Natural Language Processing: Computer Assignment #3

Due on May 12, 2023

Tohid Abdi

Problem 1

Preprocessing the Data

In the first step, we download the data using the download link and extract them. Then we put them in training and testing dataframes.

To preprocess the data, we perform the following steps:

- Convert all text to lowercase
- Expand clitic contractions
- Remove URLs
- Remove punctuation marks (including hashtags and mentions)
- Tokenization
- Stop word removal
- Stemming

We apply these operations on both the training and testing data.

Training and Evaluating Simple RNN Model

1. To perform this task, we used the tokenize function that exists in the nltk library.
2. We have used One-Hot method and Glove embeddings to convert the words into suitable vectors.
3. We have used the pad_sequence function for this task. Applying padding on the right side of the sentences causes loss of information and weakens the performance of the model. Therefore, we apply padding on the right side of the sentencs.
4. To apply RNN in this setting, we pass the text to be classified through the RNN a word at a time generating a new hidden layer at each time step. We can then take the hidden layer for the last token of the text, h_n, to constitute a compressed representation of the entire sequence. We can pass this representation h_n to a feedforward network that chooses a class via a softmax over the possible classes. Fig. 1 illustrates this approach.

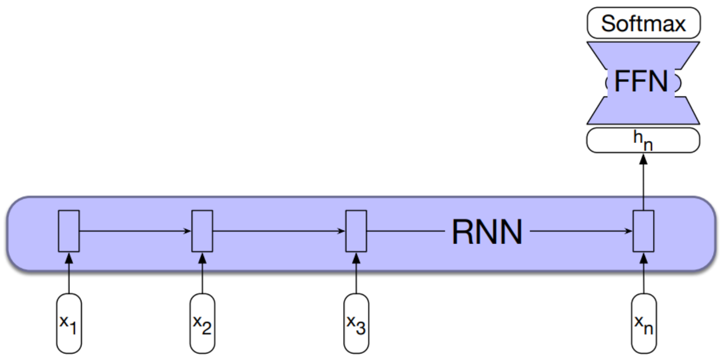

Figure 1: Sequence classification using a simple RNN combined with a feedforward network

5. Done!

Problem 1 continued on next page. . .

6. Due to the fact that the training data had two classes, positive and negative, and there was no neutral class in them, we removed data with a neutral class from the test data. The confusion matrix for the test data is as follows:

Table 1: Confusion Matrix for test data using one-hot Embedding

-	Actual 0	Actual 1
0 Predicted	102	107
1 Predicted	75	75

Table 2: Confusion Matrix for test data using Glove Embedding

-	Actual 0	Actual 1
0 Predicted	134	42
1 Predicted	43	140

Problem 2

Training and Evaluating LSTM and GRU Models

1. We use nn.LSTM layer to implement LSTM layer
2. Done!
3. We set the criterion equal to the nn.CrossEntropyLoss()
4. We have used the one-hot method and Glove embedding. Like RNN, using Glove embedding improved the performance of the model.
5. The confusion matrix for the test data is as follows:

Table 3: Confusion Matrix for LSTM Model using one-hot Embedding

-	Actual 0	Actual 1
0 Predicted	103	106
1 Predicted	74	76

Table 4: Confusion Matrix for LSTM Model using Glove Embedding

-	Actual 0	Actual 1
0 Predicted	138	42
1 Predicted	39	140

When we use the one-hot method, our model has a performance almost close to random selection, but using Glove embedding increases the accuracy of the model in separating classes.

6. We use nn.GRU layer to implement GRU layer
7. The confusion matrix for the test data is as follows:

Table 5: Confusion Matrix for GRU Model using one-hot Embedding

-	Actual 0	Actual 1
0 Predicted	103	119
1 Predicted	74	63

Problem 2 continued on next page. . .

Table 6: Confusion Matrix for GRU Model using Glove Embedding

-	Actual 0	Actual 1
0 Predicted	132	41
1 Predicted	45	141

Each epoch in LSTM model with Glove embedding takes 9 seconds, but while using GRU it only takes roughly 15 seconds.

Table 7 shows the general performance of different models.

Table 7: Accuracy of prediction with different models

-	One-Hot embedding	GloVe
Simple RNN	0.493	0.7632
LSTM	0.4986	0.7744
RNN	0.4624	0.7604