Problem Set 3

Jungwoong Yoon (jy8963)

March 24, 2023

1 Exercise 1

1. All possible hidden state sequences: (1 2 3 4), (1 1 3 4), (1 3 3 4)
2. The best hidden sequence: (1 3 3 4)
3. $P(HTT|\lambda) = \frac{5}{32}$
4. $\alpha_2(1) = \frac{1}{16}$

2 Exercise 2

2.1 Computing class likelihoods

```
HMM_list = [fee_HMM, pea_HMM , rock_HMM, burt_HMM, see_HMM, she_HMM]
wav_list = ["fee.wav", "pea.wav" , "rock.wav", "burt.wav", "see.wav", "she.wav"]

for wav in wav_list:
    print("Using {}".format(wav))

    for hmm in HMM_list:
        forward_result = hmm.forward(compute_phone_likelihoods(model, load_audio_to_melspec_tensor(wav)))
#         viterbi_result = hmm.viterbi(compute_phone_likelihoods(model, load_audio_to_melspec_tensor(wav)))
        print("{}: likelihood = {}".format(name, forward_result))

    print()
```

2.2 Implementing the Forward Algorithm

```
    def forward(self, state_likelihoods):
        # state_likelihoods.shape is assumed to be (N_timesteps, 48)
        # TODO: fill in
        time = state_likelihoods.shape[0]
        alpha = np.zeros((time, self.N_states))

        for t in range(time - 1):
            # Initialization
            if t == 0:
                for i in range(self.N_states):
                    alpha[t, i] = state_likelihoods[t, self.labels[i]] + self.pi[i]

            # Induction
            for i in range(self.N_states):
                cur = [alpha[t, j] + self.A[j, i] for j in range(self.N_states)]
                alpha[t+1, i] = logsumexp(cur) + state_likelihoods[t+1, self.labels[i]]

        # Termination (alpha_{T}(N))
        return alpha[-1, -1]
```

2.3 Implementing the Viterbi Decoding Algorithm

```python
def viterbi(self, state_likelihoods):
    # state_likelihoods.shape is assumed to be (N_timesteps, 48)
    # TODO: fill in
    time = state_likelihoods.shape[0]
    delta = np.zeros((time, self.N_states))
    psi = np.zeros((time, self.N_states), dtype=int)
    q_star = np.zeros(time, dtype=int)

    for t in range(time):
        if t == 0:
            # Initialization
            for i in range(self.N_states):
                delta[t, i] = state_likelihoods[t, self.labels[i]] + self.pi[i]
        else:
            # Induction
            for i in range(self.N_states):
                prev = [delta[t-1, j] + self.A[j, i] for j in range(self.N_states)]
                psi[t, i] = np.argmax(prev)
                delta[t, i] = prev[psi[t, i]] + state_likelihoods[t, self.labels[i]]

    # Termination
    q_star[-1] = np.argmax(delta[-1])

    # Backtrace
    for t in range(time-1, 0, -1):
        q_star[t - 1] = psi[t, q_star[t]]

    return q_star
```

2.4 Implementing Viterbi Training

```python
def viterbi_transition_update(self, state_likelihoods):
    # state_likelihoods.shape is assumed to be (N_timesteps, 48)
    # TODO: fill in
    q_star = self.viterbi(state_likelihoods)
    print(np.exp(self.A))

    for i in range(self.A.shape[0]):
        tau = 0
        gamma = 0

        for t in range(len(q_star) - 1):
            cur_state = q_star[t]
            next_state = q_star[t+1]

            if cur_state == i:
                gamma += 1

        for j in range(self.A.shape[1]):
            for t in range(len(q_star) - 1):
                cur_state = q_star[t]
                next_state = q_star[t+1]

                if cur_state == i and next_state == j:
                    tau += 1

            self.A[i, j] = tau / gamma

    self.A = np.log(self.A + self.eps)
    print(np.exp(self.A))
```

2.5 Likelihood computation

	fee HMM	pea HMM	rock HMM	burt HMM	see HMM	she HMM
fee	212.22	178.08	-94.74	-88.53	188.32	188.97
pea	252.19	270.26	12.14	75.10	238.91	237.35
rock	-61.38	-3.84	156.74	65.48	-60.25	-62.19
burt	-59.64	-17.15	114.60	218.39	-71.04	-95.40
see	76.37	78.22	-173.40	-155.18	233.89	132.22
she	77.35	91.23	-210.63	-190.74	124.16	283.93

All of the words were correctly recognized.

2.6 Optimal state sequence

Optimal hidden sequence for ”rock”: [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5]

2.7 Viterbi Update

New likelihood for "rock": 168.82 (The likelihood increased.)

Updated transition matrix (after exponentiation):

$$\begin{bmatrix} 9.3e-001 & 1.0e+000 & 1.0e+000 & 1.0e+000 & 1.0e+000 & 1.0e+000 \\ 1.0e-200 & 9.17e-001 & 1.0e+000 & 1.0e+000 & 1.0e+000 & 1.0e+000 \\ 1.0e-200 & 1.0e-200 & 9.3e-001 & 1.0e+000 & 1.0e+000 & 1.0e+000 \\ 1.0e-200 & 1.00e-200 & 1.0e-200 & 9.1e-001 & 1.0e+000 & 1.0e+000 \\ 1.0e-200 & 1.0e-200 & 1.0e-200 & 1.0e-200 & 7.5e-001 & 1.0e+000 \\ 1.0e-200 & 1.0e-200 & 1.0e-200 & 1.0e-200 & 1.0e-200 & 1.0e+000 \end{bmatrix}$$

The original transition matrix had all $a_{ij} = 0$ (before adding eps), except for those where i = j. The updated matrix is similar to the original, but for a_{ij} where i = j, its $a_{ij+1}...a_{iN}$ have probability of 1.