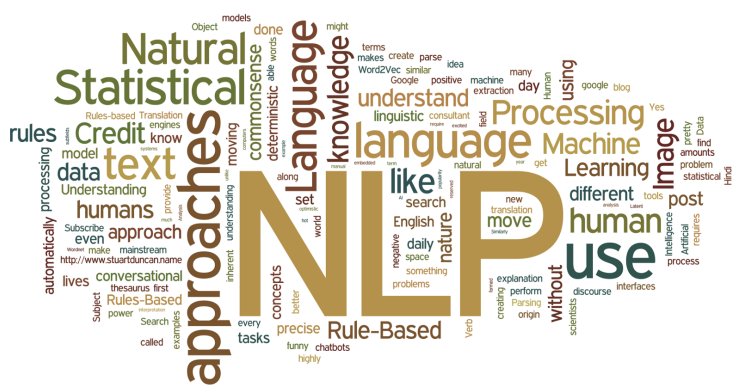
Institut Polytechnique de Paris

ENSTA Paris

MA202 - Modèles de Markov

TP 2

Les modèles HMC pour la segmentation de texte



Auteur :

Elie Azeraf

Professeur :

Wojciech Pieczynski

Année scolaire 2019-2020

1

Qu'est ce que la segmentation de texte ?

Parmi les très nombreuses tâches liées au Traitement du Langage Naturel (abrégé NLP pour Natural Language Processing), les modèles de chaînes de Markov cachées (HMC) sont particulièrement adaptées pour celle concernant la segmentation de texte.

Mais en quoi cela consiste t-il ?
Segmenter un texte consiste à labéliser chaque mot de celui-ci selon la tâche voulue. Nous avons trois types principales de segmentation que nous décrivons ci-après : Part-Of-Speech Tagging, Chunking, et Named Entity Recognition. La compréhension détaillée de ces trois tâches n'est pas nécessaires pour la suite du TP. La présentation des trois tâches est brève, absolument non exhaustive, et consiste juste à vous familiariser avec le sujet.

1.1 Part-Of-Speech Tagging

Le Part-Of-Speech (POS) Tagging consiste à labéliser chaque mot d'une phrase avec sa fonction grammaticale tel que : NOM, ADJECTIF, DÉTERMINANT, etc...
Par exemple :

— *(Bruce, Wayne, is, the, main, vigilante, of, Gotham, .)* a les labels suivants : *(NOUN, NOUN, VERB, DET, ADJ, NOUN, DET, NOUN, PUNCT)*.

— *(ENSTA, is, one, of, the, most, famous, school, in, France, .)* a les labels *(NOUN, VERB, NUM, PREP, DET, ADV, ADJ, NOUN, PREP, NOUN, PUNCT)*

Pour évaluer un modèle de POS tagging, on calcule le taux de réponses corrects (ou de réponses fausses).
Pour ce TP, nous utiliserons les datasets CoNLL 2000, CoNLL 2003, et Universal Dependencies English pour le POS Tagging (tous disponible dans le github).

1.2 Chunking

Le chunking consiste à décomposer une phrase de manière syntaxique : les composantes liées au nom, au verbe, etc...
Par exemple :

- *(Mr., Carlucci, „ 59, years, old, „ served, as, defense, secretary, in, the, Reagan, administration, .)* a les labels *(NP, NP, O, NP, NP, ADJP, O, VP, PP, NP, NP, PP, NP, NP, NP, O)*.
- *(Rockwell, said, the, agreement, calls, for, it, to, supply, 200, additional, so-called, shipsets, for, the, planes, .)* a les labels *(NP, VP, NP, NP, VP, SBAR, NP, VP, VP, NP, NP, NP, NP, PP, NP, NP, O)*

Le Chunking peut être évalué selon le score F1 (avec le label O comme étant en référence) et le pourcentage de mots bien labélisés, les deux méthodes peuvent être employées.
Dans ce TP nous regarderons le nombre de mots bien labélisés, et nous utiliserons les datasets CoNLL 2000 et CoNLL 2003 pour le Chunking.

1.3 Named-Entity Recognition

La reconnaissance d'entités nommées (NER) consiste à trouver les entités au sein d'une phrase. Une entité peut être un nom, une ville, le nom d'une société, etc...

- *(IBM, is, a, company, based, in, New, York, .)* a les labels *(ORG, O, O, O, O, O, LOC, LOC, O)*
- *(Bruce, Wayne, is, the, secret, identity, of, Batman, .)* a les labels *(PER, PER, O, O, O, O, O, PER, O)*

La NER s'évalue avec le F1 Score basé sur le label O.
Pour ce TP, nous utiliserons le dataset CoNLL 2003 pour la NER.

La personne intéressée plus en détail par ces tâches et la NLP en général peut poursuivre avec le NLTK Book : https ://www.nltk.org/book/

2

Rappel sur les chaînes de Markov cachées

Une chaîne de Markov cachées (HMC) est un double processus stochastique composé d'une séquence "cachée" et d'une séquence "observée". Le graphe de probabilité d'une HMC est donnée dans la figure ci-dessous.

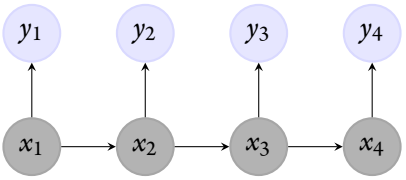


Figure 1 – Graphe de probabilité d'un HMC

Soit la séquence de taille T, nous notons $X_{1:T} = (X_1, ..., X_T)$ la séquence cachée, et $Y_{1:T} = (Y_1, ..., Y_T)$ la séquence observée. $\forall t \in \{1, ..., T\}, X_t$ prend ses valeurs dans l'espace discret Ω_X, et Y_t dans Ω_Y. La loi de probabilité d'un HMC est la suivante :

$$P(X_{1:T}, Y_{1:T}) = P(X_1)P(Y_1|X_1)P(X_2|X_1)P(Y_2|X_2)...P(X_T|X_{T-1})P(Y_T|X_T)$$

Prenons l'exemple du POS Tagging, les variables observées seraient les mots, et les variables cachées les fonctions grammaticales. Les réalisations des variables aléatoires sont notés en minuscules.

2.1 Les paramètres d'un HMC

Nous supposons prendre un HMC homogène (ses paramètres ne dépendent pas du temps), nous avons donc les trois paramètres suivants, $\forall (i, j) \in \Omega_X^2, y \in \Omega_Y$:

— Probabilité d'un état $\pi = \{\pi_i\}$ tel que : $\forall t \in \{1, ..., T\}$,

$$\pi_i = P(X_t = i)$$

— Probabilité de transition $A = \{a_{ij}\} : \forall t \in \{2, ..., T\}$,

$$a_{ij} = P(X_t = j | X_{t-1} = i)$$

— Probabilité d'émission : $\forall t \in \{1, ..., T\}$,

$$b_i(y) = P(Y_t = y | X_t = i)$$

2.2 Comment restaurer avec un HMC ? L'algorithme Forward-Backward

Supposons que nous observons les mots d'une phrase, nous avons les trois paramètres π, A, et B, nous allons voir comment retrouver nos labels grâce à l'algorithme Forward-Backward.

L'objectif de l'algorithme Forward-Backward est de calculer l'estimateur par MPM, cela consiste à calculer, $\forall t \in \{1, ..., T\}, i \in \Omega_X, P(X_t = i | \mathbf{y}_{1:T})$. Ainsi, il suffira de sélectionner i tel que cette probabilité soit la plus grande pour restaurer notre séquence.
Mais alors, comment calculer cette probabilité ?

Nous pouvoir écrire cette probabilité de la manière suivante :

$$P(X_t = i | y_{1:T}) = \frac{\alpha_i(t)\beta_i(t)}{\sum_{j \in \Omega_X} \alpha_j(t)\beta_j(t)}$$

avec $\alpha_i(t) = P(X_t = i, y_{1:t})$ et $\beta_i(t) = P(y_{t+1:T} | X_t = i)$ appelé les fonctions forward et backward, respectivement. Elles sont calculées de manière récursive.

$\forall i \in \Omega_X, t \in \{1, ..., T-1\}$, nous calculons les fonctions forward de la manière suivante :

— $\alpha_i(1) = \pi_i b_i(y_1)$,

— $\alpha_i(t+1) = b_i(y_{t+1}) \sum_{j=1}^{N} \alpha_j(t) a_{ji}$;

et les fonctions backwards :

— $\beta_i(T) = 1$,

— $\beta_i(t) = \sum_{j=1}^{T} \beta_j(t+1) a_{ij} b_j(y_{t+1})$.

Ainsi, nous avons rappelé comment restaurer une HMC avec l'algorithme Forward-Backward.

*Les questions indexées par une * se sont pas du code et peuvent être mise de côté durant la séance pour que vous puissiez vous consacrer au code uniquement (et profitez de ma présence pour vous aider^^) durant ces 3h.*

Question 1* Démontrez que les fonctions forwards et backwards se calculent bien comme ci-dessus.

3

Modéliser une phrase avec une chaîne de Markov : le principe

Nous allons, au cours de ce TP, appliquer notre modèle HMC pour un problème réel : la segmentation de texte.

Pour que tout soit clair avant d'attaquer ce TP, comment modélise-t-on ce problème ? Supposons que nous sommes dans le POS tagging, alors les variables sont donc les mots, et les variables cachées à retrouver sont les fonctions grammaticales de ces mots. Le graphe pour représenter cela est le suivant :

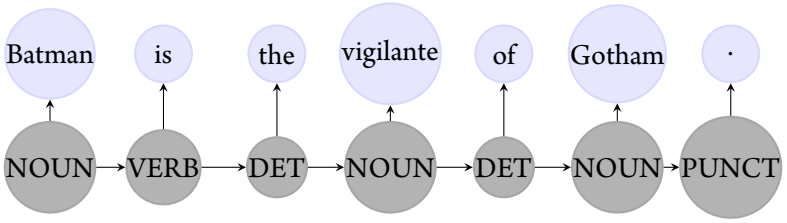


Figure 2 – Exemple d'une modélisation d'un HMC pour le POS tagging

Ainsi, nous allons appliquer notre segmentation de texte avec le modèle HMC à partir des questions suivantes.

4

Un petit mot sur le code !

Le code est disponible sur le suivant :
https ://github.com/ElieAzeraf/TD_HMC_NLP_Segmentation_IPP

Vous avez à votre disposition un code contenant trois dossier.

— Dataset : ce dossier contient les datasets et comment les charger, vous n'aurez jamais à toucher ce dossier. Sachez qu'un dataset chargé se présente sous la forme d'une liste python. Chaque élément de la liste correspond à une phrase. Une phrase est de la forme $(x_1, y_1), (x_2, y_2), ..., (x_T, y_T)$.

— Learning_Parameters : contenant le fichier qui HMC_Learning_Parameters.py comporte la fonction permettant de calculer les paramètres π, A, et B pour un dataset, vous n'avez pas besoin de modifier ce fichier non plus

— HMC : dossier principale de notre TP, vous y trouverez :

  — HMC_Forward_Backward_Restorator.py : ce fichier conporte la classe permettant d'appliquer l'algorithme Forward-Backward

  — HMC_Down_Forward_Backward_Restorator : ce fichier conporte la classe permettant d'appliquer l'algorithme Forward-Backward version Down, nous verrons cela plus tard dans le TP

  — Le dossier Result_HMC : contenant les 6 fichiers pour appliquer l'algorithme Forward-Backward de HMC, vous n'aurez pas à toucher ces fichiers

  — Le dossier Result_HMC_Down : contenant les 6 fichiers pour appliquer l'algorithme Forward-Backward version "Down", que nous verrons en détail plus tard, vous n'aurez pas à modifier ces fichiers non plus

  — Le dossier Result_HMC_Down_Bis : à compléter dans la question 8

5

Première application du HMC pour la segmentation de texte

Pour pouvoir mener à bien ce TP, l'utilisation de Python 3. avec la librairie NumPy est nécessaire.

Question 2 Dans le fichier HMC_Learning_Parameters.py du dossier Learning_Parameters se trouve la fonction pour l'estimation de nos trois paramètres π, A, et B.

1. Décrivez en détail le fonctionnement de cette fonction, comment cette estimation est-elle réalisée ?
2. Vous remarquerez que les différents paramètres sont modélisés par des dictionnaires en python, selon vous pourquoi procède t-on ainsi et non par des matrices ?
3. pouvez-vous montrer que cette estimation correspond à l'estimation des paramètres par maximum de vraisemblance ?

Question 3* A présent, prenez le fichier HMC/HMC_Forward_Backward_Restorator.py, prenez connaissance des différentes fonctions en présence. Complétez, à partir du rappel des formules de l'algorithme Forward-Backward, les fonctions :

— compute_alpha_1

— compute_alpha_t_plus_1

— compute_beta_T

— compute_beta_t

Question 4 Vous pouvez maintenant compiler l'ensemble des fichiers du dossiers Result_HMC. Présentez chacun des résultats sous forme d'un tableau, en notant bien les distinctions entre les mots connus et les mots inconnus. Que remarquez-vous ?

Question 5* Dans les quatre fonctions complétés lors de la question 2, à la fin de la fonction nous normalisons nos probabilités forward et backward que nous venons de calculer. Dans quelle but faisons-nous cela ? Pouvez-vous prouver pourquoi cela n'affecte pas les résultats de notre algorithme.

Question 6* A quoi sert le paramètre epsilon_laplace utilisé dans chacune des fonctions ?

Félicitations ! A l'issue de cette partie vous êtes capables d'appliquer le modèle HMC pour de la segmentation de textes ! A présent, voyons comment améliorer un peu les performances de notre algorithme.

6

Comment améliorer les performances de notre modèles grâce aux features ?

Au vue des résultats de notre modèle, nous pouvons être satisfait des résultats obtenus pour les mots connus (présent dans le dataset d'entraînement), néanmoins pour les mots inconnus, les résultats ne sont pas vraiment au rendez-vous.
En regardant notre algorithme Forward-Backward, le problème se situe dans l'estimation du paramètre $\forall i \in \Omega_X, b_i(y)$ lorsque y est un mot inconnu : celui-ci vaut toujours 0. Du coup, l'algorithme mis en place dans ce TD ne prend compte que du label précédent pour effectuer sa prédiction, la chaine se modélise ainsi :

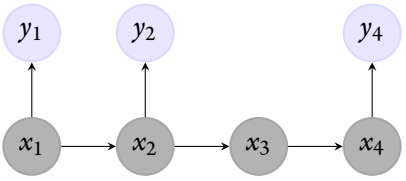


Figure 3 – Graphe de probabilité d'un HMC

Dans ce cas, y_3 étant un mot inconnu, on fait comme ci on ne l'observe pas.

Pour palier (un peu) à ce problème, nous pouvons utiliser des caractéristiques du mot pour estimer la valeur de $b_i(y)$.
Par exemple, si y est un mot inconnu mais que son suffixe de taille 3 est "ing", on se dit qu'il est probable d'avoir un verbe non ? Si y comment par une lettre en majuscules, on se dit que c'est tout de même assez probable que ce soit un nom propre, non ?
Ainsi, si y est un mot inconnu, au lieu de laisser $\forall i \in \Omega_X, b_i(y)$ à 0, nous l'estimons de la

manière suivante :

$$\hat{b}_i(y) = P(u(y), h(y), f(y), d(y), s_3(y) | X_t = i)$$

avec :

- $u(y) = 1$ si la première lettre de y est majuscule, 0 sinon
- $h(y) = 1$ si y a un hyphen (-), 0 sinon
- $f(y) = 1$ si y est le premier mot de la phrase, 0 sinon
- $d(y) = 1$ si y a un digit, 0 sinon
- $s_m(y) =$ le suffix de taille m de y

Ainsi, dans notre algorithme Forward-Backward, si y est un mot inconnu, nous estimons, $\forall i \in \Omega_X, b_i(y)$ par $\hat{b}_i(y)$

Question 7 En vous inspirant du fichier HMC_Forward_Backward_Restorator.py, complétez, avec la nouvelle règle énoncée le fichier HMC_Down_Forward_Backward_Restorator.py. Pour cela, nous donnons en argument de notre fonctions une liste de paramètre B, et une liste de Y. Pour vous aider et vous vérifier, vous pourrez utiliser les fichier du dossier Results_HMC_Down, ces fichiers ne doivent pas être modifiés.
Reportez les résultats sur un tableau comme en question 3 et commentez.

Question 8 A présent, serez-vous capable de "descendre" encore plus ? C'est-à-dire que si $\forall i, \hat{b}_i(y) = 0$, pourriez-vous faire en sorte que la restauration fasse alors une estimation pour l'ensemble des mêmes features, mais avec le suffixe de taille 2 au lieu de la taille 3 ? Puis si toutes les estimations sont encore nulles, de taille 1 ? Puis sans suffixe dans le pire des cas ?
Reportez les résultats sous forme de tableau.
Si la question précédente est bien réaliser, la question actuelle est réalisable en ne modifiant que les fichiers du dossier Result_HMC_Down_Bis que vous construirez en vous inspirant très largement des fichiers du dossier Result_HMC_Down. Commentez les résultats obtenus.

N'hésitez pas en cas de questions durant le TP, et par la suite par mail à azeraf.elie@telecom-sudparis.eu et/ou elie.azeraf@ibm.com

May the force be with you !