TP3 – **Chaînes de Markov cachées**

{Francois.Denis, Yann.Esposito}@lif.univ-mrs.fr

8 décembre 2004

☞ Il va être question d'utiliser une librairie en java qui contient tous les algorithmes relatifs aux HMM (la librairie JAHMM). Cette librairie pourra nous permettre de comprendre comment se comportent les HMM.

1 Un exemple

```
import java.util.*;
import be.ac.ulg.montefiore.run.jahmm.*;
import be.ac.ulg.montefiore.run.jahmm.toolbox.*;
import be.ac.ulg.montefiore.run.jahmm.learn.*;
import be.ac.ulg.montefiore.run.jahmm.draw.*;

public class ExempleComplet {
    static public void main(String[] argv) throws java.io.IOException {
        // --- On construit un Hmm (voir la fonction buildHmm plus loin)
        // On sauvegarde le fichier dans Init
        Hmm cible = buildHmm();
        (new HmmIntegerDrawer()).write(cible, "Cible.dot");

        // --- Génération de la séquence d'observation
        // on crée un générateur à partir du HMM
        MarkovGenerator mg = new MarkovGenerator(cible);
        // on génère une chaînes de 1000 observations
        Vector sequence = mg.observationSequence(100);
        // on crée un vecteur de séquences
        Vector sequences = new Vector();
        // on ajoute la chaine à l'ensemble de séquences
        sequences.add(sequence);

        // --- Apprentissage de Baum-Welch
        // on fabrique un BaumWelchLearner
        BaumWelchLearner bwl =
            new BaumWelchLearner(2, new OpdfIntegerFactory(2), sequences);

        // On crée un nouveau HMM
        Hmm learntHmm = buildInitHmm();
        // On sauvegarde le fichier dans Init
        (new HmmIntegerDrawer()).write(learntHmm, "InitHmm.dot");

        for (int i = 0; i < 10; i++) { // on va faire 10 itération de Baum-Welch
                                       // On crée un calculateur de la divergence de Kullback-Leibler
            KullbackLeiblerDistanceCalculator klc =
            new KullbackLeiblerDistanceCalculator(cible, learntHmm, false);
            // On affiche la divergence
            System.out.println(i + "-" + klc.distance() );
            // On fait une itération de Baum Welch
            learntHmm = bwl.iterate(learntHmm);
        }
        // On écrit le fichier final dans learnFinalHmm.dot
        (new HmmIntegerDrawer()).write(learntHmm, "learntFinalHmm.dot");

        // Calcul direct de la probabilité de la séquence oseq
```

```
        ForwardBackwardCalculator fbc =
        new ForwardBackwardCalculator(sequence, learntHmm);
        System.out.println("Probability " + fbc.probability());

        // Calcul du logarithme de la probabilité de la séquence oseq
        ForwardBackwardScaledCalculator fbsc =
            new ForwardBackwardScaledCalculator(sequence,learntHmm);
        System.out.println("Probability (scaled) " +
        Math.exp(fbsc.lnProbability()));

        // Calcul du chemin de Viterbi
        ViterbiCalculator vc =
            new ViterbiCalculator (sequence, learntHmm);
        System.out.println(toString(vc.stateSequence()));
        System.out.println("Probabilité du chemin de viterbi = " +
                           Math.exp(vc.lnProbability()));
    }

    // ---------- Le HMM cible ----------
    static Hmm buildHmm() {
        Hmm hmm = new Hmm(2); // on crée un HMM à deux états
        hmm.setPi(0, 0.65); // 65% de chance de commencer dans l'état 0
        hmm.setPi(1, 0.35); // 35% de chance de commencer dans l'état 1
        // Remarquons que la somme fait 1

        // Génération de lettre de l'état 0
        // 85% pour 0, 15% pour 1
        hmm.setOpdf(0, new OpdfInteger(new double[] {0.85, 0.15}));
        // Génération de lettre de l'état 1
        // 30% pour 0, 70% pour 1
        hmm.setOpdf(1, new OpdfInteger(new double[] {0.3, 0.7}));

        // Transitions entre états
        hmm.setAij(0, 1, 0.05); // probabilité d'aller de l'état 0 vers l'état 1 :  5%
        hmm.setAij(0, 0, 0.95); // probabilité d'aller de l'état 0 vers l'état 0 : 95%
        hmm.setAij(1, 0, 0.30); // probabilité d'aller de l'état 1 vers l'état 0 : 30%
        hmm.setAij(1, 1, 0.70); // probabilité d'aller de l'état 1 vers l'état 1 : 70%

        return hmm;
    }

    // ---------- Le HMM donné à Baum Welch ----------
    static Hmm buildInitHmm() {
        Hmm hmm = new Hmm(2);
        hmm.setPi(0, 0.90);
        hmm.setPi(1, 0.10);
        hmm.setOpdf(0, new OpdfInteger(new double[] {0.8, 0.2}));
        hmm.setOpdf(1, new OpdfInteger(new double[] {0.1, 0.9}));
        hmm.setAij(0, 1, 0.2);
        hmm.setAij(0, 0, 0.8);
        hmm.setAij(1, 0, 0.15);
        hmm.setAij(1, 1, 0.85);
        return hmm;
    }

    // -- pour l'affichage de séquences d'entiers --
    static private String toString(int[] sseq) {
        String s = "[ ";
        for (int i = 0; i < sseq.length; i++)
            s += sseq[i] + " ";
        return s + "]";
    }
    static private String toString(Vector cluster) {
```

```
        String s = "[ ";
        for (int i = 0; i < cluster.size(); i++) {
            s += cluster.elementAt(i).toString() + " ";
        }
        return s + "]";
    }
}
```

Vous pouvez récupérer le programme suivant à l'adresse :
http://www.lif.univ-mrs.fr/~esposito/pub/FD/ExempleBM.java

2 Chemins de Viterbi

☞ Lorsque qu'un HMM est suffisamment simple, on peut utiliser le chemin de Viterbi d'une séquence dans ce HMM pour classer des zones de cette séquence. Par exemple, on peut utiliser cette technique pour détecter des zones codantes ou non codantes dans les séquences d'ADN.

2.1. Copiez le fichier `ExempleComplet.java` sous le nom `ExempleViterbi.java` et modifiez-le de façon à avoir un HMM qui simule l'expérience suivante :

☞ Zenon joue à pile ou face avec deux pièces de monnaies. L'une est bien équilibrées quand à l'autre, elle possède 70% de chance de tomber sur pile et 30% de chance de tomber sur face. Zenon utilise aussi une roulette pour savoir quand il change de pièce. À chaque fois que Zenon lance une pièce il note sur quelle face elle est tombée puis, il joue à la roulette. Si il tombe sur le numéro 0 (1% de chance de se produire) alors il change de pièce, sinon il conserve la même et rejoue à pile ou face. Il commence avec la pièce non biaisée.

Plus précisément écrivez un programme qui écrit la séquence de 0 pour pile et de 1 pour face sur la sortie standard un certain nombre de fois.

2.2. Le problème va être d'essayer de deviner à quel moment les changements de pièces ont-eu lieu. Expliquez comment s'y prendre en utilisant les chemins de Viterbi.

(a) Écrivez un programme qui joue 100 fois avec la première pièce, puis 100 fois avec la seconde puis 120 fois avec la première puis 30 fois avec la deuxième.

(b) Écrivez un programme qui étant donné une séquence de 0 et de 1 donne une estimation des moments de changement de pièce.

(c) Y-a-t'il un retard dans les prédictions ?

2.3. Expliquez comment l'algorithme `forward` peut donner des réponses plus nuancées.

2.4. L'hypothèse de Viterbi souvent considérée comme correcte en pratique est que le chemin de Viterbi d'une observation portent presque tous le poids de la probabilité pour cette observation.

(a) Vérifiez si pour le HMM de l'expérience l'hypothèse est vérifiée.

(b) L'algorithme `forward` permet-il d'améliorer la qualité des si l'hypothèse de Viterbi est vérifiée ?

2.5. Quel peut-être l'intérêt d'utiliser un entraînement de Viterbi plutôt que l'algorithme de Baum-Welch ?

3 Apprentissage de Baum-Welch

3.1. Compilez et exécutez ce programme. Que fait-il ?

3.2. Que ce passe-t-il si on change beaucoup les paramètres du HMM initial ?

3.3. À partir de combien de séquences l'algorithme de Baum Welch est-il vraiment efficace ?

(a) Le nombre d'états de la cible influence-t-il ce nombre ?

(b) Le nombre d'états du HMM initial ?

(c) Les paramètres de transitions ?

(d) Les paramètres de générations ?

3.4. Est-ce que le plus important est le nombre de séquences ou la longueur des séquences ?

3.5. Que se produit-il lorsque le nombre d'états du HMM initial est plus grand que le nombre d'états du HMM cible ?

3.6. Que se produit-il lorsque le nombre d'états du HMM cible est plus grand que le nombre d'états du HMM initial ?

(a) Donnez un cas trivial dans lesquels cela n'a aucune importance ?

(b) Pouvez-vous imaginer des cas non triviaux pour lequel cela n'a pas d'importance ?

3.7. Que se produit-il lorsque le HMM initial possède des probabilités d'émission ou de transitions nulles ?