Lab4 Report

fmonteroperez1 - aca15fm

For this lab our task was to implement different algorithms to calculate the best set of POS tags (Person, Location, Organisation, Misc) for a given sentence. Using the code provided by the lecturer I implemented the Viterbi and beam algorithms for the scoring function.

1 Method, Data & Results

I initially run the original code to record the time and accuracy for the "brute forced" perceptron that analyses all the the possible tag combinations. The results for this can be found in Table 1.

Then I implemented the Viterbi algorithm in the two functions "do_viterbi" and "process_backpointer". The first one implements the algorithm, the second one traverses the pointer and returns the predicted sequence of tags. This algorithm is faster because does less calculations than the standard perceptron which had an exponential complexity. The results show this with a decrease in time from 19 seconds to 0.81 seconds. The F1 score for this implementation should have been the same as the one for the previous algorithm. This is not the case due to a limitation of my implementation of my back pointer which will return wrong tags when the scores for all combinations are 0.

	Epoch Average/s	**F1 Score**
Structured Percetron	19.0	0.87
Viterbi	0.81	0.82
Beam k=5	1.00	0.87
Beam k=3	0.65	0.87
Beam k=2	0.55	0.87
Beam k=1	0.33	0.87

Table 1: Results for Implementations

Finally I implemented the Beam search algorithm which is similar to the Viterbi but reduces the amount of calculations done when iterating over the words in a sentence by utilising a smaller set of results from previous iterations. This algorithm was tested with beam sizes 1 to 3. Utilising beam size 1 we have a greedy search which has the fastest epoch time of all the implementations at 0.33 seconds. For a more usual beam size of 3 the epoch time was 0.65 and had the same accuracy as all other Beam implementations. A Beam of 5 would be the same as the Viterbi implementation as we only have 5 tags. This implementation was slower than the Viterbi because of some overhead calculation the max and argmax of the beam as well as sorting the top-ranked POS Combinations.

Overall I belive that the difference between Viterbi and Beam will be more noticed for documents with longer sentences and/or more POS tag categories.