Assignment 3

CSE 143 Spring 2024

Alison Sun, Harsh Jha, Arnav Nepal

May 31, 2024

1 Text Classification with Neural Networks

1.1 Text Classification with RNNs

For the RNN implementation of this text classifier, we embedded the input text as a series of vectors, transforming that sequence into a single vector using a single-layer simple RNN, and then applying a feed-forward layer to obtain a label for it. We utilized the recommended hyperparameters to create and train our model. Resulting accuracies for the training and testing datasets are reported in the table below.

Train Accuracy	**Test Accuracy**
0.9582	0.7052

Table 1: Train and Test Accuracies for RNN Language Model

1.2 Text Classification with LSTMs

For the LSTM implementation of this text classifier, we followed the same experimental process, but using an LSTM model rather than an RNN model. The LSTM model performed far better than the RNN model overall. (list examples here) Resulting accuracies for the training and testing datasets are reported in the table below.

Train Accuracy	**Test Accuracy**
0.9872	0.7354

Table 2: Train and Test Accuracies for LSTM Language Model

2 Deriving the Viterbi Algorithm

2.1 Proof

Given:

$$v_j(y_j) = \max_{y_1,\ldots,y_{j-1}} \sum_{i=1}^{j} s(x,i,y_{i-1},y_i)$$

To show that:

$$v_j(y_j) = \max_{y_{j-1}} \left[s(x,j,y_{j-1},y_j) + v_{j-1}(y_{j-1}) \right]$$

To decompose the expression for $v_j(y_j)$:

$$v_j(y_j) = \max_{y_1,\ldots,y_{j-1}} \left(\sum_{i=1}^{j-1} s(x,i,y_{i-1},y_i) + s(x,j,y_{j-1},y_j) \right)$$

We can separate the last term from the rest:

$$v_j(y_j) = \max_{y_1,\ldots,y_{j-1}} \left(\sum_{i=1}^{j-1} s(x,i,y_{i-1},y_i) \right) + \max_{y_{j-1}} s(x,j,y_{j-1},y_j)$$

Notice that the first part is just $v_{j-1}(y_{j-1})$:

$$v_j(y_j) = \max_{y_{j-1}} \left(v_{j-1}(y_{j-1}) + s(x,j,y_{j-1},y_j) \right)$$

Thus, we have derived the recursive relation:

$$v_j(y_j) = \max_{y_{j-1}} \left[s(x,j,y_{j-1},y_j) + v_{j-1}(y_{j-1}) \right]$$

This completes the proof.

2.2 Time Complexity

Let N = number of states, T = length of the sequence x.
For each j, ranging from 2 to (T + 1):

- For each y_j of N possible states
  - For each $y_j - 1$ of N possible states
    - Compute $s(x, j, y_j - 1, y_j)$ + $v_j - 1(y_j - 1)$

The total number of such computations is: $T \times N \times N = T \times N^2$. Therefore, the time complexity of the Viterbi algorithm is $O(T \times N^2)$.

3 Implementing the Viterbi Algorithm

3.1 Coding the Viterbi Algorithm