RAMANATHAN 21BAI1723

```
state =['NOUN', 'VERB', 'PRON', 'ADJ']
obs = ['he', 'reads', 'books', 'football', 'quick', 'lazy', 'runs',
'the','eats']

sp = {'NOUN':0.4,'VERB':0,'PRON':0.6,'ADJ':0}
tp = {
    'NOUN': {
        'NOUN': 0.2,   # Staying a NOUN
        'VERB': 0.1,   # Transitioning to a VERB
        'PRON': 0.1,   # Transitioning to a PRON
        'ADJ': 0.6     # Transitioning to an ADJ (e.g., adjectives
often follow nouns)
    },
    'VERB': {
        'NOUN': 0.5,   # Transitioning to a NOUN (e.g., verbs often
lead to nouns)
        'VERB': 0.2,   # Staying a VERB
        'PRON': 0.1,   # Transitioning to a PRON
        'ADJ': 0.2     # Transitioning to an ADJ
    },
    'PRON': {
        'NOUN': 0.4,   # Transitioning to a NOUN (e.g., pronouns can
lead to nouns)
        'VERB': 0.3,   # Transitioning to a VERB
        'PRON': 0.3,   # Staying a PRON
        'ADJ': 0.0     # Less likely to transition to an ADJ
    },
    'ADJ': {
        'NOUN': 0.7,   # Adjectives typically modify nouns
        'VERB': 0.1,   # Less likely to transition to a VERB
        'PRON': 0.1,   # Less likely to transition to a PRON
        'ADJ': 0.1     # Staying an ADJ
    }
}

ep = {
    'NOUN': {
        'he': 0.0,       # "he" is not a NOUN
        'reads': 0.0,    # "reads" is not a NOUN
        'books': 0.4,    # "books" is a NOUN
        'football': 0.6  # "football" is a NOUN
    },
    'VERB': {
        'he': 0.0,       # "he" is not a VERB
        'reads': 0.6,    # "reads" is a VERB
        'books': 0.0,    # "books" is not a VERB
        'football': 0.0  # "football" is not a VERB
    },
    'PRON': {
        'he': 1.0,      # "he" is a PRON
        'reads': 0.0,   # "reads" is not a PRON
        'books': 0.0,   # "books" is not a PRON
        'football': 0.0  # "football" is not a PRON
    },
    'ADJ': {
        'he': 0.0,      # "he" is not an ADJ
        'reads': 0.0,   # "reads" is not an ADJ
        'books': 0.0,   # "books" is not an ADJ
        'football': 0.0  # "football" is not an ADJ
    }
}

import numpy as np
def viterbi(sentence, states, sp, tp, ep):
    num_obs = len(sentence)
    num_states = len(states)

    viterbi_table = np.zeros((num_states, num_obs))
    path_table = np.zeros((num_states, num_obs), dtype=int)

    for i, state in enumerate(states):
        viterbi_table[i, 0] = sp[state] * ep[state].get(sentence[0],
0)

    for t in range(1, num_obs):
        for i, state in enumerate(states):
            max_prob = -1
            max_state_index = 0
            for j, prev_state in enumerate(states):
                prob = (viterbi_table[j, t - 1] * tp[prev_state]
[state] * ep[state].get(sentence[t], 0))
                if prob > max_prob:
                    max_prob = prob
                    max_state_index = j
            viterbi_table[i, t] = max_prob
            path_table[i, t] = max_state_index

    best_path = [0] * num_obs
    best_path[-1] = np.argmax(viterbi_table[:, num_obs - 1])
    for t in range(num_obs - 2, -1, -1):
      best_path[t] = path_table[best_path[t + 1], t + 1]
    best_path = [state[i] for i in best_path]
    return best_path, viterbi_table

sentence = ['he', 'reads', 'football']  # Note: "eats" is not in the
observations
```

```
best_path, viterbi_table = viterbi(sentence, states, sp, tp, ep)
print("Best Path:", best_path)
print("Viterbi Table:\n", viterbi_table)
```

```
Best Path: ['N', 'P', 'R']
Viterbi Table:
 [[0.     0.108  0.    ]
 [0.     0.     0.0324]
 [0.     0.     0.    ]
 [0.6    0.     0.    ]]
```