LAB 4

Steps Followed:

1. Step Up and data loading
    - Verified the 2.x tensorflow version
    - Downloaded the data from the Kaggle as the blog data was giving error. The Kaggle data is easy to download and is derived from the blog data itself.
2. Data Preparation
    - Loading the notMNIST_small dataset.
    - Preprocessed by resizing to (28,28), converted to grayscale, normalizes pixel values and using 80/20 split for training and validation.
3. Custom RNNCell Implementation
    - Implemented GRUCell and MGUCell from scratch by subclassing BaseRNNCell.
    - Each implements their respective gate mechanisms: GRU uses update/reset gates and MGU uses a single forget gate.
4. Model Building
    - Created build_rnn_model to stack multiple custom cells.Inputs reshaped from (28, 28, 1) to (28, 28) using Lambda and the final output layer: dense with 10 units (logits).
5. Training Procedure
    - Defined compile_and_train to compile and train the models.
    - For each config: Ran 3 trials with different seeds and with layers 1,2,3 and hidden layer of 50,128,256
    - Saved all training histories into JSON files for GRU and MGU.
6. Evaluation
    - Loaded both GRU and MGU results and or each configuration, calculated: Mean validation accuracy over 3 trials, Standard deviation (± std), and Classification error (1 - accuracy).

7. Visualization
    - Plotted Validation Accuracy and Validation Loss over epochs.

Results:
Each row in the table shows results for a specific configuration of:

- Model Type i.e GRU and MGU
- Number of layers: 1l, 2l, 3l
- Hiden size: 50h, 128h, 256h

Each model is trained and evaluated 3 times with random seeds.
For each configuration, the following metrics are reported:

1. Accuracy: The average validation accuracy over the 3 trials. The higher the accuracy, the better it is.
2. Standard Deviation: The standard deviation is calculated of the validation accuracy across the 3 trials i.e. lower = more stable performance.
3. Error: The classification error, calculated as 1 - accuracy and interpreted as lower is better.

Key Findings:

Final Classification Accuracy (Validation Set):

Model	Accuracy	± Std	Error
GRU_1l_50h	0.9181	0.0010	0.0819
GRU_1l_128h	0.9292	0.0023	0.0708
GRU_1l_256h	0.9397	0.0020	0.0603
GRU_2l_50h	0.9322	0.0018	0.0678
GRU_2l_128h	0.9348	0.0032	0.0652
GRU_2l_256h	0.9379	0.0008	0.0621
GRU_3l_50h	0.9352	0.0021	0.0648
GRU_3l_128h	0.9375	0.0042	0.0625
GRU_3l_256h	0.9399	0.0020	0.0601
MGU_1l_50h	0.9066	0.0021	0.0934
MGU_1l_128h	0.9184	0.0025	0.0816
MGU_1l_256h	0.9269	0.0024	0.0731
MGU_2l_50h	0.9205	0.0009	0.0795
MGU_2l_128h	0.9282	0.0008	0.0718
MGU_2l_256h	0.9315	0.0007	0.0685
MGU_3l_50h	0.9249	0.0030	0.0751
MGU_3l_128h	0.9355	0.0035	0.0645
MGU_3l_256h	0.9396	0.0041	0.0604

Figure 1: For Final Classification Accuracy on Validation Set

1. GRU Performance: The GRU model performed well with an increased number of layers and hidden units. For GRU, with 3 layers and 256 hidden units the accuracy was 0.9399, Standard deviation of 0.0020 and Error of 0.0601. It concludes that the GRU accuracy increases consistently from 1 layer to 3 layers and from 50 to 256 hidden units.

2. MGU Performance: The MGU was second in comparison with the GRU performance. The MGU model wit 3 layers and 256 hidden units have an accuracy of 0.9396, error of 0.0604 and standard deviation of 0.0041. MGU benefits more from larger hidden sizes and deeper networks.

3. Comparison of both MGU and GRU: In the terms of accuracy, GRU outperforms MGU by a little bit difference for the most configurations. For the Stability, GRU tends to have lower standard deviation and consistently performs well across the different trials. But MGU is faster as it has fewer gates and is preferred in resource-constrained environment.

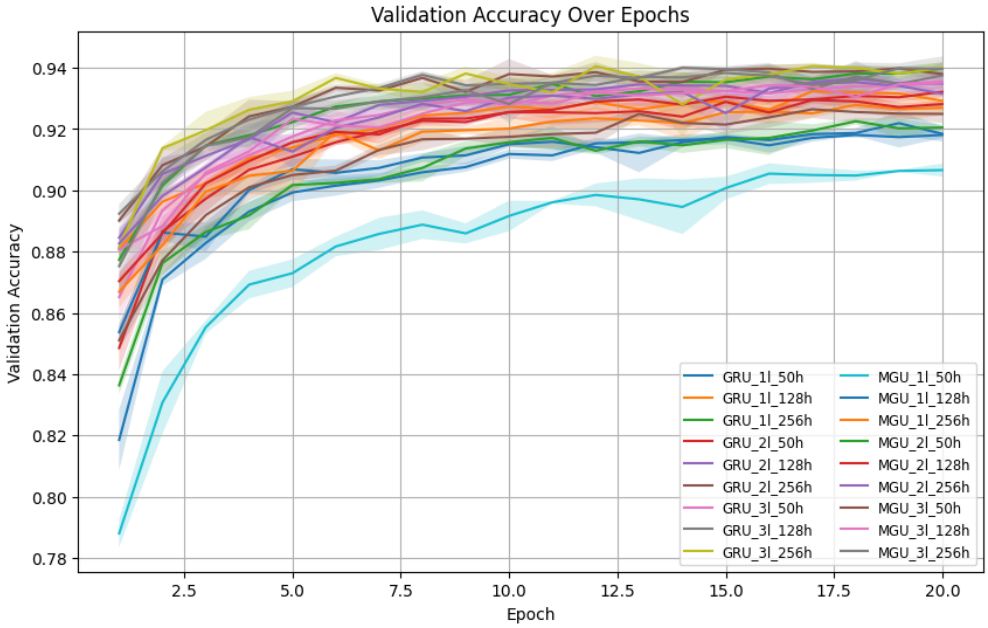

Figure 2: Validation Accuracy over Epochs

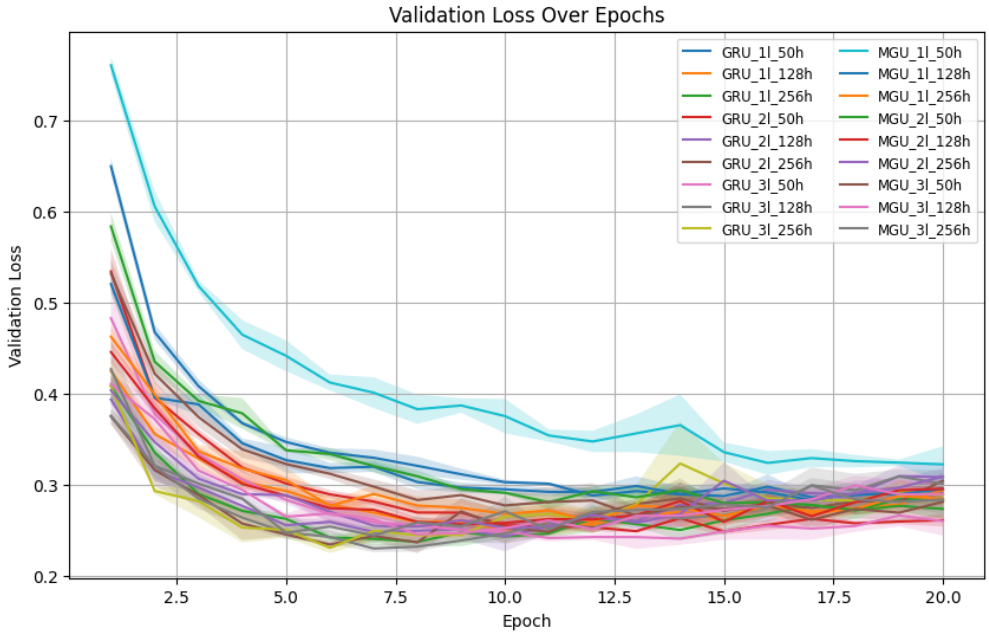

Figure 3: Validation Loss over Epochs

For the graph, here is the following explanation:

1. Validation Accuracy over Epochs : All the models improves accuracy over the 20 training epochs. GRU models reached the highest validation accuracy of 0.9399 with 3 layers and 256 hidden unit. The MGU model did perform well but had a slightly lower accuracy than GRU and more visible variance, particularly in 1 layer and 50 hidden state. The gap between GRU and MGU is more apparent in smaller models (e.g., 1 layer, 50 hidden units), but narrows as the models scale up.

2. Validation Loss over Epochs: Both GRU and MGU showed a smoth and steady decrease in loss, which confirms the effective learning. GRU models had a lower final loss for deeper variants and the MGU initial model 1 layer 5- hidden units started with higher initial loss and decreased slowly, i.e. the early training was less efficient for the lighweight configurations.

In conclusion, the GRU outperforms the MGU in both accuracy and loss, especcially for smaller models. Training stabilizes for both architectures after 10 epochs, which could help in tuning training time in the future. Both models GRU and MGU with 3 layer and 256 hidden units are the best overall performers in your experiments.

Link to google colab: Colab file here

Link to github : GitHub link here