CE7455: Deep Learning for Natural LanguageProcessing

Assignment # 2

Assignment Logistics

- **Submission Due:** 16 March 2021 (before 2:30 pm)
- **Submission:** Send it to ce7455.assignments@gmail.com with subject "A2-YourStudentID"".
- This is a **individually-graded assignment**. Although you can discuss with your peers, you have to submit your own assignment (following the university guideline).
- Softcopy submission: A CE7455A2-StudentID.zip file containing the following files and folder should be submitted: (i) Report.PDF, (ii) Readme.txt, (ii) SourceCode.
  (i) Report.PDF should contain the written part.
  (ii) Readme.txt should include instructions to run the code and explanations of sample output obtained from your code.
  (iii) SourceCode folder should contain all your source code. The libraries shall NOT be included in the softcopy submission to minimize the file size. You can also submit a **Jupyter notebook.**

1 Question One [50 marks]

Named Entity Recognition (NER) is an important information extraction task that requires to *identify* and *classify* named entities in a given text. These entity types are usually predefined like *location*, *organization*, *person*, and *time*. In this exercise, you will learn how to develop a neural NER model in Pytorch. In particular, you will learn

(a) How to prepare data (input and output) for developing a NER model.

(b) How to design and fit/train a neural NER model.

(c) How to use the trained NER model to predict named entity types for a new text.

You can use this code as your codebase and build on top of it. This code implements the CNN-LSTM-CRF NER model described in [1]. The model has an architecture as shown in Figure 1.

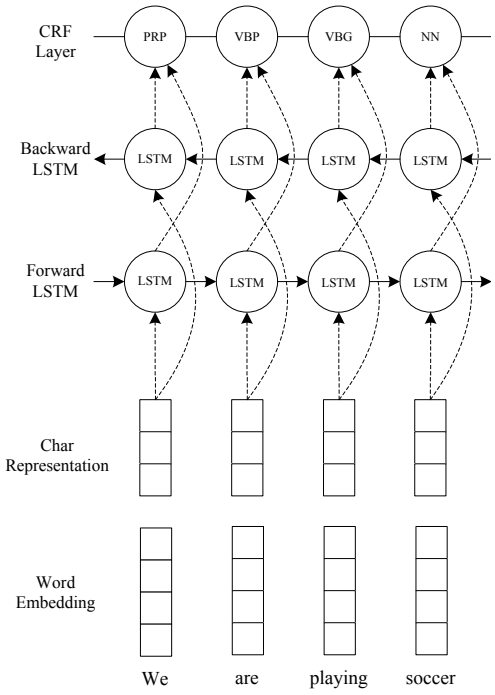

Figure 1: The CNN-LSTM-CRF Model for NER by [1]

The dataset for this exercise will be the standard CoNLL NER dataset, which is also available in the codebase (inside 'data' directory). Please do the following:

(i) Download the code repository. The dataset should have three files: *eng.train*, *eng.testb*, and *eng.testa* to be respectively used for training, testing, and validation. The dataset contains four different types of named entities: PERSON, LOCATION, ORGANIZATION, and MISC; it uses the BIO tagging scheme introduced in the lecture. Try to be familiar with the data and the BIO tagging scheme used in the data.

(ii) Run the code and see the results. The code performs basic prepossessing steps to generate tag mapping, word mapping and character mapping that you should use. You should understand the preprocessing and data laoding functions.

(iii) Go through the model part (get_lstm_features (..), class BiLSTM_CRF(nn.Module)). Notice that it implements a **charater-level encoder** with a *convolutional* neural network (CNN) and an LSTM recurrent neural network. The default setting uses a CNN for character-level encoding. Please read these implementations carefully.

(iv) The code implements a **word-level encoder** with an LSTM network. In its default setting, the output layer of the network has a CRF layer. You can change it to a regular Softmax layer. For now, leave it as it is (i.e., use CRF). Your job is to replace the LSTM-based word-level encoder with a CNN layer (convolutional layer followed by an optional max pooling layer). The CNN layer should have the same output dimensions (out_channels) as the LSTM.

(v) Report the testset results when you use only one such CNN layer in your network. Report results when you use an LSTM-based **character-level** encoder. In each case, report the number of parameters in your model.

(vi) Now increase the number of CNN layers in your network and see the impact on your results. In each case, report the results on the testset (use the validation set to select the best model).

(vii) Notice that the convoluted representations have a local receptive field over the input tokens (i.e., the input tokens that are considered to get the representation). Specifically, in a CNN composed of a series of stacked convolutional layers of window size s, the number c of context tokens incorporated into a token's representation at a given layer d, is given by $c = d(s-1)+1$. In other words, the receptive field over tokens increases *linearly* as we go deeper in the network.

Dialated convolution [3] is an idea to increase the receptive field with an exponential growth, which can be succinctly defined as:

$$c_t = W_c \oplus_{k=0}^{r} x_{t \pm \delta k} \tag{1}$$

where $\oplus$ represents vector concatenation and δ is the dialated factor; $\delta = 1$ gives simple convolution. In this exercise, you have to replace the simple convolution layers in steps (iv-v) by dialated convolution layers.[1] In particular, define a CNN with three dialated layers, where δ varies from 1 (first layer) to 3 (third layer). Report the NER results with the newly implemented dialated layers (you do not need to change anything in the network).

(viii) Now, instead of using a CRF output layer, simply use a softmax output and report the results.

(ix) Please summarize your observations.

2 Question Two [0 marks]

To help me in future planning with the assignment exercises, please mention how much time you spent on the question.

References

[1] Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany, August 2016. Association for Computational Linguistics.

[2] Emma Strubell, Patrick Verga, David Belanger, and Andrew McCallum. Fast and accurate entity recognition with iterated dilated convolutions. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2670–2680, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[3] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

[1] Dialated convolution has already been used for NER in [2]