Text miming: Home assignment 2

June 20, 2023

1 Natural Language Inference

Natural language inference is way of understanding to what extent a model is able to perform common sense reasoning. You are given a pair of sentences. The task is to identify if the second sentence logically follows from the first one. For example, consider a sentence pair -
Sentence 1: John couldn't see the stage with Billy in front of him because he is so short.
Sentence 2: John is so short.

Sentence 2 indeed logically follows *Sentence 1* hence is classified as positive.

Winograd dataset: This dataset is a standard dataset for Natural Language Understanding. The *train.tsv* file consists of 636 data points with each line representing a data point. Each line consists of *id*, *sentence 1*, *sentence 2* and *label* tab separated.

Task: Your objective is to solve this task using a recurrent neural network.

2 Named Entity Recognition

The named entity recognition (NER) tagging is the task of assigning NER tags (location, person and organization) to the words in a given sentence. For example -

'Soccer', '-', 'Japan', 'get', 'lucky', 'win', ',', 'China', 'in', 'surprise', 'defeat', '.'

tags:{'0', '0', 'LOC', '0', '0', '0', '0', 'LOC', '0', '0', '0', '0'}

CoNLL 2003 dataset: Consists of tokenized sentences and the corresponding tags. There are 9 possible tags - {'O': 0, 'B-PER': 1, 'I-PER': 2, 'B-ORG': 3, 'I-ORG': 4, 'B-LOC': 5, 'I-LOC': 6, 'B-MISC': 7, 'I-MISC': 8}. The dataset is divided into train, validation and test splits.

Task: Your objective is to design a RNN-based neural architecture for the task of named entity recognition.

3 Submission

You will have to submit your solution as a zipped folder. Your main folder should have two sub-folders each for a task. For each task, you need to submit a set of *.py* files.

1. *model.py* file that consists of the model architecture.
2. *utils.py* file that consists of dataset pre-processing as well as pytorch dataset class for the task.
3. *main.py* file for training and evaluation. It should consist of a train function and an evaluation function. Executing this file should train the model and save the model with the highest validation accuracy.
4. *test.py* file that loads the best model and reports the test accuracy.

You can have additional files in the folders if necessary. I will evaluate your submission by executing *main.py*. Additionally, you will have to submit a short report explaining the architectures of your implemented models.

You are free to use any set of hyperparameters. You can also look into related papers and adopt ideas from there. However, please cite any resource that you use.