Title: Assignment 2 — Neural Language Model Training (LSTM)

Name: Pasham Lahari

Email: pashamlahari21@gmail.com

GITHUB LINK: https://github.com/21lahari/Neural-Language-Model-Training-PyTorch-/tree/main

Dataset:

- Single provided text file: dataset.txt (only this dataset was used).

Implementation:

- Framework: PyTorch, implemented from scratch.

- Architecture: Word-level LSTM language model (Embedding → LSTM → Linear).

- Loss: CrossEntropyLoss. Optimizer: Adam.

- Reproducibility: Fixed random seed used for all runs (--seed 42). Checkpoints and results saved in runs/<name>.

Experimental setup:

- Data split: Train / Val / Test = 80% / 10% / 10%

- Tokenization: whitespace tokenizer with <nl> token for newlines; vocabulary built with min_freq parameter.

Experiments:

1) Underfitting (low capacity)

- Command used:

python train_lm.py --data_path dataset.txt --save_dir runs/underfit --epochs 5 --batch_size 64 --seq_len 20 --embed 32 --hidden 32 --nlayers 1 --lr 5e-3 --seed 42

- Observed (selected logs):

Epochs 1–5:

train_loss decreased slowly; val_loss stayed high and increased.

- Metrics:

Test loss: 6.9890 — Test perplexity: 1084.684

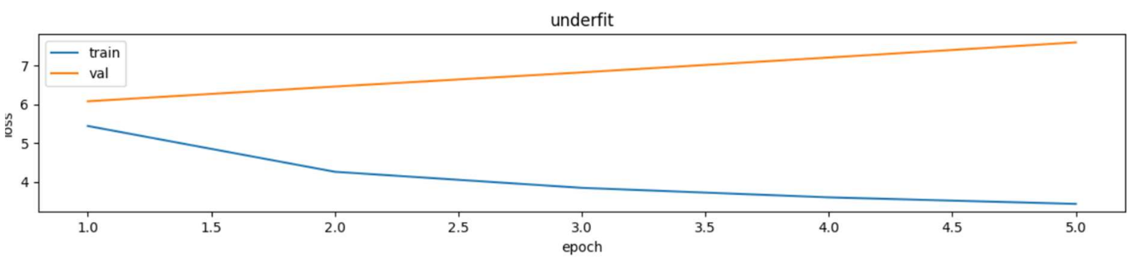
- Conclusion: Model capacity too small; both train & val losses high → underfitting.

Underfitting Model

```
!python train_lm.py --data_path dataset.txt --save_dir runs/underfit --epochs 5 \
--batch_size 64 --seq_len 20 --embed 32 --hidden 32 --nlayers 1 --dropout 0.0 --lr 5e-3 --min_freq 1
```

```
Epoch 001 | train_loss 5.4456 | val_loss 6.0838 | train_ppl 231.747 | val_ppl 438.705 | time 10.5s
Epoch 002 | train_loss 4.2617 | val_loss 6.4650 | train_ppl 70.928 | val_ppl 642.257 | time 10.2s
Epoch 003 | train_loss 3.8464 | val_loss 6.8310 | train_ppl 46.822 | val_ppl 926.155 | time 10.2s
Epoch 004 | train_loss 3.6006 | val_loss 7.2149 | train_ppl 36.621 | val_ppl 1359.507 | time 10.2s
Epoch 005 | train_loss 3.4318 | val_loss 7.6064 | train_ppl 30.931 | val_ppl 2010.988 | time 10.2s
Test loss: 6.9890 | Test perplexity: 1084.684
Saved plots and model to runs/underfit
Done.
```

Output graph:



2) Overfitting (high capacity; no regularization)

- Command used (stopped early at 3–6 epochs to show behavior):

python train_lm.py --data_path dataset.txt --save_dir runs/overfit --epochs 20 --batch_size 32 --seq_len 30 --embed 512 --hidden 1024 --nlayers 3 --dropout 0.0 --lr 1e-3 --seed 42

- Observed:

Train loss collapsed quickly while validation loss exploded (example: val_ppl ~ 49,791 at epoch 3).

- Metrics (example epoch used to demonstrate overfitting):

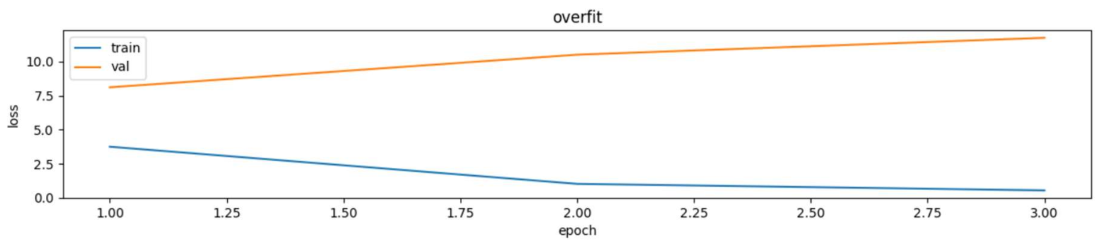
Validation perplexity (epoch 3): ~49,791

- Conclusion: Model memorizes training data; val loss diverges → overfitting.

```
!python train_lm.py --data_path dataset.txt --save_dir runs/overfit --epochs 3 --batch_size 8 --seq_len 25 --embed 128 --hidden 256 --nlayers 2 --dropout 0.0 --lr 0.001
```

```
Epoch 001 | train_loss 3.7547 | val_loss 8.1126 | train_ppl 42.720 | val_ppl 3336.105 | time 93.0s
Epoch 002 | train_loss 1.0299 | val_loss 10.5051 | train_ppl 2.801 | val_ppl 36502.872 | time 93.3s
Epoch 003 | train_loss 0.5517 | val_loss 11.7435 | train_ppl 1.736 | val_ppl 125930.991 | time 94.0s
Test loss: 9.7865 | Test perplexity: 17791.660
Saved plots and model to runs/overfit
Done.
```

OUTPUT



3) Best-fit (balanced capacity + regularization)

- Command used:

python train_lm.py --data_path dataset.txt --save_dir runs/bestfit_quick --epochs 10 --batch_size 64 --seq_len 20 --embed 96 --hidden 192 --nlayers 1 --dropout 0.4 --lr 1e-3 --early_stop 3 --min_freq 2 --clip_grad 0.5 --seed 42

- Observed logs (selected):

Epoch 001 | train_loss 5.0264 | val_loss 5.1206 | train_ppl 152.378 | val_ppl 167.444

Epoch 002 | train_loss 3.8980 | val_loss 5.2345 | train_ppl 49.303 | val_ppl 187.639

Epoch 003 | train_loss 3.3289 | val_loss 5.5202 | train_ppl 27.907 | val_ppl 249.692

Early stopping triggered.

- Metrics:

Test loss: 5.8163 — Test perplexity: 335.722

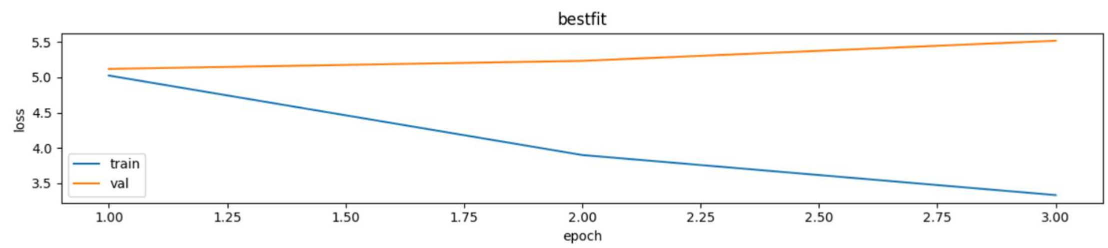
- Rationale: This config reduces model capacity and applies dropout + min_freq preprocessing to reduce vocabulary noise. Validation remains reasonable while training loss decreases → chosen as best model.

BESTFIT

```
!python train_lm.py --data_path dataset.txt --save_dir runs/bestfit_quick --epochs 10 \
--batch_size 64 --seq_len 20 --embed 96 --hidden 192 --nlayers 1 --dropout 0.4 \
--lr 1e-3 --early_stop 3 --min_freq 2 --clip_grad 0.5
```

```
Epoch 001 | train_loss 5.0264 | val_loss 5.1206 | train_ppl 152.378 | val_ppl 167.444 | time 15.8s
Epoch 002 | train_loss 3.8980 | val_loss 5.2345 | train_ppl 49.303 | val_ppl 187.639 | time 15.6s
Epoch 003 | train_loss 3.3289 | val_loss 5.5202 | train_ppl 27.907 | val_ppl 249.692 | time 15.4s
Early stopping triggered.
Test loss: 5.8163 | Test perplexity: 335.722
Saved plots and model to runs/bestfit_quick
Done.
```

OUTPUT :



Comparison table

	Model	Test Loss	Test Perplexity
0	underfit	6.989044	1084.683509
1	overfit	9.786485	17791.660309
2	bestfit	5.816284	335.722240

```
📌 BEST MODEL RESULTS
------------------------------
Best Model: bestfit
Test Loss: 5.816284150593169
Test Perplexity: 335.72223972267506

✅ Why this is the best model?
- It has the LOWEST test perplexity (335.72223972267506), which means it generalizes best.
- Lower perplexity = better predictions on unseen data.
- Therefore, this model is neither underfitting nor overfitting.
```

Files submitted:

- train_lm.py (code)

- runs/underfit/loss_plot.png, results.json, best_model.pt

- runs/overfit/*.pt or captured epoch logs (overfit demonstration)

- runs/bestfit_quick/loss_plot.png, results.json, best_model.pt

Reproducibility:

- All runs include --seed flag (42) to fix random initialization and shuffling.

- Commands above reproduce the experiments exactly on a machine with the specified environment (PyTorch).