COMP550 – Natural Language processing

Assignment 2

Badreddine SAYAH

Question 1

1. The Viterbi algorithm returns to the globally optimal state sequence for hidden Markov models.

$$arg\max_{\vec{Q}} \ P\ (\ \vec{Q}\ ,\vec{O}\ |O)$$

True.

Proof: by induction

$V(j)$ denotes the highest probability.

$$V_t(j) = arg\max_{\vec{Q}} \ \ P\ (\ \vec{Q}\ ,\vec{O}\ \ |\ O)$$

By definition:

$$V_t(j)\ =\ \max_{1\le i\le N}\ \left[V_{t-1}\ (i)a_{ij}\right] b_j(O_t\ )$$

doing term rearrangement and apply induction hypothesis:

$$B_j(O_t\ )\ \max_{1\le i\le N}\ (\ a_{ij}\ \times P\ (q_{t-1}= Q_i\ ,\ Q_1,O_{1......t-1}))$$

We can replace the transition (a_{ij}) (by its equation), move $b_j(O_t\ )$ inside the max, since it's a constant. Then e can use one-to-one Markov assumption and apply chain rule, we obtain:

$$\max_{1\le i\le N}\ \max_{\vec{Q}}\ \ \ b_j(O_t\ )\ P\ (q_{t-1}\ = Q_i\ ,\ q_t = Q_i,O_{1.....t-1})$$

$$\max_{\vec{Q}}\ b_j(O_t\ )\ P\ (q_{t-1}\ = Q_i\ ,\ q_t = Q_i, O_{t-1})$$

Bu replacing $b_j(O_t\ )$ The emission probability f O_t from the state S_j) and again using Markov assumption: we got

$$\max_{\vec{Q}} P\ (O_t\ |\ q_{t-1}\ ,\ q_t = Q_j,O_{t.1}) \times P\ (q_{t-1}\ ,\ q_t = Q_j,\ O_{t-1})$$

$$\max_{\vec{Q}} P\ (q_{t-1}\ ,\ q_t = Q_j,O_{1...t})$$

Using the arguments of the maxima; **the argmax** is stored at each step giving the best state sequence for HMM.

Question 2

<u>Non terminal categories:</u>

S	**sentence/clause**
NP	**noun phrase**
VP	**verb phrase**
N	**noun**
PN	**pronoun**
V	**verb**
DT	**determiner**
A	**adjective**
N-Pl	**noun plural**
N-M-Sg	**noun male singular**
N-F-Sg	**noun female singular**
N-M-Pl	**noun male plural**
N-F-Pl	**noun female plural**
N-Prop	**proper noun**
PR-1Sg	**pronoun first person singular**
PR-2Sg	**pronoun second person singular**
PR-3Sg	**pronoun third person singular**
PR -1Pl	**pronoun first person plural**
PR -2Pl	**pronoun second person plural**
PR -3Pl	**pronoun third person plural**
V-1Sg	**verb first person singular**
V-2Sg	**verb second person singular**
V-3Sg	**verb third person singular**
V-1Pl	**verb first person plural**
V-2Pl	**verb second person plural**
V-3Pl	**verb third person plural**
DT-Pl	**determiner plural**
DT-M-Sg	**determiner male singular**

DT-F-Sg	**determiner female singular**
A-M-Sg-Pre	**adjective male singular preceding noun**
A-M-Pl-Pre	**adjective male plural preceding noun**
A-F-Sg-Pre	**adjective female singular preceding noun**
A-F-Pl-Pre	**adjective female plural preceding noun**
A-M-Sg-Post	**adjective male singular post noun**
A-M-Pl-Post	**adjective male plural post noun**
A-F-Sg-Post	**adjective female singular post noun**
A-F-Pl-Post	**adjective female plural post noun**
DO-1Sg	**direct object first person singular**
DO-2Sg	**direct object second person singular**
DO-M-3Sg	**direct object third person male singular**
DO-F-3Sg	**direct object third person female singular**
DO-1Pl	**direct object first person plural**
DO-2Pl	**direct object second person plural**
DO-M-3Pl	**direct object third person male plural**
DO-F-3Pl	**direct object third person female plural**

2.1 Some advantages of modelling French grammar with a CFG's

- CFG's are able of modelling multiple sentences/phrases in a language using same rules with good precision.

- The rule set is quite compact and readable, moreover we can easily add new features (rules)

- Computationally tractable: we can use CFG's in a program that can 'decide' whether a sentence or phrase is grammatical or not.

2.2 Some disadvantages of modelling French grammar with a CFG's

The main disadvantage is overgeneration, we can have many rules to one terminal word.

The syntax analysis in French grammar is not enough, like (25) *"les noirs chats mangent le poisson"*

It is unlikely that the cats are eating the fish.

2.3 some aspects of French that your CFG does not handle

This CFG does not handle negation in all its forms in the French grammar, also indirect objects (like: *il lui a préter sa voiture/ he lent him his car*), multiple types of determiner and the verb tenses "imparfait, plus que parfait, future…etc" or the conjugation in different tenses .

Question 3 : Decipherment with HMMs

The table below, show the results in term of accuracy for training Hidden Markov model using NLTK module.

First, we can see that training HMM using MLE estimator gives poor results. 9.18% on cipher 1, slightly better on cipher2 even if the latter is more complex cipher.

Essentially by "fixing" the issue for unseen events where the MLE fails; using Laplace smoothing method dramatically increases the accuracies to 97.66 for cipher1, and 83.12 for the second cipher.

The plain text was improved by getting additional text from Project Gutenberg (over than 1M characters). Thus, retrieving the transitions probability improves the results on the Caesar cipher (94.02%). On the second cipher we observed an increase of four points when we consider the state sequences of the optimal path through the HMM.

By contrast, accuracy on cipher 3 do not move at all. In front of limited data available we've tried to pull more text from multiple sources (more than 6 million characters), finetuning gamma parameters of Lidstone estimation or initializing differently the HMM probabilities. We suppose that using another smoothing method like Kneser-ney smoothing can improve significantly the results. We may also consider random restarts for decipherment with HMM as in [1].

	Accuracy Built-in test function				Accuracy Best path simple (Viterbi)			
	MLE	**Laplace smoothing**	**Text improve.**	**Text improve. + Laplace smoothing**	MLE	Laplace smoothing	Text improve.	Text improve. + Laplace smoothing
Cipher 1	**9.18**	**97.66**	**94.02**	**97.66**	94.12	97.66	94.02	94.02
Cipher 2	**14.20**	**83.12**	**14.98**	**83.12**	70.64	83.11	74.80	74.80
Cipher 3	**21.30**	**21.30**	**21.30**	**21.30**	21.29	21.29	21.29	21.29

Reference: [1] "Cryptanalysis of Classic Ciphers Using Hidden Markov Models" by Rohit Vobbilisetty