School of Computing and Information Systems
The University of Melbourne
COMP90042
WEB SEARCH AND TEXT ANALYSIS (Semester 1, 2019)

Workshop exercises: Week 9

Discussion

1. What are the assumptions that go into a **Hidden Markov Model**? What is the time complexity of the **Viterbi algorithm**? Is this practical?

   (a) How can an HMM be used for POS tagging a text? For the purposes of POS tagging:

      i. How can the initial state probabilities π be estimated?

      ii. How can the transition probabilities A be estimated?

      iii. How can the emission probabilities B be estimated?

   (b) Estimate π, A and B for POS tagging, based on the following corpus:

```
1. silver-JJ wheels-NNS turn-VBP
2. wheels-NNS turn-VBP right-JJ
3. right-JJ wheels-NNS turn-VBP
```

2. Consider using the following Hidden Markov Model to tag the sentence `silver wheels turn`:
   $\pi[\text{JJ}, \text{NNS}, \text{VBP}] = [0.3, 0.4, 0.3]$

A	JJ	NNS	VBP	B	silver	wheels	turn
JJ	0.4	0.5	0.1	JJ	0.8	0.1	0.1
NNS	0.1	0.4	0.5	NNS	0.3	0.4	0.3
VBP	0.4	0.5	0.1	VBP	0.1	0.3	0.6

   (a) Visualise the HMM as a graph.

   (b) Use the **Viterbi algorithm** to find the most likely tag for this sequence.

3. What are regular grammar and regular language? How are they different?

   (a) Regular languages are closed under union, intersection and concatenation. What does it mean? Why is it important?

   (b) Draw a Finite State Acceptor (FSA) for word morphology to show the possible derivations from root forms using the words: play, played, playing; walk, walked, walking; sit, sat, sitting.

   (c) What are Weighted Finite State Acceptors (WFSAs)? When and why are they useful?

Programming

1. In the iPython notebook `WSTA_N15_hidden_markov_models.ipynb`:

    - The Viterbi algorithm is implemented with loops. Try to implement Viterbi using recursion instead.
    - Can you see the difference between the speed of the Viterbi algorithm and the exhaustive search over the lattice? How much faster is Viterbi than exhaustive search on an example problem? (hint: *time* or *clock* functions from the *time* package can be useful)