NLP Assignment 4 : Viterbi and Beam Search

180128022

March 2019

1 Introduction

In this assignment, the structured perceptron has been accelerated using two methods, Viterbi and Beam search.

2 Implementation

The new implementation has been done on the provided code. These functions have been reused: **load_dataset_sents**, **cw_ct_counts**, **phi_1**, **train_perceptron**, **test_perceptron**, **scoring** and **evaluate**.

The following functions have been changed:

1. **phi_1**: Instead of the previous implementation, the count (0 or 1) of the words has been returned depending on whether the tags present in the sentences are also present in the keys of *cw_ct_count*.

2. **scoring**: This function works on the argument value of '-v' and '-b', and returns the tag sequence.

The following functions have been added:

1. **viterbi_func**: This function takes in sentence, *cw_ct_count* and weights as arguments. At first it generates a matrix of zeros with row values as the number of *all_tag* elements and column values as the number of sentences. Running a nested loop, a list of tuples is generated for each iteration, and the **phi_1** function is called to return the values. For the first column of the matrix, the weights are multiplied by the returned values, else, the maximum of the array is calculated and added to the product of the weights and the returned values. Thereafter, the indices of the maximum values are found out and finally, the sequence of tags for those indices are returned.

2. **beam_func**: This function works in the same way as **viterbi_func**, except for the rest of the columns other than the first, the indices of maximum values are retrieved using heapq and the first value is set as the maximum value. Then, iterating through all the indices, if new maximum is found, the maximum value is updated. The rest of the function follows like above.

3 Behaviour of Each Model

The following table describes the time (in secs) required in each case:

Epoch	Structured Perceptron	Viterbi	Beam Search : k = 1	k = 2	k = 5
1	14.47	0.28	0.22	0.24	0.29
2	14.28	0.27	0.21	0.23	0.27
3	14.29	0.27	0.2	0.22	0.27
4	14.23	0.27	0.21	0.23	0.3
5	14.52	0.28	0.21	0.23	0.28

Here, **k** denotes the beam size.

4 Discussion on the Results

1. As expected, the F1 Score for Viterbi for Structured Perceptron is 0.74885 which is equal to that of the naive implementation.

2. It can be seen that the time taken for each epoch of the Structured Perceptron accelerated with the Viterbi Algorithm is much less compared to that of the Structured Perceptron, on its own. This can be attributed to the fact that Viterbi is an exact search method which supports dynamic programming through storing and re-using calculation by keeping track of the highest probability to reach each PoS tag for each word and the path.

3. It can be seen that a large enough beam size, for e.g., 5, the time taken for each epoch for Viterbi and Beam Search are almost the same.

4. The F1 score for Beam Search remains constant at 0.74885, however the speed improves noticeably for beam size 1, when compared to Viterbi, and dramatically, compared to Structured Perceptron, as 'Greedy Search' takes place by keeping only a predetermined number of best states as candidates. For larger beam sizes, the time taken is almost equal to that of Viterbi, but the convergence of Beam Search, when compared to the naive implementation, is still, a lot faster.