NLU course projects: Lab. 4 (LM)

Andrea Richichi — University of Trento
andrea.richichi@studenti.unitn.it - 257850

1. Introduction

This report presents a two-part study on neural language modeling, done independently for the NLU course. Part 1 compares a basic recurrent neural network (RNN) to a more capable long short-term memory (LSTM) model. Part 2 focuses on improving the LSTM with advanced training techniques.

The task is next-word prediction, based on context. All models are trained on the Penn Treebank (PTB) dataset, with perplexity (PPL) used as the main evaluation metric.

The goal is to better understand both standard and more refined approaches to language modeling, and to test methods for improving generalization and training stability.

2. Implementation Details

All models are trained using cross-entropy loss and evaluated on token-level perplexity. Early stopping with a patience of three epochs is used to prevent overfitting.

2.1 Part 1: RNN and LSTM Baselines

The first part compares two foundational architectures: a simple RNN and a basic LSTM.

The **RNN-based model** consists of:

- An embedding layer for token representation,
- A single-layer RNN,
- A linear layer projecting to vocabulary size.

This setup is trained with SGD and no regularization. It serves mainly as a baseline.

The **LSTM-based model** uses the same structure but replaces the RNN with an LSTM, which is better at handling long-range dependencies. I tested:

- LSTM + SGD (no dropout),
- LSTM + SGD with dropout ($p = 0.3$),
- LSTM + AdamW with dropout.

All use 300-dimensional embeddings, 200 hidden units, and gradient clipping to improve stability.

2.2 Part 2: Advanced LSTM Techniques

In the second part, I focused on improving the LSTM model with more recent and effective techniques.

First, I added **locked (variational) dropout**, which applies a consistent dropout mask across time steps, offering more stable training. Then I used **weight tying**, where the output projection shares weights with the input embeddings, reducing parameters and often helping generalization.

I also tested **NT-ASGD**, an SGD variant that starts averaging model weights once validation perplexity plateaus. This improves the final model by reducing variance.

The configurations tested were:

- LSTM + SGD with weight tying,
- LSTM + SGD + weight tying + locked dropout ($p = 0.2$),
- LSTM + NT-ASGD + dropout ($p = 0.4$) with a larger hidden size (400).

All models were trained for up to 200 epochs, using early stopping based on development perplexity.

3. Results

Part 1: As expected, the RNN model has the highest perplexity, due to its limited capacity. Switching to LSTM gives a clear improvement, and adding dropout improves generalization further. The best results come from the configuration using LSTM + AdamW + dropout.

Part 2: Weight tying reduces parameters and performs well. Locked dropout increases training stability. The NT-ASGD setup achieves the best perplexity overall, showing the benefit of combining adaptive optimization with regularization.

4. Conclusion

This project provided a hands-on exploration of different language modeling architectures. In Part 1, I confirmed that LSTM outperforms RNNs, especially when combined with dropout and AdamW. In Part 2, I showed that adding weight tying, locked dropout, and NT-ASGD leads to further gains.

Architecture	LR	Epochs	Dev PPL	Test PPL
RNN + SGD	0.5	52	181.3590	172.5554
	1.0	**42**	**168.8981**	**161.3865**
	1.5	29	173.0624	165.4523
	2.0	24	175.5462	167.4353
LSTM + SGD	1.0	72	159.1089	152.1228
	2.0	39	156.1992	150.5988
	3.0	**35**	**149.8601**	**144.7272**
	4.0	29	151.9360	145.6093
LSTM + SGD + Dropout	1.0	121	150.2659	145.3965
	2.0	94	141.9746	135.8986
	3.0	97	**132.3564**	**127.9176**
	4.0	87	131.7785	128.4177
LSTM + AdamW + Dropout	**0.0005**	**111**	**122.2937**	**112.0377**
	0.001	70	124.6117	113.3791
	0.002	36	124.3932	115.4522
LSTM + SGD + Weight Tying	1.0	114	139.8593	135.3900
	1.5	82	134.1340	130.7502
	2.0	**73**	**128.7227**	**125.4202**
	3.0	47	129.7516	126.9898
	4.0	31	133.1347	129.9217
LSTM + SGD + WT + Var Dropout	1.0	160	125.4421	121.6137
	2.0	110	112.0674	108.3275
	3.0	**70**	**105.1763**	**102.7415**
	4.0	60	109.2876	106.0312
LSTM + NTAvSGD + WT + VD	**30.0**	**32**	**97.0330**	**94.3286**

Table 1: Perplexity scores for all architectures. Best row per block is shaded in gray; best overall in yellow.