Lab 2 Report: Surname Classification Using a Character RNN

1 Task 1

1.1 Introduction

Surname classification is a sequence modeling problem where the goal is to predict the nationality or origin of a surname based on its character sequence. In this experiment, we explore different recurrent neural network (RNN) architectures, namely `nn.RNN`, `nn.LSTM`, and `nn.GRU`, while varying the input size and hidden size hyperparameters. The objective is to find the best model based on test loss and test accuracy.

1.2 Experiment Setup

We train three types of RNNs with different configurations of input size (character embedding size) and hidden size. The models are evaluated on test loss and test accuracy. The experimental settings are as follows:

Models	RNN, LSTM, GRU

Hyperparameters	Input size: {100, 150, 200}
	Hidden size: {64, 128, 256, 512}

Evaluation Metrics	Test loss, Test accuracy

Table 1: Model configurations and evaluation metrics.

1.3 Results

The test loss and accuracy for each configuration are summarized in the tables below.

RNN Results

Hidden Size	**Input Size 100**	**Input Size 150**	**Input Size 200**
64	1.58 / 53.50*	1.45 / 51.37	1.42 / 52.25
128	1.51 / 56.12	1.51 / 55.94	1.47 / 52.56
256	1.42 / 56.06	1.50 / 56.69	1.42 / 55.13
512	1.52 / 50.94	1.50 / 61.31	1.55 / 59.81

Table 2: RNN Test Loss / Accuracy

GRU Results

Hidden Size	**Input Size 100**	**Input Size 150**	**Input Size 200**
64	1.38 / 54.19*	1.47 / 57.75	1.43 / 56.88
128	1.39 / 55.25	1.55 / 59.75	1.94 / 61.50
256	1.43 / 56.56	1.69 / 60.75	1.80 / 61.25
512	1.50 / 58.00	1.59 / 59.18	1.67 / 61.06

Table 3: GRU Test Loss / Accuracy

LSTM Results

1.4 Analysis and Discussion

From the results, we observe the following trends:

Hidden Size	**Input Size 100**	**Input Size 150**	**Input Size 200**
64	1.43 / 54.19*	1.93 / 65.19	2.36 / 65.31
128	1.49 / 54.50	2.09 / 63.25	2.51 / 66.44
256	1.54 / 57.50	1.93 / 62.56	2.91 / 67.25
512	1.67 / 59.94	1.76 / 60.81	2.90 / 67.31

Table 4: LSTM Test Loss / Accuracy

- RNN: The accuracy remains relatively stable across different hidden sizes, with the highest accuracy (61.31%) observed at (input size = 150, hidden size = 512). However, the loss values do not show a clear decreasing trend, indicating limited learning capability.
- GRU: GRU outperforms standard RNN in most cases, achieving its best accuracy (61.50%) at (input size = 200, hidden size = 128). The test loss is generally lower than that of LSTM, indicating better generalization in some configurations.
- LSTM: LSTM exhibits the best accuracy overall, peaking at 67.31% (input size = 200, hidden size = 512). However, it also has the highest test loss in some cases, suggesting potential overfitting.

Overall, LSTM achieves the highest accuracy but may require further regularization to prevent overfitting. GRU provides a balance between performance and generalization, making it a competitive choice. Standard RNNs perform the worst due to their limited capacity to handle long-range dependencies.

2 Task 2

2.1 Introduction

In this experiment, we compare different methods of extracting the final representation from the RNN outputs: the original approach using *column_gather()*, a mean aggregation method, and a max aggregation method implemented using *column_summation()*.

2.2 Methodology

We modify the original RNN model by replacing *column_gather()* with *column_summation()* using both "mean" and "max" modes. The modified function aggregates all valid hidden states, ignoring those generated from padding tokens. The evaluation metrics used are the model loss and accuracy.

2.3 Results

Table 5 presents the results obtained for each method.

Method	Loss	Accuracy (%)
Original (*column_gather()*)	1.58	53.50
Mean Aggregation	1.36	59.13
Max Aggregation	1.37	56.75

Table 5: Performance comparison of different output aggregation methods in the RNN model.

2.4 Discussion

From the results, it is evident that using mean aggregation achieves the lowest loss (1.36) and the highest accuracy (59.13%). This suggests that averaging over all valid hidden states provides a more informative representation compared to selecting only the last valid hidden state (original method) or using the maximum hidden state value. The max aggregation method, while improving over the original approach, does not perform as well as the mean method.