CSCI544: Homework Assignment №4

Due on March 22, 2023 (before class)

Introduction

This assignment gives you hands-on experience on building deep learning models on named entity recognition (NER). We will use the CoNLL-2003 corpus to build a neural network for NER. The same as HW2, in the folder named *data*, there are three files: *train*, *dev* and *test*. In the files of *train* and *dev*, we provide you with the sentences with human-annotated NER tags. In the file of *test*, we provide only the raw sentences. The data format is that, each line contains three items separated by a *white space* symbol. The first item is the index of the word in the sentence. The second item is the word type and the third item is the corresponding NER tag. There will be a blank line at the end of one sentence. We also provide you with a file named *glove.6B.100d.gz*, which is the GloVe word embeddings [1].

We also provide the official evaluation script *conll03eval* to evaluate the results of the model. To use the script, you need to install *perl* and prepare your prediction file in the following format:

$$\text{idx} \quad \text{word} \quad \text{gold} \quad \text{pred} \tag{1}$$

where there is a white space between two columns. *gold* is the gold-standard NER tag and *pred* is the model-predicted tag. Then execute the command line:

$$\text{perl} \quad \text{conll03eval} \; < \; \{predicted\ file\}$$

where $\{predicted\ file\}$ is the prediction file in the prepared format.

Task 1: Simple Bidirectional LSTM model (40 points)

The first task is to build a simple bidirectional LSTM model (see slides page 43 in lecture 12 for the network architecture) for NER.

Task. Implementing the bidirectional LSTM network with PyTorch. The architecture of the network is:

$$\text{Embedding} \rightarrow \text{BLSTM} \rightarrow \text{Linear} \rightarrow \text{ELU} \rightarrow \text{classifier}$$

The hyper-parameters of the network are listed in the following table:

embedding dim	100
number of LSTM layers	1
LSTM hidden dim	256
LSTM Dropout	0.33
Linear output dim	128

Train this simple BLSTM model with the training data on NER with SGD as the optimizer. Please tune other parameters that are not specified in the above table, such as batch size, learning rate and learning rate scheduling.

What are the precision, recall and F1 score on the dev data? (hint: the reasonable F1 score on dev is 77%.

Task 2: Using GloVe word embeddings (60 points)

The second task is to use the GloVe word embeddings to improve the BLSTM in Task 1. The way we use the GloVe word embeddings is straight forward: we initialize the embeddings in our neural network with the corresponding vectors in GloVe. Note that GloVe is case-insensitive, but our NER model should be case-sensitive because capitalization is an important information for NER. You are asked to find a way to deal with this conflict. What are the precision, recall and F1 score on the dev data? (hint: the reasonable F1 score on dev is 88%.

Bonus: LSTM-CNN model (10 points)

The bonus task is to equip the BLSTM model in Task 2 with a CNN module to capture character-level information (see slides page 45 in lecture 12 for the network architecture). The character embedding dimension is set to 30. You need to tune other hyper-parameters of CNN module, such as the number of CNN layers, the kernel size and output dimension of each CNN layer. What are the precision, recall and F1 score on the dev data? Predicting the NER tags of the sentences in the test data and output the predictions in a file named *pred*, in the same format of training data. (hint: the bonus points are assigned based on the ranking of your model F1 score on the test data).

Submission

Please follow the instructions and submit a zipped folder containing:

1. A model file named *blstm1.pt* for the trained model in Task 1.

2. A model file named *blstm2.pt* for the trained model in Task 2.

3. Predictions of both dev and test data from Task 1 and Task 2. Name the file with *dev1.out*, *dev2.out*, *test1.out* and *test2.out*, respectively. All these files should be in the same format of training data.

4. You also need to submit your python code and a README file to describe how to run your code to produce your prediction files. In the README file, you need to provide the command line to produce the prediction files. (We will execute your cmd to reproduce your reported results on dev).

5. A PDF file which contains answers to the questions in the assignment along with a clear description about your solution, including all the hyper-parameters used in network architecture and model training.

References

[1] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 con-*

ference on empirical methods in natural language processing (EMNLP), pages 1532–1543, 2014.