Assignment 2 - CS753

Ankur Verma (20D070014)
Chinmay Pimpalkhare (200100115)
Saksham Katiyar (20D100022)
Vikhyat Agrawal (200260058)

April 2024

Part I

(A)

The usual recurrence equation for the Viterbi Algorithm is given by

$$v_t(j) = \max_{i=1}^{N} \ v_{t-1}(i) a_{ij} b_j(o_t)$$

In our case we have,

$$a_{ij} = \begin{cases} q, \ \forall i \neq j \\ p, \ otherwise \end{cases}$$

Thus, instead of calculating max over all states of the previous iteration for each of the 'j's of the current iteration, we can simply compute and store

$$m = \max_{i=1}^{N} \ v_{t-1}(i)$$

Then, for every 'j', at a particular time step 't',

$$v_t(j) = \max(mq, \ v_{t-1}(j) \ p) \times b_j(o_t)$$

Taking max over all 'i', while calculating 'm', works because when finally we take max of the two quantities in the last equation, then since p>q (given), it takes care of the earlier inclusion of i(=j).
Thus, now the algorithm runs in O(NT) as it will take O(N) only once for calculating 'm' at every time step and O(T) for the 'T' time steps.

(B)

Let us define two variables as below,
$V_t(q, l) \implies$ Probability of getting to q such that $\{t - l + 1, t - l + 2, ..., t\}$ only has q as the traversed state
And so for same state transitions,

$$V_t(q, l) = V_{t-1}(q, l - 1) a_{qq} P(O_t|q)$$

$V_t(q) \implies$ Probability of getting to state q at time t such that there is a sequence of states of k-consecutive runs.

Now, there are two possible cases. The first case is that the sequence from $[0, \ldots, t-1]$ already has a $k-$consecutive run, in which case whatever states we traverse starting from time t do not really matter in terms of satisfying the constraint of $k-$consecutiveness. The second case is when the states traversed at time steps $[t-k+1, \ldots, t-1]$ are all q. In the second case, the sequence on reaching q satisfies our required condition at time t and hence becomes a valid sequence. Therefore, the update step for $V_t(q)$ should take the maximum considering both of these conditions This can then be mathematically represented as

$$V_t(q) = \max\left(V_{t-1}(q, k-1)a_{qq}P(O_t|q) \quad , \quad \max_{q'} V_{t-1}(q')a_{q'q}P(O_t|q)\right)$$

The term $P(O_t|q)$ is common, hence, the recurrence term can be expressed as

$$V_t(q) = P(O_t|q)\max\left(V_{t-1}(q, k-1)a_{qq} \quad , \quad \max_{q'} V_{t-1}(q')a_{q'q}\right)$$

Now, we also need to define the base case for the $V_t(q, l)$ term and that comes out to be

$$V_t(q, 1) = P(O_t|q)\max(\max_{l} V_{t-1}(q, l)a_{qq}, \ \max_{l,q'} V_{t-1}(q', l)a_{q'q})$$

And,

$$V_1(q, 1) = \Pi(q)P(O_1|q)$$

where $\Pi(q)$ denotes the probability of starting at state q

Further, note that for $t < k$, it is impossible to have a sequence of $k-$consective runs ending in q at time t, therefore

$$V_t(q) = 0 \quad \forall \quad t \in \{1, 2, \ldots k-1\}$$

Finally, returning $\mathbf{max}\ \mathbf{V_T(q)}$ gives us the desired result.

(C)

In the extended HMM problem, we have been given that it is possible to now emit a sequence of l observations on transitioning into a particular state. Further, we have been given the pair of functions L, B where

$$L(i, l) = \Pr(l|i)$$

and

$$B(i, o_1, o_2, \ldots, o_l) = \Pr(o_1, \ldots, o_l|i, l)$$

Note that L, B are stationary functions, hence does not depend on the time at which we reach the state. We then define

$$w_t(j, k) = \max_{i} v_{t-k}(i)a_{ij}L(i, k)B(i, o_{t-k+1}, \ldots, o_t) \quad \text{where} \quad k \in [1, l_{\max}]$$

Note that $w_t(j,k)$ looks at all the paths ending at state j such that the last k observations were emitted on transition to state i. Naturally, k varies from 1 to $l_{\max}$. We can obtain the Viterbi recurrence term $v_t(j)$ which represents the probability of the most likely state sequence that ends in state j while emitting the observation sequence $o_1, \ldots, o_t$ by taking a max over k

$$v_t(j) = \max_k w_t(j,k) \quad k \in [1, l_{\max}]$$

Part I

Part B (Syllabes in WFST-ASR

Defining C and L Here, we first consider what the input and output spaces of C and L need to be.
C takes in *syllables* as input and outputs *monophones*
L takes in *monophones* as input and outputs *words*

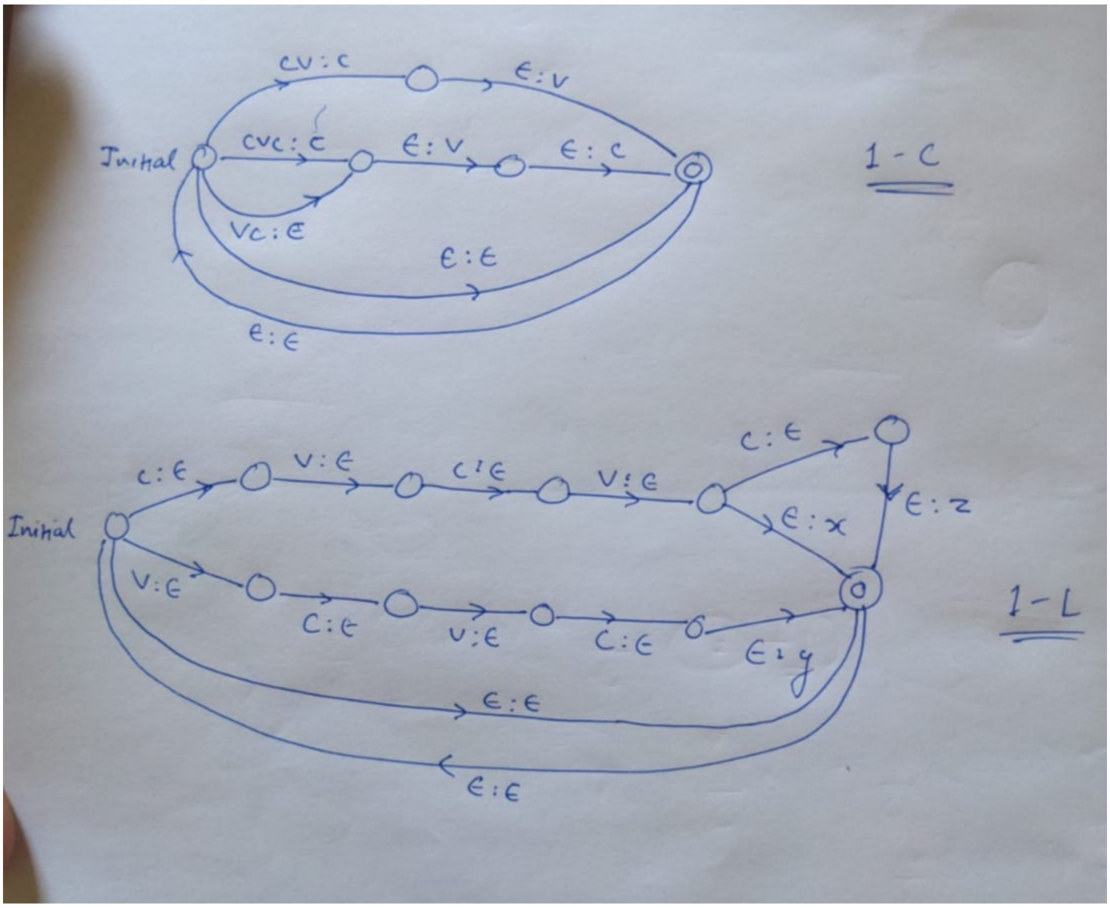

Figure 1: The C and L for when we do not want to straddle words

Contextualized Syllabes Here, **C** goes from *contextualized syllabes* to *monophones*. Consider the mapping $M : (S,P) \longrightarrow S$. We observe that this mapping gives us back the syllables that make up the word. Hence, we need to change the input labels in the earlier WFST to get the one for this case.

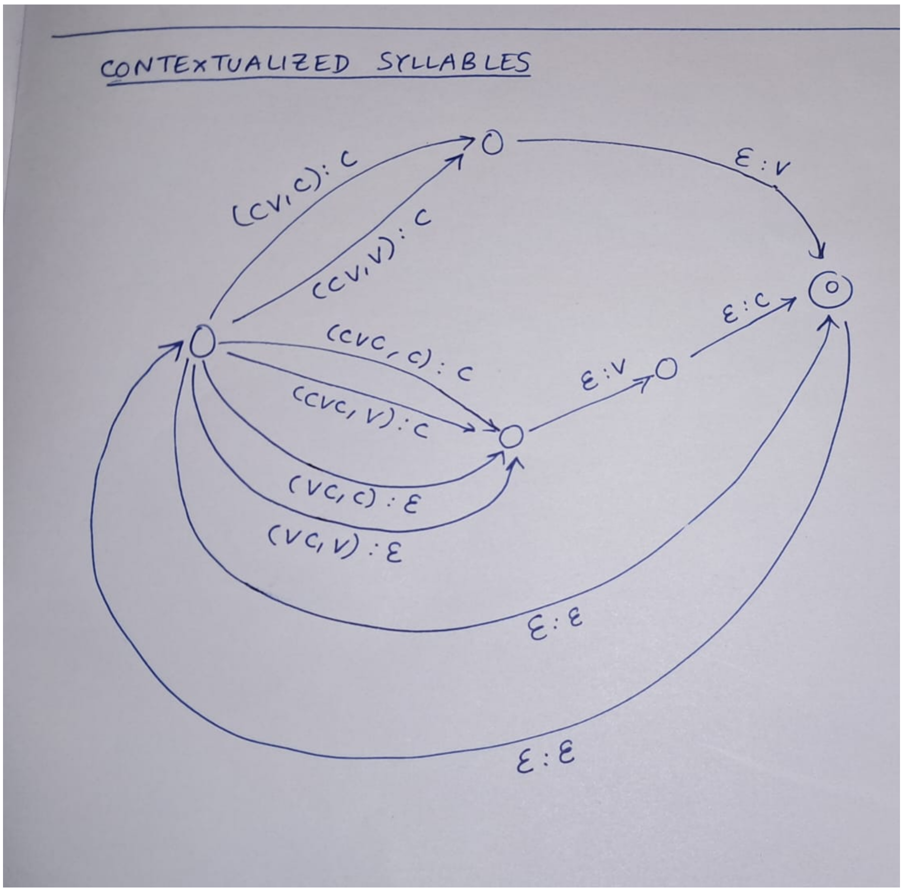

Figure 2: C for the contextualized case