MP 4 for CS510

Name:Jiayi Liu NetID:jiayi4

November 22, 2017

1 Forward Algorithm and Backward Algorithm [30 pts]

a. Forward Algorithm

$\alpha_1(V) = 0.1$

$\alpha_1(N) = 0.05$

$\alpha_2(V) = 0.0085$

$\alpha_2(N) = 0.013000000000000003$

$\alpha_3(V) = 0.006960000000000001$

$\alpha_3(N) = 0.0009900000000000002$

$\alpha_4(V) = 0.0004671$

$\alpha_4(N) = 0.0019674000000000002$

P(print line hello world) = 0.0024345

b. Backward Algorithm

$\beta_1(V) = 0.015920000000000004$

$\beta_1(N) = 0.01685$

$\beta_2(V) = 0.122$

$\beta_2(N) = 0.1075$

$\beta_3(V) = 0.3$

$\beta_3(N) = 0.35$

$\beta_4(V) = 1$

$\beta_4(N) = 1$

P(print line hello world) = 0.0024345000000000005

2 Viterbi Algorithm[70 pts]

a.The initial start probabilities are:

NNP 0.5602442510068858

NNS 0.011952708847602963

DT 0.34473171365467065

IN 0.017279459529686892

JJ 0.019929842795894503

VBN 0.006548005716512927

RB 0.009640119527088476

NN 0.028738469533584513

CC 0.0006236195920488502

VBZ 0.0003118097960244251

The top 10 words with the highest output probabilities for each POS tag for data/test_0 are:

NNP

('County', 0.015251594827261567), ('New', 0.008812032566862239), ('England', 0.007742538016012291 5),
('River', 0.006748359982827834), ('India', 0.006296460876834898), ('South', 0.006175954448570116),
('District', 0.006153359493270469), ('US', 0.005693928735510985), ('Crambidae', 0.005181776415385658),
('School', 0.004948295210622642)

VBZ

('is', 0.8492857708657883), ('has', 0.031447205132944314), ('consists', 0.007974018430016107),
('includes', 0.006363372323290999), ('serves', 0.005386423045441344), ('lies', 0.005175190769149526),
('features', 0.004647110078419983), ('contains', 0.0027196155572571487), ('plays', 0.002376363108282946),
('provides', 0.0021915348665276055)

DT

('the', 0.4105331225750826), ('a', 0.3305647363126886), ('The', 0.16726541169708292), ('an',
0.05095559706854433), ('This', 0.015030895243569478), ('A', 0.00731426929156488), ('this',
0.0044546630262968815), ('both', 0.0022273315131484408), ('all', 0.0020261531829285817),

('An', 0.0014226181922690042)

NN

('family', 0.08389095971617191), ('moth', 0.05478959895809943), ('village', 0.020680828131315398), ('district', 0.01836798850316612), ('state', 0.015808146584631966), ('town', 0.013270759419769166), ('area', 0.010666007993892307), ('station', 0.01054250684869987), ('genus', 0.01043023308034311), ('film', 0.009846409484887951)

IN

('of', 0.4028518815808112), ('in', 0.31978217416623383), ('by', 0.05721075908525741), ('on', 0.034279238868690685), ('as', 0.02664589137038261), ('at', 0.024489667437870847), ('for', 0.024111934778160756), ('with', 0.019878181217243496), ('from', 0.019390276531784628), ('near', 0.008357335096085746)

NNS

('species', 0.20187223095612405), ('stars', 0.02651136201229098), ('roles', 0.02193797341717879), ('forests', 0.01207660425896813), ('moths', 0.007931970844647706), ('sports', 0.0067171644990710305), ('areas', 0.006574246105473774), ('people', 0.006431327711876518), ('services', 0.006288409318279262), ('spiders', 0.006288409318279262)

JJ

('American', 0.03236222987118981), ('historic', 0.02796210212554558), ('small', 0.023845853589297753), ('Indian', 0.019161846634257122), ('former', 0.015116567900358398), ('geologic', 0.01383911145807459), ('civil', 0.012845534225187184), ('public', 0.010645470352365068), ('national', 0.009829317625350413), ('southern', 0.009687378020652213)

VBN

('located', 0.24616422188501097), ('known', 0.06938121733265891), ('based', 0.06651492159838139), ('owned', 0.03827347833417636), ('been', 0.02554375316135559), ('used', 0.02554375316135559), ('situated', 0.024785027819929185), ('named', 0.018799527904232), ('operated', 0.014668689934243804), ('written', 0.013994267408531444)

CC

('and', 0.9019375966524151), ('or', 0.08032384244519239), ('but', 0.009824433730555809), ('either', 0.0010916037478395342), ('both', 0.0010006367688529063), ('And', 0.0008187028108796507), ('et', 0.0007277358318930229), ('moth', 0.0006367688529063949), ('plus', 0.0005458018739197671), ('But', 0.00036386791594651143)

RB
('also', 0.19706869150626877), ('currently', 0.06003884866678439), ('now', 0.05244570015892636),
('well', 0.03337453646477132), ('as', 0.027017481900052976), ('not', 0.023132615221613986),
('commonly', 0.021543351580434397), ('just', 0.01889457884513509), ('approximately', 0.017481900052975455)
('only', 0.016952145505915592)
In addition, the results for this answer is saved in **test_0_common.txt**.

b. By using the Viterbi Algorithm, I generate the tagging results for data/test_0_tagging based on the hmm I trained in 2a. The results for this answer is saved in **test_0_results.txt**.

c. The algorithm in 2b could not deal with this problem. To solve this problem, I add another word **'UNK'**. Here, users could import a new parameter minFreq, in our training set, if the frequency of a word is smaller than minFreq, it would be changed to 'UNK'. Then, when we use Viterbi Algorithm, if the word in test sentence does not appear in our training set vocabulary, this word would be changed to 'UNK'.
By using the implemented Viterbi Algorithm, we generate the tagging results for data/test_1_tagging based on the hmm I trained here. The results for this answer is saved in **test_1_results.txt**.
Additionally, I also generate the top 10 words with the highest output probabilities for each POS tag for test_1. The most common words results is saved in **test_1_common.txt**.

d. The accuracy of test_0 is 0.9733(here I set the minFreq as 0, I keep all words in training data).
The accuracy of test_1 is 0.8495(if I set the minFreq for UNK as 1, for those words who own frequency lessthan 1, I would change them to UNK).