ANLP 662 – Homework 2

Ekraam Sabir
USC-ID: 1428-2892-90
esabir@usc.edu

1. Transliteration

Q. 1.

I tried the commands mentioned in the question and additionally I tried the following commands:

```
echo '"WHALE" "BONES"' | carmel -sliOEQk 5 eword-epron.wfst
Input line 1: "WHALE" "BONES"
        (313822 states / 313822 arcs reduce-> 18/18)
W EY L B OW N Z 1
HH W EY L B OW N Z 1
Derivations found for all 1 inputs

echo '"WHALEBONES"' | carmel -sliOEQk 5 eword.wfsa
Input line 1: "WHALEBONES"
        (0 states / 0 arcs)
Empty or invalid result of composition with transducer "eword.wfsa".
No derivations found for 1 of 1 inputs

echo '"WHALE" "BONES"' | carmel -sliOEQk 5 eword.wfsa
Input line 1: "WHALE" "BONES"
        (3 states / 2 arcs)
WHALE BONES 3.31194845095128e-11
Derivations found for all 1 inputs

echo '"WHALE"' | carmel -sliOEQk 5 eword.wfsa
Input line 1: "WHALE"
        (2 states / 1 arcs)
WHALE 2.9808918e-06
Derivations found for all 1 inputs
```

I learnt the following things:

The unigram wfsa eword.fsa can accept any combination of multiple words provided they are individually accepted by it and are separated by a space.

Some English words are more likely than others.

eword-epron.wfst will give all the possible pronunciations as equally likely (Weights are not included in this).

The eword.fsa accepts words as single strings. It doesn't form them character by character.

Q. 2.

```
        echo '"B" "EH" "R"' | carmel -sriIEQk 5 eword-epron.wfst
Input line 1: "B" "EH" "R"
        (11 states / 14 arcs)
BEHR 1
BEAR 1
BARE 1
BAHR 1
BAER 1
Derivations found for all 1 inputs
```

All their probabilities are 1 i.e. they are equally likely according to the wfst. But we can look further at the probability of the words themselves and say that the probability of the pronunciation depends on the probability of the words directly since the pronunciation has no preference of its own. In that case the ordered probability would be:

```
BEAR 0.00010622814
BARE 1.4904459e-05
BEHR 1.3549508e-06
BAER 2.167921e-06
BAHR 5.4198027e-07
```

Q. 3.

Words that sound similar to WHERE are WHERE, WEAR and WARE.

```
        echo '"WHERE"' | carmel -sliOEQk 10 eword-epron.wfst epron-eword.wfst
Input line 1: "WHERE"
        (209215 states / 209215 arcs reduce-> 12/12)
        (32 states / 38 arcs reduce-> 14/16)
WHERE 1
WEAR 1
WHERE 1
WARE 1
Derivations found for all 1 inputs
```

Word sequences that sound similar to ICE CREAM are ICE CREAM, EYE SCREAM, AY SCREAM and AI SCREAM

```
        echo '"ICE" "CREAM"' | carmel -sliOEQk 10 eword-epron.wfst epron-eword.wfst
Input line 1: "ICE" "CREAM"
        (313821 states / 313820 arcs reduce-> 12/11)
        (42 states / 46 arcs reduce-> 24/26)
ICE CREAM 1
EYE SCREAM 1
AY SCREAM 1
AI SCREAM 1
Derivations found for all 1 inputs
```

Q. 4.

For "A" "N" "J" "I" "R" "A" "N" "A" "I" "T" "O"

```
        echo '"A" "N" "J" "I" "R" "A" "N" "A" "I" "T" "O"' | carmel -sriIEQk 5 eword.wfsa eword-
epron.wfst epron-jpron.wfst
Input line 1: "A" "N" "J" "I" "R" "A" "N" "A" "I" "T" "O"
        (77 states / 161 arcs reduce-> 17/101)
        (2105 states / 2854 arcs reduce-> 1323/2009)
        (1150 states / 1663 arcs reduce-> 1053/1566)
ANGELA NIGHT 1.12372952408388e-12
ANGELA MIGHT 3.21800667874742e-13
ANGELA KNIGHT 1.28848693621702e-13
ANGELA NATO 1.54319592476387e-14
ANGELS NIGHT 5.40240866293639e-15
Derivations found for all 1 inputs
```

For "S" "U" "CH" "I" "I" "B" "E" "N" "R" "A" "R" "U" "Z" "U"

```
    echo '"S" "U" "CH" "I" "I" "B" "E" "N" "R" "A" "R" "U" "Z" "U"' | carmel -sriIEQk 5 eword.wfsa
eword-epron.wfst epron-jpron.wfst
    Input line 1: "S" "U" "CH" "I" "I" "B" "E" "N" "R" "A" "R" "U" "Z" "U"
        (67 states / 148 arcs reduce-> 25/106)
        (1198 states / 1623 arcs reduce-> 703/1055)
        (611 states / 871 arcs reduce-> 559/819)
    STEPHEN RAILS 7.72810488417023e-16
    STEVEN RAILS 5.46007428264408e-16
    STEPHEN RAWLS 1.24013742025023e-16
    STEPHEN RILES 9.50105404353684e-17
    STEPHEN LARS 9.46141651711666e-17
    Derivations found for all 1 inputs
```

For "D" "O" "N" "A" "R" "U" "D" "O" "T" "O" "R" "A" "N" "P" "U"

```
    echo '"D" "O" "N" "A" "R" "U" "D" "O" "T" "O" "R" "A" "N" "P" "U"' | carmel -sriIEQk 5
eword.wfsa eword-epron.wfst epron-jpron.wfst
    Input line 1: "D" "O" "N" "A" "R" "U" "D" "O" "T" "O" "R" "A" "N" "P" "U"
        (97 states / 188 arcs reduce-> 26/117)
        (2435 states / 3415 arcs reduce-> 1463/2214)
        (1241 states / 1770 arcs reduce-> 1128/1657)
    DONALD TRUMP 2.94676250208325e-14
    DONALD TRAMP 1.60706143668415e-15
    DONALD TRUMPS 9.60787650092132e-17
    DONALD TRUMPED 3.59441914484107e-17
    DONALD AUTO LUMP 5.02471578143643e-18
    Derivations found for all 1 inputs
```

For "SH" "Y" "E" "R" "I" "R" "U" "S" "A" "N" "D" "O" "B" "A" "A" "G" "U"

```
    echo '"SH" "Y" "E" "R" "I" "R" "U" "S" "A" "N" "D" "O" "B" "A" "A" "G" "U"' | carmel -sriIEQk
5 eword.wfsa eword-epron.wfst epron-jpron.wfst
    Input line 1: "SH" "Y" "E" "R" "I" "R" "U" "S" "A" "N" "D" "O" "B" "A" "A" "G" "U"
        (104 states / 203 arcs reduce-> 26/125)
        (2077 states / 2823 arcs reduce-> 1208/1825)
        (1044 states / 1497 arcs reduce-> 956/1409)
    SHERRILL SANDBERG 1.48823003208928e-16
    SHARE IL SANDBERG 6.02591635312725e-17
    SHARE ILL SANDBERG 5.59619159329042e-17
    CHERYL SANDBERG 4.53160710529496e-17
    SHARE OIL SANDBERG 1.26386844616831e-17
    Derivations found for all 1 inputs
```

2. Part of Speech Tagging

Q. 5.

Unigram model made the following predictions:

"That" "is" "a" "test" "."

DT VBZ DT NN . 4.26180685943196e-13
IN VBZ DT NN . 4.08666411177919e-14
WDT VBZ DT NN . 1.75142747647527e-14
DT VBZ DT VB . 1.09277098959992e-14
DT VBZ DT VBP . 1.09277098959817e-14

It gets it right on the 3rd try.

"The" "fly" "knows" "how" "to" "fly" "."

DT VB VBZ WRB TO VB . 1.03065155125578e-22
DT VB VBZ WRB TO NN . 1.71775258542448e-23
DT NN VBZ WRB TO VB . 1.71775258542449e-23
DT VB VBZ WRB TO VBP . 1.71775258541717e-23
DT VBP VBZ WRB TO VB . 1.71775258541717e-23

It gets it right in the 3rd try.

"The" "company" "has" "agreed" "to" "release" "its" "tax" "returns" "since" "1985" "," "and" "those" "of" "its" "affiliates" "and" "partnerships" "."

DT NN VBZ VBD TO NN PRP$ NN NNS IN CD , CC DT IN PRP$ NNS CC NNS . e^-132.86849923327
DT NN VBZ VBN TO NN PRP$ NN NNS IN CD , CC DT IN PRP$ NNS CC NNS . e^-133.9411360355
DT NN VBZ VBD TO NN PRP$ NN VBZ IN CD , CC DT IN PRP$ NNS CC NNS . e^-134.94794077495
DT NN VBZ VBN TO NN PRP$ NN VBZ IN CD , CC DT IN PRP$ NNS CC NNS . e^-136.02057757721
DT NN VBZ NN TO NN PRP$ NN NNS IN CD , CC DT IN PRP$ NNS CC NNS . e^-136.506085392993

It doesn't get this one right in the top solutions. The closest it comes to the right answer is the 2nd solution with 2 mistakes in it NN in place of VBN and DT in place of WDT.

Bigram model made the following predictions:

"That" "is" "a" "test" "."

DT VBZ DT NN . 4.97833673471794e-12
WDT VBZ DT NN . 6.58358484129539e-13
IN VBZ DT NN . 1.73399725572839e-14
DT NNS DT NN . 2.41022721032565e-15
DT VBZ JJ NN . 1.46760538477302e-15
It gets the right answer in the 2nd best solution.

"The" "fly" "knows" "how" "to" "fly" "."

DT NN VBZ WRB TO VB . 9.10574303997976e-21
DT NN VBZ WRB TO NN . 7.27552028263019e-23
DT VBP VBZ WRB TO VB . 1.65015837656453e-23
DT VB VBZ WRB TO VB . 2.41186736547203e-24
NNP NN VBZ WRB TO VB . 1.99047721519876e-24

It gets the right answer in the 1st attempt.

"The" "company" "has" "agreed" "to" "release" "its" "tax" "returns" "since" "1985" "," "and" "those" "of" "its" "affiliates" "and" "partnerships" "."

DT NN VBZ VBN TO NN PRP$ NN NNS IN CD , CC DT IN PRP$ NNS CC NNS . e^-125.6321218716
DT NN VBZ VBN TO NN PRP$ NN VBZ IN CD , CC DT IN PRP$ NNS CC NNS . e^-128.23625321128
DT NN VBZ VBD TO NN PRP$ NN NNS IN CD , CC DT IN PRP$ NNS CC NNS . e^-129.80757030657
DT NN VBZ VBN TO NN PRP$ NN NNS RB CD , CC DT IN PRP$ NNS CC NNS . e^-131.43441649972
DT NN VBZ VBN TO NN PRP$ NN VBZ RB CD , CC DT IN PRP$ NNS CC NNS . e^-131.59221785523

It's best solution is the first one but it still makes the same mistakes and doesn't get it right.

Q. 6.

I attribute the errors in the bigram system to sparse data. The mistakes the system makes is that it identifies "release" as "NN" instead of "VB" and "those" as "DT" instead of "WDT". If we inspect the training data, we can see that there are no instances where we can find "release/VB" or "those/WDT" as pairs, but there are quite a few cases of "release/NN" and "those/DT" in the training. If the system has not seen these cases then there is effectively no path through the tag-to-word.wfst to get these as possible outputs.

Q. 7.

Two thoughts come to my mind to correct these errors:

1. Collect more data. With more data we might end up getting these cases naturally. More training data is win win :D
2. Apply some sort of smoothing so that even the unseen cases have some non-zero probability attached to them.

Viterbi

The outputs of Viterbi algorithm for the 3 sentences are:

```
$ python viterbi.py That is a test .
5 tokens receive from user for processing
['That', 'is', 'a', 'test', '.']
['DT', 'VBZ', 'DT', 'NN', '.']
Score/probability: 4.97833673475e-12
Total cell count for this sentence: 270
Total non zero cells: 12
```

```
$ python viterbi.py The fly knows how to fly .
7 tokens receive from user for processing
['The', 'fly', 'knows', 'how', 'to', 'fly', '.']
['DT', 'NN', 'VBZ', 'WRB', 'TO', 'VB', '.']
Score/probability: 9.10574303994e-21
Total cell count for this sentence: 378
Total non zero cells: 14
```

```
$ python viterbi.py The company has agreed to release its tax returns since 1985 , and those of
its affiliates and partnerships .
20 tokens receive from user for processing
['The', 'company', 'has', 'agreed', 'to', 'release', 'its', 'tax', 'returns', 'since', '1985', ',', 'and',
'those', 'of', 'its', 'affiliates', 'and', 'partnerships', '.']
['DT', 'NN', 'VBZ', 'VBN', 'TO', 'NN', 'PRP$', 'NN', 'NNS', 'IN', 'CD', ',', 'CC', 'DT', 'IN', 'PRP$', 'NNS',
'CC', 'NNS', '.']
Score/probability: 2.74576093185e-55
Total cell count for this sentence: 1080
Total non zero cells: 34
```

Q. 8.

A table of the first two columns of Q[i][j] (showing only non-zero entries):

	That	is
WDT	2.6949e^-6	0 prev-state (None)
VBZ	0	**1.8630e^-6 prev-state(DT)**
DT	**0.00091**	0 prev-state (None)
NNS	0	1.0880e^-8 prev-state(DT)
IN	3.7730e^-5	0 prev-state (None)

Q. 9.

Over all 3 sentences, I get ~3.47% non-zero cells.

Q. 10.

Yes, my most probable answer from Viterbi matches the most probable answer from Carmel. I have not implemented k-best paths in Viterbi so I cannot compare and answer for the k-best solutions from Carmel.

3. Code-Breaking

Q. 11.

The decipherment string is “we can freeze your car until science discovers a way to repair it”.