CSCI544: Homework Assignment – 4

Name: Ashish Bhagchandani
USC ID: 4690271015

1. Read train data, where each line contains three space-separated values index, word, tag. Where sentence are append to sentences list and tag are append to tags list.
2. Read dev data, where each line contains three space-separated values index, word, tag. Where sentence are append to sentencesdev list and tagdev are append to tags list.
3. Further, build word2idx and tag2idx dictionary. Here <pad> is pre append to both word2idx and tag2idx and <unk> is pre append to word2idx.

Task 1: Simple Bidirectional LSTM model

4. PyTorch implementation of a Bidirectional LSTM (BLSTM) model
    a. The model consists of an embedding layer to map each word index to a fixed-size embedding vector, a BLSTM layer to capture contextual information from the input sequence, a fully connected layer with an ELU activation function and dropout regularization, and a linear layer to produce the final output logits.
5. Create data loader, It takes in a list of sentences and their corresponding tags, and returns them as a pair of tensors where each sentence and tag is represented as a sequence of integers based on the provided word2idx and tag2idx dictionaries.
6. The collate_fn function takes in a batch of sentences and tags, pads them to the maximum length, and returns them along with the corresponding lengths. The train_dataset instance is created using the NERDataset class and the train_loader instance is created using the DataLoader class to load the data in batches for training.
7. **Batch size = 32**
8. Creates an instance of the BLSTMModel

    model = BLSTMModel(vocab_size=len(word2idx), embedding_dim=100, hidden_dim=256, output_dim=len(tag2idx)-1, dropout=0.33)

9. Creates an instance of the Stochastic Gradient Descent optimizer for the BLSTMModel with **0.1 learning rate and 0.9 momentum**. Further instance of the Cross Entropy Loss function to calculate the loss during training and a learning rate scheduler that will adjust the learning rate based on the model's F1 score on dev data.
    a. **Learning rate scheduler is ReduceLROnPlateau, mode='max', factor='0.1', patience='15', verbose=True**
10. Trains the model for **120 epochs** and save the model when **max_f1 > f1** and print f1 score of training and dev data at every epoch.
11. Load the model with best f1 score on dev data
12. Apply best model on dev data and generate dev1.out file with index, word, tag, pred_tag
13. result from running the script conll03eval.txt < dev1.out

```
D:\Subject\CSCI 544\hw4>perl conll03eval.txt < dev1.out
processed 51577 tokens with 5942 phrases; found: 5160 phrases; correct: 4437.
accuracy:  95.73%; precision:  85.99%; recall:  74.67%; FB1:  79.93
              LOC: precision:  90.39%; recall:  82.96%; FB1:  86.52  1686
             MISC: precision:  88.34%; recall:  75.60%; FB1:  81.47  789
              ORG: precision:  78.94%; recall:  67.93%; FB1:  73.03  1154
              PER: precision:  85.24%; recall:  70.85%; FB1:  77.38  1531
```

Accuracy	95.73%
Precision	85.99%
Recall	74.67%
FB1	79.93

14. Read test data, where each line contains three space-separated values index, word, tag. Where sentence are append to sentences list and tag are append to tags list
15. Apply best model on test data and generate test1.out file with index, word, pred_tag

Task 2: Using GloVe word embeddings

16. Reads GloVe embedding file and creates dictionary where each word in the file is associated with its corresponding pre-trained embedding vector.
17. Initializes an embedding weight matrix of size len(word2idx) and 100. Then check each word in the glove embeddings if the word is there lower or original it gets its embedding else if word is not present lower or original then zeros are generated of dimension (len(word2idx, 100)).
18. Creates an instance of the nn.Embedding layer, and then assigns the embedding_weights to the layer's weight parameter.
19. PyTorch implementation of a Bidirectional LSTM (BLSTM) model
    a. The model consists of an glove embedding layer, a BLSTM layer to capture contextual information from the input sequence, a fully connected layer with an ELU activation function and dropout regularization, and a linear layer to produce the final output logits.
20. Takes in a list of sentences and their corresponding tags, and returns them as a pair of tensors where each sentence and tag is represented as a sequence of integers based on the provided word2idx and tag2idx dictionaries
21. The collate_fn function takes in a batch of sentences and tags, pads them to the maximum length, and returns them along with the corresponding lengths. The train_dataset instance is created using the NERDataset class and the train_loader instance is created using the DataLoader class to load the data in batches for training.
22. **Batch size = 32**
23. Creates an instance of the BLSTMModel with glove embedding
    model = BLSTMModel(vocab_size=len(word2idx), embedding_dim=100, hidden_dim=256, output_dim=len(tag2idx), dropout=0.33, embedding_layer=embedding_layer)
24. Creates an instance of the Stochastic Gradient Descent optimizer for the BLSTMModel with **0.1 learning rate** and **0.9 momentum**. Further instance of the Cross Entropy Loss function to calculate the loss during training and a learning rate scheduler that will adjust the learning rate based on the model's F1 score on dev data.
    a. **Learning rate scheduler is ReduceLROnPlateau, mode='max', factor='0.7', patience='15', verbose=True**
25. This code trains the model for 120 epochs and save the model when max_f1 > f1 and print f1 score of training and dev data at every epoch
26. Load the model with best f1 score on dev data
27. Apply best model on dev data and generate dev1.out file with index, word, tag, pred_tag
28. result from running the script conll03eval.txt < dev2.out

```
D:\Subject\CSCI 544\hw4>perl conll03eval.txt < dev2.out
processed 51577 tokens with 5942 phrases; found: 5726 phrases; correct: 5178.
accuracy:  97.67%; precision:  90.43%; recall:  87.14%; FB1:  88.76
              LOC: precision:  95.12%; recall:  91.24%; FB1:  93.14  1762
             MISC: precision:  86.55%; recall:  80.26%; FB1:  83.29  855
              ORG: precision:  87.88%; recall:  82.70%; FB1:  85.21  1262
              PER: precision:  89.50%; recall:  89.74%; FB1:  89.62  1847
```

Accuracy	97.67%
Precision	90.43%
Recall	87.14%
FB1	88.76

29. Read test data, where each line contains three space-separated values index, word, tag. Where sentence are append to sentences list and tag are append to tags list
30. Apply best model on test data and generate test2.out file with index, word, pred_tag

Task 3: Bonus (CNN-BiLSTM)

31. Created char2idx for character embedding
32. Implemented code for CNN-BiLSTM. The architecture includes convolutional layers to extract character-level features, and a bidirectional LSTM layer
33. Created a data loader in which sentences, tags and chars are converted into indices list. The data loader sorts the batch by descending sentence length, pads the word and character sequences, and returns them as tensors.
34. **Batch size = 32**
35. Model initialization

    model = CNNBiLSTM(vocab_size=len(word2idx), char2idx=char2idx, char_embed_dim=30, word_embed_dim=100, hidden_dim=256, num_layers=2, kernel_sizes=[3, 5], output_dim=len(tag2idx), dropout=0.33)
36. Train function is created with 10 epochs 0.1 learning rate and 0.9 momentum
37. Creates an instance of the Stochastic Gradient Descent optimizer for the BLSTMModel with 0.1 learning rate and 0.9 momentum. Further instance of the Cross Entropy Loss function to calculate the loss during training and a learning rate scheduler that will adjust the learning rate based on the model's F1 score on dev data.
    a. **Learning rate scheduler is ReduceLROnPlateau, mode='max', factor='0.7', patience='15', verbose=True**