Hidden Markov Models

1 What are Markov Chains?

- A Markov chain is a model that tells us something about the probabilities of sequences of random variables, states, each of which can take on values from some set.
- A Markov chain makes a very strong assumption that if we want to predict the future in the sequence, all that matters is the current state.
- Markov Chain consists of: a set of N states
  , a transition probability matrix A, each aij representing the probability of moving from state i to state j, an initial probability distribution over states
- Markov Assumption: when predicting the future, the past doesn’t matter, only the present.

2 Hidden Markov Models

- Here, the tags are hidden because they are not observed
- It consists of: a set of N states, a transition probability matrix, a sequence of T observations, a sequence of observation likelihoods and an initial probability distribution over states
- 3 Approximations of Hidden Markov Models :Words are independent of the words around them, Words depend only on their POS tags, not on the neighbouring POS tags, A tag is dependent only on the previous tag

3 Likelihood Computation

- Given an HMM λ = (A,B) and an observation sequence O, determine the likelihood P(O|λ).
- The forward algorithm computes the observation probability by summing over the probabilities of all possible hidden state paths that could generate the observation sequence, but it does so efficiently by implicitly folding each of these paths into a single forward trellis
- The three factors that are needed in extending the previous paths to compute the forward probability at time t are: previous path probability, transition probability, state observation likelihood
- The stages of forward algorithm are: initialization , recursion, termination
- $$P(w_i|t_i) = \frac{C(t_i, w_i)}{C(t_i)}$$

  Word Likelihood Probability
- $$P(t_i|t_{i-1}) = \frac{C(t_{i-1}, t_i)}{C(t_{i-1})}$$

  Tag Transition Probability

4 Viterbi Algorithm

- Viterbi is a kind of dynamic programming Viterbi algorithm that makes uses of a dynamic programming trellis. Viterbi also strongly resembles another dynamic programming variant, the minimum edit distance algorithm
- For the input: – State (or tag) transition probabilities (A)
  – Observation (or word) likelihoods (B)
  – An observation sequence O = (o1o2···oT)
- For the output, most probable state sequence Q = (q1q2···qT) together with its probability
- Viterbi Algorithm formula: $v_t(j) = \max_{i=1}^{N} v_{t-1}(i)\, a_{ij}\, b_j(o_t)$
- • vt-1(i) – the previous Viterbi path probability from the previous time step t – 1 (i.e., the previous word)
  • aij – the transition probability from previous state qi (i.e., the previous word having POS tag i) to current state qj (i.e., the current word having POS tag j)
  • bj(ot) – the state observation likelihood of the observation symbol ot (i.e., word at position t) given the current state j (i.e., the j POS tag)

5 HMM training: The Forward-Backward Algorithm

- For the training, we will be estimating tag transition probabilities(p(ti|ti-1)) and word likelihoods (p(wi|ti))
- For the testing which is the predicting part, we will estimate the best sequence of tags for a sentence.
- Both of the above probabilities can be trained with the Baum-Welch or forward-backward algorithm.
- The Forward-Backward algorithm finds the best-guess states for each timestep which are these will be different from the Viterbi states: we train our HMM so that it is on average the best fit over all these probabilistic guesses.
- The forward-backward algorithm has two steps: the expectation step, or E-step, and the maximization step, or M-step. In the E-step, we compute the expected state occupancy count γ and the expected state transition count ξ from the earlier A and B probabilities. In the M-step, we use γ and ξ to re-compute new A and B probabilit