Tutorial: Statistical Language Processing (CL III)
Summer 2024

Assignment 3: Due Monday, June 24, 2024 - 17:00

Submission

See the general rules which apply to all assignments.

Submit the following files to Moodle (1 submission per group):

- Encoder.py
- Decoder.py
- lang.py
- utils.py
- trainer.py
- evaluator.py
- Your best model (just 1 please)
- Discussion as pdf

Translation with an Encoder-Decoder Model

In this assignment you will train and evaluate an encoder-decoder model to translate an utterance represented in IPA (International Phonetic Alphabet) to English.

Examples:

j ʊ ɹ d æ n s ɪ ŋ represents *you're dancing*
aɪ g oʊ aʊ t represents *i go out*

As in Assignment 2, you will keep track of the best model during training and save it. To determine the best model, you can simply use the lowest loss.

After training, the best model is saved and then evaluated on the dev data. The BLEU score is used for evaluation, which is a standard metric for translation tasks.

Overview of the steps:

- Implement load_tsv_data() in utils.py to read training data in data/train.tsv
- Implement the Lang class in lang.py
- Implement the Encoder and Decoder models
- Implement the training methods in train.py
- Train and save a baseline model as described in main() in trainer.py
- Implement the evaluation code in evaluator.py
- Evaluate both your baseline model and the given baseline model, on the data in data/dev.tsv. You should get a bleu score $\approx .4$ for both.
- Experiment with different hyperparameters. You should be able to achieve a bleu score $> .75$

You will need to do some tensor manipulation. Some operations that may be useful are: tensor.view(), tensor.cat(), tensor.squeeze(), tensor.unsqueeze().

Dataset

We will use a modified version of the Phonemized Child Directed Speech Dataset data. The modified train and dev data for this assignment are provided in the Data directory in .tsv files. Each file has 2 columns: *phonemized_utterance* and *processed_gloss*. No special preprocessing is required - simply split the inputs and outputs on whitespace.

The train and dev sets were carefully chosen using small, fixed vocabularies. This means that you do not need to handle OOV words, although normally you *do* need to do that.

You will find the train, and dev sets in the data directory of the starter code, as well as the fixed vocabularies.

Starter Code

Directory Structure

- Models:
  - save your models here
  - evaluation of baseline-model-given.pt on the dev data should get a bleu score of ≈ .43
- Data:
  - train and dev splits as .tsv
  - phone_idx_map.json fixed vocab for input language
  - eng_idx_map.json fixed vocab for output language
- UnitTestData: files used for unit tests

Code Files

- constants.py: Defines sentence markers, MAX_LENGTH for the decoder, model dictionary keys.
- utils.py: Load tsv data
- lang.py: Lang class for storing mappings of input and output languages
- Encoder.py: encoder
- Decoder.py: decoder
- trainer.py: Instantiate, train, and save model.
- evaluator.py: Evaluate models using the BLEU metric.

Discussion

Train and evaluate the baseline model and at least 5 additional models with hyperparameters of your choice. You should be able to achieve a bleu score greater than .75. Report your results and include a brief discussion.

Point Distribution: Total Points: 60

- utils (Total: 2 pts)
  - load_tsv_data() (2 pts)
- Lang (Total: 5 pts)
  - load_vocab() (1 pt)
  - words_to_tensor() (1 pt)
  - tensor_to_words() (1 pt)
  - get_state_dict() (1 pt)
  - load_state_dict() (1 pt)
- Encoder (Total: 8 pts)
  - forward_step() (2 pts)
  - forward() (5 pts)
  - init_hidden() (1 pt)
- Decoder (Total: 10 pts)
  - forward_step() (2 pts)
  - forward() (8 pts)
- Trainer (Total: 18 pts)
  - _generate_io_tensor_pairs() (1 pt)
  - load_data() (1 pt)
  - train_one() (5 pts)
  - train() (5 pts)
  - main() (6 pts)
- Evaluator (Total: 13 pts)
  - load_model() (4 pts)
  - load_data() (1 pt)
  - evaluate_sentence() (2 pts)
  - evaluate() (2 pts)
  - main (4 pts)
- Discussion (4 pts)