COL 774: Machine Learning
Assignment 4

Team Name: Inbox
Name: Suraj Reddy
Entry Number: 2025AIY7586
Name: Sai Gangadhar
Entry Number: 2025AIY7588

1 RNN Implementation

Training setting:

- dataset setup: train has 80k points and test has 20k points.
- validation is not used.
- trained for 20 epochs as mentioned in Assignment.

Hyperparameter	Value
batch_size	32
epochs	20
lr	1×10^{-4}
dropout	0
embedding_dim	128
hidden_dim	512
nof_rnn_layers	2
optimizer	adam
pad_idx	0
sos_idx	1
eos_idx	2
teacher_forcing_ratio	0.5
attention	256

Table 1: Training hyperparameters used for the RNN model.

Comments

- **Poor Generalization:** While the RNN achieves strong training performance (94.17% token accuracy and 72.27% sequence accuracy), its test metrics drop sharply, with a high test loss (2.4199) and reduced token accuracy (68.45%). This indicates significant overfitting and difficulty in generalizing to unseen mazes.

Metric	Value
Train Loss	0.1910
Test Loss	2.4199
Train Token Accuracy	94.17%
Test Token Accuracy	68.45%
Train Sequence Accuracy	72.27%
Test Sequence Accuracy	63.59%
Train F1 Score	94.17%
Test F1 Score	68.45%

Table 2: Final Training and test performance metrics.

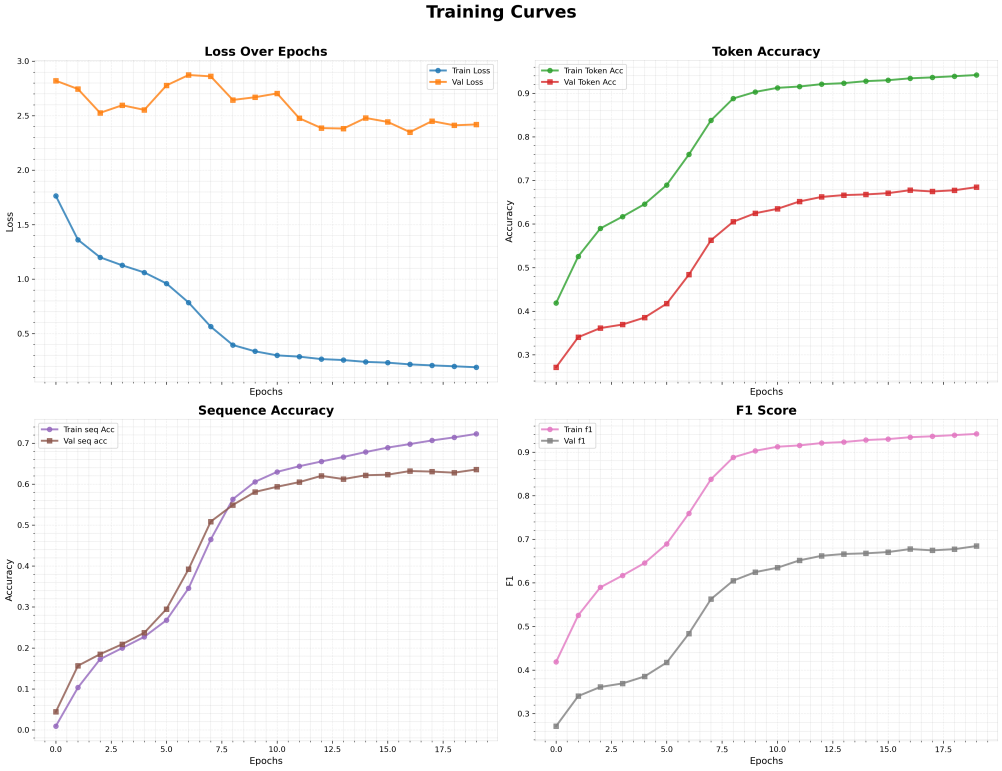

Figure 1: Training Curves.

- **Struggles with Long-Range Dependencies:** The large gap between training and testing performance suggests that the RNN is unable to reliably capture long-range structural relationships in the maze representation. This limitation results in unstable predictions and lower overall sequence accuracy during evaluation.

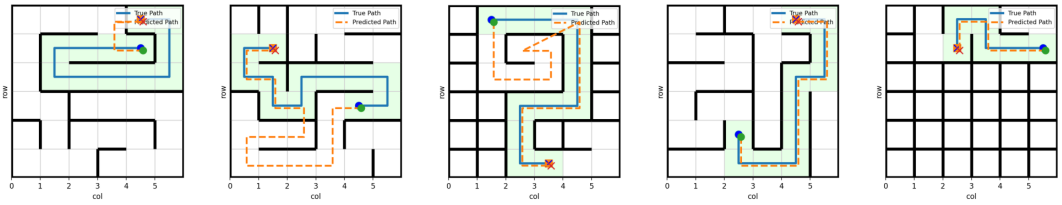

Figure 2: 5 Random prediction visualisation

2 Training Transformer

Training setting:

- dataset setup: train has 72k points and test has 20k points.
- validation is used.
- trained for max 40 epochs with early stop of PATIENCE 5.

Hyperparameter	Value
D_MODEL	128
NHEAD	8
NUM_LAYERS	6
DIM_FEEDFORWARD	512
DROPOUT	0.1
EPOCHS	40
PATIENCE	4
LR	5×10^{-4}

Table 3: Transformer Hyperparameters

Metric	Value
Train Loss	0.1514
Test Loss	0.2229
Train Token Accuracy	93.09%
Test Token Accuracy	88.99%
Train Sequence Accuracy	58.21%
Test Sequence Accuracy	47.00%
Train F1 Score	93.08%
Test F1 Score	89.95%

Table 4: Final Training and test performance metrics.

Comparison of RNN and Transformer Models. From Tables 1 and 2, we observe clear differences in how the RNN and Transformer models behave during training and evaluation.

- **Generalization:** The Transformer achieves significantly better generalization, as indicated by its low test loss (0.2229) compared to the RNN (2.4199). Similarly, the Transformer attains much higher test token accuracy (88.99%) and test F1-score (89.95%), whereas the RNN drops to 68.45%. This shows that the Transformer is more robust to unseen mazes and captures long-range dependencies in the maze structure more effectively.

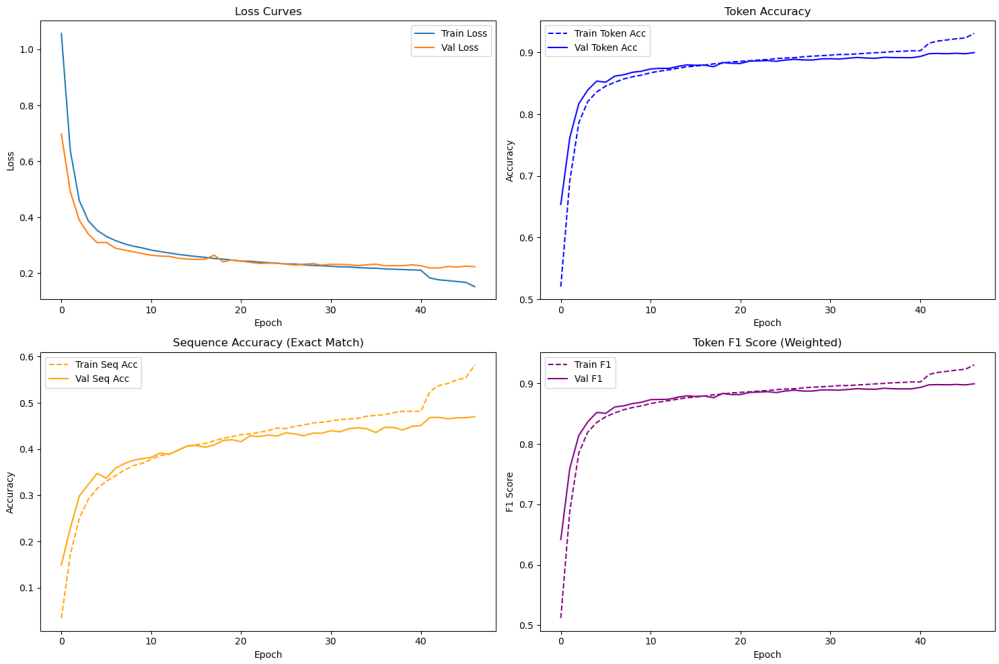

Figure 3: Training Curves

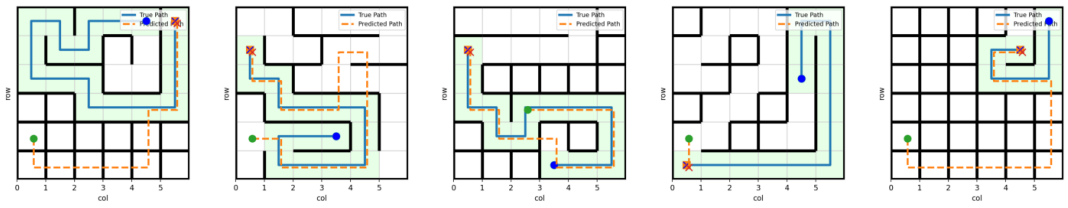

Figure 4: Transformer : 5 Random prediction visualisation

- **Token-Level vs Sequence-Level Performance:** Although the Transformer performs better at the token level, both models struggle with full sequence accuracy. The Transformer improves test sequence accuracy to 47.00%, but this is still relatively low, indicating that exact path reconstruction remains challenging. The RNN performs even worse at 63.59% token accuracy and 63.59% F1 but achieves slightly higher sequence ac-

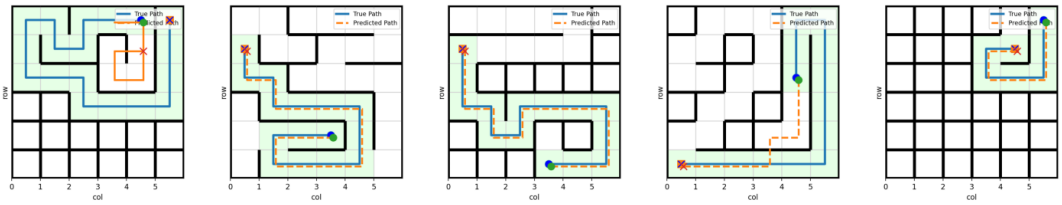

Figure 5: RNN : 5 Random prediction visualisation

curacy (63.59%) than its token agreement would suggest, implying that when it predicts correctly, it often gets full path segments right.

- **Training Dynamics:** The RNN shows a much larger gap between train and test performance, indicating overfitting. Its train loss is very low (0.1910) and train accuracy is high (94.17%), but this does not translate to test performance. The Transformer, on the other hand, maintains a small train–test gap across all metrics, showing stable training and better inductive bias for this task.

- **Overall Observation:** The Transformer clearly outperforms the RNN in all critical metrics, particularly in generalization and token-level correctness. This aligns with the expectation that attention-based architectures handle long-range structural dependencies—such as maze layouts—much better than recurrent models.