

CS310 Natural Language Processing
Assignment 3: Recurrent Neural Networks for Language Modeling
Total points: 50

Tasks

Train an RNN and LSTM language models on *Harry Potter* and evaluate their perplexities.

Submit

- The modified notebook files `A3_lm_rnn.ipynb` and `A3-lm_lstm.ipynb`.
- A `write-up document` in Word/PDF reporting your results in Task 3 and 4.

Requirements

1. (10 points) Data preprocessing and loading.
 - a) Build the vocabulary and load the data to integer tensor. Feel free to use any helper tools, including the `utils.py` files used in previous labs, and packages such as `torchtext`, or `dataloader` from PyTorch.
 - b) You can use a simple space tokenizer or a basic tokenizer from `nltk`.
(<https://www.nltk.org/howto/tokenize.html>)
2. (15 points) Model implementation.
 - a) Use `torch.nn.RNN` module to implement the vanilla RNN model.
 - b) Use `torch.nn.LSTM` module to implement the LSTM-based model.
 - c) Bidirectional and multi-layer can be used; determine the hyper-params yourself.
3. (15 points) Evaluation and generation.
 - a) Use a 90%-5% train-test split.
 - b) Train the models for sufficient number of epochs; compare the two models' final perplexity scores on the *test* set.
 - c) Generate 5 pairs of sentences using greedy search; compare the sentences generated from RNN and LSTM, starting with the same prefix but different follow-ups.
For example, $s_{RNN} = \text{Harry looked at}$, $s_{LSTM} = \text{Harry looked over ...}$, etc.
4. (10) Use only the LSTM model, compare the perplexity on two conditions: randomly initialized embeddings vs. with pretrained embeddings (using the “glove-wiki-gigaword-200” embeddings downloaded using `gemsim`).
 - a) Plot the training loss curves.
 - b) Report the final perplexity scores on test set.