

01.112 Machine Learning

Fall 2018

Design Project

Rayson Lim Jun Kai (1002026)

Angelia Lau Kah Mun (1002417)

Kimberlyn Loh (1002221)

Table of Contents

[1 Estimating only Emission Parameters of Hidden Markov Model 2](#_Toc531171600)

[1.1 Approach 2](#_Toc531171601)

[1.2 Results 3](#_Toc531171602)

[2 Estimating Hidden Markov Model 4](#_Toc531171603)

[2.1 Approach 4](#_Toc531171604)

[2.2 Results 5](#_Toc531171605)

[3 Implementation of Second-Order Hidden Markov Model 6](#_Toc531171606)

[3.1 Approach 6](#_Toc531171607)

[3.2 Results 7](#_Toc531171608)

[4 Design Challenge 8](#_Toc531171609)

[5 Conclusion 8](#_Toc531171610)

# Estimating only Emission Parameters of Hidden Markov Model

## Approach

The estimation of emission parameters of a Hidden Markov Model can be calculated by using Maximum Likelihood Estimate (MLE), whereby

where is the number of times the observation was emitted from the tag and is the number of times the tag appears in the training data.

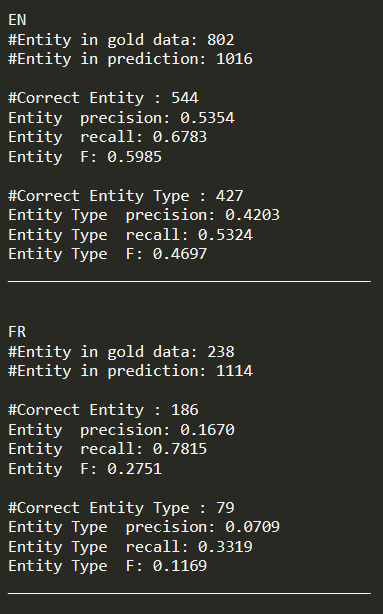
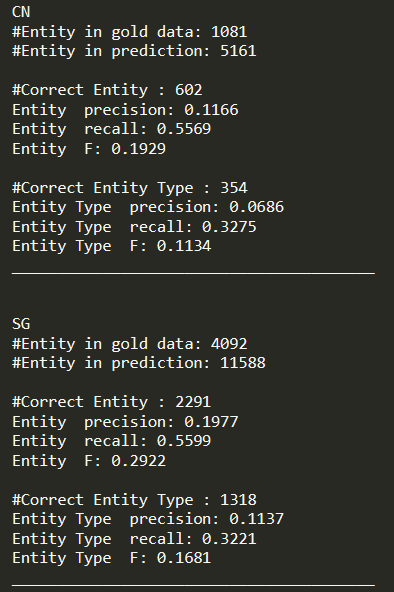
To account for words that appear in the test set but not in the training set, a special word token is used to replace words that are not in the training set. The emissions probability then becomes

where is a value that we can choose. This is a smoothing technique. We are essentially assuming that there is a certain chance of emitting as a rare event.

Given the training file, the function will calculate , utilising the formula above and return a dictionary in the form of . With the dictionary from the function, the tag generation for the test set can be performed, whereby for each word, the tag with the maximum probability is taken. In addition, words that have never appeared in training data before are replaced with the token.

## Results

The results of the above implementation of just estimating the emission is as shown below.

# Estimating Hidden Markov Model

## Approach

As observed in the previous section, utilising only emission parameters leads to low accuracy in the test set as we are assuming that every tag is independent of the tag before it. This is not a valid assumption in languages. Hence, the transition parameters can be estimated, using MLE, to be as followed.

where is the number of times the two successive tag appears and is the number of times the tag appears in the training data. In addition, Hidden Markov Model can be defined as follows.

Thus, with the estimate of transition and emission parameters, the tags can be generated. However, as brute force enumeration is computationally expensive with a time complexity of , dynamic programming like the Viterbi Algorithm with a time complexity of is implemented for decoding instead. In addition, the space complexity of the Viterbi Algorithm is .

Implementing the Viterbi algorithm results in the numerical underflow problem. This is due to the product of very small probabilities. To resolve that, we use log probabilities and sum them instead.

Our Viterbi Algorithm implementation is as follows:

1. Base Case:

u

START 🡪

u 🡪

v

1. Recursive Case:
2. Final Case:

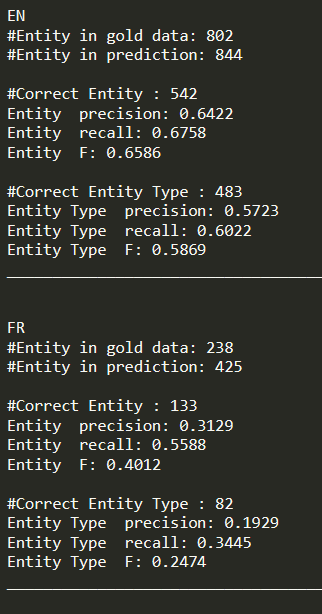
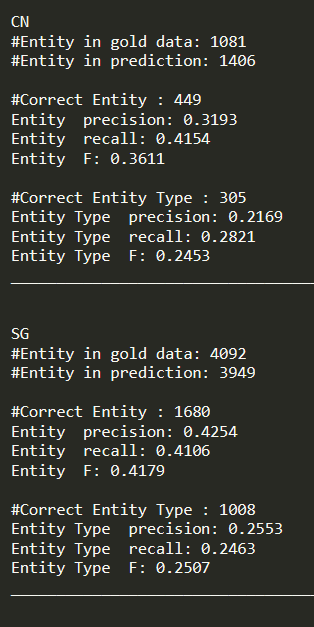
v

🡪 STOP

1. Backtracking to get the optimal sequence of tag: The parent of each node is saved. We traverse the from the node all the way to the node, saving the sequence of parents. The reverse of this sequence will give the most likely tag sequence.

## Results

The results of the above implementation of Hidden Markov Model is as shown below. As expected, the accuracy of the result have improved.

# Implementation of Second-Order Hidden Markov Model

## Approach

HMM can be extended into second-order dependencies and by parameterised into

Therefore, the emission parameter does not change whereas the transmission parameter will be changed to be as followed.

where is the number of times the successive tag appears and is the number of times the successive tag appears in the training data. In addition, the Viterbi algorithm is changed to be as followed.

The Viterbi Algorithm is as followed:

1. Base Case:

u

START 🡪 START 🡪

1. Recursive Case:

u 🡪

v 🡪

w

1. Final Case:

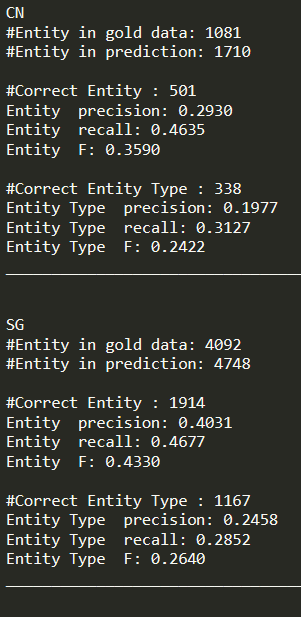
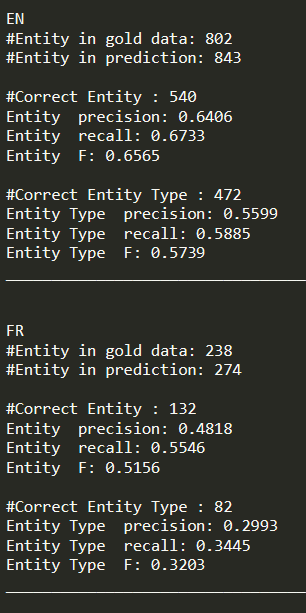
v

🡪 STOP

1. Backtracking to get the optimal sequence of tag

## Results

The results of the above implementation of second-order Hidden Markov Model is as shown below.



# Design Challenge

# Conclusion