# NLP Assignment #2 Report

Submitted by:

Jigna Reshamwala [425008356]

This report contains details about NLP Assignment#3. In this assignment, we have done POS tagging using Viterbi Algorithm. Also, the program implements forward algorithm. Following is the sample output for the POS tagging of the sentence: mark has fish

1. ***How to compile and run the code***

My code is tested for Python 2.7.12. It might give errors with Python 3.

To compile and run the code following are the two methods:

Using IDE

* Import the project in any Python IDE
* Edit Run Configurations and set script parameters for ViterbiAlgorithm.py to: data/probs.txt data/sents.txt
* Run the script to get the output

From Terminal

* Open Terminal
* Go the folder NLP\_HW3/python
* Type in the following command:

python ViterbiAlgorithm.py ../data/probs.txt ../data/sents.txt

***Testing another test set:***

You can use python ViterbiAlgorithm.py data/Testprobs.txt data/Testsents.txt to test the test data.

1. ***Results and Analysis***

PROCESSING SENTENCE: bears fish

FINAL VITERBI NETWORK

P(bears=noun) = 0.0160000000

P(bears=verb) = 0.0020000000

P(bears=inf) = 0.0000000100

P(bears=prep) = 0.0000000100

P(fish=noun) = 0.0001232000

P(fish=verb) = 0.0007280000

P(fish=inf) = 0.0000000440

P(fish=prep) = 0.0000004800

FINAL BACKPTR NETWORK

P(fish=noun) = verb

P(fish=verb) = noun

P(fish=inf) = verb

P(fish=prep) = noun

BEST TAG SEQUENCE HAS PROBABILITY=0.0003640000

fish->verb

bears->noun

FORWARD ALGORITHM RESULTS

P(bears=noun) = 0.0160000000

P(bears=verb) = 0.0020000000

P(bears=inf) = 0.0000000100

P(bears=prep) = 0.0000000100

P(fish=noun) = 0.0001233287

P(fish=verb) = 0.0007280145

P(fish=inf) = 0.0000000442

P(fish=prep) = 0.0000005300

PROCESSING SENTENCE: mark has fish

FINAL VITERBI NETWORK

P(mark=noun) = 0.0720000000

P(mark=verb) = 0.0060000000

P(mark=inf) = 0.0000000100

P(mark=prep) = 0.0000000100

P(has=noun) = 0.0000004620

P(has=verb) = 0.0014040000

P(has=inf) = 0.0000001320

P(has=prep) = 0.0000021600

P(fish=noun) = 0.0000864864

P(fish=verb) = 0.0000000210

P(fish=inf) = 0.0000000309

P(fish=prep) = 0.0000000351

FINAL BACKPTR NETWORK

P(has=noun) = verb

P(has=verb) = noun

P(has=inf) = verb

P(has=prep) = noun

P(fish=noun) = verb

P(fish=verb) = noun

P(fish=inf) = verb

P(fish=prep) = verb

BEST TAG SEQUENCE HAS PROBABILITY=0.0000432432

fish->noun

has->verb

mark->noun

FORWARD ALGORITHM RESULTS

P(mark=noun) = 0.0720000000

P(mark=verb) = 0.0060000000

P(mark=inf) = 0.0000000100

P(mark=prep) = 0.0000000100

P(has=noun) = 0.0000004627

P(has=verb) = 0.0014040182

P(has=inf) = 0.0000001327

P(has=prep) = 0.0000023100

P(fish=noun) = 0.0000866446

P(fish=verb) = 0.0000000379

P(fish=inf) = 0.0000000309

P(fish=prep) = 0.0000000351

PROCESSING SENTENCE: mark bears fish

FINAL VITERBI NETWORK

P(mark=noun) = 0.0720000000

P(mark=verb) = 0.0060000000

P(mark=inf) = 0.0000000100

P(mark=prep) = 0.0000000100

P(bears=noun) = 0.0000924000

P(bears=verb) = 0.0009360000

P(bears=inf) = 0.0000001320

P(bears=prep) = 0.0000021600

P(fish=noun) = 0.0000576576

P(fish=verb) = 0.0000042042

P(fish=inf) = 0.0000000206

P(fish=prep) = 0.0000000234

FINAL BACKPTR NETWORK

P(bears=noun) = verb

P(bears=verb) = noun

P(bears=inf) = verb

P(bears=prep) = noun

P(fish=noun) = verb

P(fish=verb) = noun

P(fish=inf) = verb

P(fish=prep) = verb

BEST TAG SEQUENCE HAS PROBABILITY=0.0000288288

fish->noun

bears->verb

mark->noun

FORWARD ALGORITHM RESULTS

P(mark=noun) = 0.0720000000

P(mark=verb) = 0.0060000000

P(mark=inf) = 0.0000000100

P(mark=prep) = 0.0000000100

P(bears=noun) = 0.0000925442

P(bears=verb) = 0.0009360122

P(bears=inf) = 0.0000001327

P(bears=prep) = 0.0000023100

P(fish=noun) = 0.0000578162

P(fish=verb) = 0.0000042243

P(fish=inf) = 0.0000000206

P(fish=prep) = 0.0000000262

PROCESSING SENTENCE: mark likes to fish for fish

FINAL VITERBI NETWORK

P(mark=noun) = 0.0720000000

P(mark=verb) = 0.0060000000

P(mark=inf) = 0.0000000100

P(mark=prep) = 0.0000000100

P(likes=noun) = 0.0000004620

P(likes=verb) = 0.0000046800

P(likes=inf) = 0.0000001320

P(likes=prep) = 0.0000021600

P(to=noun) = 0.0000000004

P(to=verb) = 0.0000000000

P(to=inf) = 0.0000010193

P(to=prep) = 0.0000003861

P(fish=noun) = 0.0000000263

P(fish=verb) = 0.0000000535

P(fish=inf) = 0.0000000000

P(fish=prep) = 0.0000000000

P(for=noun) = 0.0000000000

P(for=verb) = 0.0000000000

P(for=inf) = 0.0000000000

P(for=prep) = 0.0000000031

P(fish=noun) = 0.0000000002

P(fish=verb) = 0.0000000000

P(fish=inf) = 0.0000000000

P(fish=prep) = 0.0000000000

FINAL BACKPTR NETWORK

P(likes=noun) = verb

P(likes=verb) = noun

P(likes=inf) = verb

P(likes=prep) = noun

P(to=noun) = verb

P(to=verb) = noun

P(to=inf) = verb

P(to=prep) = verb

P(fish=noun) = prep

P(fish=verb) = inf

P(fish=inf) = inf

P(fish=prep) = noun

P(for=noun) = verb

P(for=verb) = noun

P(for=inf) = verb

P(for=prep) = verb

P(fish=noun) = prep

P(fish=verb) = noun

P(fish=inf) = verb

P(fish=prep) = noun

BEST TAG SEQUENCE HAS PROBABILITY=0.0000000001

fish->noun

for->prep

fish->verb

to->inf

likes->verb

mark->noun

FORWARD ALGORITHM RESULTS

P(mark=noun) = 0.0720000000

P(mark=verb) = 0.0060000000

P(mark=inf) = 0.0000000100

P(mark=prep) = 0.0000000100

P(likes=noun) = 0.0000004627

P(likes=verb) = 0.0000046801

P(likes=inf) = 0.0000001327

P(likes=prep) = 0.0000023100

P(to=noun) = 0.0000000006

P(to=verb) = 0.0000000000

P(to=inf) = 0.0000010196

P(to=prep) = 0.0000004320

P(fish=noun) = 0.0000000294

P(fish=verb) = 0.0000000536

P(fish=inf) = 0.0000000000

P(fish=prep) = 0.0000000000

P(for=noun) = 0.0000000000

P(for=verb) = 0.0000000000

P(for=inf) = 0.0000000000

P(for=prep) = 0.0000000051

P(fish=noun) = 0.0000000003

P(fish=verb) = 0.0000000000

P(fish=inf) = 0.0000000000

P(fish=prep) = 0.0000000000

Process finished with exit code 0

1. ***Any known bugs, problems, or limitations of the program***

The program can run only for four POS tags – "noun", "verb", "prep", "inf". This is because the bigram and lexical generation probabilities are given in the same file and same format and there is no way to distinguish between the two types of probabilities other than hard coding the POS tags. Also, because only 10 digits after decimal point are to be printed, values smaller than 1\*10^-10 are printed as all zeros 0.0000000000

.