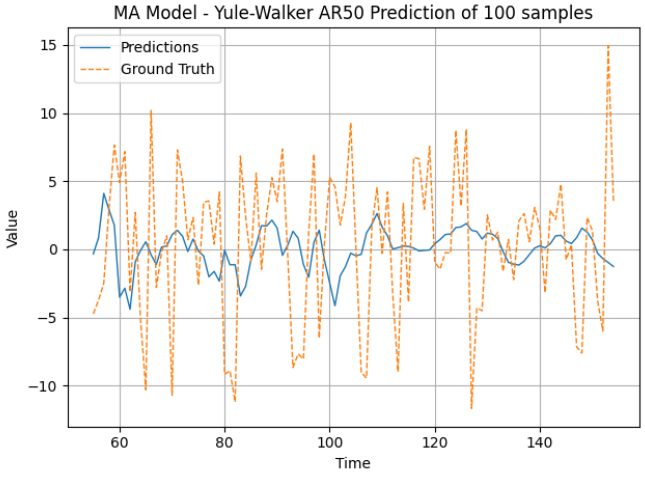
AR model of 5 coefficients:



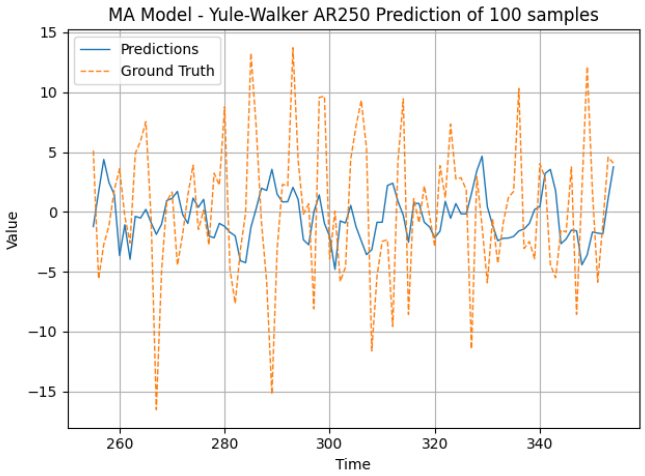
AR model of 10 coefficients:



AR model of 50 coefficients:

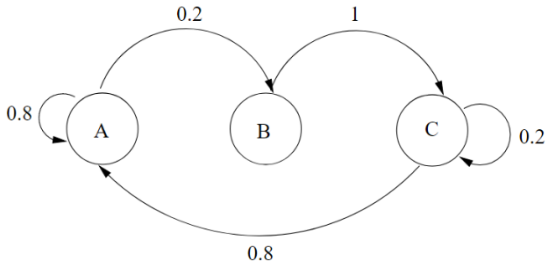


AR model of 250 coefficients:



As we can see, if we add more coefficients to the AR process, the better we generalize the MA process, which is consisting with the theory that MA process is AR process in order ∞.

Markov Chain



a. Calculation of the stationary distribution of the Markov Chain is:

The transition matrix is: $P = \begin{bmatrix} 0.8 & 0.2 & 0 \\ 0 & 0 & 1 \\ 0.8 & 0 & 0.2 \end{bmatrix}$

We can calculate the stationary vector $v = [v1, v2, v3]$ by solving the equation:

$$vP = v$$

Thus we get: $v = \left[\frac{5}{7.25}, \frac{1}{7.25}, \frac{1.25}{7.25}\right] = [0.689, 0.137, 0.172]$

b. The entropy rate of a stationary Markov process of order one is dependent of the last sample,

$$H(\mathcal{X}) = \lim_{n\to\infty} \frac{1}{n} H(X^n) = H(X_1|X_0) = \sum P(X_1 = i) \cdot H(X_2|X_1 = i) =$$

c. As for this Markov process we get:

$$\ldots = -\sum P(X_1 = i) \cdot \sum P(X_2 = j|X_1 = i) \cdot \log\big(P(X_2 = j|X_1 = i)\big) =$$

$$0.689 \cdot H(0.8,0.2,0) + 0.137 \cdot H(0,0,1) + 0.172 \cdot H(0.8,0,0.2) = 0.4315\ [bits]$$

d. The stationary distribution of this Markov Process is

$$p(A) = 0.689, p(B) = 0.137, p(C) = 0.172$$

The empirical distribution of the samples that we generated is calculated by the following equation:

$$p(x) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}_{\{X_i = x\}}, for\ x \in \{A, B, C\}$$

Thus, we get:

$$p(A) = 0.6878, p(B) = 0.13883, p(C) = 0.17337$$

e. The empirical entropy rate is obtained by using the empirical distribution probability, and the new transition matrix is as follows:

$$P = \begin{bmatrix} 0.79815 & 0.201846 & 0 \\ 0 & 0 & 1 \\ 0.80081 & 0 & 0.199180 \end{bmatrix}$$

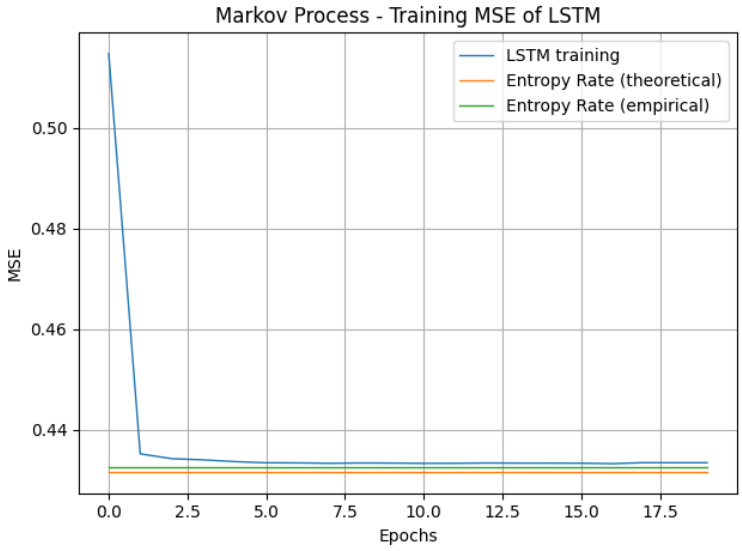
Thus,

$$H(\mathcal{X}) = 0.6878 \cdot H(0.79815,0.201846,0) + 0.13883 \cdot H(0,0,1) + 0.17337 \cdot H(0.80081,0,0.199180) = 0.43249\ [bits]$$

Which is close to the theoretical entropy rate (calculated with log based e).

f. After training LSTM network (as seen in the next figure) we got the following results:

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 5, 32)             4352
_________________________________________________________________
lstm_1 (LSTM)                (None, 16)                3136
_________________________________________________________________
dense (Dense)                (None, 3)                 51
=================================================================
Total params: 7,539
Trainable params: 7,539
Non-trainable params: 0
_________________________________________________________________
```



g. As we can see, the log loss converges to 0.4334 [bits] which is almost the empirical entropy rate we calculated before, thus it is the minimum loss we can achieve.

h. Let us calculate the **entropy rate for the i.i.d. case**:

$$H(\mathcal{X}) = \lim_{n\to\infty}\frac{1}{n}H(X^n) = \lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}H(x_i) = \lim_{n\to\infty}\frac{1}{n}n\cdot H(x) = H(x)$$

for $x_i \sim x\ ;\ i \in \{1,2,\dots n\}$.

The **negative log loss** is as follows:

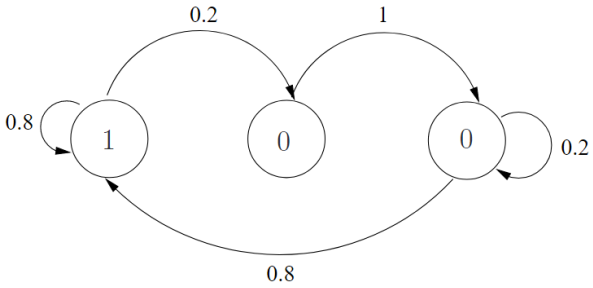
$$C(\mathcal{X}) = \frac{1}{N}\sum_{i=Samples}\left(\sum_{c=Class} -\log\big(Q(x_i)\big)\cdot P(x_i)\right)$$

for N the number of samples, $Q(x_i)$ is the network's prediction and $P(x_i)$ is the true distribution.

$$C(\mathcal{X}) \xrightarrow[n\to\infty]{} \mathbb{E}_P\big[-\log\big(Q(x)\big)\big] = H_{CE}\big(P(x),Q(x)\big) \geq H\big(P(x)\big) = H(x)$$

Example out vector:

Hidden Markov Model (Q4)



a. This is a hidden markov model.
If we take the previous question markov model, and define the probability of this process as $x(t)$ we can define a new probability $y(t)$ as follows:

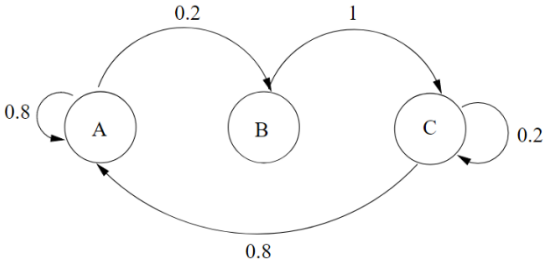
$$p(y_t|x_t = A) = \{1,0\} = \{p(y_t = 1|x_t = A, p(y_t = 0|x_t = A\}$$

$$p(y_t|x_t = B) = \{0,1\} = \{p(y_t = 1|x_t = B, p(y_t = 0|x_t = B\}$$

$$p(y_t|x_t = C) = \{0,1\} = \{p(y_t = 1|x_t = C, p(y_t = 0|x_t = C\}$$

In addition, we can observe that the new markov model is dependent on the last sample of the original markov model, which is the definition of HMM.

b. 1.



If we look at the previous markov model we define with A, B and C, we can define a one-to-one bijective function from this model to the HMM model consists with only 1 and 0's.

If the HMM has value of 1, then for sure we are at state A.

If HMM is at current state 0 and before is 1, then we are at state B.

When HMM is at current state 0 and before is 0, then we are at state C.

Thus there is one-to-one objective function from the Markov Model to HMM.

2. Proving that two r.v. that have one-to-one bejective function has the equal entropies:

$$H(X^n) =_{\{given\ Z^n, X^n is\ deterministic\}} = H(X^n) - H(X^n|Z^n) = I(X^n; Z^n)$$
$$= H(Z^n) - H(Z^n|X^n) =_{\{given\ X^n, Z^n is\ deterministic\}} = H(Z^n)$$

Where the equality to the Mutual Information is from the definition.

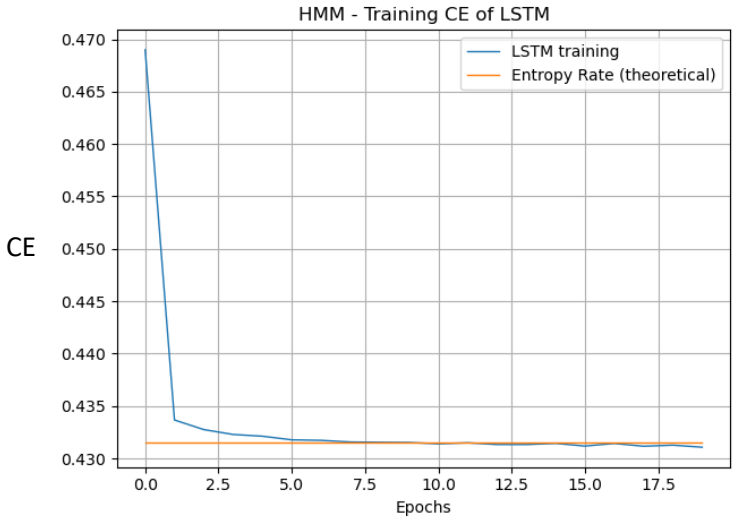
3. Now will prove that the entropy rate of X^n, Z^n is the same:

$$H(\mathcal{X}) = \lim_{n\to\infty}\frac{1}{n}H(X^n) =_{\{from\ seciton\ 2\}}= \lim_{n\to\infty}\frac{1}{n}H(Z^n) = H(\mathcal{Z})$$

c. We trained the LSTM model on the HMM model:

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 32)                4352
_________________________________________________________________
dense (Dense)                (None, 2)                 66
=================================================================
Total params: 4,418
Trainable params: 4,418
Non-trainable params: 0
_________________________________________________________________
```

We used 5 past samples to train it and got the following results,



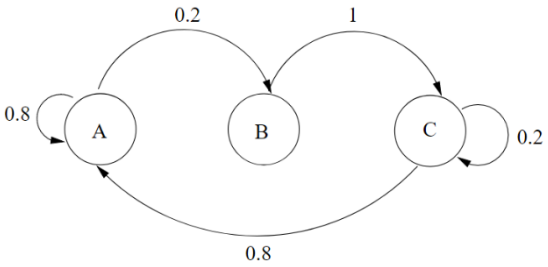
d. We can conclude that the cross entropy loss has reached it's minimum because the entropy rate is as of the previous question, which is 0.4315 [bits].

The results are presented here:

```
The model as viewed transition matrix is:
Trnasition | out=0 | out =1 |
-------------+---------+----------+
P(x|1):   [0.29367 0.70633]
P(x|0):   [0.30501 0.69486]
==============================
The model hidden transition matrix is:
Trnasition | out=0 | out =1 |
-------------+---------+----------+
P(x|1←1):  [0.29017 0.70998]
P(x|1←0):  [0.30791 0.69207]
P(x|0←1):  [0.29    0.70994]
P(x|0←0):  [0.31696 0.68309]
```

Hidden Markov Model (Q 5)

Considering the following markov model:



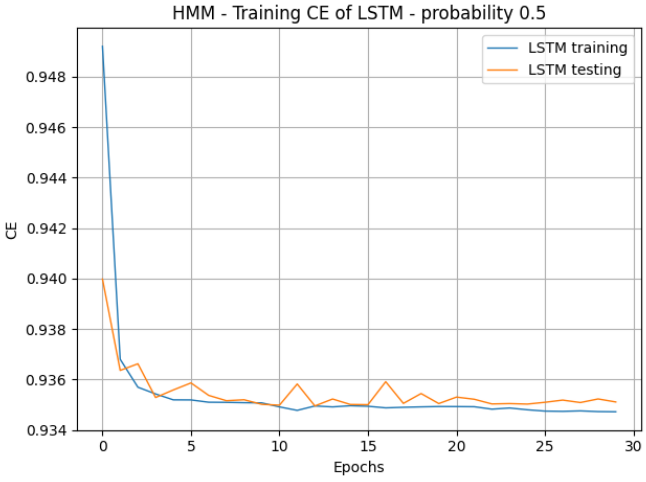
If the state is A the output of the channel Y is distributed according to $Bernouli(p)$. If the state is B or C the output is distributed according to $Bernouli(1-p)$ for the following probabilities - $p \in \{1,0.9,0.8,0.7,0.6,0.5\}$. As we can see given the probability $p = 1$ we get the hidden markov model in Q4 which we proved that we can identify each state using two last time step.

We created a model using '*past*' samples from previous time steps, to predict the correct state of each one of them, by using LSTM.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 5, 32)             4352
_________________________________________________________________
lstm_1 (LSTM)                (None, 5, 16)             3136
_________________________________________________________________
dense (Dense)                (None, 5, 3)              51
=================================================================
Total params: 7,539
Trainable params: 7,539
Non-trainable params: 0
_________________________________________________________________
Train on 100000 samples, validate on 100000 samples
```

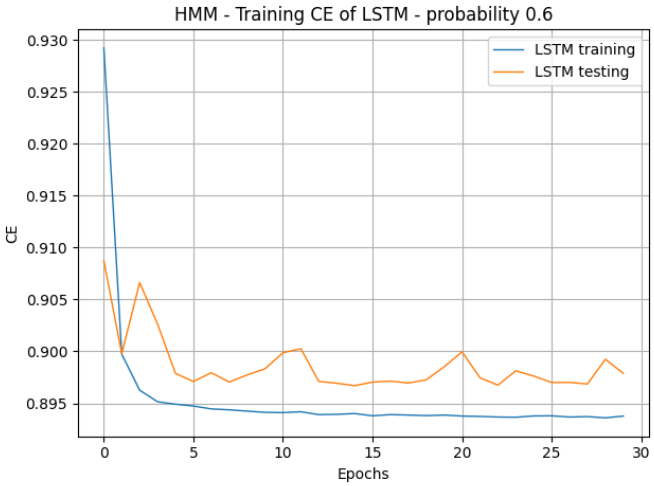
The output is different now, because we want to predict the current state and not the next. Thus, using the all the outputs of the LSTM will help us to correct the net in addition to the time dependent values.

We will expect to see that for higher probability of p, the lower the cross entropy. As for $p = 0.5$ we will expect that the model will classify the state by random, because there is no information to figure out the correct state.

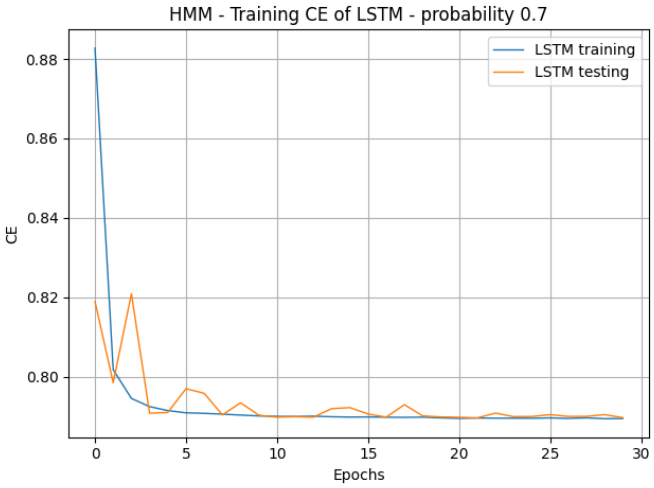


loss: 0.9347 - accuracy: 0.4599 -
val_loss: 0.9351 - val_accuracy: 0.4541

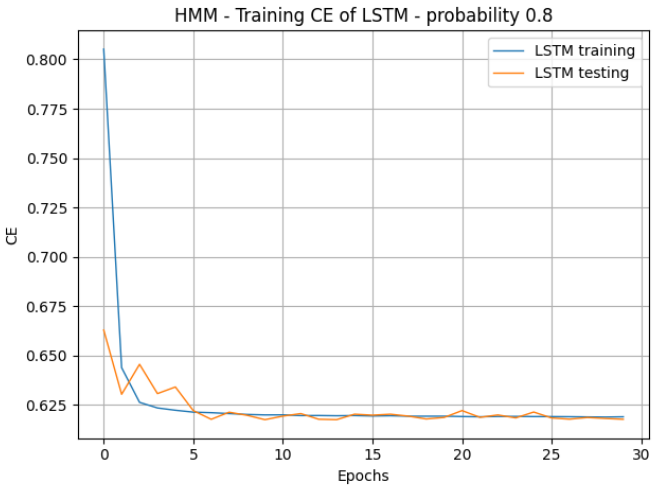
loss: 0.8938 - accuracy: 0.5582 -
val_loss: 0.8979 - val_accuracy: 0.5541



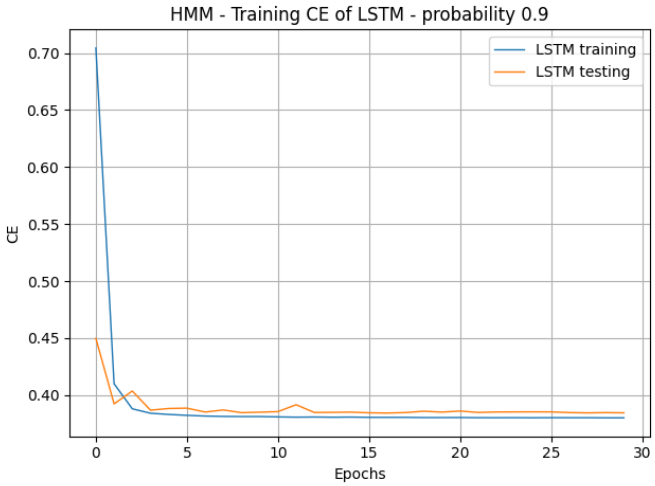
loss: 0.7894 - accuracy: 0.6479 -
val_loss: 0.7897 - val_accuracy: 0.6487



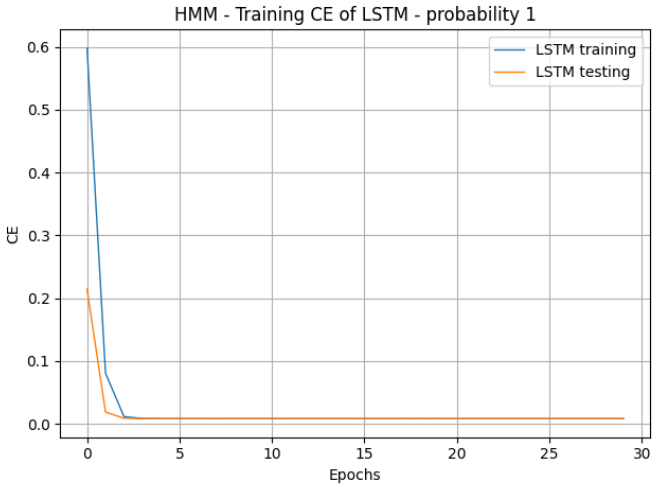
loss: 0.6190 - accuracy: 0.7422 -
val_loss: 0.6177 - val_accuracy: 0.7424



loss: 0.3812 - accuracy: 0.8630 -
val_loss: 0.3850 - val_accuracy: 0.8621



loss: 0.0085 - accuracy: 0.9968 -
val_loss: 0.0084 - val_accuracy: 0.9969



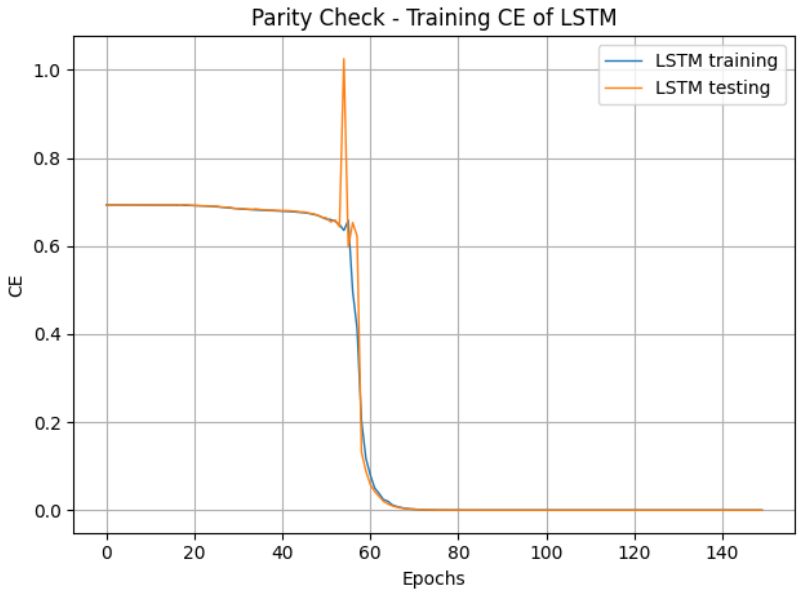
The results are as we expected.

Parity Check

Built a neural network that gets an arbitrary sequence of binary numbers and the output is a binary sequence such that it indicates if the number of '1' till now is even or odd.

Working with samples at length 100 each, and trained the model for 150 epochs

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 100, 64)           16896
_________________________________________________________________
dense (Dense)                (None, 100, 1)            65
=================================================================
Total params: 16,961
Trainable params: 16,961
Non-trainable params: 0
_________________________________________________________________
Train on 1000 samples, validate on 100 samples
```



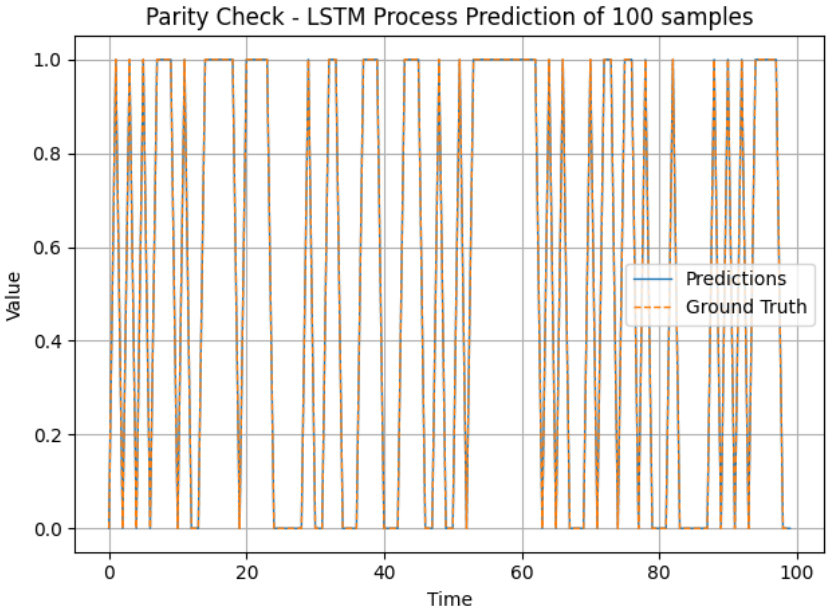
As we can see after 60 epochs the loss dropped to definitely zero.

```
Epoch 150/150 - loss: 1.4592e-08 - accuracy: 1.0000 - val_loss: 1.5658e-08 - val_accuracy: 1.0000

Test loss:              0.000000

Test accuracy:          100.00%
```

We can see that the prediction is perfect:

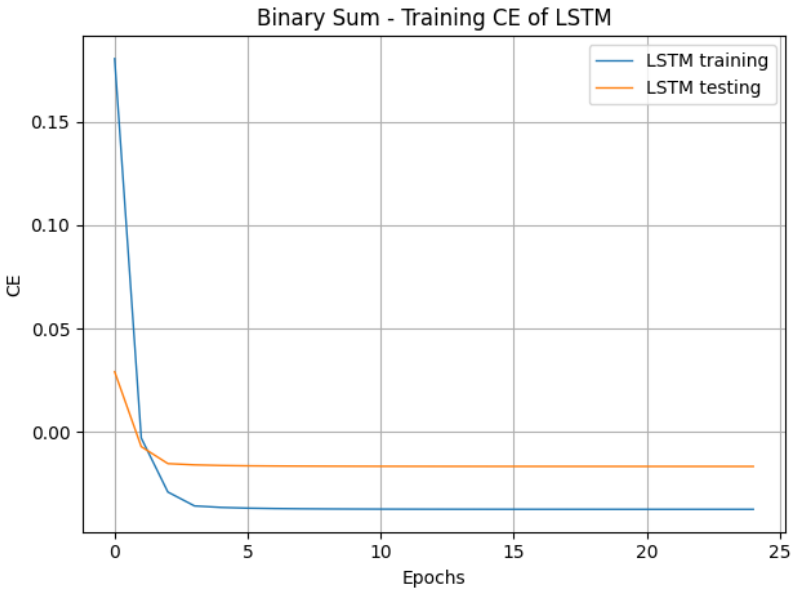


Summing two numbers

Built a neural network that gets two sequences of binary number and the output is a binary sequence that is a a sum of the two.

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, None, 128)         67072
_________________________________________________________________
dense_1 (Dense)              (None, None, 1)           129
=================================================================
Total params: 67,201
Trainable params: 67,201
Non-trainable params: 0
_________________________________________________________________
Train on 10000 samples, validate on 100 samples
```

The accuracy on the train data is 0.9975 and for the validation is 0.9989.



The predictions are great as well: