Analyse du Comportement des utilisateurs exploitant une base de données vidéo

Sylvain Mongy*

*Univ. de Lille1, Bât. M3 59655 Villeneuve d'Ascq Cedex FRANCE
mongy@lifl.fr,
http://www-rech.enic.fr/MIIRE

Résumé. Dans cet article, nous présentons un modèle de fouille des usages de la vidéo pour améliorer la qualité de l'indexation. Nous proposons une approche basée sur un modèle à deux niveaux représentant le comportement des utilisateurs exploitant un moteur de recherche vidéo. Le premier niveau consiste à modéliser le comportement lors de la lecture d'une vidéo unique (comportement intra vidéo), le second à modéliser le comportement sur l'ensemble d'une session (comportement inter video). A partir de cette représentation, nous avons développé un algorithme de regroupement, adapté à la nature particulière de ces données. L'analyse des usages de la vidéo nous permet d'affiner l'indexation vidéo sur la base de l'intérêt des utilisateurs.

1 Introduction

De par le développement rapide des techniques de stockage et de diffusion, les vidéos, notamment digitalisées, sont de plus en plus nombreuses et accessibles. En particulier, les agences de presse, les diffuseurs TV, les agences de publicité travaillent sur des ressources vidéo grandissantes. Pour être à même de travailler sur de tels volumes, des technologies adaptées doivent être mises en oeuvre. La « fouille des usages de la vidéo », qui cherche à analyser les comportements des utilisateurs sur des ensembles de vidéo est l'une des techniques clé émergentes pour optimiser les accès aux vidéos.

Dans cet article, nous proposons d'analyser le comportements des utilisateurs d'un moteur de recherche vidéo pour améliorer la qualité de l'indexation textuelle. Notre objectif est de comprendre pourquoi et comment chacune des séquences vidéo est visionnée. Par exemple, les utilisateurs recherchant des vidéos concernant le mot-clé « montagne » visionnent successivement les vidéos (18, 73, 29) qui sont retournées dans cet ordre par le moteur de recherche. Si l'on note que dans la majeure partie des cas, la vidéo 29 est visionnée totalement alors que les vidéos 18 et 73 ne le sont que partiellement, on en déduit que, selon l'utilisateur, le concept de « montagne » est mieux exprimé par la vidéo 29 que par les vidéos 18 et 73. En conclusion, la vidéo 29 doit être proposée en premier aux utilisateurs lors des futures recherches sur le concept « montagne ». Son poids dans la vidéo 29 s'en trouve augmenté et celui des vidéos 18 et 73 réduit.

Dans ce papier nous présentons une approche qui combine usage intra-vidéo et usage inter-vidéo pour générer des profils de visite sur un moteur de recherche vidéo dans le contexte de

montage de films d'entreprise (figure 1). Nous proposons une nouvelle approche qui définit deux types de données de log collectées par le moteur de recherche. Le premier concerne la manière dont un utilisateur visionne les séquences vidéo (lecture, pause, avance rapide, stop...). A ce niveau, nous définissons le « visionnage d'une séquence vidéo » comme unité de comportement. Le second type de log trace les transitions entre les différents visionnages. Cette partie regroupe l'écriture des requêtes, la présentation des résultats et les visionnages successifs des séquences vidéo. A ce niveau plus général, nous introduisons la « session » comme unité de comportement.

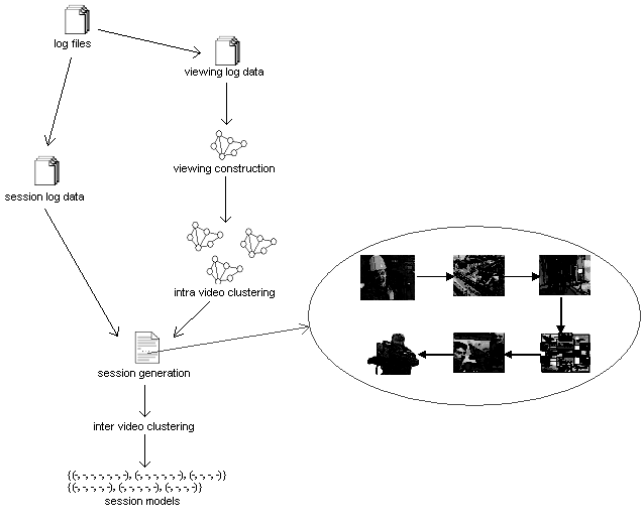

FIG. 1 – *Description globale du processus de fouille de données vidéo.*

Un comportement intra-vidéo est modélisé par un modèle de Markov du premier ordre non caché. Ce modèle est construit en utilisant les différentes actions proposées aux utilisateurs lors des visionnages (lecture, pause, avance rapide, retour rapide, saut, stop). Nous proposons une technique de regroupement de ces modèles (K-Models). Cette technique est une adaptation de la technique classique des K-Means [MacQueen (1966)] adaptée à l'utilisation de modèles en lieu et place des moyennes. Nous caractérisons ainsi plusieurs comportements type (lecture totale, lecture partielle, survol...). Grâce à ces comportements type, nous sommes à même de savoir quelle fut l'utilité ou l'importance d'une séquence vidéo lors d'une visite (si elle est la plus importante ou seulement le résultat d'une recherche infructueuse...)

Un comportement inter-vidéo est modélisé par une session. Cette session est une séquence ordonnée des visionnages des séquences vidéo. Chaque visionnage étant représenté par l'identifiant de la vidéo et le modèle de comportement qui lui correspond pour cette visite. Pour regrouper ces sessions, nous nous sommes basés sur une technique de regroupement hiérarchique ascendante que nous avons adaptée à ces données particulières.

Cet article est organisé comme suit. La section 2 présente l'état de l'art dans le domaine de la fouille des usages de la vidéo et en spécifie les particularités. La section 3 décrit le contexte applicatif de notre approche qui est le montage de films. Elle présente ensuite notre modélisation à deux niveaux des comportements des utilisateurs exploitant un moteur de recherche vidéo. Enfin, la section 4 présente les premiers résultats obtenus sur des données test. La section 5 regroupe les conclusions et introduit les futures lignes directrices de notre travail.

2 Fouille des usages de la vidéo

Quelques approches existent dans ce domaine précis. Les travaux les plus proches peuvent être classés en deux catégories.

La première concerne l'analyse des comportements des utilisateurs sans considérer le contenu de la vidéo. Ces travaux permettent de produire des statistiques sur le comportement des utilisateurs et l'analyse des fréquences des accès vidéo. Par exemple dans [Reuther et Meyer (2002)] on analyse l'usage d'un système d'enseignement multimédia par les étudiants. Cette analyse se base sur les types de personnalités des étudiants. En effet, les besoins et les attentes en apprentissage dépendent des caractéristiques du type de personnalité de l'étudiant. Pour conduire cette analyse, les auteurs ont développé un programme permettant d'extraire les actions effectuées par les étudiants sur le système multimédia et construit des profils utilisateurs permettant de tracer ce que fait chaque étudiant chaque fois qu'il utilise le système. Ces profils utilisateurs comportent les statistiques suivantes : le nombre de sessions de visionnage des vidéos, le nombre total de secondes passées à visionner les vidéos, le nombre de sessions ayant duré plus de 20 minutes, la durée moyenne d'une session, le nombre moyen de commandes par minute lors d'une session, (lecture, pause, saut...). En se basant sur les statistiques collectées sur les divers types de personnalité des étudiants, les auteurs analysent comment le système d'enseignement multimédia peut être amélioré.

Dans [Acharya et al. (2000)] une analyse des données tracées obtenues à partir des accès utilisateurs à des vidéos sur le Web est présentée. Les auteurs examinent des propriétés telles que : les variations journalières des requêtes des utilisateurs utilisant des accès vidéo. Ils proposent de tirer avantage de ces propriétés pour la conception des systèmes multimédia tels que : les systèmes de proxy, de cache et les serveurs de vidéo. A titre d'exemple leur analyse a montré que généralement les utilisateurs visionnent la première portion d'une vidéo pour savoir s'ils sont intéressés ou pas. S'ils sont intéressés, ils poursuivent le visionnage, sinon ils l'interrompent. Cette analyse suggère que mettre en cache les premières minutes d'une vidéo permettrait d'améliorer les performances d'accès au serveur vidéo.

La seconde catégorie concerne l'analyse du comportement de l'utilisateur sur une vidéo unique.

Un projet, réalisé par Microsoft [Yu et al. (2003)], développe un modèle centré « utilisateur », combinant l'analyse de contenu de la vidéo et la fouille de données log vidéo afin de générer des résumés vidéo. Le modèle utilise l'expérience des visionnages antérieurs pour guider les futurs visionnages. La contribution essentielle du travail réside dans la pondération des plans « ShotRank » de la vidéo, et l'utilisation de cette pondération dans l'extraction des résumés de la vidéo. Le « ShotRank » mesure la probabilité de visionner un plan donné durant l'exploration de la base de vidéo. La pondération des plans est calculée par un algorithme d'analyse des liens, et utilise les interactions de l'utilisateur (« Vote ») pour évaluer l'utilité et l'importance

de chaque plan. Des expériences avec des vérités de terrain ont confirmé que la pondération du plan « ShotRank » estime l'utilité et l'importance de chaque plan de vidéo, et améliore l'exploration des futurs visionnages.

Un autre projet similaire, réalisé par IBM [Syeda-Mahmood et Ponceleon (2001)], utilise également les données de log pour déterminer les type de comportement des utilisateurs visionnant des données vidéo. En se basant sur les actions réalisées et sur l'implémentation de modèles de Markov cachés, les auteurs proposent de définir des types de comportement tels que : - curieux ; - recherchant quelque chose en particulier ; - a trouvé un passage intéressant...

Le point non traité dans les travaux précédents est la non-corrélation des comportements généraux des utilisateurs avec leur comportement sur chacune des séquences vidéo. Ils ne prennent pas en compte les actions réalisées durant les visionnages lors de l'analyse de la navigation entre les vidéos (la session). Les concepts de navigation et de recherche dans une grande base de données vidéo ne sont pas définis. De plus, il n'existe à notre connaissance aucun jeu de données ou banc d'essai relatifs aux données log vidéo, ce qui ne simplifie pas la tâche de validation des résultats.

Deux points importants discriminent notre approche des travaux actuels. Tout d'abord, il n'existe aucun outil analysant l'utilisation complète d'une base de données vidéo. Les seuls travaux que nous avons référencés dans le domaine de l'analyse vidéo ne considèrent qu'une vidéo à la fois.

Ensuite, nous avons développé une technique de regroupement qui correspond à la nature de nos données. En effet, nombre de techniques du web mining utilisent des algorithmes basés uniquement sur les distances et l'approche par voisinage produisant des résultats difficiles à analyser. Il n'est pas rare de retrouver deux éléments totalement différents dans une même classe pour peu qu'ils soient connectés par une chaîne de voisins très proches les uns des autres. Nous proposons ici d'introduire un modèle de classe qui capitalisera les informations données par tous les éléments d'une même classe, éléments qui devront correspondre à ce modèle.

3 Notre approche

3.1 Contexte

L'un des besoins des professionnels de l'audiovisuel est de pouvoir retrouver facilement des séquences vidéo particulières dans d'importantes bases de films afin de les réutiliser pour le montage de nouveaux films. Notre approche met en oeuvre un moteur de recherche adapté à ce besoin. Les recherches s'appuient sur une indexation des séquences vidéo. Mais beaucoup d'informations peuvent également être extraites des usages et utilisées pour optimiser la pertinence des séquences retournées par le moteur de recherche.

Pour réaliser cette analyse, nous devons tout d'abord définir ce qu'est une utilisation du moteur de recherche vidéo. Un tel comportement peut être divisé en trois parties. - 1. Ecriture d'une requête : l'utilisateur définit les attributs sur lesquels porte la recherche et entre une valeur pour chacun d'entre-eux. Ces attributs sont utilisés pour trouver les vidéos pertinentes dans la base de vidéos indexée - 2. Exploitation des résultats : les séquences vidéo retrouvées sont présentées dans l'ordre de leur proximité aux attributs. - 3. Visionnage des séquences sélectionnées : l'utilisateur visionne les séquences qui lui semblent les plus pertinentes. Ce

visionnage est réalisé dans un lecteur offrant les fonctionnalités usuelles d'un lecteur vidéo (lecture, pause, avance rapide, retour rapide, stop, saut).

Les groupes de séquences visionnées forment des sessions. Ces dernières correspondent à une visite d'un utilisateur. Elles sont composées de plusieurs groupes recherche - exploitation et visionnage des résultats.

3.2 Collecte des données

Toutes ces données sont tracées et écrites dans des fichiers de log. Pour les créer, nous avons défini un langage adapté basé sur XML. La grammaire du langage de trace d'une session est le suivant. Une première partie contient la requête exécutée et la liste des séquences vidéo retournées. Une seconde partie trace le visionnage des séquences.

Tout comme les logs web, notre outil trace les actions de tous les utilisateurs. Pour les regrouper sous forme de sessions, nous avons développé un convertisseur XSLT (eXtensible Stylesheet Language Transformation) [w3c]. Ce convertisseur extrait et regroupe les sessions depuis les fichiers de log dans un format XML. La suite de l'article présente sous quelle forme sont modélisées les sessions.

3.3 Modélisation du comportement : un modèle à deux niveaux

A partir des données collectées précédemment, nous générons deux modèles pour représenter le comportement des utilisateurs. Le premier montre comment un utilisateur a visionné une séquence vidéo (lecture, pause, avance rapide, saut, stop). A ce niveau, nous définissions le « visionnage d'une séquence vidéo » comme une unité de comportement. Le second trace les transitions entre chaque visionnage. Cette partie regroupe les requêtes, les résultats et les séquences visionnées successivement. A ce niveau plus général, nous proposons la « session » comme unité de comportement.

Une session est une liste de séquences vidéo visionnées. L'intérêt des logs vidéo est de permettre de définir l'importance de chaque séquence pour chaque session. Plus que de les caractériser par un simple poids, comparable à l'exploitation du temps passé sur chaque document [Wang et Zaiane (2002)], nous allons ici construire plusieurs comportements type (lecture totale, aperçu, ouverture-fermeture, lecture partielle...).

3.3.1 Modélisation et regroupement des comportements intra-vidéo

Un comportement intra-vidéo est modélisé par un modèle de Markov du premier ordre (figure 2). Ce modèle représente les probabilités d'exécuter une action à chaque seconde du visionnage d'une vidéo. Les sommets correspondent aux différentes actions accessibles aux utilisateurs durant le visionnage. Ces actions sont Lecture, Pause, Stop, Avance Rapide, Retour Rapide, Saut, Stop. Par exemple, la transition de l'état Lecture vers l'état Pause de la figure (figure 2) signifie que quand un utilisateur regarde une séquence, il y a une probabilité de 8% qu'il effectue une pause la seconde suivante.

Ce modèle est totalement déterminé par les paramètres suivants :

V_i les sommets. N est l'ensemble des actions proposées à l'utilisateur lors d'un visionnage. Nous avons défini $N = \{lecture, pause, avancerapide, retourrapide, saut, stop\}$.

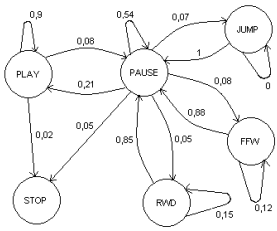

FIG. 2 – *Modèle du comportement intra-vidéo.*

π_i la probabilité de démarrer dans l'état i.

A_{ij} la probabilité de passer d'un état V_i à un état V_j la seconde suivante. Cette discrétisation du temps (avec la seconde comme unité) permettant de prendre en compte le paramètre temps dans un modèle de Markov a été introduite par [Branch et al. (1999)]. Elle permet de considérer le temps sans ajouter de paramètre supplémentaire.

Sa complexité limitée (nombre d'états peu important, premier ordre, modèle non caché) permet de proposer une méthode de regroupement efficace de ces comportements. Nous allons ici introduire la technique des K-Models. Cette technique correspond à une adaptation de la méthode bien connue des K-Means pour utiliser des modèles en lieu et place des moyennes. Nous essayons de découvrir K classes dans un ensemble de visionnages (actions exécutées sur la séquence vidéo par un utilisateur) par partitionnement de l'espace. Chaque classe est représentée par l'un des modèles de Markov décrits précédemment. La différence réside dans l'utilisation des probabilités en lieu et place des distances pour associer les visionnages aux classes. Nous calculons la probabilité qu'un visionnage ait été généré par les modèles. Nous l'assignons ensuite à la classe ayant la plus grande probabilité de l'avoir généré.

Un tel algorithme se découpe en trois phases : initialisation, allocation, maximisation.

Initialisation La phase d'initialisation, tout comme pour les K-Means, apparaît ardue. En effet, il est indispensable de définir les bons modèles pour débuter le regroupement. Dans notre cas, nous avons quelques connaissances sur les classes recherchées. Tout d'abord, V_i et π_i sont déjà définis. Le problème restant est de déterminer les probabilités A_{ij}. Même si les données vidéo disponibles sont très variées, les manières de les visionner sont relativement limitées et constantes quel que soit le contenu des vidéos. Après quelques tests sur des jeux de données réelles, nous sommes capable de définir les modèles initiaux des classes, assez hétérogènes et proche des modèles escomptés. Ils sont au nombre de quatre et correspondent aux profils suivants : 1. visionnage total ; 2. visionnage d'une scène précise ; 3. aperçu général ; 4. fermeture rapide.

Allocation Pour chaque visionnage $e = (e_1..e_l)$ e_i appartenant à N, de longueur l, pour chaque classe définie par un modèle k, nous calculons la probabilité que k ait généré e. e est associé à la classe avec la plus grande probabilité.

Maximisation Chaque modèle k représentant une classe c de cardinalité m est mis à jour en fonction des données appartenant à c. Cette mise à jour correspond à compter chaque

transition dans chaque élément e_i et d'attribuer ces comptes aux probabilités de transition du modèle. Pour chaque cluster c, les probabilités sont mises à jour de cette manière.

A partir de ces modèles, nous construisons un vecteur v_e de comportement pour chaque visionnage. Ce vecteur correspond aux probabilités que le visionnage ait été généré par chacun des modèles (1).

$$v_e = [\begin{smallmatrix} i=1 \\ v_i \\ i=K \end{smallmatrix}], \forall i, v_i = p(e|k_i) \quad (1)$$

3.3.2 Modélisation et regroupement des comportements inter-vidéo

A partir des données initiales et des vecteurs de comportement générés par le regroupement intra-vidéo, nous construisons une représentation séquentielle des sessions. Une session est une séquence ordonnée dans le temps des vidéos visionnées. Chaque visionnage est caractérisé par l'identifiant de la vidéo et par le vecteur de comportement qui lui est associé. nous avons développé un algorithme de regroupement qui répond aux besoins suivants : - chaque élément appartenant à une classe a au moins une partie commune avec les autres éléments de la classe ; - le niveau d'homogénéité des classes repose sur la définition de paramètres entrés par l'utilisateur. Ces paramètres sont présentés ci-après.

Ces besoins ont conduit à la représentation suivante des classes : une classe c est représentée par un ensemble de S sessions s_c de longueur minimale l. Une session s est attribuée à une classe si elle correspond au moins à p des S sessions. Une session s correspond à s_c si s_c est une sous-séquence extraite de s (2).

$$isSubsequence((s_1..s_n),(s'_1..s'_m)) = \exists i \leq n \mid \{ \begin{smallmatrix} s'_1 = s_i \\ isSubsequence((s_i..s_n),(s'_2..s'_m)) \end{smallmatrix} \quad (2)$$

Ainsi, nous assurons l'homogénéité des classes et le fait qu'il y ait un facteur commun entre tout élément d'une même classe. De fait nous évitons de créer des classes composées d'éléments totalement différents, connectés par une chaîne de proches voisins généralement produites par les techniques de regroupement basées sur les distances [Guha et al. (1998)]. La longueur minimale d'une séquence représentative et le nombre de séquences nécessaires pour modéliser une classe sont donnés par l'analyste, lui permettant d'obtenir des classes d'une généricité en rapport avec ses besoins.

L'algorithme de regroupement lui-même est un algorithme hiérarchique ascendant. Il débute en considérant des petits groupes de sessions comme classes et fusionne itérativement les classes les plus proches. Le regroupement prend fin lorsque le niveau d'homogénéité requis est atteint. Son originalité réside dans la représentation des classes. Les techniques de regroupement de séquences usuelles ne traitent qu'une unique séquence pour représenter une classe. Nous avons développé un outil capable de comparer et de fusionner des classes représentées non plus par une unique séquence mais par un ensemble de séquences.

Comparaison des classes Soient $C_1 = (s_{11}..s_{1S})$ et $C_2 = (s_{21}..s_{2S})$ deux classes.

$$d(C_1, C_2) = \sum_{i=1}^{S} \begin{smallmatrix} j=S \\ min \\ j=1 \end{smallmatrix} (d(s_{1i}, s_{2j})) \quad (3)$$

Cette fonction de distance (3) est basée sur la comparaison des sessions, comparaison elle-même basée sur l'extraction de la plus longue sous-séquence commune. Etant donnée $I = [(i_1, i_2)]$ la liste de longueur l des indices des éléments sélectionnés des deux sessions comparées s_1 et s_2, la distance entre elles est définie par (4) où v_{xy} est le vecteur de comportement du y^{me} élément de la session x. (5) est la fonction de distance entre deux vecteurs de comportement.

$$d(s_1, s_2) = \frac{\sum_{i_1,i_2} d(v_{1i1}, v_{2i2})}{l} \tag{4}$$

$$d(v_1, v_2) = \frac{\sum_{i=1}^{K} |v_{1i} - v_{2i}|}{K} \tag{5}$$

Fusion des classes La fonction de fusion est également basée sur l'extraction de la plus longue sous-séquence commune. Elle extrait les plus longues sous-séquences en comparant les séquences des modèles 2 à 2. En fusionnant les séquences 2 à 2, nous assurons que la proportion p est conservée jusqu'à la fin de l'exécution. Soient $c_1 = (s_{11}..s_{1S})$ et $c_2 = (s_{21}..s_{2S})$ deux classes à fusionner (6).

$$merge(c_1, c_2) = [\underset{i=1,j=1}{\overset{i=S,j=S}{merge}} (s_{1i}, s_{2j})] \tag{6}$$

i et j sont choisis pour maximiser la longueur des séquences fusionnées. Pour fusionner deux séquences, l'algorithme extrait la plus longue sous-séquence commune sans tenir compte du vecteur de comportement. Une fois cette sous-séquence créée, il fusionne les comportements correspondants des deux séquences en calculant la moyenne sur chaque élément du vecteur. Soient $v_1 = (v_{11}..v_{1K})$ et $v_2 = (v_{21}..v_{2K})$ deux comportements à fusionner, le résultat de la fusion est donné par (7).

$$merge(v_1, v_2) = [\underset{i=1,j=1}{\overset{i=K,j=K}{\frac{(v_{1i}+v_{2j})}{2}}}] \tag{7}$$

4 Resultats expérimentaux

Cette partie met en avant les deux avantages de notre approche comparée aux techniques usuelles. Tout d'abord, nous allons voir comment l'analyse du comportement intra-vidéo permet de différencier des groupes de sessions composées des même séquences vidéo mais visionnées de manière différente. Ensuite, nous allons montrer l'avantage de décrire les classes par un groupe de sessions comparé à une simple extraction de sous-séquence.

4.1 Création des jeux de données

En raison de la nouveauté des données exploitées ici, nous avons réalisé nos expérimentations sur des jeux de données de test.

La création des jeux de données de test se présente en deux phases. Premièrement, nous avons créé les modèles de comportement intra-vidéo. Nous avons défini quatre comportements type (lecture totale, survol rapide, lecture d'un passage précis, fermeture rapide). De nouvelles expérimentations sur des données réelles nous permettront de valider totalement ces modèles. A partir de ces modèles, nous avons créé aléatoirement un vecteur de comportement pour chaque visionnage des sessions vidéo.

Ensuite, pour générer les sessions vidéo, nous avons défini pour chaque classe une source de génération avec un ensemble de séquences d'identifiants vidéo. Pour chaque classe, nous avons généré les sessions en fusionnant aléatoirement ces séquences. Nous avons enfin ajouté dans ces sessions entre 5% et 20% de bruit en introduisant des identifiants de vidéo n'ayant pas de lien avec le contenu de la classe dans les séquences. Nous avons finalement généré différents jeux de données composés de 2000 sessions. Chacune est composée de 5 à 20 visionnages correspondant à des comportements de 10 à 20 actions de base (lecture, pause...). Les fichiers de tests sont donc composés d'environ 100.000 à 800.000 actions de base. La figure 3 présente quelques sessions générées pour le deuxième scénario ci-après. Une session est une séquence de vidéos visionnées à la suite d'une navigation par un utilisateur dans un intervalle de temps donné.

```
session 1: (18, 73, 29, 41, 23, 17, 25, 12, 19, 49, 87, 10, 129, 2, 73, 32, 6, 91, 301, 302, 65 ,303, 304)
session 2: (18, 73, 59, 29, 41, 301, 302, 303, 304, 62, 17, 25, 12, 19, 87, 10, 129, 2, 73, 92, 32 ,91, 31)
session 3: (129, 2, 13, 73, 32, 91, 301, 301, 303, 304, 18, 73, 29, 41, 16, 17, 25, 12, 19, 87, 11, 10)
session 4: (301, 302, 303, 69, 304, 129, 2, 73, 32, 60 ,91 ,17, 25, 12, 19, 87, 10, 18, 73, 29, 41)
session 5: (129, 2, 73, 34, 32, 91, 18, 73, 29, 33, 41, 301, 302, 303, 304, 17, 25, 12, 11, 19, 87, 10)
session 6: (1, 17, 25, 12, 19, 87, 10, 13, 301, 302, 303, 304, 18, 73, 29, 41, 129, 2, 73, 32, 91, 40)
session 7: (129, 2, 73, 32, 91, 17, 25, 12, 19, 13, 87, 10, 18, 73, 29, 14, 41, 301, 302, 303, 304, 28)
session 8: (18, 73, 29, 41, 129, 2, 73, 32, 91, 17, 25, 12, 19, 87, 10, 301, 302, 303, 304)
```

FIG. 3 – *Exemple des sessions analysées.*

La suite présente deux exécutions correspondant à des scénari prédéfinis qui mettent en avant les avantages de notre technique. Considérons dans la suite une base de données de vidéo en rapport avec la nature contenant des vidéos de montagnes et de volcans.

4.2 Exploitation du comportement intra-vidéo

Ce scenario est basée sur l'affirmation suivante. La base vidéo n'est pas correctement indexée et de nombreuses vidéos traitant des volcans sont indexées comme des vidéos de montagne.

Nous avons généré deux classes. La première correspond à une recherche sur les volcans et retourne uniquement une seule vidéo complètement visionnée. Elle correspond à la séquence (1, 2, 3, 4, 5, 6, **10**) dans laquelle seulement la vidéo **10** est visionnée (représentée en gras dans la table 1). La seconde est le résultat d'une recherche sur les montagnes et toutes les vidéos sont visionnées complètement (table 1). Elle correspond à la séquence (**1**, **2**, **3**, **4**, **5**, **6**) dans laquelle toutes les vidéos sont visionnées totalement. Avec une approche classique, ne prenant pas en compte le comportement intra-vidéo, les deux classes ne sont pas découvertes et le regroupement retourne une seule classe {(1, 2, 3, 4, 5)}.

Avec notre approche à deux niveaux, le regroupement est à même de découvrir que les vidéos ont été exploitées différemment et les deux classes sont effectivement découvertes. Pour

	vidéos fréquemment accédées	approche intra et inter-vidéo	approche inter-vidéo
rech. sur les volcans	(1, 2, 3, 4, 5, 6, **10**)	(1, 2, 3, 4, 5, 6, **10**)	(1, 2, 3, 4, 5, 6)
rech. sur les montagnes	(**1**, **2**, **3**, **4**, **5**, **6**)	(**1**, **2**, **3**, **4**, **5**, **6**)	

TAB. 1 – *Description des classes du premier jeu de données*

toute valeur inférieure à 6 de la longueur minimale des séquences représentatives, les classes {(1, 2, 3, 4, 5, 6)} et {(1, 2, 3, 4, 5, 6, 10)} sont découvertes.

4.3 Modélisation des classes par plusieurs séquences

Cette expérimentation met en avant l'avantage de modéliser les classes avec un ensemble de séquences au lieu d'une seule séquence. Nous avons généré des données correspondant à 4 classes. Trois d'entre elles correspondent à des recherches sur des stations de ski et sont composées d'une séquence spécifique et de trois autres partagées : Montagne, correspondant à la séquence (18, 73, 29, 41) ; Snowboard (17, 25, 12, 19, 87, 10) ; Ski (129, 2, 73, 32, 91). La dernière classe est également composée d'une recherche sur la montagne (18, 73, 29, 41) et d'une autre sur le trekking (2, 3, 4, 8). Pour ce jeu de données, toutes les vidéos ont été visionnées complètement (table 2).

	Resultats		
vidéos fréquemment accédées	minSize = 2 or 3	minSize = 4	minsize = 1
(18, 73, 29, 41) (17, 25, 12, 19, 87, 10) (129, 2, 73, 32, 91) (301, 302, 303, 304)		{(18, 73, 29, 41) (17, 25, 12, 19, 87, 10) (129, 2, 73, 32, 91) (301, 302, 303, 304)}	
(18, 73, 29, 41) (17, 25, 12, 19, 87, 10) (129, 2, 73, 32, 91) (401, 402, 403, 404)	{(18, 73, 29, 41) (17, 25, 12, 19, 87, 10) (129, 2, 73, 32, 91)}	{(18, 73, 29, 41) (17, 25, 12, 19, 87, 10) (129, 2, 73, 32, 91) (401, 402, 403, 404)}	{(18, 73, 29, 41)}
(18, 73, 29, 41) (17, 25, 12, 19, 87, 10) (129, 2, 73, 32, 91) (501, 502, 503, 504)		{(18, 73, 29, 41) (17, 25, 12, 19, 87, 10) (129, 2, 73, 32, 91) (501, 502, 503, 504)}	
(18, 73, 29, 41) (2, 3, 4, 8)	{(18, 73, 29, 41) (2, 3, 4, 8)}	{(18, 73, 29, 41) (2, 3, 4, 8)}	

TAB. 2 – *Regroupement du second jeu de données*

En positionnant le nombre de séquences représentatives à 2 ou 3, les sessions correspondant à l'une des trois premières classes sont fusionnées pour former une classes « sessions traitant des stations de ski ». Avec une valeur de 4, cette classe est divisée et chaque classe source est découverte. Pour ces valeurs, la classe correspondant au trekking est correctement analysée et

n'est pas fusionnée avec les autres données. Mais si nous positionnons le nombre de séquences à 1, nous plaçant ainsi dans le cadre d'une simple extraction de sous-séquence, toutes ces données sont fusionnées dans une unique classe et la différence entre les stations de ski et le trekking n'est pas détectée lors du regroupement. On voit finalement l'intérêt d'une approche multi-séquences. Elle permet de retrouver des éléments correspondant à plusieurs recherches réalisées les unes à la suite des autres dans un ordre variable et de les reconstituer dans une même classe. Sur notre exemple, on peut ainsi reformer une classe générique correspondant au travail sur les stations de ski (minSize = 2 ou 3).

4.4 Exploitation des modèles générés

Une fois ces modèles générés, l'objectif est de s'appuyer sur les connaissances extraites pour optimiser la qualité de l'indexation. Plusieurs applications sont envisagées.

Pondération des termes de l'index Pour des recherches courantes, nous pourrons mettre à jour l'ordre des vidéos retournées en fonction de l'usage fait de ces vidéos par les utilisateurs précédents. Par exemple, dans la recherche concernant la montagne et retournant les vidéos (18, 73, 29, 41), si l'on note que la vidéo 29 est systématiquement la plus regardée, elle devra apparaître en tête de liste lors des recherches contenant les vidéos 18, 73 et 41.

Détection des erreurs d'indexation Si une vidéo n'est jamais visionnée lors d'une requête précise (comme dans notre premier exemple), c'est que son indexation a été mal réalisée. Dans ce cas, il faut découvrir avec quel donnée sémantique ne correspond pas au contenu et la supprimer. Dans la recherche précédente, si la vidéo 41 n'est jamais visionnée alors qu'elle est pourtant retournée, cela signifie qu'elle ne correspond pas au concept de montagne pour les utilisateurs et cette notion doit disparaitre de son indexation.

Correction des erreurs d'indexation Lorsqu'une erreur d'indexation est découverte, il est également possible de tenter de la corriger plutôt que de la supprimer. Pour ce faire, il faut trouver, en fouillant les modèles, avec quelles autres vidéos la vidéo mal indexée est exploitée pour en déduire les bonnes données d'indexation.

5 Conclusion et travaux futurs

Nous proposons ici un modèle de représentation des utilisateurs exploitant un moteur de recherche vidéo à deux niveaux. Le premier niveau conduit à modéliser les comportements relatifs au visionnage d'une séquence vidéo (intra-vidéo), le second à modéliser les comportements observés sur un ensemble de séquence vidéo (inter-vidéo). A partir de cette représentation, nous avons développé une technique de regroupement qui permet de retourner les vidéos les plus souvent retournées et les modes de visionnage de ces vidéos. Ces derniers nous permettent de pondérer l'importance et de mettre à jour les termes de l'index.

Notre futur objectif est d'utiliser les profils découverts pour optimiser le moteur de recherche. En effet, ces résultats nous permettront de repérer les données mal indexées et de proposer aux utilisateurs les séquences vidéo en relation avec l'historique de leur session.

Remerciements

Je tiens à remercier particulièrement mon directeur de thèse Chabane Djeraba et Fatma Bouali pour leur support et leurs conseils avisés tout au long de mon travail.

Références

W3c. *http ://www.w3.org/Style/XSL*.

Acharya, S., B. Smith, et P. Parnes (2000). Characterizing user access to videos on the world wide web. *Proc. of Multimedia computing and networking*.

Branch, P., G. Egan, et B. Tonkin (1999). Modeling interactive behaviour of a video based multimedia system. *IEEE International Conference on Communications 2*, 978–982.

Guha, S., R. Rastogi, et K. Shim (1998). An efficient clustering algorithm for large databases. *SIGMOD*.

MacQueen, J. (1966). Some methods for classification and analysis of multivariate observations. *Proc. Fifth Berkeley Symp. University of California Press 1*, 281–297.

Reuther, A. I. et D. G. Meyer (2002). The effect of personality type on the usage of a multimedia engineering education system. *Frontiers in Education. FIE'02 1*, T3A7–T3A12.

Syeda-Mahmood, T. F. et D. Ponceleon (2001). Learning video browsing behavior and its application in the generation of video previews. *Proceedings of ACM MM*.

Wang, W. et O. R. Zaiane (2002). Clustering web sessions by sequence alignment. *13th International Workshop on Database and Expert Systems Applications (DEXA'02)*.

Yu, B., W.-Y. Ma, K. Nahrstedt, et H.-J. Zhang (2003). Video summarization based on user log enhanced link analysis. *MM'03*.

Summary

The analysis of user behaviors in large video databases is an emergent problem. The growing importance of video in every day life (ex. Movie production) is bound to the importance of video usage. In order to cope with the abundance of available videos, users of these videos need intelligent software systems that fully utilize the rich source information hidden in user behaviors on large video data bases to retrieve and navigate through videos. In this paper, we present a framework for video usage mining to generate user profiles on a video search engine in the context of movie production. We suggest a two levels model based approach for modeling user behaviors on a video search engine. The first level aims at modeling and clustering user behavior on a single video sequence (intra video behavior), the second one aims at modeling and clustering user behavior on a set of video sequences (inter video behavior). Based on this representation we have developed a two phase clustering algorithm that fits these data.