Project 1: Language Modeling Report

Yash Dagade*

Introduction

In this assignment, I built upon a 2-layer GRU language model for the WikiText-2 dataset. Initially, I started with a baseline configuration (hidden size 768, embedding size 128, dropout 0.5, batch size 64, and no learning rate schedule), obtaining a validation perplexity (PPL) of around 130. This was following the directions to a tee. I then introduced multiple modifications, including higher hidden/embedding sizes, embedding dropout, weight decay, a learning rate scheduler, L2 regularization using Adam, and a larger batch size. Below I describe the major changes and present the final results.

Modifications and Rationale

1. Larger GRU and Embedding Dimensions. I increased the GRU hidden size from 768 to 1024 and then 2048, and the embedding size from 128 to 512, aiming to capture more capacity. Although bigger models risk overfitting, I combined them with heavier dropout and weight decay.

2. Higher Dropout (up to 0.55). I found that raising the GRU dropout layers from 0.5 to 0.55 gave a slight improvement in validation perplexity, though it slowed convergence. Embedding dropout (p=0.1) also helped reduce overfitting.

3. Weight Decay (L2 Regularization). I set `weight_decay` in Adam to about 3×10^{-6} or 1×10^{-6}. This penalizes large weights, preventing memorization on a relatively small dataset. I think this probably helped a lot!

4. Larger Batch Size. Switching from a batch size of 64 to 256 enabled faster training and improved perplexity (due to more stable gradient estimates). This also helped more than expected, decreasing the perplexity by a few points.

5. Learning Rate Schedule. A `StepLR` halving the LR every 5 epochs worked best, preventing plateau or divergence. Without scheduling, the model often stalled around 130 PPL.

6. Early Stopping. If validation PPL did not improve for 3 consecutive epochs, I stopped training to avoid unnecessary overfitting or computation.

Results

Table 1 shows some representative configurations.

Configuration	Hidden	Embed	Dropout	WDecay	Batch	Val PPL	Comments
Baseline	768	128	0.50	–	64	130.9	
+ LR Scheduler	768	128	0.50	–	64	128.1	Some improvement
+ Batch=256	768	128	0.50	–	256	125.2	Faster training
+ Embedding Dropout	768	128	0.50	1×10^{-6}	256	123.5	Reduced overfitting
Final	1024	512	0.55	3×10^{-6}	256	117.5	After StepLR

Table 1: Example final validation perplexities on WikiText-2 with different modifications.

*I would like to acknowledge Pranav Ponnoswamy for providing access to H200 GPUs from the GT GPU Maker Space Computer Block. It allowed me to iterate exponentially faster

With all modifications, the final perplexity reached around 117–118. Notably, deeper networks (6 layers—even with residual connections to avoid vanishing gradient problems and different nonlinearities) or extreme batch sizes (512) provided diminishing returns. I want to acknowledge also trying ensemble networks, but I suppose I did not implement them well enough, since they gave basically the same results with three times the compute time.

Overall, the most important factors were the LR schedule, embedding dropout, and weight decay to mitigate overfitting on this small dataset, combined with a larger batch size for stable gradients and faster epochs.