Viterbi and Beam Search

Vijeta Agrawal

$26^{th} March$ 2019

1 Aim

The main focus of this report is to show the results of accelerated named entity recogniser (NER) using structured perceptron with two methods, Viterbi and Beam search. For each word in a sequence, the named entity recogniser predicts one of the following labels:

- O: not part of a named entity
- PER: part of a person's name
- LOC: part of a location's name
- ORG: part of an organisation's name
- MISC: part of a name of a different type (miscellaneous, e.g. not person, location or organisation)

2 Introduction and Background

Named-entity recognition (NER) (also known as entity identification, entity chunking and entity extraction) is a subtask of information extraction that seeks to locate and classify named entity mentions in unstructured text into pre-defined categories such as the person names, organizations, locations, medical codes, time expressions, quantities, monetary values, percentages, etc. En.wikipedia.org. (2019)

Viterbi Search or Dynamic Programming is a method for solving a complex problem by breaking it down into a collection of simpler subproblems, solving each of those subproblems just once, and storing their solutions using a memory-based data structure (array, map,etc)

Beam search is a heuristic search algorithm that explores a graph by expanding the most promising node in a limited set. Beam search is an optimization of best-first search that reduces its memory requirements.

3 Training & Test Sets

The training and test sets consists of sentences up to 5 words long, each from the data used in the shared task of CoNLL-2003, which concerns language-independent named entity recognition.

4 Feature Extraction - Phi_1

The cw_ct_counts function takes the training data as an input which is a list of tuples and extracts word_tag features for each tuple. After extracting the features, it calculates the frequency of each feature in the training data set.

Phi_1 function checks if the given tuple is present in the feature space or not.

5 NER Models

This report will feature the results of accelerated Structured Perceptron with Viterbi and Beam search models. The models performs training and testing using Structured Perceptron algorithm with cw_ct_count feature space.

Viterbi model accelerates the structured perceptron by calculating the scores once and storing them, then calculates the maximum and stores its index in backpointer matrix, which enables the algorithm to go back and get the maximum scoring sequence.

Beam Search uses the viterbi approach with a slight difference. While calculating the maximum score, it takes the indices of top k maximum scores from previous column and treats them as possible candidates, it then searches for maximum in those indices. It might reduce accuracy of the model but improves the time complexity by a large number for extremely huge data sets.

6 Results

The below table shows the time taken by different models to run each iteration, the average time taken per iteration and the f1_score for each model.

Search Type	*Iteration 1*	*Iteration 2*	*Iteration 3*	*Iteration 4*	*Iteration 5*	*Average Time taken in seconds*	*F1 Score*
Beam Search b=1	0.21	0.19	0.19	0.19	0.19	0.194	0.74885
Beam Search b=3	0.22	0.2	0.21	0.2	0.2	0.206	0.74885
Beam Search b=5	0.24	0.21	0.21	0.21	0.21	0.216	0.74885
Viterbi Search	0.27	0.25	0.25	0.25	0.25	0.254	0.74885
Standard Structured Perceptron	15.59	16.67	16.71	17.15	17.01	16.626	0.74885

7 Model Performance

7.1 Viterbi Search

It is evident from the above table that the speed of algorithm by using Viterbi Search to calculate the argmax, increases by many folds. It makes sense beacause we are performing much less calculations per iteration than in Standard Structured Perceptron and using the previously calculated values whenever required. The f1_score remains the same as of Standard Structured Perceptron since it is an exact search.

7.2 Beam Search

The speed of the algorithm improves further when we use beam search with viterbi because it searches for argmax in top k indices instead of all. The f1_score with beam search might reduce when we take a lower beam value as it is an inexact search but in our case it doesn't. As shown in below equation, we are adding the maximum value instead of multiplying, the index of the max value remains the same which will be found even if beam value is '1'.

$$B[y, n] = argmax_{y' \in \mathcal{Y}} V\left[y', n-1\right] + \mathbf{w} \cdot \phi\left(y, y', \mathbf{x}\right)$$

Below graph shows the comparison in search speed with three different beam values. As we reduce the beam size, the search becomes faster but the f1_score remains the same.

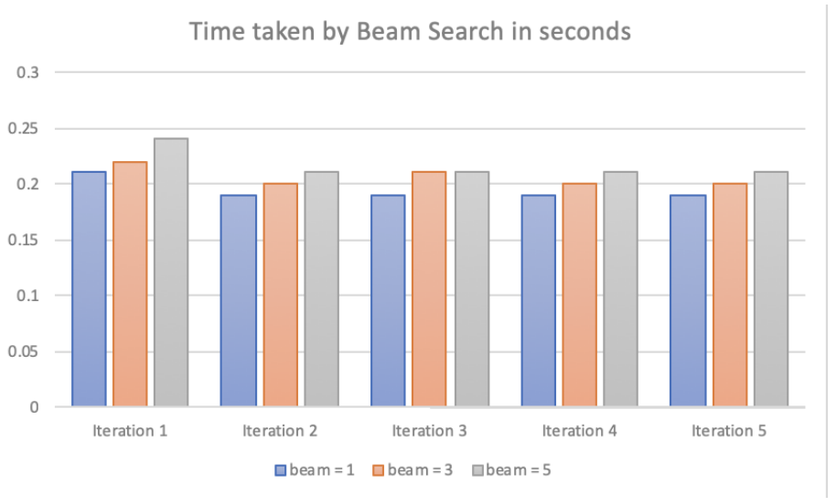

Figure 1: Comparison with different beam values

8 Conclusion

When we apply Viterbi search, the algorithm becomes much faster than Standard Structured Perceptron but returns the same f1_score because of its exact nature.

when we apply Beam search, the algorithm becomes slightly more faster than Viterbi, and as we reduce the beam size the speed increases. In our case, f1_score remains same for each beam size even though its an inexact search because of the above stated reason.