# 50.007 Machine Learning

# Design Project

Samuel Lim 1000971

Lin Jun Sheng

Shaun Chiang

## Part 2

## Part 3

For part 3, we were tasked to calculate the transmission parameters as well as run the Viterbi algorithm with the emission and transmission parameters

1. Transmission Parameters

For transmission parameters, we first spilt our training data set into its respective sequences.

(Insert diagram of sequences split from data set)

After splitting the data set, we create a 7 x 7 2D-array to store the results in the following format.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Next From | O | I-positive | B-positive | I-neutral | B-neutral | I-negative | B-negative | End |
| O |  |  |  |  |  |  |  |  |
| I-positive |  |  |  |  |  |  |  |  |
| B-positive |  |  |  |  |  |  |  |  |
| I-neutral |  |  |  |  |  |  |  |  |
| B-neutral |  |  |  |  |  |  |  |  |
| I-negative |  |  |  |  |  |  |  |  |
| B-negative |  |  |  |  |  |  |  |  |
| Start |  |  |  |  |  |  |  |  |

We then start by filling out the count for the table. Each time a case is observed, we increment the count. I.e. if we see a case of O 🡪 B-negative, we increment the table entry of [0, 6] by one. Note that the first dimension of the array is stored as rows, meaning entry [0] is row 1, entry [3] is row 4 etc.

Once the table is complied, we calculate the sum of each row, and then calculate a new 7 x 7 2D array, this time containing the transmission parameters using:

1. Viterbi

For Viterbi, we used the following structure.

(insert diagram of 4D array and how we ran through the array)

We generated a 4D array where the structure of the array is as follows:

* 1st Dimension: List of all sequences
* 2nd Dimension: List of all words in each sequence
* 3rd Dimension: List of all labels in each word
* 4th Dimension: List of each possible combination with the labels of the previous layer per label

**Start**

At the start of every sequence, we only calculate up to the 3rd dimension, and add the scores based on and . We multiply the two scores and add it to the array.

**Middle layers**

We then run through each label, with seven in total, with the seven labels of the previous layer. We multiply the score from each label from the previous layer with the transmission and emission score of the current layer, and store it in the 4th dimension of the array. This will give us a total of 49 scores per layer (sum of 3rd and 4th dimension). After running through all the seven labels of the previous layer, we compare the best label for layer given label for layer .

Once we finish running through all 49 possible combinations, and have the 7 best label for layer given each label in layer as well as their respective scores then append the label of into the selected sequence.

**Note:** that the pointer is at layer but we are appending the label for layer .

**End**

At the end, we do something similar to the middle layers, but we only calculate till the 3rd dimension. We then add the final label of the sequence and then add an ‘End’ label as well.

1. Results

**SG**

Entity in gold data: 4779

Entity in prediction: 4041

Correct Entity: 301

Entity precision: 0.0745

Entity recall: 0.0630

Entity F: 0.0683

Correct Sentiment: 132

Sentiment precision: 0.0327

Sentiment recall: 0.0276

Sentiment F: 0.0299

**CN**

Entity in gold data: 935

Entity in prediction: 1660

Correct Entity: 20

Entity precision: 0.0120

Entity recall: 0.0214

Entity F: 0.0154

Correct Sentiment: 6

Sentiment precision: 0.0036

Sentiment recall: 0.0064

Sentiment F: 0.0046

**ES**

Entity in gold data: 1326

Entity in prediction: 2262

Correct Entity: 98

Entity precision: 0.0433

Entity recall: 0.0739

Entity F: 0.0546

Correct Sentiment: 42

Sentiment precision: 0.0186

Sentiment recall: 0.0317

Sentiment F: 0.0234

**EN**

Entity in gold data: 662

Entity in prediction: 769

Correct Entity: 33

Entity precision: 0.0429

Entity recall: 0.0498

Entity F: 0.0461

Correct Sentiment: 12

Sentiment precision: 0.0156

Sentiment recall: 0.0181

Sentiment F: 0.0168

## Part 4

## Part 5