# Délinéarisation automatique de flux de télévision

Gaël Manson

N° d'ordre : 4128 ANNÉE 2010



**THÈSE / UNIVERSITÉ DE RENNES 1**

*sous le sceau de l'Université Européenne de Bretagne*

pour le grade de

**DOCTEUR DE L'UNIVERSITÉ DE RENNES 1**

*Mention : Informatique*

**École doctorale MATISSE**

présentée par


# Gaël Manson

préparée à l'unité de recherche TEXMEX – UMR 6074 IRISA / INRIA
et à Orange Labs – France Télécom R&D, Rennes
Composante universitaire : IFSIC


---

# Délinéarisation automatique de flux de télévision

**Thèse à soutenir à l'IRISA, Rennes en juillet 2010**

devant le jury composé en partie de :

**Nozha BOUJEMAA**
Directrice de recherche, INRIA / rapporteur

**Bernard MERIALDO**
Professeur, EURECOM / rapporteur

**Patrick GROS**
Directeur de recherche, INRIA / directeur de thèse

**Sid-Ahmed BERRANI**
Ingénieur de recherche, Orange Labs / encadrant

# Table des matières

# Introduction générale

*Avec la télévision, il n'y a plus qu'un flux permanent,*
*un ruissellement ininterrompu, tout coule et rien ne reste.*
*L'œil ne contemple plus, il avale.*

Alain Finkielkraut, philosophe, février 1999.

Cette thèse porte sur la délimitation et l'étiquetage des éléments audiovisuels diffusés en continu à la télévision. À ce jour, les chaînes de télévision sont nombreuses et variées [1]. Elles transmettent sans arrêt, 24 heures sur 24, de manière linéaire, des flux d'images et de sons. Cette linéarité génère des contraintes temporelles. En effet, les éléments ne sont diffusés qu'à des moments précis. La délimitation des éléments dans les flux permet alors de s'affranchir de ces contraintes. Les éléments délimités peuvent ainsi être classés, étiquetés, indexés et réutilisés à travers les principales applications suivantes :

- la régulation et la réglementation des diffusions. Ceci est effectué en France par le Conseil supérieur de l'audiovisuel (CSA), ou au niveau européen par l'*European Platform of Regulatory Authorities* (EPRA), ou la *Federal Communications Commission* (FCC) aux États Unis ;
- la vérification ou l'examen des diffusions pour la pige publicitaire (*TNS Media Intelligence*) ;
- la mesure d'audience (Médiamétrie, *Nielsen Media Research*) ;
- l'archivage et la documentation des diffusions, comme à l'Institut national de l'audiovisuel (INA) ou à *Beeld en Geluid* aux Pays Bas ;
- la télévision à la demande (TVoD – *TV on demand*).

La dernière application est visée plus particulièrement par cette thèse. Cette application se traduit par de nouveaux services et modes de consommation de la télévision. Nous pouvons citer notamment :

- les *nPVR* : ce sont les enregistreurs numériques personnels connectés à internet. Les enregistrements sont commandés à distance puis effectués par un opérateur tiers. Les éléments enregistrés sont ensuite récupérés par internet ;

1. La *Federal Communications Commission* chargée de réguler les diffusions télévisuelles américaines a comptabilisé 971 chaînes TV numériques en juin 2009 aux États Unis. (`http://www.fcc.gov/dtv/`)

– le *start-over* : ce service permet de revenir au début d'un élément en cours de diffusion ;

– la *catch-up-TV* : c'est la possibilité de revoir des éléments diffusés plusieurs heures, plusieurs jours ou plusieurs semaines plutôt.

Les utilisateurs de ces nouveaux services obtiennent le choix de regarder, quand il leur convient, des éléments diffusés, sélectionnés avec soin ; l'œil peut à nouveau contempler.

## Problématique

Un flux télévisuel est un flux audiovisuel, c'est à dire une suite ininterrompue d'images et de sons. Il est produit par une chaîne de télévision. Il peut également être accompagné de quelques informations de description, appelées métadonnées, fournies par la chaîne. Ces métadonnées sont soit diffusées avec le flux, soit disponibles sur internet. Notre problématique consiste à identifier dans un tel flux TV, de façon automatique et précise, le début, la fin et le titre ou la catégorie de chaque jeu, journal, magazine, film, documentaire, publicité, bande annonce, etc.

Nos travaux appartiennent au domaine général de la **structuration** des flux TV. Nous cherchons à délimiter les bornes des éléments diffusés. Cependant, différents niveaux de granularité dans la structuration des flux existent. En effet, certains éléments diffusés peuvent eux-mêmes appartenir à un élément plus imposant, comme une météo insérée dans un magazine ou un épisode d'une série diffusée sur une soirée en deux épisodes de deux parties. Les éléments diffusés peuvent aussi être regroupés par thème. Quelques éléments peuvent posséder une structure propre. Par exemple, les journaux télévisés sont constitués de plateaux présentateurs et de reportages qui peuvent aussi être regroupés par thème ou par sujet d'actualité. De plus, certains éléments sont liés, comme un parrainage ou une bande annonce d'une prochaine diffusion.

Dans notre contexte, nous visons la télévision à la demande. Nous proposons donc de considérer les éléments diffusés suivant deux notions : les inter-programmes et les programmes.

1) **un inter-programme (IP)** est un élément diffusé généralement à caractère publicitaire. Nous différencions six catégories d'inter-programmes qui sont : la publicité, la bande annonce, le parrainage, le jingle (marque de début ou de fin de la diffusion de publicités), l'auto-promotion pour la chaîne et la campagne d'intérêt général [2].

2) **un programme (P)** est un ensemble d'éléments consécutifs qui ne sont pas des inter-programmes et qui sont liés par une même charte audiovisuelle. Il est principalement à valeur culturelle, informative ou divertissante. Il peut être constitué de plusieurs parties séparées par des coupures « publicitaires » contenant des inter-programmes. Cela peut être un film, un épisode d'une série, un jeu, un journal, la météo, un clip, un magazine, un documentaire, etc. Chaque programme possède un titre qui

2. Par exemple, l'institut national de prévention et d'éducation pour la santé a lancé en 2006 une campagne d'information sur le thème « Le sucre, le sel et le gras ne sont pas toujours là où on le pense ». (`http://www.mangerbouger.fr`)

le qualifie. Nous choisissons de différencier deux types de programmes : les programmes courts qui ont une durée totale inférieure à 5 minutes et les autres programmes, les programmes longs.

Un flux télévisuel est composé de programmes entiers, de parties de programmes et d'inter-programmes assemblés consécutivement lors de la production du flux. À la réception d'un flux, les informations relatives aux bornes des programmes et des inter-programmes sont perdues. Le flux devient ainsi un unique continuum naturellement compréhensible par la perception humaine.

Dans la suite de cette thèse, nous appelons « **délinéarisation** » **d'un flux de télévision continu** la capacité à découvrir la structure sous-jacente d'un flux TV en différents programmes et inter-programmes. Cette délinéarisation est dite **automatique** lorsque la découverte s'effectue sans intervention humaine. La délinéarisation est aussi dite **basée sur le contenu** lorsque seul le contenu audiovisuel et les métadonnées éventuelles du flux sont utilisés. La délinéarisation requiert de délimiter et d'étiqueter les programmes et les inter-programmes. Ces derniers sont automatiquement extraits du flux télévisuel continu et linéaire comme cela est illustré sur la figure 1.

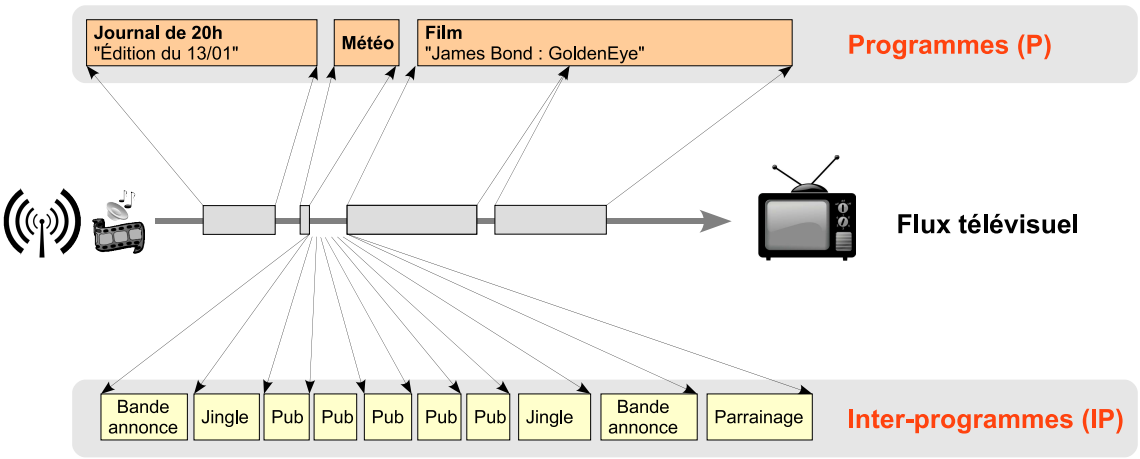



FIGURE 1 – Délinéarisation d'un flux TV. Délimitation et extraction des inter-programmes et des programmes.

Nous nous limitons dans cette thèse à la délinéarisation automatique basée sur le contenu. Les problèmes liés à notre problématique de recherche sont multiples. Ils peuvent s'exprimer dans les points suivants :

- comment délimiter automatiquement les programmes et les inter-programmes qui composent un flux TV ?
- Comment organiser/identifier/classer/nommer les programmes et les inter-programmes très variés ?
- Comment rendre l'approche la plus générique possible pour faire face à la diversité des chaînes TV ?
- Comment gérer la continuité des flux TV ininterrompus ?

## Intérêt d'un système automatique basé sur le contenu

La manière la plus directe pour effectuer une délinéarisation automatique précise pourrait être de réutiliser les informations de production. Les chaînes connaissent les programmes et les inter-programmes qu'elles insèrent dans leurs flux. Ceux-ci sont ainsi souvent inscrits au préalable dans un conducteur d'antenne lors de la phase de production. Il paraît donc facile de conserver ces informations qui fournissent une délinéarisation inhérente à la production. Ces informations peuvent, de plus, être proprement communiquées dans les flux à travers des protocoles tels le PDC [ETS98]. Nous verrons plus en détail certains de ces protocoles dans le chapitre 1.

Il s'avère, malheureusement, que ce n'est pas la technologie qui freine l'utilisation de la délinéarisation inhérente à la production mais les chaînes elles-même. Une application non évoquée précédemment de la délinéarisation est la possibilité de retirer tous les écrans publicitaires. Étant donné que la publicité est une source principale de financement des chaînes TV sans abonnement, le moyen de l'éviter est par conséquent dissimulé et non communiqué dans le flux.

Les informations issues de cette délinéarisation inhérente à la production pourraient cependant être distribuées à des organismes publics (CSA, INA) ou vendues à des sociétés sans être communiquées dans le flux. Malheureusement, ces informations ne sont pas forcément évidentes à recueillir pour toutes les chaînes[3]. Il faut savoir que le système de production et de diffusion de la télévision implique de nombreux acteurs entre lesquels le dialogue peut se révéler hautement compliqué et bruité. Cette délinéarisation inhérente à la production nécessite aussi d'être normalisée pour s'appliquer à de très nombreuses chaînes. Ceci implique un effort de standardisation entre les chaînes. La réponse à cet effort est non immédiate et réclame une réelle motivation. Bien que les nouveaux services dérivant de la délinéarisation automatique soient certains, ceux-ci bouleversent le système économique actuel du monde de la télévision. Les opérateurs des nPVR peuvent notament s'insérer entre les téléspectateurs et les chaînes TV. Ces opérateurs peuvent alors remplacer la publicité des chaînes par la leur.

Actuellement, les organismes et sociétés tels que le CSA, TNS Media Intelligence, l'INA ou Médiamétrie utilisent les résultats d'une délinéarisation majoritairement manuelle. Celle-ci est en place depuis de nombreuses années et traite en parallèle tous les flux des grandes chaînes. Néanmoins, elle possède un coût élevé en personnes et en moyens. Elle nécessite beaucoup d'attention et elle impose un travail fastidieux non exempt d'erreurs. Le lecteur trouvera en annexe B le bilan d'une expérience de délinéarisation manuelle que nous avons réalisée.

L'automaticité est indispensable pour le traitement en parallèle d'un très grand nombre de flux TV. Sans la participation des chaînes, la délinéarisation basée sur le contenu devient la principale solution de délinéarisation automatique. Nous pouvons noter toutefois que l'évolution et le succès de cette technologie peut sérieusement influencer la position actuelle des chaînes TV.

3. Plusieurs accords et coopérations avec certaines grandes chaînes françaises nous ont montré la complexité de conserver proprement les conducteurs d'antenne.

## Aperçu des méthodes existantes

Comme présenté dans la définition de notre problématique, le contenu à disposition pour la délinéarisation est constitué d'images, de sons et éventuellement de métadonnées.

Les métadonnées fournies par les chaînes correspondent à des grilles de programmation. Elles peuvent être assimilées à des résultats de délinéarisations grossières des flux. Elles visent à informer le téléspectateur des programmes qui seront diffusés dans la journée. Certains programmes peuvent manquer, d'autres peuvent être annoncés par erreur. Les horaires sont approximatifs et ils n'indiquent pas les coupures publicitaires. Malgré tout, ces métadonnées contiennent des noms et parfois des descriptions détaillées des programmes diffusés.

De manière générale, les méthodes de délinéarisation se concentrent dans un premier temps sur les signaux audiovisuels. Ensuite, les résultats obtenus sur les flux audiovisuels peuvent être alignés avec les métadonnées pour faciliter l'étiquetage des programmes [Nat07].

Les méthodes de délinéarisation se séparent en deux groupes. Les approches du premier groupe vont se concentrer sur les programmes pour tenter de les extraire directement dans les flux. Ces approches reposent sur la détection des bornes des programmes. Les bornes d'un programme peuvent être identifiées en découvrant des ruptures d'homogénéité [EKSJ08] ou bien en détectant les génériques par reconnaissance dans un ensemble de référence, par l'étude des répétitions quotidiennes [LLXT05] ou par des caractéristiques communes [WDL+08]. L'inconvénient de ces approches est qu'elles sont spécifiques aux programmes récurrents ou aux programmes non divisés par des coupures publicitaires. Ces approches se contentent, de plus, des programmes. Ces méthodes ne s'appliquent pas aux inter-programmes qui ne sont pas délimités individuellement.

Les approches du deuxième groupe vont se concentrer à l'inverse sur les inter-programmes individuellement. Le reste du flux est dans ce cas constitué de programmes ou de parties de programmes. L'avantage de ces dernières approches est de pouvoir délimiter tous les inter-programmes et tous les programmes. La reconstruction des programmes entiers à partir de leur parties est cependant très peu étudiée. Les approches se limitent donc à la détection des inter-programmes. Elles s'inspirent fortement des méthodes de détection des publicités. La frontière entre les notions de publicité et d'inter-programme est en effet très floue. Les bandes annonces et les parrainages sont parfois considérés comme des publicités [CBF06]. Ainsi, quelques approches [DJN+02] emploient des caractéristiques intrinsèques (images monochromes, silences, absences de logo) pour la détection des inter-programmes. Ces caractéristiques imposent des *a priori* trop restrictifs sur les flux à traiter. Ces caractéristiques sont appropriées pour certaines chaînes particulières à un moment donné. C'est pourquoi, les méthodes les plus efficaces détectent les inter-programmes par une seule caractéristique qui est leur propriété de répétition. De la même manière que les publicités, les inter-programmes sont en grande majorité rediffusés et donc répétés. Des méthodes comme celle proposée dans [Nat07] reconnaissent les inter-programmes rediffusés grâce à un ensemble de référence. Cependant, l'ensemble de référence repose sur une délimitation manuelle des inter-programmes à reconnaître ultérieurement. La mise à jour de l'ensemble de référence est alors nécessaire pour gérer la continuité des flux. Pour être automatisée, celle-ci nécessite la détection d'inter-programmes inconnus. Par conséquent, d'autres méthodes [TBA08, WLQ+09, ZZZY08] proposent directement de détecter auto-

matiquement les inter-programmes inconnus sans ensemble de référence. Ces méthodes recherchent pour cela toutes les occurrences de répétitions dans le flux. Malheureusement, toutes les occurrences des répétitions ne sont pas des occurrences de répétitions d'inter-programmes. Les occurrences des répétitions détectées requièrent donc d'être classées. Ce problème n'a pas été traité efficacement par la communauté. Les méthodes existantes utilisent seulement des approches heuristiques basées sur des seuils fixes de durées.

## Présentation de l'approche

Par rapport aux méthodes existantes, nous nous fixons comme contraintes de délinéariser en entier, automatiquement et en continu de nombreux flux TV variés et ininterrompus. Ces contraintes sont ambitieuses et nous montrons à travers des expériences dans quelle mesure nous y répondons.

Nous nous focalisons donc sur les quatre contraintes clés suivantes :

1) l'efficacité,

2) la généricité,

3) l'automaticité,

4) la continuité.

Pour délinéariser entièrement le flux et délimiter à la fois les programmes et les inter-programmes, notre approche se base sur la détection des inter-programmes. De plus, nous proposons une méthode pour reconstruire les programmes à partir de leur parties délimitées. Pour la généricité et la variété des flux, notre approche n'emploie pas de caractéristique intrinsèque et préfère la propriété unique de répétition des inter-programmes. Pour l'automaticité, notre approche ne possède pas d'ensemble de référence créé et mis à jour manuellement à travers lequel les répétitions sont reconnues. Toutes les occurrences de répétitions dans le flux sont détectées puis les occurrences de répétitions d'inter-programmes sont identifiées grâce à des règles pérennes préalablement apprises. Enfin, pour la continuité, notre approche propose un traitement périodique.

Le schéma global de notre approche est illustré dans la figure 2. Les flux sont traités périodiquement. À l'issue de chaque période, la délinéarisation est effectuée sur la portion de flux correspondant à cette période de temps. Les éléments délimités peuvent ensuite être stockés dans des catalogues pour servir aux différentes applications dont la télévision à la demande (TVoD). Pour le cas particulier de la TVoD, nous ajoutons la possibilité de déclencher la délinéarisation à la demande. Lors d'une demande spéciale de délinéarisation, la portion de flux depuis la dernière délinéarisation est traitée.

Pour chaque portion de flux traitée, résultat d'une demande ou d'une échéance périodique, nous analysons le flux TV suivant trois niveaux appelés niveaux de délinéarisation. Chacun de ces niveaux représente une étape vers une délinéarisation automatique. Ces

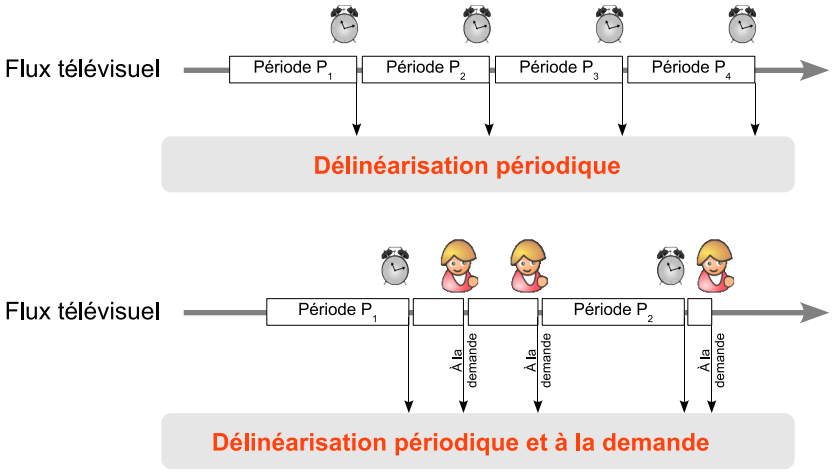



FIGURE 2 – Schéma global de l'approche. La délinéarisation proposée est périodique mais elle peut aussi être obtenue à la demande.

trois niveaux de délinéarisation sont représentés dans la figure 3 et décrits ci-dessous.

- **Niveau 1 : le découpage du flux en segments.** Toutes les occurrences des répétitions du flux sont détectées. Ces occurrences forment des segments et les séparations entre ces segments forment d'autres segments. Le flux est alors découpé en segments. Un segment issu d'une occurrence d'une répétition est appelé un segment « répété ».
- **Niveau 2 : la classification des segments.** Les segments découpés sont séparés en segments de programme et en segments d'inter-programme. Des règles de classification sont définies pour cela à partir des caractéristiques de répétitions des segments et des relations entre les segments. Ces règles sont apprises grâce à la programmation logique inductive. Ce niveau classe les segments du niveau 1. Les segments d'inter-programme peuvent aussi être classés plus précisément suivant leur catégorie : publicité, bande annonce, parrainage, jingle, etc.
- **Niveau 3 : l'extraction des programmes.** Les segments de programme appartenant à un même programme sont réunifiés. Les programmes entiers sont ainsi reconstruits. Chaque programme est ensuite étiqueté par un nom issu des métadonnées éventuelles. Ce niveau repose sur les résultats de classification du niveau 2.

La détection des inter-programmes par les répétitions est la méthode qui nous parait la plus prometteuse. Elle permet de détecter la majorité des inter-programmes de manière complètement automatique. Elle est générique et elle ne nécessite pas d'intervention manuelle. Le fait qu'un inter-programme est majoritairement rediffusé et donc répété est flagrant sur la majorité des chaînes TV. Les inter-programmes sont généralement à but commercial. C'est le cas pour les publicités, les parrainages et les bandes annonces, soit pour plus de 95 % des inter-programmes selon l'étude que nous avons réalisée sur quatre

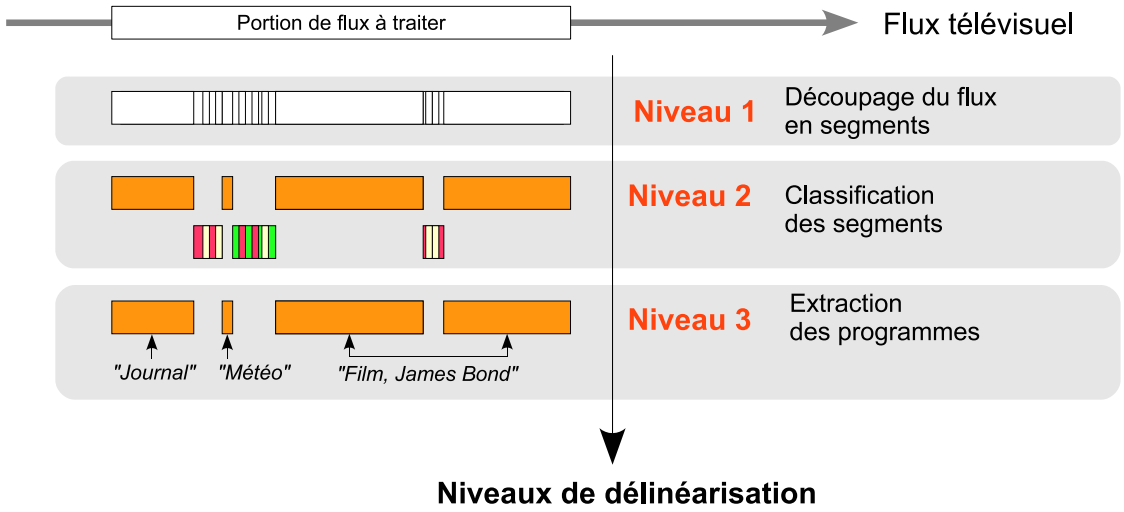



FIGURE 3 – Présentation des différents niveaux de délinéarisation.

semaines sur France 2 et TF1 (c.f. annexe A). La finalité des inter-programmes est d'être vus par un grand nombre de téléspectateurs. Mais les inter-programmes sont aussi des éléments courts, d'une durée moyenne inférieure à 30 secondes. Pour augmenter la visibilité des inter-programmes, il est par conséquent indispensable qu'ils soient rediffusés.

L'utilisation des occurrences des répétitions du flux pour la détection des répétitions d'inter-programmes et pour la délinéarisation n'est pas immédiate. Il y a deux raisons principales à cela. Premièrement, il peut persister quelques inter-programmes qui ne se répètent pas sur la portion de flux analysée. Notre découpage en segments à partir des occurrences des répétitions permet, cependant, de détecter la majeure partie de ces inter-programmes. Ceux-ci apparaissent généralement dans les segments qui séparent deux segments répétés issus de deux occurrences de répétitions d'inter-programmes. Ce découpage permet par ailleurs de délimiter les programmes et les parties de programmes. Deuxièmement, toutes les occurrences détectées des répétitions ne sont pas forcément des occurrences de répétitions d'inter-programmes. Les meilleurs exemples sont la répétition de reportages (ou de morceaux de reportages) entre le journal de 13h et le journal de 20h en France ou la répétition d'un générique d'une série diffusée chaque jour. Des filtres basés sur la durée des segments répétés ne sont pas suffisants.

Notre classification des segments permet alors de séparer efficacement les segments de programme des segments d'inter-programme. Notre technique de classification analyse les distributions des occurrences des répétitions d'inter-programmes. Intuitivement, il nous semble possible de différencier une publicité qui va se répéter plusieurs fois par jour à des horaires différents sur plusieurs jours, d'un générique qui va se répéter tous les jours à la même heure. Notre technique emploie, de plus, les informations relationnelles et contextuelles qui relient les différents segments du flux. Ces informations révèlent la structure sous-jacente du flux. L'idée est alors de découvrir des règles de la forme « *Un segment est un inter-programme si ce segment est entouré de deux autres inter-programmes et que ce*

*segment se répète un certain nombre de fois.* ». Ces informations sont donc décrites par des faits logiques et des règles de classification sont apprises et généralisées grâce à la programmation logique inductive. L'objectif est ici d'apprendre une structure pérenne, c'est à dire des règles très génériques et applicables longtemps pour conserver l'automaticité de notre système.

À propos de la réunification des parties d'un même programme, nous considérons qu'un programme entier partage un thème ou un sujet principal. Ce fil conducteur se retrouve aussi dans le mode de réalisation ou de production d'un programme. Nous recherchons donc un ensemble de propriétés homogènes dans les différentes parties d'un même programme et nous utilisons pour cela des informations de couleur, de visage ou de rythme. Cette homogénéité d'un programme tout au long de sa durée est appropriée pour rassembler ou réunifier différentes parties de programmes séparées. Au final, la dernière étape de délinéarisation des flux se fait en alignant les métadonnées sur nos programmes ainsi reconstruits pour les étiqueter. Quand les métadonnées n'existent pas, l'étiquetage reste manuel.

## Résumé des contributions

Notre contribution principale est de proposer un système complet de délinéarisation automatique. Tous les inter-programmes et les programmes sont délimités périodiquement et sans recours à une intervention manuelle. Nous avons appliqué nos travaux plus particulièrement à l'extraction de programmes et nous avons évalué rigoureusement nos résultats. Cette contribution principale s'appuie sur trois contributions élémentaires apportées aux différents niveaux de délinéarisation.

La première contribution élémentaire est la conception d'une technique de détection des occurrences des répétitions du flux audiovisuel pour le découpage. Cette contribution porte sur le niveau 1 de délinéarisation. La technique proposée est explicitement destinée à détecter les inter-programmes répétés. Elle détecte des répétitions inconnues, c'est à dire sans caractéristique audiovisuelle particulière, de durées inconnues et répétées un nombre inconnu de fois. Elle est aussi capable de traiter des flux audiovisuels sans interruption. Cette contribution se base sur les travaux de *clustering* de Berrani [Ber04] afin d'adapter sa technique de micro-*clustering* au problème. Notre technique est validée et testée expérimentalement sur de réels flux audiovisuels.

La deuxième contribution élémentaire est la conception d'une méthode originale de classification des segments préalablement découpés. Cette méthode permet, dans un premier temps, de séparer les segments d'inter-programme des segments de programme et peut, par extension, déterminer la catégorie des inter-programmes. Cette contribution porte sur le niveau 2 de délinéarisation. L'originalité de la méthode provient de l'exploitation d'informations relationnelles et contextuelles pour modéliser l'enchaînement des programmes et des inter-programmes. Elle emploie pour cela la programmation logique inductive. L'efficacité de la méthode est aussi validée et testée expérimentalement sur de réels flux audiovisuels.

La troisième et dernière contribution élémentaire de notre travail est la conception d'une technique nouvelle de réunification de segments de programmes. Cette contribution porte sur le niveau 3 de délinéarisation. À notre connaissance, ce sujet n'a pas été abordé

explicitement dans la littérature spécialisée. Cette technique permet de reconstruire, proprement et de manière automatique, sans métadonnées, les programmes diffusés.

## Plan de la thèse

Cette thèse est écrite de manière à décrire les différents travaux qui nous ont été nécessaires pour la délinéarisation automatique d'un flux télévisuel. Elle est organisée en quatre chapitres dont trois qui reprennent les différents niveaux de délinéarisation atteints progressivement.

Le premier chapitre traite des techniques existantes d'analyse de flux télévisuels pour la délinéarisation. Il détaille en particulier les travaux déjà évoqués et il expose les différents outils et pistes disponibles dans la réalisation de notre travail de recherche.

Le deuxième chapitre présente notre méthode de détection des occurrences des répétitions pour le découpage des flux en segments. Ce chapitre donne aussi les résultats de l'évaluation de cette méthode. Il montre comment atteindre le niveau 1 de délinéarisation à partir des répétitions. C'est notre première contribution.

Le troisième chapitre explique notre méthode de classification des segments. Il fournit également l'expérimentation de cette méthode. Ce chapitre nous permet d'atteindre le niveau 2 de délinéarisation. C'est notre deuxième contribution.

Le quatrième chapitre propose une méthode de réunification des segments préalablement identifiés comme segments de programme. Ce chapitre permet d'accéder enfin au niveau 3 de délinéarisation. C'est notre troisième contribution. Ce chapitre présente aussi notre contribution principale à travers l'étude expérimentale du système entier de délinéarisation automatique.

Nous concluons enfin cette thèse par une synthèse des travaux présentés et un examen de quelques perspectives de recherche à court ou à long terme.

# Chapitre 1

# Méthodes existantes pour la délinéarisation automatique

**Sommaire**



L'objectif de la délinéarisation automatique d'un flux télévisuel est de délimiter par un processus informatique tous les programmes et inter-programmes qui composent ce flux puis de leur donner un titre ou une catégorie. Les données à disposition pour cela sont les signaux audio et vidéo et les éventuelles métadonnées qui décrivent la grille de programmation de la chaîne.

Ce chapitre présente les principales méthodes existantes pour la délinéarisation automatique d'un flux télévisuel. La première partie du chapitre est dédiée à l'exploitation des éventuelles métadonnées. La deuxième partie du chapitre expose des méthodes basées sur le contenu audiovisuel.

## 1.1 Exploitation des métadonnées

Les métadonnées associées aux flux télévisuels sont des informations de description relatives aux grilles de programmation des flux. Elles sont fournies par les chaînes de télévision suivant leurs prévisions de diffusion. L'objectif de ces métadonnées est de renseigner les téléspectateurs sur les programmes prochainement diffusés ou en cours de diffusion. Ces métadonnées contiennent, par conséquent, des horaires approximatifs de programmes, des

titres de programmes et éventuellement des résumés plus ou moins détaillés comportant la liste des acteurs, des invités ou des participants, les thèmes abordés, etc.

De manière directe, les métadonnées peuvent servir à réaliser une délinéarisation grossière des flux TV. Les horaires et durées indiqués s'interprètent alors comme des indications de début et de fin de segments de programmes. Ces segments de programmes possèdent des titres et des descriptions. Les segments qui séparent les segments de programmes deviennent des segments de groupe d'inter-programmes. Malheureusement, le résultat de la délinéarisation est imprécis, incomplet et il n'est pas toujours disponible. De plus, les inter-programmes n'y sont pas présents individuellement.

Les principaux travaux autour des métadonnées ne modifient pas les métadonnées. Ils les utilisent pour les personnaliser et recommander certains programmes aux téléspectateurs [LKYH05, RGL+04], pour créer des résumés vidéo des programmes [KSY07], ou pour simplement indexer les programmes [LGS06].

Nous présentons, tout d'abord, les différents types de métadonnées qui existent. Ensuite nous étudions la précision et la qualité de ces métadonnées. Enfin, nous présentons une approche originale qui permet d'enrichir ces métadonnées.

### 1.1.1 Présentation des métadonnées

Il y a principalement deux types de métadonnées : les métadonnées qui sont associées et diffusées avec le flux audiovisuel ou bien les métadonnées qui peuvent être récupérées à la demande auprès de sites internet ou d'organisations spécialisées.

#### 1.1.1.1 Métadonnées diffusées avec le flux

Les métadonnées diffusées avec le flux dépendent des standards et des modes de diffusion. Dans le cas de la télévision analogique européenne, les métadonnées sont disponibles dans le format télétexte [ETS97b]. Le télétexte peut contenir des informations, la météo et des guides des programmes diffusés sur une semaine. Les informations textuelles sont cachés dans le signal analogique à travers des données non audiovisuelles diffusées (système VBI, *Vertical Blanking Interval*). Ces données dites inactives correspondent aux temps libres entre deux balayages du faisceau d'électrons de la télévision analogique.

En ce qui concerne les flux numériques, la norme DVB [ETS97a] (*Digital Video Broadcasting*) inclut directement le transport d'informations textuelles (DVB-SI) en plus des images et du son. Ces données sont alors « multiplexées » ou encore fusionnées pour former le signal numérique binaire transmis. Les métadonnées numériques transportées utiles pour la délinéarisation sont connues sous le nom *Event Information Table* (EIT). Elles sont de deux types :

- l'EIT *schedule* qui offre des guides des programmes diffusés sur une semaine. En pratique, ces EIT sont rarement disponibles ;
- l'EIT *present and follow* qui renseigne sur le programme en cours de diffusion et sur le programme suivant. Ces EIT sont les plus utilisées.

Les EIT peuvent être remplacées par un système de description de métadonnées plus avancé comme le standard TV-Anytime [1] [ETS09]. Cependant, la précision et la disponibilité de ces métadonnées dépend toujours du bon vouloir des chaînes.

1. Le standard est issu du TV-Anytime Forum, association d'organisations internationales qui

Pour assurer la précision temporelle des métadonnées plusieurs protocoles ont été inventés [BS02]. En particulier, le *Program Delivery Control* [ETS98] (PDC) du télétexte analogique permet d'envoyer des marqueurs de programmes. À chaque programme est associé un marqueur spécifique qui est inclus dans le télétexte tout au long de la diffusion du programme. Le début et la fin d'un programme correspondent simplement au début et à la fin de la diffusion du marqueur. Dans le format numérique DVB, des champs de descriptions sont prévus pour le transport des marqueurs PDC. Le format DVB possède aussi son propre protocole de marquage à travers les tables RST (*Running Status Table*) mises à jour continuellement. Ces tables RST contiennent une information binaire signalant si un programme identifié dans les EIT par son *Event ID* est en cours ou non. Les tables RST sont malheureusement rarement implémentées [Toz04]. Comme ces marqueurs signalent aussi la fin de la publicité avec le début du programme, très peu de chaînes les diffusent.

#### 1.1.1.2 Métadonnées disponibles en dehors du flux

Les métadonnées disponibles sur internet ou auprès de sociétés sont généralement des guides des programmes prévus sur une semaine. Ils sont appelés *Electronic Program Guides* (EPG). Nous pouvons citer par exemple *tvtv*[2] comme fournisseur payant de guides de programmes sur internet pour les enregistreurs numériques personnels. Il existe des serveurs d'EPG payants pour les entreprises comme *emapmedia*[3] ou *plurimedia*[4]. Et il y a aussi quelques logiciels gratuits tels *ZGuideTV*[5] qui récupèrent les guides de programmes sur des sites internet de magazines comme *Télépoche* ou *Téléloisirs* en France. Il est à noter que la dénomination EPG représente aussi, souvent, un format générique de guide de programmes quelque soit son mode de communication (télétext, DVB, web).

Dans la suite de la thèse, l'**EIT** désignera plus précisément les informations de délinéarisation obtenues par le service EIT *present and follow*. Elles sont fournies par les chaînes au fur et à mesure de la diffusion. L'**EPG** désignera un guide électronique des programmes d'une semaine. Il est fourni par les chaînes environ une fois par semaine et il est rarement mis à jour, une fois diffusé.

### 1.1.2 Étude expérimentale

Nous savons que, principalement à cause de la publicité, les chaînes ne fournissent pas d'horaires précis pour la délinéarisation dans les métadonnées. Il est cependant intéressant d'étudier l'imprécision de ces horaires et, de manière plus générale, la qualité de ces métadonnées. Nous nous intéressons uniquement, dans notre étude, aux grandes chaînes TV françaises. Nous ne proposons pas une étude des métadonnées internationales qui serait difficile et extrêmement coûteuse à mener. Par ailleurs, ces métadonnées restent très

travaillent sur les spécifications de services centrés sur les enregistreurs numériques personnels (`http://www.tv-anytime.org`).

2. `http://www.tvtv.com`
3. `http://www.emap.com`
4. `http://www.plurimedia.fr`
5. `http://www.zguidetv.net`

précises et complètes dans certains pays, notamment en Grande Bretagne avec le groupe de chaînes publiques anglaises *BBC*[6] non financé par la publicité.

Notre étude est réalisée sur les données à partir desquels notre corpus[7] de travail a été construit. Nous présentons deux expériences. Dans la première expérience, nous regardons simplement l'incomplétude de l'EIT et de l'EPG sur une journée entière sur une chaîne. Dans la deuxième expérience, nous estimons l'imprécision temporelle des horaires des programmes annoncés sur cette même chaîne.

#### 1.1.2.1 Incomplétude des programmes de l'EPG et de l'EIT

Nous avons récupéré l'EIT et l'EPG d'une journée entière sur la chaîne France 2. Nous avons manuellement délimité les programmes de la journée et nous présentons dans le tableau 1.1 les différences entre les programmes annoncés de l'EPG et de l'EIT et les programmes diffusés selon les résultats de notre délinéarisation manuelle qui constitue la vérité terrain (VT).

| | VT | EPG | EIT |
|---|---|---|---|
| Nombre de programmes de durée $< 5$ minutes | 24 | 5 | 22 |
| Nombre de programmes de durée $\geq 5$ minutes | 25 | 26 | 25 |
| Nombre de programmes total | 49 | 31 | 47 |

Tableau 1.1 – Nombres de programmes délimités manuellement dans la vérité terrain (VT), présents dans l'EPG et présents dans l'EIT sur une journée de France 2.

Le tableau 1.1 permet d'illustrer l'incomplétude des informations fournies par la chaîne France 2. Nous remarquons que les programmes courts, d'une durée inférieure à 5 minutes, peuvent être délaissés par les chaînes. En effet, mis à part la météo ou les résultats du loto présents dans l'EPG et l'EIT, ces programmes courts ne font généralement pas partie des programmes les plus attendus par les téléspectateurs. Des exemples de ces programmes courts sont des jeux, des informations culturelles ou pratiques ou des clips musicaux. Ils servent parfois à combler des « trous » entre deux programmes longs lors de la production des flux et sont ainsi difficilement prédictibles.

De plus, trois programmes annoncés dans l'EPG ne sont pas diffusés. De manière similaire, quatre programmes de l'EIT ne sont pas diffusés. Les programmes annoncés ne correspondent donc pas toujours aux programmes diffusés. La majorité de ces erreurs proviennent de programmes annoncés pour la nuit. Il n'y a en effet pas suffisamment de téléspectateurs la nuit pour que les chaînes fassent l'effort de respecter leurs métadonnées. Cependant, ces erreurs sont parfois dues à des changements non signalés dans la programmation des chaînes suite à des évènements de l'actualité. Dans notre cas, ces erreurs étaient dues à de mauvais programmes annoncés la nuit.

Notre étude réalisée sur une journée et sur une chaîne sert seulement d'illustration de l'incomplétude de l'EPG et de l'EIT. Une autre étude de l'incomplétude est proposée dans [Pol07]. Cette autre étude porte sur l'EPG, sur quatre années et sur quatre chaînes. Elle ne réalise cependant qu'une estimation de l'incomplétude en se basant sur un EPG modélisé. L'EPG réel n'a pas pu, en effet, être utilisé à cause de la durée de l'étude. Les

6. `http://www.bbc.co.uk/rd/projects/eca/accurate-recording/index.shtml`
7. Ce corpus est détaillé en annexe A.

résultats donnent toutefois une approximation de l'incomplétude de l'EPG à environ 65 %. Cette autre étude permet surtout d'analyser l'évolution des guides de programmes sur plusieurs années. L'auteur de cette étude montre que les guides de programmes restent globalement très stables sur le long terme. Ceci révèlle une structure pérenne dans les guides de programmes.

#### 1.1.2.2 Imprécision temporelle de l'EPG et de l'EIT

La deuxième expérience que nous proposons à pour but d'estimer l'imprécision temporelle des horaires annoncés dans l'EPG et dans l'EIT. L'imprécision temporelle représente le décalage entre l'horaire annoncé et l'horaire de diffusion. Pour mesurer ce décalage, nous avons utilisé une technologie de reconnaissance de vidéos déjà implémentée et validée au sein de notre laboratoire. Cette technologie est aussi utilisée dans des services opérationnels. Elle permet de détecter le début des programmes en reconnaissant dans le flux leur générique. Nous avons sélectionné pour notre expérience des programmes récurrents possédant un générique de début stable comme les journaux, les magazines, les jeux ou les séries télévisées. L'estimation de l'imprécision porte ainsi sur cette sélection de programmes.

Les génériques des programmes sont stockés dans un ensemble de vidéos de référence et ils sont reconnus lorsqu'ils sont rediffusés dans le flux. Cette reconnaissance permet de détecter les horaires de diffusion de manière automatique à la seconde près. Nous avons préalablement vérifié que la précision et le rappel en terme de reconnaissance de la technologie sont de pratiquement 100 %. Presque toutes les reconnaissances signalées étaient bien des reconnaissances de génériques de l'ensemble de référence et presque tous les génériques à reconnaître étaient bien reconnus.

Cette méthode a été appliquée sur 5 mois de flux vidéo de septembre 2006 à janvier 2007 sur la chaîne France 2. Nous avons sélectionné les instants du flux correspondant approximativement à la fois à une reconnaissance d'un générique d'un programme et à un début de ce programme dans l'EPG ou l'EIT. De cette manière, nous avons été capables d'estimer l'imprécision temporelle. Les résultats sont présentés dans le tableau 1.2. Ces résultats montrent que les métadonnées de l'EIT sont globalement plus précises que les métadonnées de l'EPG selon le type des programmes considérés. Malheureusement les métadonnées de l'EIT restent très imprécises. Même les journaux qui sont des programmes fidélisant un maximum de téléspectateurs ne sont précis qu'à la minute près en moyenne.

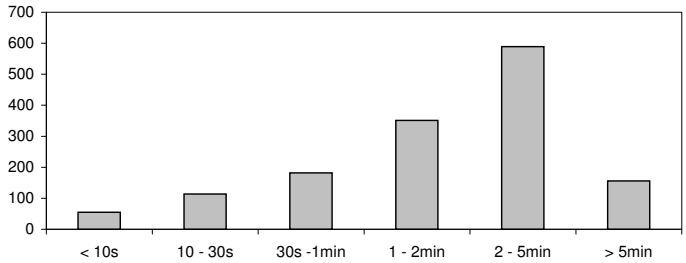
| | Journaux (13h + 20h) | | Tous les programmes | |
|---|---|---|---|---|
| | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ |
| Imprecision de l'EPG | 2m 16s | 36s | 3m 38s | 10m 18s |
| Imprecision de l'EIT | 56s | 29s | 2m 30s | 3m 48s |

Tableau 1.2 – Moyenne ($\mu$) et écart-type ($\sigma$) de l'imprécision temporelle de l'EPG et de l'EIT par rapport à la reconnaissance automatique des génériques.

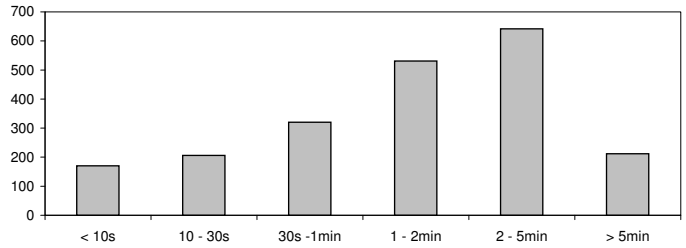
Pour mieux illustrer ces résultats, nous présentons aussi dans la figure 1.1 les distributions moyennes des imprécisions temporelles sous forme d'histogrammes. Il apparaît que moins de 10 % des programmes de l'EPG ou de l'EIT seulement sont annoncés à 10 secondes près de leur instant réel de diffusion. Par contre, 40 % des programmes de l'EPG ou de l'EIT sont annoncés avec plus de 2 minutes de retard ou d'avance sur leur instant

réel de diffusion.

Ces deux expériences réalisées sur France 2 montrent dans quelle mesure les métadonnées peuvent être imprécises et incomplètes pour une grande chaîne TV française.




(a) Distribution de l'imprécision de l'EPG




(b) Distribution de l'imprécision de l'EIT

FIGURE 1.1 – Distribution des imprécisions temporelles de l'EPG et de l'EIT par rapport à la reconnaissance automatique des génériques pour un ensemble de programmes considérés.

### 1.1.3 Enrichissement des métadonnées

Parmi les travaux exploitant les métadonnées, nous avons identifié une seule méthode de prédiction des métadonnées. Cette méthode issue des travaux de Poli [Pol07] enrichit les métadonnées en les corrigeant grâce à des prédictions. Ces nouvelles métadonnées « fiabilisées » fournissent ensuite les horaires et les durées des programmes pour la délinéarisation. L'auteur part du constat que les guides des programmes sont stables d'une semaine sur une autre tout au long de l'année. Il propose donc une modélisation statistique afin de décrire cette stabilité. Les guides de programmes sont représentés par l'intermédiaire d'un modèle de Markov caché contextuel et d'un arbre de régression. Le modèle de Markov modélise l'enchaînement des programmes entre eux par des états correspondant aux genres des programmes. Le modèle utilisé est dit contextuel par sa particularité à prendre en compte l'heure et le jour de la semaine des programmes pour passer d'un état à un autre. L'arbre de régression permet, lui, d'encadrer à partir d'un arbre de décision la durée d'un programme en fonction du jour de la semaine où il est diffusé, de son horaire de diffusion et de son genre. Le modèle statistique global est entrainé par les guides de programmes

précis établis manuellement sur les années précédentes [8]. Le modèle permet par la suite de prédire les guides de programmes et ainsi d'enrichir et de modifier le guide courant. Les résultats présentés dans cette solution sont très intéressants. Ils permettent de situer à la minute près la majorité des programmes diffusés.

La limite principale de cette méthode réside dans la quantité de données construites à la main nécessaires à l'apprentissage du modèle. Ces données représentent plusieurs années de délinéarisations manuelles. Elles doivent être établies pour chaque nouvelle chaîne et elles peuvent ne pas exister pour des flux anciens déjà enregistrés.

### 1.1.4 Synthèse

Les métadonnées fournies par les chaînes dans les flux TV peuvent finalement être :

- erronées ;
- très imprécises temporellement (de l'ordre de plusieurs minutes) ;
- incomplètes (surtout face aux programmes courts et aux inter-programmes) ;
- non disponibles (en particulier pour les enregistrements anciens d'archives).

Les métadonnées seules sont donc insuffisantes pour la délimitation des programmes et des inter-programmes. Elles offrent néanmoins une solution facile pour donner un titre à certains programmes extraits automatiquement. En particulier, pour les programmes longs qui bénéficient *a priori* de métadonnées de meilleure qualité, cette solution est plus fiable que des méthodes basées sur la transcription de la parole [JH01, MBD+06].

Les métadonnées peuvent être enrichies grâce à la méthode de prédiction de Poli [Pol07]. Cette méthode requiert cependant un grand nombre de délinéarisations manuelles de flux passés. Les résultats de ces délinéarisations nécessaires sont souvent non disponibles ou trop longues et coûteuses à créer.

À travers la méthode de prédiction des métadonnées de [Pol07], une propriété importante des guides de programmes est révélée. Cette propriété est la stabilité des guides de programmes sur le long terme. Elle traduit la présence d'une certaine structure. C'est un point clé pour la délinéarisation.

## 1.2 Exploitation des signaux audio et vidéo

Nous nous intéressons maintenant aux différents outils connus et aux pistes envisagées autour de la délinéarisation de flux télévisuels en exploitant les signaux audio et vidéo.

Les premiers travaux pertinents sont les approches classiques du domaine de l'audiovisuel qui permettent de découper les flux audiovisuels et de classer les segments découpés. Elles peuvent contribuer de manière implicite à une délinéarisation automatique. Nous étudions dans la suite les techniques de découpage en plans, en scènes ou en segments de musique, de silence et segments de parole. Puis, nous regardons les techniques de classification de divers segments basés sur leur contenu audiovisuel. Ces techniques proposent de classer des plans, des scènes ou des morceaux pré-délimités en genres comme une publicité, du sport, des informations, etc.

8. Ces travaux ont été réalisés à l'INA. Ils bénéficient de l'archivage de plus de quatre années de guides de programmes précis établis à la main.

En plus de ces travaux, les premières recherches explicites à propos de la délinéarisation d'un flux se concentrent sur la publicité. Les publicités partagent de nombreuses propriétés communes au travers de caractéristiques intrinsèques ou le fait qu'elles sont rediffusées massivement. Ce sont donc des éléments plus faciles à extraire. Par ailleurs, il existe des applications industrielles précises à la détection des publicités comme la pige ou la régulation [SM04].

À la suite de ces travaux, d'autres recherches se sont intéressées à la délimitation des programmes TV. Cela amène les approches dédiées à la délinéarisation des flux basée sur le contenu.

### 1.2.1 Approches apparentées à la délinéarisation

Les approches apparentées sont les approches classiques à la base d'une majorité de travaux d'indexation de flux audiovisuels [HWJ94, SW05a]. Ces approches peuvent théoriquement être appliquées à la délinéarisation d'un flux TV dans le sens ou elles permettent de découper des « segments » et de les classer par la suite.

Il est largement admis dans la communauté [RHM99] qu'un flux peut être vu comme un assemblage de plans vidéo. Un plan vidéo se définit [9] comme « une prise de vues effectuée sans interruption ». Ainsi, la détection des discontinuités entre les contenus des images successives du flux vidéo permet de détecter facilement les ruptures entre les plans [ZKS93, KC01]. Mis à part les transitions progressives telles les fondus, les plans sont relativement bien détectés [Han02].

Suivant le même principe, le flux vidéo peut aussi être réduit à un ensemble d'images clés [FBGS04, Wol96]. Ces images clés correspondent à des images particulières du flux. Elles résument à elles seules des portions de flux. Ces images clés peuvent être une ou plusieurs images sélectionnées dans les plans [ZLSW95].

Les plans peuvent ensuite être regroupés selon certains critères de similarité. Par exemple, ils peuvent partager un objet commun [GFT98], des personnages communs, des dialogues [Aok05], des paysages similaires ou, de manière plus subjective, une histoire commune. Les groupes de plans forment des scènes dont la définition reste ambigüe et dépend des applications [YYL96, KY98, Han02, RS05].

Les détections des plans ou des scènes ne conduisent pas directement à une délinéarisation du flux TV. Les segments issus des plans ou des scènes sont plus courts que les programmes et les inter-programmes. Parfois un inter-programme ou un programme court peut correspondre à un seul segment. Parfois un programme peut en contenir une centaine. Il n'y a pas de règles suivant les genres permettant de savoir combien de segments seront nécessaires. Les segments requièrent donc des traitements supplémentaires.

Dans le cadre de la délinéarisation, les techniques de classification classent les segments découpés en un nombre fini de genres [BC08]. Ces genres sont généralement les informations, les dessins-animés, le sport, les films et la publicité. Ce sont des classes très spécifiques. L'extension de ces techniques de classification à tous les programmes et à tous les inter-programmes est compliquée car les programmes et les inter-programmes sont trop hétérogènes [RMX+02].

Le flux audio peut aussi être découpé. Les segments audio sont des portions de silence, de parole et de musique [SC00, ZJK01, LZJ02, WLH00]. Les segments audio permettent

9. Définition issue du trésor de la langue française informatisé (`http://atilf.atilf.fr`).

notamment de structurer les journaux télévisés qui se divisent en segments de parole du présentateur et en segments de reportage [PFGM04, MN03]. De la même manière que pour les segments vidéo, les segments audio ne conduisent pas directement à une délinéarisation du flux TV.

### 1.2.2 Approches consacrées à la détection de la publicité

La détection de la publicité est la cible de nombreux intérêts industriels [SM04] : pour le contrôle des diffusions et le respect des contrats publicitaires ou pour remplacer une publicité par une autre publicité plus ciblée. La détection de la publicité ne fournit qu'une délinéarisation partielle de la télévision. Néanmoins, les techniques utilisées peuvent s'étendre, comme nous le verrons dans la section 1.2.3.2, aux inter-programmes et ainsi fournir une délinéarisation plus complète.

Pour détecter les publicités, nous distinguons trois grandes techniques : l'utilisation de caractéristiques intrinsèques (les couleurs, le contexte), la reconnaissance dans une collection de publicités connues, ou encore la détection grâce aux répétitions.

#### 1.2.2.1 Utilisation de caractéristiques intrinsèques

Les principales caractéristiques utilisées pour les publicités sont les images monochromes et les silences qui séparent les publicités [SMOM02], la fréquence des changements de scènes ou de plans [LKE97] ou l'absence de logo de la chaîne [AFAT04]. L'avantage de ces méthodes est leur efficacité à traiter en temps réel les flux télévisuels. Leur limite dépend du choix de leurs hypothèses. Les images monochromes sont rarement incluses au début et à la fin des coupures publicitaires (ce qui exclut la détection des premières et dernières publicités), elles peuvent aussi ne pas exister bien que cela soit une obligation légale dans certains pays [10]. Les fréquences de changement de plans entre les publicités ressemblent de manière forte aux fréquences de changement de plans de certains films ou séries d'actions. Par ailleurs, certaines chaînes n'affichent aucun logo.

D'autres approches s'orientent vers la classification de plans ou de scènes [HLZ05]. Elles peuvent utiliser pour cela des caractéristiques multimodales [DWZ+06]. Ces méthodes se heurtent à l'hétérogénéité des contenus télévisuels. Un simple plan pris au milieu d'une publicité peut être très similaire d'un plan issu d'un magazine, d'un documentaire, d'un reportage de journal ou d'un film.

#### 1.2.2.2 Reconnaissance

Lienhart *et al.* [LKE97] proposent en plus de l'utilisation des caractéristiques intrinsèques de construire un ensemble de référence contenant les publicités connues. Lors de la rediffusion des publicités celles-ci, peuvent être alors facilement reconnues. Lienhart *et al.* reconnaissent les publicités par une recherche exhaustive du dernier morceau de flux diffusé dans l'ensemble de référence.

Pour optimiser la recherche dans l'ensemble de référence, des techniques d'indexation multidimensionnelle peuvent être appliquées [SG07, LZZJ07]. De manière plus générale, tout système de détection de similarité ou de copies est applicable à ce problème [OKH02,

10. Dans les pays européens par exemple.

HK02, CS04]. Joly [Jol05] propose un état de l'art sur les techniques efficaces de détection de copies vidéo dans une grande collection de références audiovisuelles.

L'inconvénient de ces méthodes est qu'elles réclament un ensemble de référence créé manuellement. De plus, cet ensemble de référence doit être mis à jour régulièrement afin de pouvoir détecter les nouvelles publicités. Lienhart *et al.* proposent d'ajouter dans leur collection de publicités tous les segments courts encadrés par deux publicités. Li *et al.* [LZZJ07] proposent une mise à jour plus précise. Des publicités candidates sont découpées suivant les images monochromes, les logos et la fréquence des plans. Puis, ces publicités candidates sont comparées à toutes les autres publicités candidates d'un historique. Cette dernière technique amène la détection des publicités grâce à leurs répétitions.

#### 1.2.2.3 Détection à partir des répétitions

La détection des publicités à partir des répétitions consiste à identifier toutes les occurrences des répétitions du flux. Ensuite, les publicités répétées sont sélectionnées parmi toutes ces occurrences de répétitions. Ces techniques sont efficaces car les publicités sont majoritairement rediffusées.

Les occurrences des répétitions détectées par Duygulu *et al.* [DCH04] sont construites à partir d'images clés. Chacune des images clés est associée à un groupe de $k < 50$ autres images clés pratiquement identiques en réalisant un parcours séquentiel des autres images clés. À partir de ces groupes d'images clés identiques, des suites d'images clés sont construites. Ces suites contiennent des images clés ordonnées chronologiquement qui appartiennent respectivement aux mêmes groupes. Ces suites d'images forment les occurrences des répétitions. Pour sélectionner enfin les publicités à travers ces occurrences des répétitions, Dyugulu *et al.* utilisent un *Support Vector Machine* (SVM) et des caractéristiques basés sur la transformée de Fourier du signal audio et un histogramme de couleur.

Gauch *et al.* [GS06] utilisent une méthode de hachage perceptuel pour optimiser la recherche des occurrences des répétitions. Toutes les images du flux sont hachées, c'est à dire transformée dans une signature binaire. Pour chaque image d'un plan la méthode retourne alors un ensemble d'images possédant la même signature de hachage. Ces images appartiennent aussi à des plans. Pour chaque plan considéré, un ensemble de plans candidats est donc retourné de cette manière. La similarité de ces plans candidats avec le plan considéré est ensuite vérifiée plus précisément par une technique d'alignement. Les suites de plans répétés dans le même ordre forment enfin les occurrences des répétitions. Pour sélectionner les publicités depuis l'ensemble des occurrences des répétitions, Gauch *et al.* classent initialement un certain nombre d'occurrences manuellement. Ensuite, ils calculent les $p$ plus proches voisins de chaque occurrence des répétitions suivant les images monochromes environnantes, la fréquence des plans et des moments de couleurs. Si la moitié de ses $p$ plus proches voisins sont des publicités alors l'occurrence de répétition est une publicité.

### 1.2.3 Approches dédiées à la délinéarisation des flux

Comme nous l'avons présenté en introduction, un flux TV est constitué de programmes et d'inter-programmes. Les techniques de délinéarisation se séparent en deux groupes. Le premier groupe se concentre sur l'extraction des programmes. Le second groupe se focalise à l'inverse sur les inter-programmes.

Les techniques spécifiques aux programmes ne permettent pas d'extraire individuellement les inter-programmes. Pourtant ceux-ci s'avèrent importants pour l'archivage ou la pige. À l'opposé, une fois les inter-programmes proprement extraits, le reste du flux forme naturellement des segments de programmes ou de parties de programmes. Les techniques spécifiques aux inter-programmes sont donc plus adaptées pour la délinéarisation du flux. De plus, les caractéristiques des inter-programmes sont moins hétérogènes que les programmes. Par exemple, leurs durées sont plus semblables et ils partagent plus de propriétés communes dont leur rediffusion. Ils paraissent donc plus faciles à détecter.

Nous présentons d'abord les approches qui se focalisent sur l'extraction des programmes puis les approches d'extraction des inter-programmes qui, par extension, délinéarisent complètement les flux.

#### 1.2.3.1 Extraction des programmes uniquement

Liang *et al.* [LLXT05] choisissent de tirer profit de la répétition des génériques de début et de fin des programmes quotidiens. Pour cela, ils comparent plusieurs journées et sélectionnent les plans répétés d'une journée sur l'autre dans des intervalles de temps limités. Cette approche reste limitée aux programmes quotidiens ou hebdomadaires qui possèdent des génériques stables.

Une autre technique se base sur le principe des génériques. Wang *et al.* [WDL+08] définissent un détecteur d'images particulières qu'ils nomment les *Program Oriented Informative iMages* (POIM). Ces POIM représentent des images contenant des logos sur des fonds monochromes avec du texte en gros caractère. Typiquement, ces POIM apparaissent dans des génériques de programmes et dans des fins de publicités. Wang *et al.* combinent alors ces POIM avec des changements déterminés dans le signal audio et le contenu textuel pour déterminer plus précisément les POIM pertinents. Ils arrivent ainsi à découper le flux en segments de programmes et en segments d'inter-programmes. Néanmoins, ces POIM ne peuvent délimiter que les publicités se terminant par des logos et certains génériques de programmes. Il reste donc tous les autres inter-programmes à délimiter. Les auteurs s'arrêtent, de plus, au découpage en parties de programmes et en inter-programmes. Pour manipuler ces segments, une solution de représentation automatique est proposée, basée notamment sur les POIM. Les programmes entiers ne sont donc pas extraits complètement.

Suivant la solution de El-Khoury *et al.* [EKSJ08], chaque programme possède des propriétés homogènes. Cette homogénéité d'un programme permet alors de l'extraire par détection de rupture de continuité. Pour modéliser cette homogénéité, El-Khoury *et al.* appliquent une technique de segmentation de locuteurs basée sur le GLR (*Generalized Likelihood Ratio*) et sur le BIC (*Bayesian Information Criterion*). La technique détecte le point de changement le plus probable entre deux régions homogènes. Elle est appliquée sur des caractéristiques vidéo, puis audio, et les 2 découpages sont fusionnés. El-Khoury obtient ainsi une segmentation en parties de programmes. Les auteurs mentionnent dans [EKSJ08] que les inter-programmes sont trop courts pour être détectés par cette méthode.

Haidar [Hai05] propose une solution basée sur les matrices de similarités. Haidar propose d'extraire certains programmes en comparant deux journées consécutives dans une matrice de similarité. Pour cela, une distance de similarité basée sur le style est développée. Cette similarité gère aussi bien la similarité totale (identification des copies), que la similarité partielle (quantitative, qualitative ou temporelle). La matrice de similarité

présentée permet de voir ressortir les programmes quotidiens d'une journée sur l'autre. Cependant l'extraction de ces programmes n'est pas effectuée.

Enfin, nous rappelons la méthode de Poli [Pol07] d'enrichissement des métadonnées déjà vue dans la section 1.1.3. Cette méthode permet de prédire et de corriger les guides de programmes fournis dans les métadonnées. Cela fournit un encadrement à la minute près des débuts et fins des programmes. Pour détecter les débuts et fins précis, Poli n'a plus qu'à identifier quelques transitions simples (images monochromes, silences et logos). Les débuts et fins approximatifs sont donc recalés et Poli réalise une extraction des programmes parfois à la seconde près.

#### 1.2.3.2 Extraction des inter-programmes

La frontière entre les notions d'inter-programme et de publicité est ambigüe et souvent mal définie dans certains travaux [DJN+02, CBF06, ZZZY08]. Ces travaux traitent de la détection de publicité (*advertisement*) en incluant les auto-promotions (*self-advertisements*). Les « publicités » considérées regroupent donc les publicités, les bandes annonces et parfois les parrainages : soit la majorité des inter-programmes.

Dimitrova *et al.* [DJN+02] utilisent des caractéristiques intrinsèques pour détecter des publicités et des bandes annonces. Les caractéristiques utilisées sont le changement de ratio des images (4 : 3 ou 16 : 9), les images monochromes et les transitions de plans rapprochés. Ces caractéristiques restent trop spécifiques. Elles s'appliquent difficilement à tous les inter-programmes.

La propriété principale utilisée pour la détection des inter-programmes est leur caractéristique à être en grande partie rediffusés. Comme pour les publicités, nous retrouvons donc des techniques de détections des inter-programmes par reconnaissance dans un ensemble de référence ou par détection des répétitions. Ces techniques sont présentées dans les 2 sous-sections suivantes.

##### *Reconnaissance*

L'une des méthodes pionnières de délinéarisation à partir des inter-programmes est la méthode de Naturel [Nat07].

Pour reconnaître un inter-programme, Naturel [Nat07] construit manuellement un Ensemble de Vidéos de Référence (EVR). Cet EVR contient des plans de vidéos classés manuellement en programmes ou inter-programmes. Grâce à une technique de hachage perceptuel, la méthode de Naturel reconnaît par la suite les plans rediffusés. Ces plans sont alors classés par leur reconnaissance dans l'EVR. Toute solution de hachage perceptuel préalablement envisagée pour les publicités convient également pour cette solution [OKH02, HK02, CS04]. Le hachage permet une reconnaissance en temps réel des rediffusions.

L'inconvénient de ces techniques est la construction et la mise à jour de l'ensemble de référence. Naturel [Nat07] montre que ses résultats de reconnaissance chutent lorsque son EVR devient trop « vieux ». Il propose ainsi une solution pour mettre à jour automatiquement cet EVR à partir de règles simples. Un exemple de règle est que les groupes de plans encadrés par des plans d'inter-programmes sont insérés dans l'EVR comme nouveaux plans d'inter-programmes. Ces règles réduisent finalement la chute des résultats.

Les plans des bandes annonces posent cependant problème. Ceux-ci sont dans un premier temps étiquetée comme « inter-programme » puis certains plans du programme qu'elles annoncent sont par reconnaissance étiquetés comme des plans d'inter-programmes. Il en découle une sursegmentation et une chute des résultats. Les nouveaux plans correspondant à des bandes annonces doivent donc être détectés. Pour cela, Naturel utilise un différé de plusieurs jours à une semaine. Ce différé permet d'attendre la diffusion du programme qui permet de décider si une suite de plans est une bande annonce ou non. Au final, la mise à jour de l'EVR est une tâche compliquée et contraignante. Pour être efficace, l'approche nécessite une technique rigoureuse de détection des inter-programmes inconnus.

Les inter-programmes reconnus conduisent ensuite à la délimitation des parties de programmes. Pour étiqueter ces parties de programmes, Naturel [Nat07] projette les métadonnées sur le flux. L'auteur aligne les segments de programmes découpés avec les segments de programmes issus des horaires et des durées donnés dans les guides des programmes. L'alignement est effectué par une méthode de type *Dynamic Time Warping*. Autrement dit, une distance d'édition est calculée pour déterminer le nombre d'opérations minimales d'insertion, de suppression et de substitution pour transformer une suite de segments en une autre. Lorsqu'un segment découpé est aligné avec un segment du guide des programmes, le segment découpé en récupère un titre. Dans cette solution, les métadonnées ne sont pas modifiées. Une seule des parties d'un programme séparé en plusieurs parties est alignée avec le segment adéquat dans le guide des programmes. Afin de pallier ce problème, Naturel divise tout segment du guide des programmes en 2 ou 3 morceaux suivant leur longueur. Cette astuce n'a été testée que sur la chaîne publique France 2 dans laquelle très peu de programmes sont séparés par des coupures publicitaires. De manière plus générale, l'EPG requiert d'être modifié dynamiquement en fonction des segments délimités. Au final, la solution s'avère efficace pour étiqueter les programmes longs (de plus de 5 minutes) diffusés dans la journée. Par contre, les programmes courts restent souvent mal étiquetés.

#### *Détection à partir des répétitions*

La détection des inter-programmes par leur rediffusion est une méthode complètement automatique. Elle ne requiert pas de traitement manuel et détecte des inter-programmes inconnus qui peuvent ensuite remplir des bases de références. Pour la détection des répétitions seules, de nombreuses techniques ont déjà été proposées [Her06, FC03, PGGM04, YMWL08, DL09, CN05, WLY05, YTX07] dont celles pour la détection des publicités [DCH04, GS06]. Nous verrons plus en détail ces techniques dans le chapitre 2. Cependant, toutes les répétitions ne sont pas des inter-programmes, il faut donc classer ces répétitions.

Covell *et al.* [CBF06] étendent à l'audio la technique présentée par Gauch *et al.* [GS06] pour la détection des publicités basée sur la vidéo. Le flux est découpé en morceaux de 5 secondes qui sont insérés dans un tableau par hachage perceptuel sur l'audio. Pour chaque morceau inséré, tous les morceaux possédant la même signature binaire hachée sont récupérés. Ces morceaux candidats sont comparés par des descripteurs vidéo afin de valider ou non la répétition du morceau. Le morceau répété est ensuite étendu le plus possible pour obtenir une occurrence d'une répétition. Ensuite, les inter-programmes sont sélectionnés parmi les occurrences des répétitions à partir d'un seuil minimal et d'un seuil maximal sur la durée de l'occurrence.

Zeng *et al.* [ZZZY08] découpent le flux TV suivant les silences audio. Puis, chaque

début de segment est comparé de manière exhaustive aux autres grâce à une signature binaire. Les répétitions sont ainsi détectées. Ensuite, les occurrences des répétitions dont la durée n'est ni trop longue et ni trop courte forment les inter-programmes. Les trous entre les occurrences des répétitions forment les parties de programmes si leur durée dépasse un certain seuil. Pour reconstruire les programmes en entier, Zeng *et al.* utilisent une règle simple. Selon eux, les coupures d'inter-programmes séparant des parties d'un même programme sont plus courtes que celles séparant deux programmes différents. Cela n'est pas vrai pour tous les corpus.

La technique de Wang *et al.* [WLQ+09] détecte et reconnait aussi les inter-programmes à partir de descripteurs audio. Elle découpe l'enveloppe énergétique du signal audio en segments à partir des sommets et des creux de l'énergie. Les segments sont triés suivant leur longueur puis seuls les débuts des segments de longueurs similaires sont comparés de manière exhaustive pour détecter les répétitions. Pour la classification des occurrences des répétitions en inter-programmes, Wang *et al.* effectuent une annotation manuelle qu'ils justifient par le faible nombre d'éléments à annoter.

Si les travaux présentés visent explicitement la détection d'inter-programmes, seul Zeng *et al.* abordent une solution de délinéarisation à partir de ceux-ci. Un dernier point important est que les travaux ne traitent généralement que des inter-programmes se répétant et non de la détection des quelques inter-programmes qui ne se répètent pas.

### 1.2.4 Synthèse

Nous avons présenté les principales méthodes autour de la délinéarisation de flux TV. Le découpage classique en plans ou en scènes et la classification ne permettent pas d'extraire les programmes et les inter-programmes. Les segments découpés sont généralement trop courts et doivent être fusionnés. De plus, les caractéristiques audiovisuelles seules ne sont pas suffisantes pour décider si ces segments appartiennent à un parrainage, une publicité, une bande annonce, un reportage, un magazine ou à un film. Le contenu de la télévision est dans ce sens trop hétérogène. Des méthodes particulières à la délinéarisation sont nécessaires.

Pour une délinéarisation complète, c'est à dire une extraction des inter-programmes et de tous les programmes. L'approche basée sur la détection des inter-programmes est la plus intéressante. En effet, chaque inter-programme est délimité et les parties de programmes sont déduites naturellement dans le reste du flux. De plus, les inter-programmes sont globalement plus homogènes que les programmes. Une propriété forte des inter-programmes largement utilisée est le fait que ceux-ci sont rediffusés.

Pour utiliser la rediffusion des inter-programmes, deux méthodes existent : la reconnaissance de répétitions connues et la détection des répétitions inconnues. La reconnaissance implique la gestion d'un ensemble de référence créé manuellement. Par conséquent, dans l'optique d'une délinéarisation automatique, la détection des répétitions inconnues s'impose.

De nombreux travaux proposent des solutions de détection automatique des répétitions. Cependant, l'utilisation des occurrences de ces répétitions pour la délinéarisation a concentré beaucoup moins d'efforts. Bien que les inter-programmes soient rediffusés majoritairement, certains inter-programmes ne sont pas rediffusés. Certaines répétitions comme les reportages ou les flash-backs ne sont pas des inter-programmes. Les simples seuils em-

ployés sur la durée des occurrences des répétitions ne suffisent pas et une méthode efficace de classification est nécessaire.

Dans cette thèse, nous allons proposer une solution pour délinéariser entièrement un flux TV de manière automatique à partir des répétitions. Ce dernier point résume toutes nos contributions.

# Chapitre 2

# Découpage du flux en segments à partir des répétitions

**Sommaire**



Le premier niveau de délinéarisation est le niveau de découpage du flux audiovisuel. Il permet de délimiter des segments dans la portion de flux analysée de notre approche globale de délinéarisation. Cela est illustré dans la figure 2.1. L'objectif de ce niveau est de construire des segments pertinents pour les niveaux de délinéarisation suivants.

La pertinence des segments se situe dans leur capacité à délimiter les bornes des programmes et des inter-programmes. Nous choisissons de découper le flux télévisuel en segments à partir des répétitions qu'il contient. Ce découpage se base sur le constat que les

inter-programmes du flux sont majoritairement rediffusés. Ils forment donc des occurrences de répétitions dans le flux. À partir des occurrences des répétitions du flux, nous délimitons la majeure partie des inter-programmes. Les morceaux de flux entre ces inter-programmes délimitent le reste des inter-programmes et des parties de programmes ou des programmes entiers.

Les inter-programmes sont répétés pour plus de visibilité afin d'augmenter leur audience et leur impact. Cela est évident pour les publicités, les parrainages et les campagnes d'intérêt général. Cela est vrai aussi pour les bandes annonces qui communiquent les futurs évènements importants des chaînes. L'habillage (les logos et les jingles) est répété aussi pour conserver une certaine régularité. Il sert de repère aux téléspectateurs.

Dans ce chapitre, nous présentons le premier niveau de délinéarisation de notre solution de délinéarisation. Ce niveau de délinéarisation est constitué de deux méthodes : une méthode de détection des répétitions et une méthode de découpage proprement dit à partir des occurrences de ces répétitions. Nous rappelons que notre solution est soumise aux quatre contraintes clés suivantes :

1) l'efficacité : pour délinéariser tous les inter-programmes et les programmes ;
2) la généricité : pour délinéariser tous types de flux ;
3) l'automaticité : pour délinéariser automatiquement les flux ;
4) la continuité : pour délinéariser en continu des flux ininterrompus.

Nous commençons par préciser la notion de répétition et nous présentons les principales méthodes existantes pour la détection des répétitions. Puis, nous détaillons notre méthode de détection des répétitions du flux. Nous expliquons ensuite notre méthode de découpage proprement dit du flux en segments à partir des occurrences des répétitions. Enfin, les deux méthodes sont rigoureusement évaluées sur des flux TV réels. Nous cherchons alors à vérifier nos quatre contraintes clés pour ce niveau de délinéarisation.

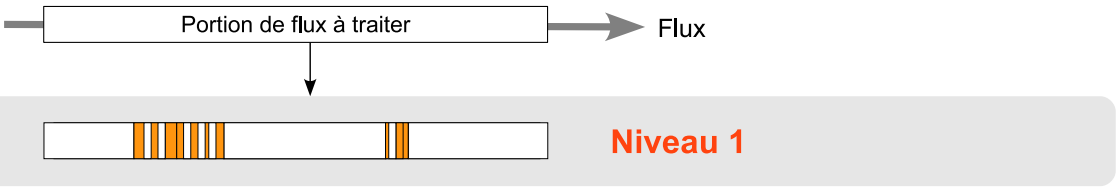



FIGURE 2.1 – Niveau 1 : découpage en segments à partir de la détection des répétitions.

À la fin de ce chapitre, nous proposons, en plus, une méthode originale de détection de bandes annonces à partir des répétitions détectées. Cette méthode permet de détecter certains inter-programmes parmi les répétitions détectées. Elle propose ainsi une transition vers notre méthode de classification des segments issus des répétitions, présentée dans le chapitre 3 suivant.

## 2.1 Définition des répétitions

Le domaine de la détection de « répétitions » est très vaste. Certains biologistes étudient les reproductions de chaînes de nucléotides dans la macromolécule d'ADN [KCO+01], des

musiciens recherchent des thèmes musicaux sous forme de suites répétées de notes [LHC99] et d'autres encore [MAEA05] analysent les duplications contenus dans des flux de données binaires.

Dans le domaine de l'audiovisuel, une répétition peut être une copie d'une vidéo [Jol05, Pou09]. Elle peut alors représenter seulement une réplique entière quasiment à l'identique d'une vidéo ou bien, de manière plus générale, un extrait d'une vidéo ayant subi diverses transformations visuelles (rotation, décalage, incrustation, etc.). Nous devons donc, dans un premier temps, bien définir notre problème particulier de détection des répétitions dans un flux audiovisuel pour la détection des inter-programmes rediffusés.

Nous recherchons en réalité des rediffusions d'inter-programmes pour pouvoir ensuite découper les flux TV. Un inter-programme constitue un contenu. Lorsque ce contenu est diffusé plusieurs fois et est donc répété, chaque diffusion de ce contenu définit une occurrence d'une répétition. La répétition englobe ici toutes les apparitions de ce contenu. Cela est illustré par la figure 2.2.

Les rediffusions d'inter-programmes subissent les mêmes principales transformations dues à la transmission et à la réception du flux. Ce sont donc des répliques quasiment à l'identique. Nous ne cherchons pas des occurrences d'une répétition ayant subi des transformations différentes. La détection de copies robustes à de nombreuses transformations définit un cadre beaucoup plus général. Elle révèle, de plus, une structure et un découpage trop fins du flux qui incluent la structure **intra**-programme [Pou09].

Comme les inter-programmes du flux ne sont pas connus *a priori*, les rediffusions sont des occurrences de répétitions « inconnues » :

- sans caractéristique audiovisuelle spécifique ;
- répétées un nombre indéterminé de fois ;
- répétées quasiment à l'identique ;
- non connues dans un ensemble de référence ;
- de n'importe quelle durée, de quelques secondes à plusieurs minutes ;
- fondues dans un flux continu.

Grâce à ces répétitions inconnues, nous découpons les flux TV en segments. Néanmoins, il est important de noter que toutes les répétitions détectées ne sont pas des inter-programmes. Certains programmes sont rediffusés à l'identique. Les répétitions peuvent également contenir des génériques d'émissions ou de séries qui sont des rediffusions quasi identiques d'un même contenu. Il peut y avoir aussi certains morceaux de reportages d'actualité qui sont rediffusés dans la même journée.

## 2.2 Méthodes existantes de détection des répétitions

De nombreuses solutions existent pour la détection des répétitions telles que nous les avons définies [Her06, FC03, PGGM04, YMWL08, DL09, CN05, YTX07]. Nous avons vu de plus dans le chapitre 1 des solutions orientées pour la détection des répétitions de publicité [DCH04, GS06] et celles pour la détection des répétitions d'inter-programmes précisément [CBF06, ZZZY08, WLQ+09].

Une partie de ces techniques se limite à la détection des répétitions dans un document audiovisuel fini ou dans une portion finie de flux. La méthode la plus simple pour cela

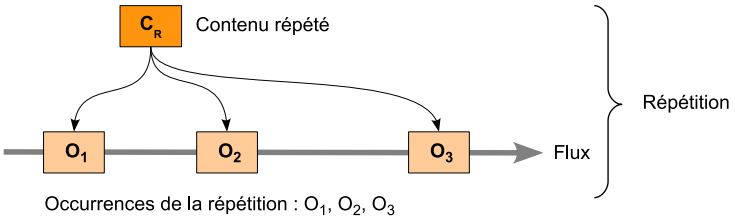



FIGURE 2.2 – Illustration d'une répétition et de ses occurrences.

consiste à parcourir de manière exhaustive le document pour y détecter des images quasi identiques ou des segments quasi identiques [DCH04, WLQ+09]. Une variante de cette approche compare le document avec lui même par l'intermédiaire d'une matrice de similarité [FC03]. Pour éviter de parcourir exhaustivement tout le document, une méthode de recherche des plus proches voisins peut être appliquée [YMWL08, YTX07]. La détection des répétitions est ensuite restreinte à la détection des répétitions parmi les plus proches voisins. L'ensemble de ces approches rencontre, cependant, des difficultés face à de larges documents audiovisuels.

Afin de pouvoir traiter un flux potentiellement infini en continu, une solution est de limiter la recherche exhaustive des répétitions à un historique fini glissant en temps réel sur le flux [Her06]. Pour optimiser la recherche exhaustive, Herley [Her06] découpe le flux en morceaux entrecroisés de $L$ secondes. Il compare alors les morceaux en mesurant la corrélation des énergies des signaux audio. Il compare en fait chaque nouveau morceau diffusé avec tous les morceaux de l'historique glissant. La recherche s'arrête lorsqu'une corrélation est identifiée. Cette corrélation est alors étendue au maximum afin de délimiter les bornes des occurrences de la répétition détectée. La taille de l'historique limite les performances de la méthode. Elle doit en effet être suffisamment petite pour pouvoir effectuer la recherche de corrélation d'un morceau avant la diffusion du morceau suivant.

Au lieu de considérer un flux potentiellement infini, celui-ci peut être vu comme une portion très large de flux ou encore comme une très grande base de données audiovisuelles. La technique majoritairement employée pour détecter efficacement les répétitions dans une telle grande base de données audiovisuelles est le *hachage perceptuel* [PGGM04, GS06, CBF06, ZZZY08, DL09, CN05]. Pour cela, des images, des plans ou des morceaux de $n$ secondes sont transformés en des signatures binaires par une fonction de hachage. Ces signatures sont telles que des éléments similaires selon la perception humaine obtiennent la même signature. La détection des répétitions est réduite à la détection de signatures binaires répétées qui sont plus faciles à indexer et à retrouver. Au final, les signatures consécutives répétées dans le même ordre définissent les occurrences des répétitions du flux. Malheureusement, la fonction de hachage n'a pas de réciproque et une même signature peut également représenter des éléments fortement différents. Il est alors nécessaire de vérifier les répétitions obtenues par d'autres mesures de similarité. La difficulté de ces dernières méthodes est, en plus, de définir *a priori* une fonction de hachage qui reflète la similarité entre les images, les plans ou des morceaux de $n$ secondes.

Notre position par rapport à ces travaux est de proposer un système original de détection des répétitions capable de traiter un flux TV potentiellement infini en continu. Nous utilisons pour cela une technique de *clustering* incrémental de données multidimensionnelles. Notre méthode utilise un historique limité qu'elle traite et met à jour de manière périodique ou à la demande.

## 2.3 Notre méthode de détection des répétitions

Nous souhaitons détecter des répétitions telles que nous les avons définies : répétées un nombre indéterminé de fois, répétées quasiment à l'identique, de n'importe quelle durée, de quelques secondes à plusieurs minutes, et fondues dans un flux.

Nous présentons notre méthode en deux étapes. Nous exposons d'abord une méthode de détection des répétitions sur une portion finie du flux. Puis, nous présentons les modification de cette méthode pour gérer des flux en continu.

### 2.3.1 Fonctionnement général de la méthode sur une portion de flux

Le fonctionnement générale de notre méthode est illustré dans la figure 2.3. La portion de flux analysée est d'abord décrite par un ensemble d'unités. Ces unités représentent des composants à la base de tous les éléments diffusés dans le flux. Elles sont telles que quel que soit l'emplacement d'une occurrence d'une répétition dans le flux celle ci reste composée des mêmes unités. La répétition est donc révélée par les unités. La détection des répétitions débute donc par la détection des unités répétées. Ensuite, il suffit d'analyser pour chaque paire d'unités voisines chronologiquement si les unités sont répétées également dans un même voisinage. Les répétitions sont donc reconstruites par les réplications de suites d'unités.

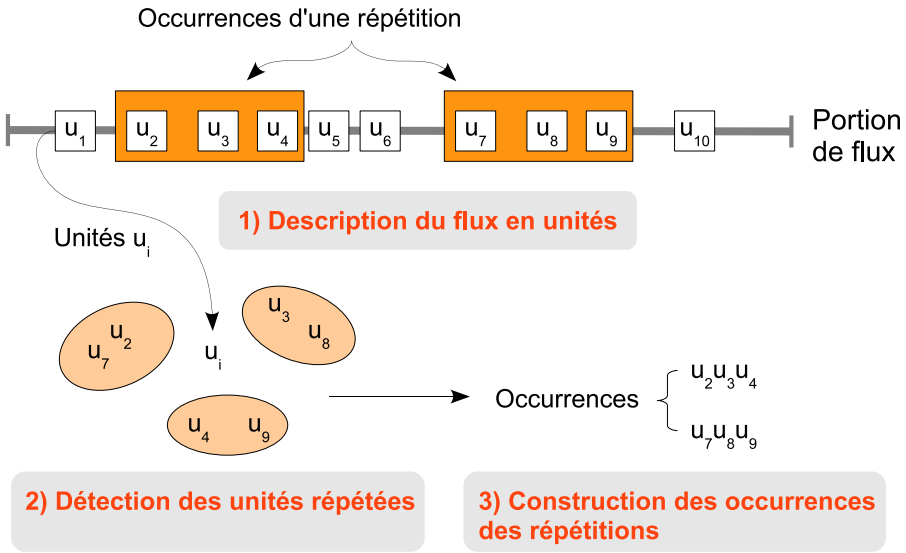



FIGURE 2.3 – Stratégie générale de la détection des répétitions.

La détection des répétitions dans notre contexte se divise donc en trois étapes : le choix d'une bonne description en unités, la détection des unités répétées quasi à l'identique et la construction des occurrences des répétitions à partir des unités répétées détectées.

Notre méthode est issue de la technique développée par Berrani et Lechat [BL06]. Cette technique permet de détecter des répétitions sur une portion finie d'un flux. Nous avons modifié et amélioré la technique afin de pouvoir traiter en continu les flux TV. Par ailleurs, nous avons réimplémenté la technique et nous avons introduit des optimisations aux différentes étapes de la détection des répétitions.

### 2.3.2 Description en unités « images clés »

Nous avons choisi de détecter les répétitions en utilisant les informations visuelles seulement. Ce choix est validé par les résultats présentés dans la section 2.5. L'approche pourrait cependant s'appliquer au signal audio et il pourrait être intéressant dans une suite à cette thèse de comparer les occurrences de répétitions audio, vidéo et audiovisuelles pour la délinéarisation.

Nous décrivons la portion de flux TV analysée à partir de l'unité « image clé ». Ce sont donc des images clés répétées que nous recherchons. Les répétitions vidéo que nous construisons par la suite sont alors basées sur ces images clés.

Afin de délimiter les bornes précises des occurrences des répétitions, nous avons besoin de connaître aussi toutes les images qui entourent nos images clés. Nous décrivons donc le flux vidéo à la fois par les images qui le composent et par un ensemble d'images clés. La figure 2.4 illustre la description du flux.

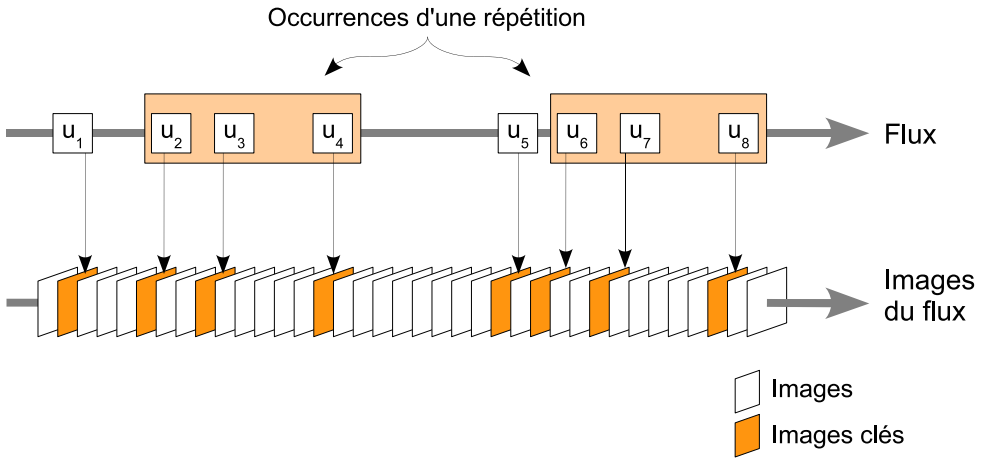



FIGURE 2.4 – Description du flux par la totalité de ses images et par des images clés.

Chaque image contient des informations visuelles importantes que nous réduisons à un vecteur de caractéristiques appelé descripteur. Suivant notre approche, nous gérons deux types de descripteurs : les descripteurs d'images $D_I$ et les descripteurs d'images clés $D_C$. Dans notre méthode, le flux est décrit au fur et à mesure. Les différents descripteurs utilisés sont calculés et stockés en temps réel sur le disque.

#### 2.3.2.1 Calcul des images clés

Notre technique de détection des images clés segmente, dans un premier temps, la vidéo en plans. Notre méthode de segmentation en plans analyse les discontinuités de l'évolution de la similarité colorimétrique entre les images consécutives [Han02]. Nous n'avons pas évalué rigoureusement les performances de notre méthode de segmentation en plan. Notre système repose, en effet, sur des images clés et non sur les frontières des plans. La seule contrainte imposée sur l'algorithme de segmentation en plan est que celui-ci segmente chaque occurrence des répétitions suivant les mêmes plans. À partir de ces plans, des images principales qui les résument sont extraites en tant qu'images clés. Nous avons appliqué pour cela le test statistique de Page Hinkley [Pag54, Hin71, Bas88]. Ce test étudie les points de rupture dans l'évolution des histogrammes de couleurs des images d'un plan. De cette façon, deux plans répétés fournissent des images clés également répétées. Nous donnons dans le tableau 2.1 les nombres d'images clés moyens par seconde calculés sur les chaînes France 2 et TF1 pour une semaine de flux vidéo. Il y a légèrement plus d'images clés calculées sur TF1 que sur France 2. Cela est dû aux différences entre les deux chaînes montrées en annexe A.

| | Nombre d'images clés moyen par seconde |
|---|---|
| France 2 | 0,4073 |
| TF1 | 0,4671 |

Tableau 2.1 – Nombres d'images clés moyens par seconde calculés sur une semaine de flux vidéo.

#### 2.3.2.2 Détection des images monochromes et des silences

Un flux télévisuel contient en particulier de nombreuses micro-coupures. Ces micro-coupures sont parfois des artefacts dûs au processus de production et d'assemblage des contenus TV. Mais elles peuvent être parfois ajoutées volontairement afin de bien marquer des séparations, par exemple entre des publicités sur certaines chaînes TV. Les micro-coupures se traduisent généralement par des silences dans le flux audio accompagnés d'images monochromes.

Notre méthode est théoriquement insensible à ces micro-coupures qui apparaissent naturellement dans le découpage en plans. Néanmoins, les images monochromes issues de ces micro-coupures forment des groupes d'images clés identiques nombreux et inutiles pour la détection des répétitions. Nous avons donc supprimé ces images monochromes lors des micro-coupures.

Pour détecter les micro-coupures, nous avons employé la méthode de détection des séparations proposée par Naturel [Nat07]. Les silences sont d'abord identifiés puis les images monochromes autour de ces silences sont analysées. Nous choisissons empiriquement un premier seuil sur l'énergie des trames du signal audio en dessous duquel les silences sont identifiés. Nous établissons un second seuil sur l'entropie des histogrammes de luminance des images pour séparer les images monochromes du reste des images.

#### 2.3.2.3 Descripteurs des images clés : les $D_C$

Les images clés servent à la construction des occurrences des répétitions. Leurs descripteurs doivent permettre de détecter des images clés quasiment à l'identique. Pour cela, il suffit que les détails de ces images ne soient pas comparés. Nous empruntons nos descripteurs aux techniques de représentations et de compression des images. Une technique classique pour supprimer les détails consiste à ne conserver que les basses fréquences des images dans le domaine fréquentiel. Pour obtenir les fréquences des images nous appliquons la transformée en cosinus discrète (DCT – *Discrete Cosine Transform*). Nous ne recherchons pas de descripteurs robustes aux transformations complexes comme des rotations, de grosses suppressions, des redimensionnements, etc.

Les descripteurs des images clés sont appelés $D_C$. Ils sont obtenus en divisant les images en $3 \times 2$ blocs réguliers. Cette première division en blocs est appliquée afin d'obtenir des descripteurs robustes à de légères incrustations dans les images. Chacun des blocs est ensuite sous-échantilloné à une matrice de taille $8 \times 8$. Une DCT est calculée sur ces blocs et les 5 premiers cœfficients DCT sont conservés (le cœfficient direct et 4 cœfficients alternatifs des basses fréquences). Un $D_C$ est donc un vecteur composé de 30 dimensions. Finalement, pour comparer deux images clés, nous pouvons calculer une distance euclidienne entre leurs descripteurs $D_C$ respectifs.

#### 2.3.2.4 Descripteurs des images : les $D_I$

Les descripteurs des images $D_I$ servent seulement à délimiter les bornes précises des occurrences des répétitions détectées. Ce descripteur peut donc être plus grossier et moins précis que les descripteurs d'images clés $D_C$. Ces descripteurs sont obtenus en divisant les images en $2 \times 2$ blocs réguliers. Chacun des blocs est ensuite sous-échantillonné à une matrice de taille $8 \times 8$. Une DCT est calculée sur ces blocs et les 16 premiers cœfficients DCT sont conservés (le cœfficient direct et 15 cœfficients alternatifs des basses fréquences). Chaque cœfficient est ensuite binarisé. Un $D_I$ est donc composé de 64 bits. L'avantage de coder les $D_I$ par un descripteur binaire est de pouvoir rapidement comparer les descripteurs par une distance de *Hamming* qui compte juste le nombre de bits différents.

### 2.3.3 Détection des images clés répétées

Notre méthode de détection des images clés répétées est issue d'une technique de *clustering*. L'algorithme de cette technique, de manière générale, partitionne un ensemble de vecteurs multidimensionnels en fonction de la proximité de ces vecteurs selon une distance euclidienne. Le résultat de l'algorithme de *clustering* est un ensemble de groupes de vecteurs. Ces groupes sont appelés *clusters*, ils contiennent des vecteurs proches selon la distance utilisée.

Dans notre contexte, nous utilisons les descripteurs $D_C$ des images clés pour construire des *clusters* d'images clés. Ces descripteurs fournissent des vecteurs de 30 dimensions. Nous cherchons, de plus, des *clusters* d'images clés tels que le nombre d'images clés par *cluster* soit déterminé par le nombre d'occurrences d'une répétition à détecter. Ainsi, si une répétition est répétée 3 fois par des occurrences décrites chacune par 5 images clés, nous cherchons alors à découvrir 5 *clusters* de 3 images clés.

Les méthodes de *clustering* permettent de partitionner des vecteurs dans un ensemble d'un nombre fixé de *clusters* ou dans un ensemble d'un nombre indéterminé de *clusters*. Comme nous ne connaissons pas le nombre de répétitions à détecter, nous cherchons un nombre indéterminé de *clusters*. Pour créer ces *clusters*, nous pouvons alors suivre une approche ascendante ou une approche descendante. L'approche ascendante part des vecteurs et fusionne progressivement ces vecteurs en *clusters*. L'approche descendante part à l'inverse de l'espace multidimensionnel entier qu'elle divise progressivement en régions et sous-régions afin d'isoler des groupes de vecteurs. Pour notre problème, nous recherchons des *clusters* très compacts. La compacité des *clusters* garantit en effet le fait que les images clés regroupées soient très proches entre elles et distinctes de celles des autres *clusters*. La compacité permet donc de détecter des images clés répétées quasiment à l'identique. Nous avons donc choisi une approche ascendante. Nous avons plus précisément utilisé l'algorithme de micro-*clustering* présenté dans la thèse de Berrani [Ber04] [1].

L'alogrithme de micro-*clustering* employé vérifie un certain nombre de critères. Premièrement, l'algorithme est capable de considérer de très nombreux vecteurs possédant un très grand nombre de dimensions. Ceci est nécessaire afin de pouvoir traiter les milliers d'images clés des portions de flux TV à analyser. Deuxièmement, l'algorithme forme des *clusters* compacts. Cela assure de détecter des images clés quasi-identiques. Troisièmement, l'algorithme est capable d'identifier le « bruit ». Cela signifie que l'algorithme reconnaît les images clés « isolées » qui ne se regroupent avec aucune autre image clé. Au final, l'algorithme de micro-*clustering* permet de construire un très grand nombre de *clusters* pouvant être de très petite taille. Les *clusters* ainsi construits peuvent donc refléter toutes les répétitions dont les répétitions à 2 ou 3 occurrences seulement. De plus, l'identification du bruit permet de traiter l'importante quantité de flux non rediffusé qui fournit des images clés qui ne se regroupent pas en *clusters*. L'annexe B donne la proportion de flux non rediffusé dans le flux. Celle-ci est environ 80 % pour France 2 et pour TF1 sur une semaine.

L'algorithme de micro-*clustering* utilisé construit les *clusters* par insertions successives des images clés dans l'ensemble des images clés déjà analysées et regroupées en *clusters*. Chaque *clusters* possède un centre et un rayon. Le centre est le centre de gravité des images clés du *cluster*. Le rayon est le rayon d'une hypersphère englobante de l'ensemble des images clés du *cluster*. Dans un premier temps, un rayon maximal $Rmax$ des *clusters* est fixé empiriquement à un rayon initial $Rmax_0$. Nous expliquerons dans la section 2.5 comment le déterminer. Ensuite, un *cluster* de rayon nul (ne contenant qu'une seule image clé) est construit pour chaque image clé à traiter. Le premier *cluster* de rayon nul est sélectionné. Il forme le premier *cluster*. Puis, le deuxième *cluster* de rayon nul est inséré. Si le rayon résultant de la fusion avec le premier *cluster* est inférieur au rayon maximal $Rmax$ alors la fusion est effectuée. Le deuxième *cluster* est alors dit absorbé par le *cluster* existant. Si le rayon résultant est supérieur au rayon maximal $Rmax$, le deuxième *cluster* de rayon nul est inchangé. Les *clusters* de rayon nul restants sont insérés de la même manière. Lorsque tous les *clusters* ont été insérés, le rayon maximal $Rmax$ peut être incrémenté et les *clusters* sont insérés à nouveau au cours de nouvelles passes. Lors des passes successives, les *clusters* de rayon nul utilisés initialement sont remplacés progressivement par les *clusters* déjà existants. Suite à l'augmentation du rayon $Rmax$, de nouvelles fusions sont effectuées

1. Le lecteur intéressé trouvera également un état de l'art détaillé sur les méthodes de *clustering* dans cette même thèse [Ber04].

et le nombre de *clusters* global décroit inévitablement. Le rayon $Rmax$ est finalement incrémenté jusqu'à ce qu'il atteigne une valeur d'arrêt $Rmax_{stop}$ fixée empiriquement, ou bien tant que le nombre de *clusters* est supérieur à un seuil. L'algorithme présenté est détaillé dans l'annexe D.

### 2.3.4 Construction des occurrences des répétitions à partir des images clés répétées

Les images clés répétées sont regroupées dans des *clusters*. Nous cherchons alors à construire des suites réitérées d'images clés répétées. Pour une image clé répétée $p$ fois, nous cherchons à savoir si ses images clés voisines précédentes ou suivantes ne sont pas aussi répétées $p$ fois. Le cas échéant, nous cherchons à vérifier que ces images répétées voisines restent voisines pour les $p$ fois. Cela est illustré dans la figure 2.5 à travers deux *clusters* $c_i$ et $c_j$ de 3 images clés chacun localisées aux instants respectifs $ti_1, ti_2, ti_3$ et $tj_1, tj_2, tj_3$. Les images clés répétées de $c_i$ et de $c_j$ restent voisines.

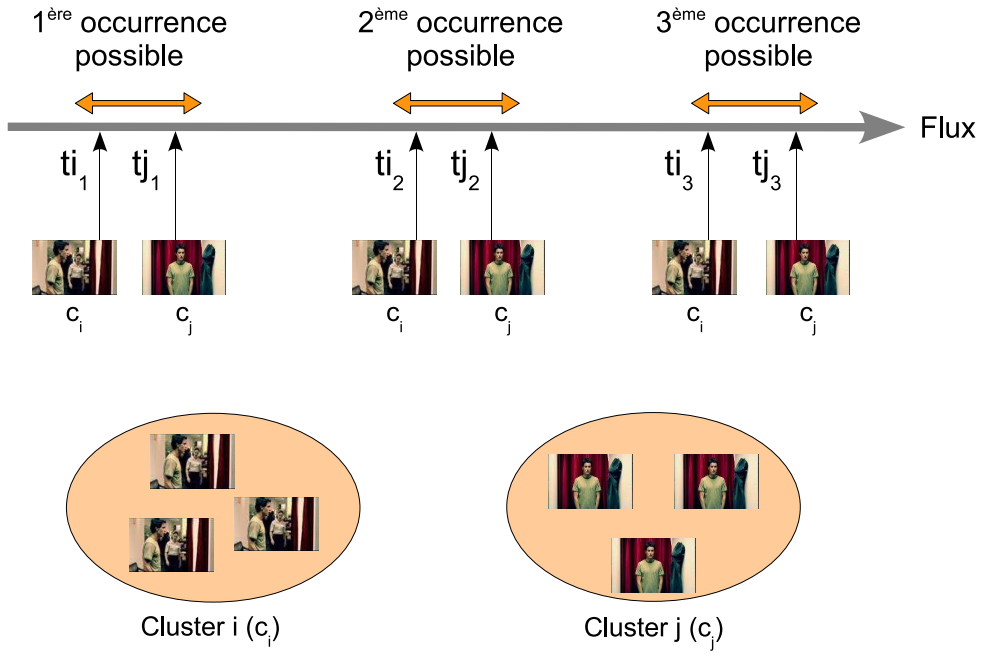



FIGURE 2.5 – Illustration et localisation des images clés de deux *clusters* $c_i$ et $c_j$. Ces deux *clusters* sont aussi dits similaires.

Le micro-*clustering* nous permet de proposer une approche originale pour la construction des occurrences des répétitions. À l'issu du micro-*clustering*, nous obtenons un grand nombre de *clusters* d'images clés compacts. Les images clés qui composent les occurrences d'une répétition sont répétées autant de fois qu'il y a d'occurrences dans la répétition. En d'autres termes, une répétition de $p$ occurrences et dont les occurrences se composent de $n$ images clés sera idéalement répartie sur $n$ *clusters* de $p$ images clés. De plus, les distributions temporelles des $p$ images clés des $n$ *clusters* seront les mêmes d'un *cluster* sur l'autre.

Nous illustrons cela par un exemple. Une publicité $A$ se répète 3 fois dans une journée à 10h, à 13h et à 20h. Les occurrences de la répétition de $A$ durent environ 30 secondes et sont composée de 4 images clés. Alors nous devrions idéalement découvrir 4 *clusters* ($c_1$, $c_2$, $c_3$ et $c_4$) de 3 images clés similaires chacune localisée vers 10h, vers 13h et vers 20h. De plus, toutes les images clés de $c_1$ précéderont les images clés de $c_2$ qui précéderont les images clés de $c_3$, etc.

L'algorithme de construction des occurrences des répétitions se base sur ces principes. Les *clusters* sont donc traités par groupes de *clusters* de même nombre d'images clés. Pour chaque groupe de *clusters* de même nombre d'images clés, les distributions temporelles des images clés des *clusters* sont comparées grâce à une matrice de similarité entre *clusters*. Deux *clusters* avec les mêmes distributions temporelles sont dits similaires, ils contiennent alors des images clés d'occurrences d'une même répétition. Cela est montré dans la figure 2.5.

#### 2.3.4.1 Construction d'une matrice de similarité entre *clusters*

La matrice de similarité est une matrice $\mathcal{M}_{sim}$ dans laquelle les cellules de coordonnées $i$ et $j$ valent 1 si et seulement si les *clusters* $c_i$ et $c_j$ sont similaires et 0 sinon. Le *cluster* $c_i$ contient $p_i$ images clés localisées aux instants $ti_1, \ldots, ti_{p_i}$ et le *cluster* $c_j$ contient $p_j$ images clés localisées aux instants $tj_1, \ldots, tj_{p_j}$.

La similarité entre deux *clusters* reflète le fait que les images clés des deux *clusters* suivent les mêmes distributions temporelles. Elle dépend de l'ordre des instants de ces images clés et des écarts entre ces instants.

Nous définissons en premier la notion d'antériorité. Un *cluster* est antérieur à un autre si ses images clés précèdent respectivement celles de l'autre. Pour les deux *clusters* $c_i$ et $c_j$ :

$$c_i \text{ est antérieur à } c_j \Leftrightarrow p_i = p_j = p \text{ et } \forall k \in [1, p-1]\ ti_k < tj_k < ti_{k+1},\ ti_p < tj_p.$$

Nous définissons ensuite l'entrelacement. Les deux *clusters* $c_i$ et $c_j$ sont dit entrelacés si l'un est antérieur à l'autre et si la différence maximum entre deux instants d'images clés correspondantes reste inférieure à un certain seuil $D_{inter}$. Celui-ci est de l'ordre de quelques secondes à quelques minutes. Il sert à ne pas rechercher des *clusters* entrelacés trop écartés. $D_{inter}$ est aussi appelé distance inter-*clusters* maximale. Elle reflète le besoin de proximité temporelle des images clés d'une même occurrence. Plus les *clusters* sont éloignés et plus le contenu qui sépare ces *clusters* à de chance de ne pas être similaire. L'entrelacement entre deux *clusters* $c_i$ et $c_j$ est donc :

$$entrelacement(c_i, c_j) = \begin{cases} 1 & \text{si } c_i \text{ est antérieur à } c_j \text{ ou } c_j \text{ est antérieur à } c_i \\ & \text{et si } \max_{1 \leq k \leq p_i = p_j = p} |ti_k - tj_k| < D_{inter} \\ 0 & \text{sinon.} \end{cases}$$

Pour deux *clusters* entrelacés $c_i$ et $c_j$, nous pouvons enfin définir un score de similarité « *inter-clusters* ». Les *clusters* les plus similaires sont ceux pour lesquels les distributions temporelles des images clés sont les plus similaires. Autrement dit, les différences respectives entre les images clés des deux *clusters* sont les plus régulières. Cela révèle la régularité des occurrences d'une répétition. Deux *clusters* $c_i$ et $c_j$ sont très proches, si la différence $|ti_1 - tj_1|$ est la même que $|ti_2 - tj_2|$ qui est la même que $|ti_3 - tj_3|$, etc. Le score de

similarité est proche de 0 si les images clés ne sont pas régulières et proche de 1 si la régularité est établie. Pour mesurer cette régularité, nous avons choisi le score $SC_{inter-cluster}$, ci-dessous, qui traduit ces propriétés pour deux clusters $c_i$ et $c_j$ entrelacés.

$$SC_{inter-cluster}(c_i, c_j) = exp\left(-\left(\frac{\sum_{k=2}^{p_i=p_j=p} |\,|ti_k - tj_k| - |ti_{k-1} - tj_{k-1}|\,|}{p-1}\right)\right)$$

À partir de ce score de similarité nous pouvons enfin établir un seuil de similarité $SC_{min}$ à partir duquel deux *clusters* sont déclarés similaires. La similarité entre deux *clusters* $c_i$ et $c_j$ est alors définie dans la matrice $\mathcal{M}_{sim}$ par :

$$\mathcal{M}_{sim}(i,j) = \begin{cases} 1 & \text{si } entrelacement(c_i, c_j) = 1 \text{ et } SC_{inter-cluster}(c_i, c_j) > SC_{min} \\ 0 & \text{sinon.} \end{cases}$$

Nous devons apporter quelques nuances. Il peut arriver qu'un *cluster* ne comporte pas que des images clés d'occurrences d'une répétition mais aussi des images similaires correspondant à des images clés de la même occurrence. Ceci peut se produire lors de scènes particulières comme des dialogues, mais aussi lors d'erreurs dans la phase de segmentation en plan. Afin de l'éviter, nous prétraitons les *clusters* pour supprimer les images clés temporellement très proches dans les *clusters*. Deux images clés sont temporellement très proches si la différence entre leurs instants est inférieure à $D_{max}$.

Le choix des valeurs des paramètres $D_{inter}$, $SC_{min}$ et $D_{max}$ est expliqué dans la section 2.5.

#### 2.3.4.2 Construction des occurrences des répétitions à partir des *clusters* similaires

La construction des occurrences des répétitions s'effectue en regroupant d'abord les *clusters* en groupes de *clusters* similaires.

Ces groupes de *clusters* similaires sont des groupes de *clusters* possédant les mêmes distributions temporelles de leurs images clés. Chaque image clé de chacun de ces *clusters* appartient en réalité à une occurrence d'une même répétition. Cela est illustré par la figure 2.5. Ces groupes définissent donc les occurrences des répétitions.

Pour construire les groupes de *clusters* similaires, nous parcourons la matrice de similarité des *clusters* selon l'algorithme 1. Nous trions préalablement les *clusters* d'images clés du plus peuplé au moins peuplé. Puis, en commençant par les *clusters* les plus peuplés, nous sélectionnons tous les *clusters* ayant la même cardinalité $p$. Ces *clusters* ayant la même cardinalité nous permettent de rechercher des *clusters* entrelacés et similaires. Pour cela, nous lisons simplement la matrice de similarité. Par la suite, les ensembles de tous les *clusters* similaires à au moins un autre *cluster* du même ensemble définissent les groupes de *clusters* similaires.

Nous utilisons les notations suivantes dans l'algorithme 1 :

$A \leftarrow B$ : affecter $B$ à la variable $A$

$L \leftarrow L, X$ : ajouter l'élément $X$ à la fin de la liste $L$

$Taille(L)$ : retourne le nombre d'éléments de la liste $L$

$L[i]$ : retourne le $i^e$ élément de la liste $L$

**construction-groupes**

▷ **Entrées:**
$L_{clusters}$ : liste des clusters sélectionnés de même cardinalité ;
$\mathcal{M}_{sim}$ : matrice de similarité des clusters ;

◁ **Sorties:**
$L_{groupes}$ : liste de groupes de clusters similaires ;

**Début**

$L_{groupes} \leftarrow \emptyset$ ;
**tant que** $Taille(L_{clusters}) > 0$ **faire**
  $E_{tmp} \leftarrow L_{clusters}[0]$ ;
  $Pile \leftarrow L_{clusters}[0]$ ;
  **tant que** $Taille(Pile) > 0$ **faire**
    $c_i \leftarrow$ enlever $Pile[0]$ ;
    **pour tout** $c_j$ de $L_{clusters}$ **faire**
      **si** $(\mathcal{M}_{sim}(i,j) = 1)$ et $(c_j \notin E_{tmp})$ **alors**
        $Pile \leftarrow Pile, c_j$ ;
        $E_{tmp} \leftarrow E_{tmp}, c_j$ ;
      **fin si**
    **fin pour**
  **fin tant que**
  **pour tout** $c_k$ de $E_{tmp}$ **faire**
    enlever $c_k$ de $L_{clusters}$ ;
  **fin pour**
  $L_{groupes} \leftarrow L_{groupes}, E_{tmp}$ ;
**fin tant que**
trier $L_{groupes}$ suivant la taille des ensembles $E_{tmp}$ de $L_{groupes}$ ;

**Fin**

Algorithme 1 – Construction de groupes de *clusters* similaires.

Pour construire les occurrences des répétitions, les *clusters* des groupes de *clusters* similaires sont triés suivant leur antériorité. Les images clés du *cluster* le plus antérieur du groupe de *clusters* similaires délimitent des bornes inférieures des occurrences des répétitions. Les images clés du *cluster* le moins antérieur du groupe de *clusters* similaires délimitent des bornes supérieures des occurrences des répétitions.

Les bornes des occurrences construites des répétitions sont à ce stade des images clés. Nous proposons donc une procédure d'extension dans la section 2.3.4.3 suivante. Cette extension ajuste les bornes prédélimitées à partir des images qui entourent les images clés.

Après la construction de chaque occurrence de répétition, nous mettons la matrice de similarité à jour afin de ne pas prendre en compte des *clusters* qui ont déjà servi à la construction d'occurrences de répétitions. Pour mettre la matrice à jour, nous mettons à zéro les scores de similarité relatifs aux *clusters* déjà employés.

Les *clusters* qui ne possèdent pas de *clusters* similaires peuvent aussi générer des occurrences d'une répétition. Ce sont des groupes de *clusters* similaires à un seul élément. Chaque image clé de ces *clusters* définit directement à la fois une borne inférieure et une borne supérieure d'une occurrence de la répétition. L'extension est ensuite appliquée normalement. La génération des occurrences de répétitions à partir de ces *clusters* solitaires fournit un grand nombre de courtes répétitions. Le découpage à partir de ces répétitions contient donc plus de segments. Suivant les performances recherchées, cette utilisation des *clusters* solitaires peut être optionnelle. Nous étudions cette option dans la section 2.5.

#### 2.3.4.3 Extension des occurrences des répétitions

La procédure d'extension consiste à examiner les images qui entourent les bornes prédéterminées des occurrences construites des répétitions. Si la similarité des images voisines des bornes inférieures (ou supérieures) de chaque occurrence d'une répétition est établie, alors ces images sont ajoutées aux occurrences de la répétition. Cela est illustré dans la figure 2.6. Nous n'avons pas utilisé les frontières des plans pour réaliser l'extension. En effet, nous ne souhaitons pas dépendre des performances de la segmentation en plans. Notre système repose ainsi seulement sur des images clés et les images qui les entourent.

Nous allons détailler la méthode d'extension des bornes inférieures prédéterminées des occurrences d'une répétition. La méthode d'extension des bornes supérieures suit le même principe. Nous notons $tinf_k, k \in [0, p-1]$ les bornes inférieures des $p$ occurrences d'une répétition.

Pour décider de la similarité entre toutes les images précédant ou suivant les bornes, nous utilisons les descripteurs $D_I$. Nous notons $D_I(t)$ le descripteur de l'instant $t$. Ces descripteurs sont des vecteurs de bits. Deux images sont alors similaires si leurs vecteurs de bits sont similaires. Pour comparer deux vecteurs de bits $v_1$ et $v_2$, nous employons la distance de *Hamming* : $d_h(v_1, v_2)$. Cette distance correspond au nombre de bits différents entre les deux vecteurs.

Les occurrences peuvent être étendues suivant un seuil de dissimilarité maximal $h_s$. La formule d'extension est alors :

$$\max_{(k,l)\in[0,p-1]^2, k\neq l} d_h(D_I(tinf_k - 1), D_I(tinf_l - 1)) < h_s \Rightarrow \forall k \in [0, p-1], tinf_k = tinf_k - 1$$

Cette formule d'extension traduit la comparaison deux à deux de toutes les bornes inférieures des occurrences d'une répétition.

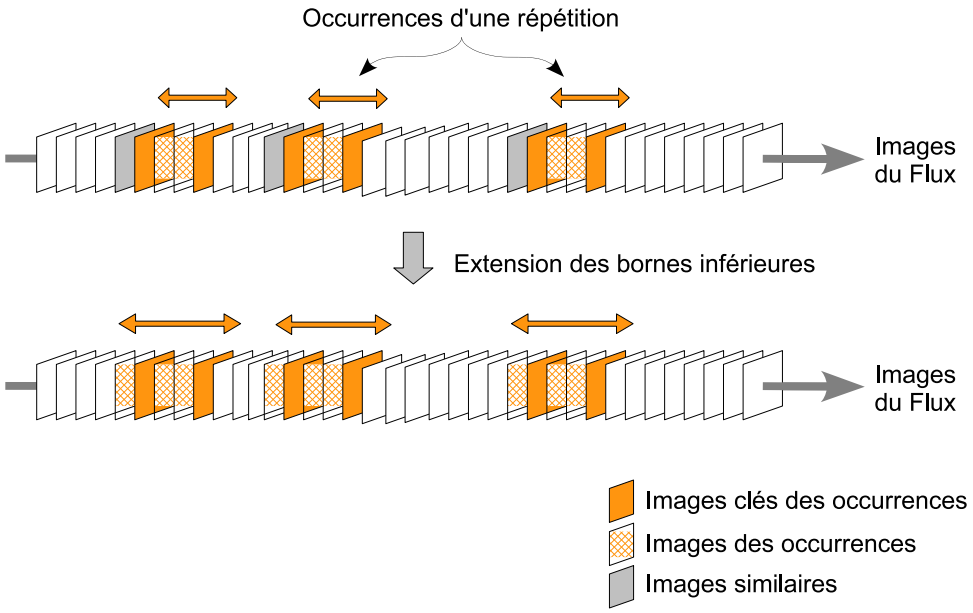



FIGURE 2.6 – Exemple d'extension des bornes inférieures d'occurences d'une répétition.

Afin d'être plus flexible dans l'extension, nous autorisons un faible décalage entre les images voisines des bornes. La formule d'extension devient finalement :

$$
\begin{aligned}
&L_{tmp} \leftarrow \emptyset \\
&\forall k \in [0, p-1] \\
&\max_{l \in [0,p-1], k \neq l} \min \begin{pmatrix} d_h(D_I(tinf_k - 1), D_I(tinf_l - 1)) \\ d_h(D_I(tinf_k - 1), D_I(tinf_l)) \end{pmatrix} < h_s \Rightarrow L_{tmp} \leftarrow L_{tmp}, k \\
&\forall k \in L_{tmp} \Rightarrow tinf_k = tinf_k - 1
\end{aligned}
$$

Cette dernière formule comptabilise les bornes inférieures similaires deux à deux à une image près, puis elle étend chacune de ces bornes.

### 2.3.5 Gestion de la continuité

Nous avons présenté jusque là la version « statique » de notre méthode. Pour une portion de flux TV fixée, nous détectons un ensemble de répétitions. Une propriété fondamentale des flux TV est leur diffusion en continu. Notre méthode doit donc pouvoir faire facilement face à la diffusion ininterrompue de nouveaux contenus audiovisuels.

Nous rappelons que le flux est décrit par des images clés. Deux possibilités sont à étudier pour le traitement du flux en continu :

– l'analyse « infinie ». Les images clés sont ajoutées au fur et à mesure sur un ensemble d'images clés déjà conservées. Les images clés répétées après un grand intervalle de temps peuvent être détectées ;

– l'analyse « avec historique ». Les images clés sont ajoutées au fur et à mesure sur un ensemble d'images clés conservées pendant un historique de temps limité. Seules les images clés répétées pendant l'historique de temps limité peuvent être détectées.

Suivant nos connaissance *a priori* des flux télévisés, la distance temporelle maximale entre deux occurrences d'une répétition d'inter-programme est généralement de l'ordre de la semaine. Nous identifions seulement de rares types d'inter-programmes pouvant être rediffusés à l'identique un mois après leur première diffusion. Par conséquent, l'analyse « avec historique » nous semble la plus appropriée.

De plus, nous donnons dans l'annexe B, la proportion de flux non rediffusé dans le flux en une semaine. Cette proportion est de 83,90 % pour France 2 et de 79,73 % pour TF1. Elle permet d'estimer qu'environ 80 % des contenus de la télévision sont diffusés une unique fois. L'analyse « infinie » implique donc de conserver une quantité très importante d'images clés qui possèdent une très forte probabilité de ne jamais servir. Malgré les capacités de l'algorithme de *clustering* à identifier le bruit et à créer des *clusters* compacts, ces images clés conservées ont plus de chance d'être incluses avec d'autres *clusters* et risquent de détériorer les résultats. Pour éviter cela, nous conservons donc notre premier choix d'une analyse « avec historique ». La taille de l'historique devient cependant une propriété critique de notre système. Nous expliquons grâce à des expériences, dans la section 2.5, comment déterminer cette taille.

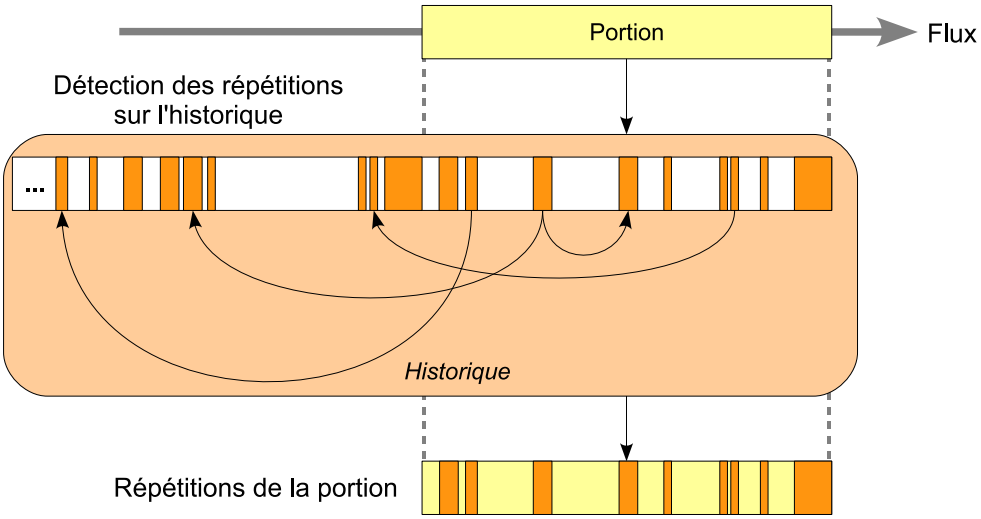



FIGURE 2.7 – Calcul des répétitions d'une portion par rapport à un historique.

Afin de traiter le flux en continu, notre méthode est relancée périodiquement. De manière périodique donc, les répétitions sont détectées sur une portion du flux. La portion analysée est en réalité ajoutée à un historique dans lequel toutes les répétitions sont détectées. Ainsi, une occurrence d'une répétition présente dans la portion analysée peu être associée avec une ancienne occurrence de cette répétition préservée dans l'historique. Finalement, pour le découpage, nous conservons seulement les répétitions possédant des

occurrences sur la portion de flux analysée. La figure 2.7 schématise ce traitement.

La gestion de la continuité dans notre système est réalisée par l'algorihtme 2. Cet algorithme se divise en deux étapes. La première étape supprime des images clés existantes pour mettre à jour l'historique. La seconde étape ajoute de nouvelles images clés à regrouper par *clustering* dans l'historique. C'est ce que nous appelons *clustering* incrémental. Nous le schématisons dans la figure 2.8.

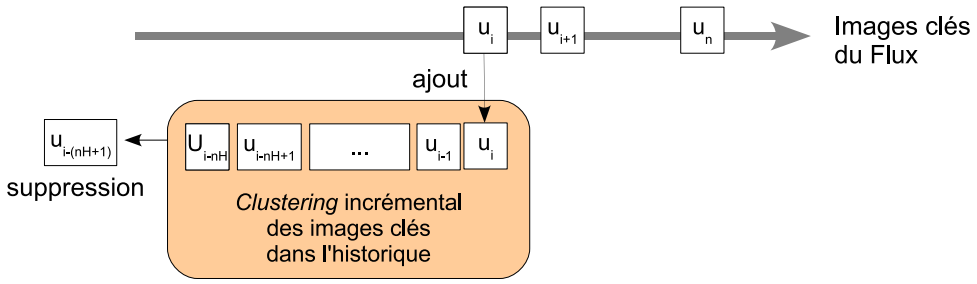



FIGURE 2.8 – Schéma du *clustering* incrémental des images clés.

Par sa nature, le micro-*clustering* utilisé est déjà incrémental. L'étape d'ajout d'une nouvelle image clé à regrouper s'effectue donc par l'algorithme 8 présenté en annexe D. L'ajout d'une nouvelle image clé s'effectue naturellement sur l'ensemble des *clusters* existants. Il suffit de sélectionner le *cluster* capable de fusionner avec le *cluster* de rayon nul constitué par la nouvelle image clé. Si aucun *cluster* ne convient, le nouveau *cluster* de rayon nul forme un *cluster* à lui tout seul. Avant d'insérer les nouvelles images clés, il est possible de les regrouper préalablement en *clusters*. Cette option engendre des fusions de *clusters* qui créent des *clusters* plus compacts que lorsque les images clés sont insérées une à une via des *clusters* de rayon nul. En effet, pour un rayon maximal $Rmax$ donné, un *cluster* issu de la fusion de deux *clusters* est finalement plus compact qu'un autre, car le rayon résultant de la fusion est une sur-estimation du rayon réel (cf. annexe D).

L'étape de suppression des images clés déjà existantes est aussi simple. À chaque image clé est associé son instant précis de diffusion. Il suffit de parcourir toutes les images clés conservées et de retirer les images clés dont l'instant de diffusion est trop ancien. Lorsque l'on retire une image clé d'un *cluster*, le rayon du *cluster* reste inchangé. Ce rayon devient surestimé et le *cluster* reste plus compact.

Traiter un flux en continu impose aussi d'enregistrer le flux en continu. Un problème évident est la capacité du système de stockage des enregistrements et des descripteurs. Face à la taille des images et des sons du flux audiovisuel même compressé (entre 7 et 8 Go par jour dans notre système avec le format de compression Xvid), la taille des descripteurs $D_I$ et $D_C$ utilisés (entre 20 et 30 Mo) est négligeable. Le problème majeur reste dans la capacité d'enregistrement d'un flux TV.

## 2.4 Notre méthode de découpage à partir des répétitions

La méthode présentée précédement nous permet de détecter l'ensemble des répétitions du flux TV. À partir de ces répétitions, nous découpons le flux en segments.

**clustering-incrémental**

▷ **Entrées:**
$H$ : historique d'images clés :$u_{i-nH}, \ldots, u_{i-1}$ de taille $nH$
$C_H$ : ensemble des *clusters* d'images clés de l'historique $H$
$t_H$ : instant limite de l'historique $H$
$P_F$ : portion du flux à traiter d'images clés $u_i, \ldots, u_{i+n}$ ;

◁ **Sorties:**
$L_{repetitions}$ : liste des répétitions dans $P_F$ ;
$H$ : historique mis à jour ;
$C_H$ : ensemble des *clusters* de l'historique mis à jour ;
$t_H$ : instant limite de l'historique mis à jour

**Début**
$t_H \leftarrow t_H +$ durée de $P_F$ ;
**pour tout** $u_k \in H$ **faire**
  **si** instant de $u_k < t_H$ **alors**
    supprimer $u_k$ de $H$ et de $C_H$ ;
  **fin si**
**fin pour**
**pour tout** $u_k \in P_F$ **faire**
  insérer $u_k$ par *clustering* dans $C_H$ ;
  $H \leftarrow H, u_k$ ;
**fin pour**
$L_{repetitions} \leftarrow$ construction des répétitions sur $C_H$ ;
**pour tout** occurrence $x$ des répétitions de $L_{repetitions}$ **faire**
  **si** $x \notin P_F$ **alors**
    supprimer $x$ des répétitions de $L_{repetitions}$ ;
  **fin si**
**fin pour**

**Fin**

Algorithme 2 – Détection des répétitions par *clustering* incrémental d'une portion du flux dans un historique.

La technique de découpage à partir des répétitions est la suivante. Nous la schématisons dans la figure 2.9. Les occurrences des répétitions définissent les premiers segments dans le flux. Nous appelons ces segments issus d'occurrences de répétitions des segments « répétés ». Ensuite, les morceaux du flux entre les segments répétés déjà définis forment les autres segments du flux. Nous obtenons ainsi un flux découpé en segments.

Nous imposons une durée minimum aux segments. Cette durée est cependant courte, elle vaut une demi seconde. Elle sert à éviter de prendre en compte des micro-morceaux qui engendrent des micro-segments trop courts. Ces micro-morceaux sont donc ignorés.

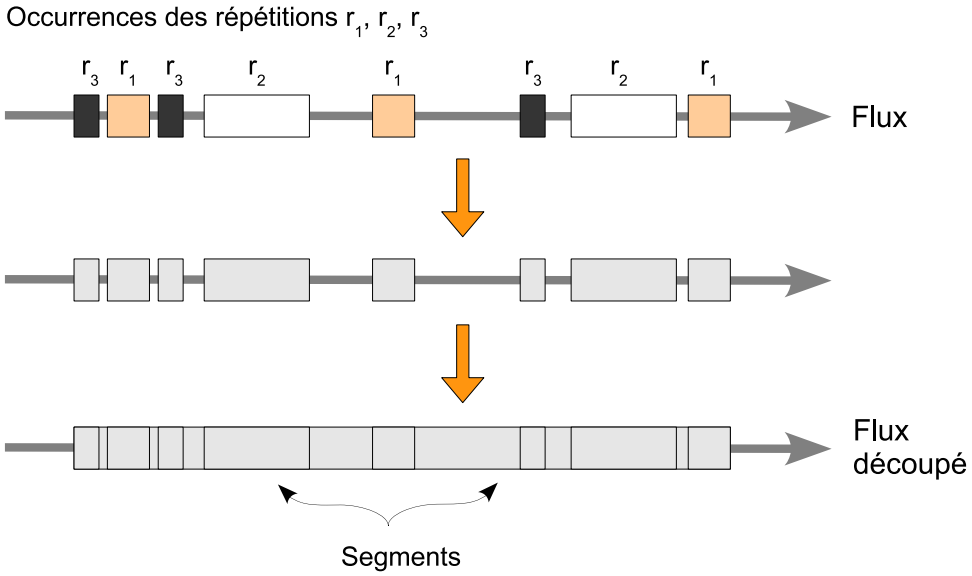



FIGURE 2.9 – Schéma de découpage du flux à partir des occurrences des répétitions.

Notre méthode de construction des occurrences des répétitions ne limite ni la durée des occurrences ni le nombre d'occurrences d'une répétition. Il peut arriver des cas où les occurrences des répétitions sont imbriquées. Cela engendre des segments imbriqués, ce qui complexifie la segmentation.

Le premier cas est illustré dans la figure 2.10. Des occurrences de la répétition $r_2$ sont nettement incluses dans les occurrences de $r_1$. La distance entre les bornes inférieures des occurrences de $r_2$ et de $r_1$ qui se chevauchent est supérieure à un seuil fixé à une seconde. De la même manière, la distance entre les bornes supérieures des occurrences de $r_2$ et de $r_1$ qui se chevauchent est supérieure à ce même seuil. De manière générale, les occurrences des répétitions représentent des diffusions de contenus entiers. Pour conserver les occurrences de $r_1$ entières, les occurrences de $r_2$ incluses dans celles de $r_1$ sont supprimées. Elles fusionnent avec les occurrences de $r_1$.

Le deuxième cas est illustré dans la figure 2.11. Des occurrences de la répétition $r_2$ sont localisées près des bornes inférieures ou supérieures des occurrences de $r_1$. Ce cas est généralement dû à la phase d'extension puisque aucun test d'intersection avec les occurrences préalablement détectées n'est effectué. Dans ce cas, les occurrences de $r_1$ sont

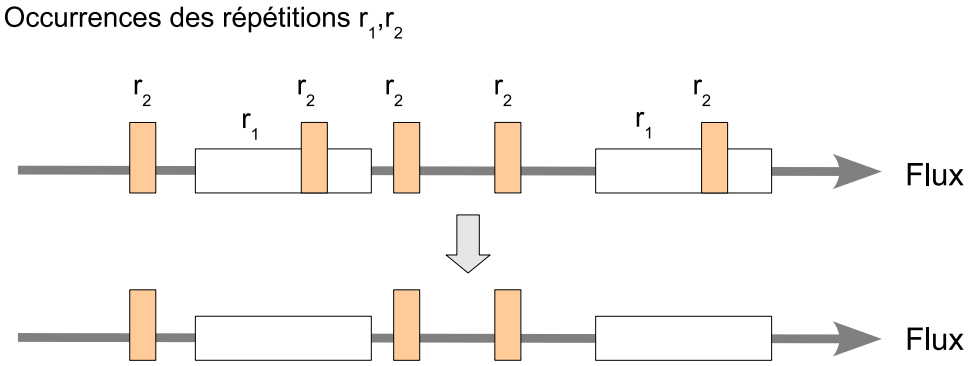



FIGURE 2.10 – Les occurrences des répétitions $r_1$ et $r_2$ sont imbriquées. Les occurrences de $r_2$ sont nettement incluses dans celles de $r_1$.

réduites de manière à ne plus inclure des occurrences de $r_2$.

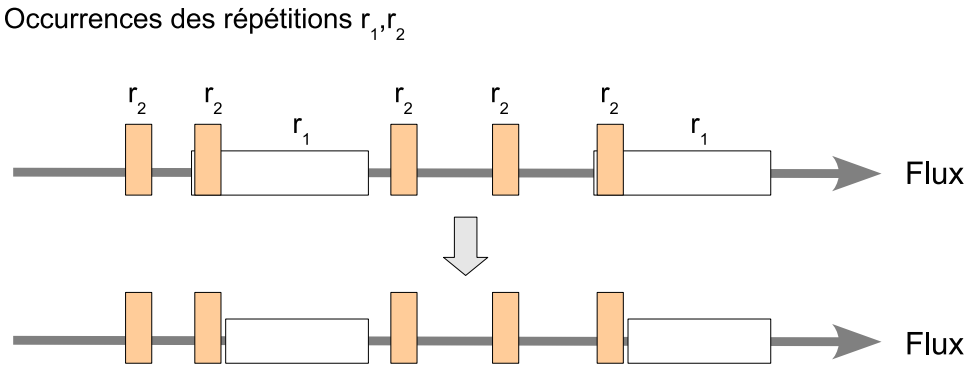



FIGURE 2.11 – Les occurrences des répétitions $r_1$ et $r_2$ sont imbriquées. Les occurrences de $r_2$ sont localisées près d'une des bornes de celles de $r_1$.

Le cas où le nombre d'occurrences de la répétition $r_2$ est inférieur strictement par rapport au nombre d'occurrences de $r_1$ n'est pas possible car toutes les occurrences de $r_1$ contiennent un contenu quasi identique.

Nous savons maintenant découper un flux TV en fonction des répétitions qu'il contient.

## 2.5 Résultats

Nous avons évalué rigoureusement nos deux méthodes de détection des répétitions pour la détection des inter-programmes rediffusés et de découpage du flux à partir des répétitions. Nos évaluations se sont d'abord concentrées sur la vérification de nos quatre contraintes à savoir l'efficacité, la généricité, l'automaticité et la continuité. Les expériences proposées montrent dans quelle mesure nous satisfaisons ces contraintes. De plus,

les évaluations nous permettent aussi d'expliquer empiriquement le fonctionnement des principales étapes de notre système.

Nous présentons le contexte expérimental dans lequel se place notre évaluation, le protocole d'évaluation puis les expériences et les résultats.

### 2.5.1 Contexte expérimental

Le corpus utilisé lors des expériences est composé de 8 semaines de flux TV : 4 semaines proviennent de la chaîne France 2 et 4 semaines proviennent de la chaîne TF1. Le corpus est décrit dans l'annexe A. Le lecteur est invité à s'y référer.

Les programmes et les inter-programmes de ce corpus sont précisément délimités dans une vérité terrain que nous avons manuellement réalisée. Cette vérité terrain est présentée en annexe B. Elle permet d'évaluer notre méthode de découpage en segments. En plus, nous avons regroupé dans cette vérité terrain, sur une semaine de France 2 et sur une semaine de TF1, les inter-programmes en groupes d'inter-programmes quasi identiques. Autrement dit, nous avons déterminé manuellement les occurrences des répétitions d'inter-programmes. Nous n'avons pas déterminé manuellement les occurrences de toutes les répétitions existantes car c'est une tâche pratiquement impossible. Au final, les deux semaines particulières permettent d'évaluer notre méthode de détection des répétitions pour la détection des inter-programmes. Il est important de noter que nous n'évaluons pas notre méthode de détection des répétitions par sa capacité à détecter toutes les répétitions mais par sa capacité à détecter seulement les répétitions d'inter-programmes. Les répétitions d'inter-programmes sont l'objectif de notre méthode de détection des répétitions.

Tous les algorithmes de notre solution ont été implémentés en C++ sous l'environnement de développement Microsoft Visual Studio 2008. Ces algorithmes ont été exécutés sur des ordinateurs de type PC possédant 4 Go de mémoire centrale et un processeur Intel Xeon 5160 à deux cœurs cadencés à 3 GHz.

### 2.5.2 Protocole d'évaluation

Pour évaluer notre système, nous employons des mesures de précision et de rappel empruntées au domaine de la recherche d'information. La précision ($P$) est le nombre d'éléments pertinents détectés par un système de recherche sur l'ensemble total des éléments détectés par le système. Le rappel ($R$) est le nombre d'éléments pertinents détectés par un système de recherche sur l'ensemble total des éléments pertinents existants. La pertinence d'un élément est définie en utilisant la vérité terrain.

$$P = \frac{\text{Détectés } \cap \text{ Pertinents}}{\text{Détectés}} \qquad R = \frac{\text{Détectés } \cap \text{ Pertinents}}{\text{Pertinents}}$$

Pour combiner la précision et le rappel nous utilisons la moyenne harmonique ($F$) entre la précision $P$ et le rappel $R$. Cette moyenne harmonique est aussi appelée F-mesure. Elle exprime au mieux la « moyenne » des deux ratios $P$ et $R$ considérés avec la même importance.

$$F = \frac{2PR}{P+R}$$

Nous évaluons d'abord notre méthode de détection des répétitions d'inter-programmes puis notre méthode de découpage du flux.

#### 2.5.2.1 Détection des répétitions d'inter-programmes

La pertinence d'une répétition détectée s'effectue grâce à la vérité terrain qui recense toutes les occurrences des répétitions d'inter-programmes recherchées. Nous définissons un moyen de comparaison entre une occurrence d'une répétition de la vérité terrain et une occurrence d'une répétition détectée par notre système. Pour cela, nous avons choisi de comparer les plans des occurrences. Un plan issu d'une occurrence détectée est considéré comme pertinent s'il appartient à une occurrence d'une répétition d'un inter-programme recensé dans la vérité terrain.

Les plans détectés sont les plans des occurrences des répétitions détectées. La précision $P_{plan}$ évalue les plans détectés pertinents par rapport à l'ensemble des plans détectés. Le rappel $R_{plan}$ évalue les plans détectés pertinents par rapport à l'ensemble des plans pertinents. La moyenne harmonique associée est notée $F_{plan}$.

Par cette mesure, nous n'évaluons pas la qualité du regroupement des occurrences des répétitions. Nous évaluons seulement la proportion des plans détectés des occurrences des répétitions d'inter-programmes.

#### 2.5.2.2 Découpage du flux

Le découpage d'un flux fournit un ensemble de segments qui représentent des programmes, des parties de programmes ou des inter-programmes. Pour évaluer la justesse de ce découpage, nous évaluons la capacité de notre technique à détecter les bornes des programmes, des parties de programmes et des inter-programmes. Nous comparons donc l'ensemble des bornes des segments découpés et l'ensemble des bornes des segments de la vérité terrain. Nous manipulons donc seulement un ensemble de bornes qui sont des instants dans le flux.

Dans notre contexte, le découpage ne s'évalue pas à l'image près. En effet, nous ne détectons pas des frontières de plans mais des bornes de programmes ou d'inter-programmes dont les positions exactes peuvent être très ambigües. Nous expliquons dans l'annexe B que notre vérité terrain est précise à la seconde près seulement. De plus, la détermination des bornes des répétitions détectées par l'extension automatique peut inclure les images monochromes qui séparent parfois certains inter-programmes. Par contre, ces images monochromes ne sont pas incluses dans les bornes des inter-programmes de la vérité terrain. Au final, nous autorisons que la différence entre la borne détectée automatiquement et la borne choisie dans la vérité terrain varie jusqu'à deux secondes.

À 2 secondes près, une borne détectée est pertinente si cette borne se trouve à moins de 2 secondes d'une borne de la vérité terrain. Inversement, une borne de la vérité terrain est détectée si cette borne se trouve à moins de 2 secondes d'une borne détectée. Pour évaluer nos résultats en prenant en compte cette tolérance, nous calculons la précision $P_{2s}$, le rappel $R_{2s}$ et la moyenne harmonique associée $F_{2s}$.

### 2.5.3 Expériences 1 : étude de l'automaticité et de la généricité

La première contrainte clé de notre solution de délinéarisation est l'automaticité. Nous recherchons en effet un système capable de délinéariser les flux télévisuels de manière complètement automatique. Nous vérifions donc cette contrainte sur ce premier niveau de

délinéarisation composé d'une méthode de détection des répétitions et d'une méthode de découpage à partir des répétitions.

Ce premier niveau de délinéarisation tel que nous l'avons présenté est *a priori* automatique dans le sens où il ne requiert aucune étape manuelle lors de son application. Il nécessite cependant d'être configuré afin de déterminer de nombreux paramètres. Si ces paramètres dépendent de trop nombreuses conditions avant chaque application, alors le niveau de délinéarisation requiert des configurations manuelles et il perd en automaticité. Nous souhaitons donc étudier ces paramètres. De plus, nous cherchons à savoir si ces paramètres dépendent de la chaîne à analyser. De cette manière nous vérifions, en plus, la généricité de ce niveau de délinéarisation.

Nous rappelons pour commencer les quatre principales étapes de ce premier niveau de délinéarisation :

1) le flux est d'abord décrit par un ensemble d'images clés ;

2) les images clés sont regroupées par similarités dans des *clusters* ;

3) les occurrences des répétitions sont construites à partir des *clusters* ;

4) les occurrences des répétitions sont utilisées pour découper le flux.

Le cœur de ce niveau de délinéarisation réside dans les étapes 2 et 3. En effet, l'étape 1 calcule simplement des descripteurs d'images clés sur le flux et l'étape 4 de découpage à partir des répétitions ne requiert pas de paramètres critiques. Les descripteurs d'images clés sont bien définis, ils ne dépendent pas de paramètres.

Nous cherchons donc à étudier les paramètres du *clustering* puis ceux pour la construction des occurrences des répétitions.

#### 2.5.3.1 Choix du rayon des *clusters*

Le *clustering* dépend de trois paramètres : le rayon maximal initial des *clusters* ($Rmax_0$), le pas d'incrémentation du rayon maximal ($\Delta_{Rmax}$) et le rayon maximal qui constitue le critère d'arrêt ($Rmax_{stop}$). Une manière simple d'étudier l'impact de ces paramètres sur les *clusters* est de définir un rayon maximal initial nul, un pas d'incrémentation petit et de ne pas fixer de critère d'arrêt.

Nous réalisons cette expérience sur 33813 images clés qui décrivent 24 heures de flux télévisuel sur France 2. Le rayon maximal $Rmax$ est initialisé à $Rmax_0 = 0$ et $\Delta_{Rmax}$ est fixé à $0,005$. Pour chaque valeur du rayon maximal $Rmax$, nous calculons le nombre de *clusters* de l'algorithme de *clustering*. Les résultats sont représentés par la courbe de la figure 2.12. Initialement chaque descripteur forme un *cluster* puis les *clusters* fusionnent mutuellement. Comme nous pouvions nous y attendre, le nombre de *clusters* décroit rapidement quand le rayon maximal augmente.

Nous avons ensuite évalué la qualité des *clusters* générés pour chaque valeur du rayon $Rmax$. Pour cela, nous fixons intuitivement les paramètres de la construction des répétitions à partir des *clusters* d'images clés et nous construisons les répétitions pour chaque ensemble de *clusters* créé pour chaque valeur de rayon $Rmax$. Les paramètres de la construction des répétitions sont étudiés dans la section 2.5.3.4 suivante.

Nous évaluons d'abord la détection des répétitions d'inter-programmes. La figure 2.13 montre l'évolution en fonction du rayon $Rmax$ de la précision $P_{plan}$ et du rappel $R_{plan}$ de la détection des plans de répétitions d'inter-programmes. Quand le rayon $Rmax$ augmente,

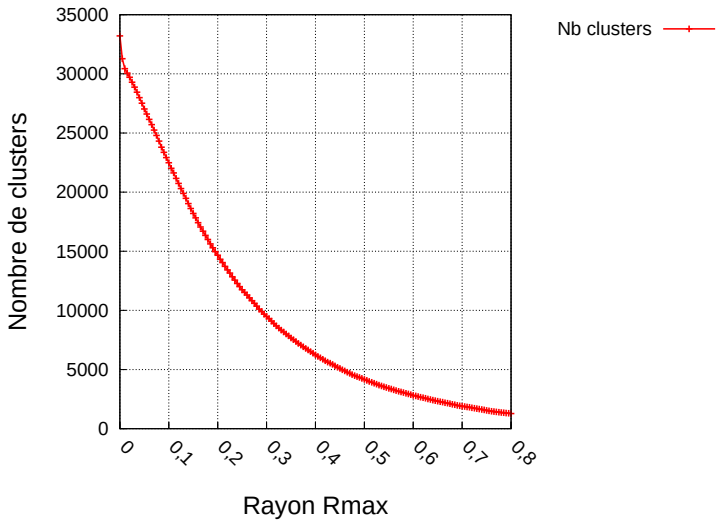
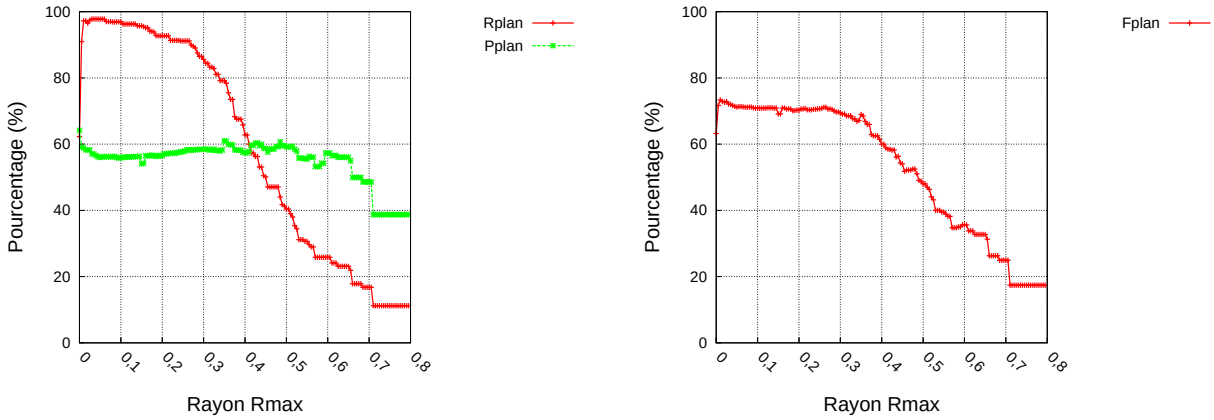



FIGURE 2.12 – Nombre de *clusters* créés en fonction de la variation du rayon maximal *Rmax* des *clusters* sur 24 heures sur France 2.



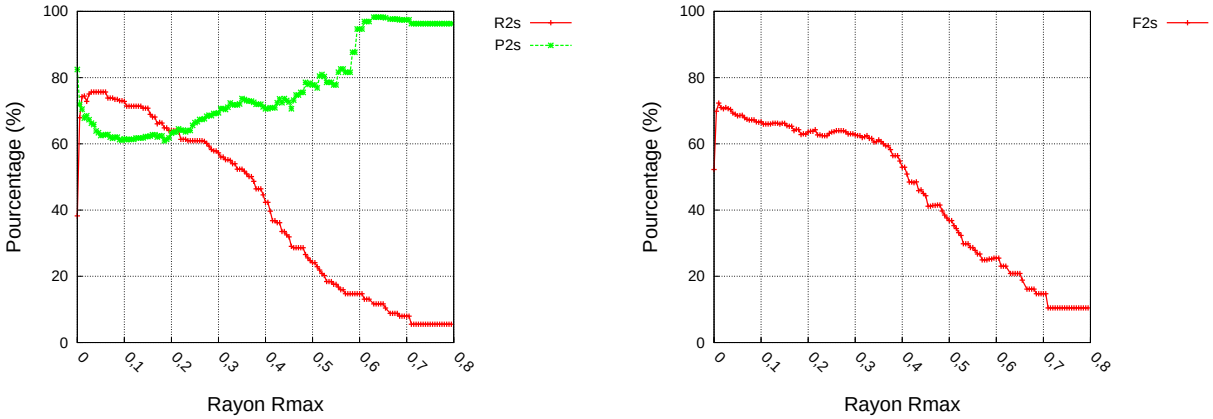

(a) Selon la précision $P_{plan}$ et le rappel $R_{plan}$ (b) Selon la F-mesure $F_{plan}$

FIGURE 2.13 – Évaluation de la détection des répétitions d'inter-programmes pour chaque ensemble de *clusters* créé pour chaque valeur du rayon maximal *Rmax* sur 24 heures sur France 2.

les *clusters* grossissent et deviennent globalement moins compacts. Les images clés se ressemblent alors de moins en moins, et il devient plus difficile de construire des répétitions. Par conséquent, le rappel diminue. Au contraire, la précision augmente car le taux d'inter-programmes parmi le peu de répétitions détectées augmente.

La figure 2.13 montre les deux courbes de précision et de rappel en présentant l'évolu-

tion de la F-mesure $F_{plan}$ en fonction du rayon $Rmax$. Les courbes révèlent une tendance. Les détections d'inter-programmes sont mieux détectées pour un rayon $Rmax$ petit mais non nul. Les courbes montrent aussi que les rappels sont élevés et les précisions plus basses. Les répétitions d'inter-programmes sont bien détectées mais de nombreuses autres répétitions sont détectées et ne sont pas des inter-programmes. Une analyse plus détaillée des répétitions détectées est proposée dans la section 2.5.5.1.




(a) Selon la précision $P_{2s}$ et le rappel $R_{2s}$ (b) Selon la F-mesure $F_{2s}$

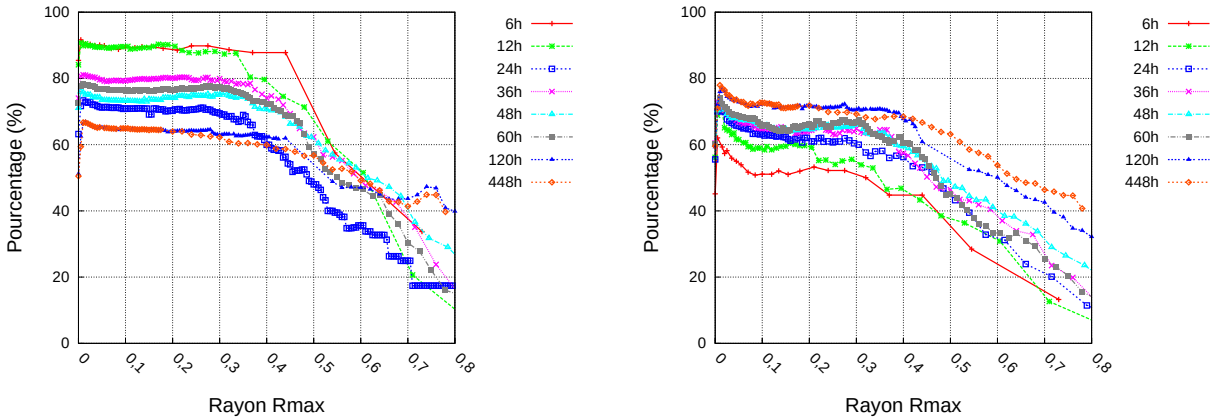
FIGURE 2.14 – Évaluation du découpage pour chaque ensemble de *clusters* créé pour chaque valeur du rayon maximal $Rmax$ sur 24 heures sur France 2.

Nous évaluons ensuite le découpage du flux. La figure 2.14 montre l'évolution de la précision $P_{2s}$, du rappel $R_{2s}$ et de la F-mesure $F_{2s}$ en fonction du rayon $Rmax$. Ces courbes sont analogues aux courbes de la figure 2.13. Lorsque le rayon $Rmax$ est petit, nous détectons plus de répétitions d'inter-programmes. Par conséquent, le découpage est meilleur. La tendance est confirmée.

Pour confirmer ces résultats, nous avons refait cette expérience sur différentes journées du flux. Tous les résultats obtenus sont également similaires aux résultats de l'expérience précédente. Nous déterminons ainsi les paramètres du *clustering* d'une journée sur France 2 : le rayon maximal initial est fixé à $Rmax_0 = 0$, le pas d'incrémentation est fixé à $\Delta_{Rmax} = 0,005$ et, suivant la lecture des courbes, le critère d'arrêt est fixé au rayon maximal petit mais non nul $Rmax_{stop} = 0,01$.

### 2.5.3.2 Variation de la quantité de flux à traiter

Nous cherchons à présent à paramétrer le *clustering* d'une quantité différente d'une journée de flux sur France 2. Nous étudions donc l'évolution des paramètres du *clustering* en fonction de la quantité de flux à traiter. Pour cela, nous réalisons l'expérience précédente avec différentes quantité de flux à traiter. Pour 6h, 12h, 24h, 36h, 48h, 60h, 120h et 448h de flux, nous initialisons le rayon $Rmax$ à zéro, nous gardons un pas d'incrémentation petit de $\Delta_{Rmax} = 0,005$ et nous faisons varier le $Rmax$ sans critère d'arrêt. Les résultats sont montrés dans la figure 2.15.




(a) Selon la F-mesure $F_{plan}$ (b) Selon la F-mesure $F_{2s}$

FIGURE 2.15 – Évaluation de la détection des répétitions d'inter-programmes et du découpage pour chaque ensemble de *clusters* créé pour chaque valeur du rayon maximal $Rmax$ sur France 2 pour différentes quantité de flux.

Les courbes de la figure 2.15 sont superposées. Elles permettent de voir que la tendance dégagée pour 24 heures, lors de l'expérience précédente, se vérifie aussi pour toutes les quantités de flux à partitionner considérées. Le choix d'un rayon maximal $Rmax$ petit mais non nul est à chaque fois la meilleure configuration. Nous pouvons donc conserver les paramètres préalablement fixés dans l'expérience précédente pour toutes les quantités de flux à traiter, jusqu'à une semaine de flux.

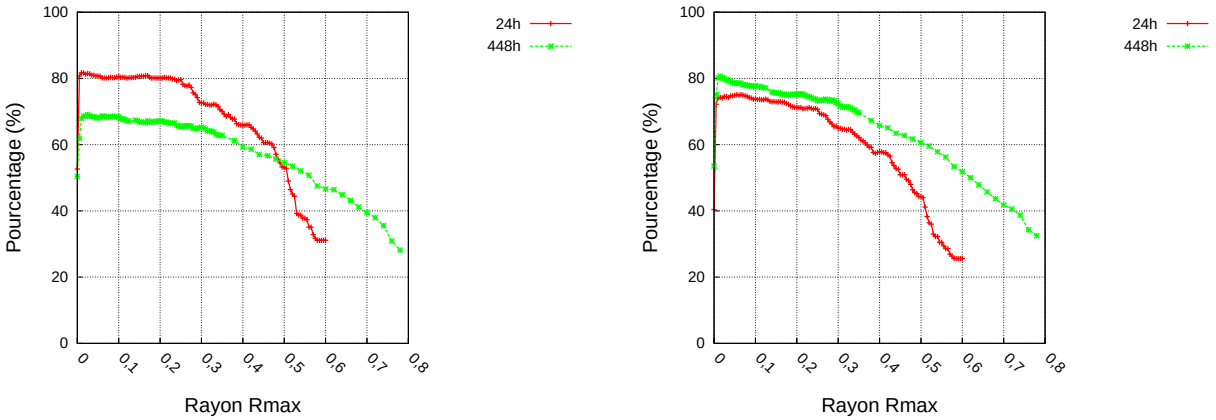
Les courbes montrent, par ailleurs, que la F-mesure $F_{plan}$ chute avec l'augmentation de la quantité de flux. Cela est dû à la détection progressive de rediffusion de programmes longs. Les programmes longs rediffusés fournissent beaucoup de plans répétés qui ne sont pas des inter-programmes et qui détériorent la précision $P_{plan}$. Ces rediffusions de programmes longs ne forment par contre qu'un seul segment dans le découpage. Cela n'a donc pas d'incidence sur $F_{2s}$.

#### 2.5.3.3 Extension de l'étude au flux d'une autre chaîne TV

Nous évaluons dans cette expérience si des paramètres spécifiques à la chaîne TV sont nécessaires. Nous réalisons, pour cela, sur du flux de la chaîne TF1, le même type d'expérience que les expériences précédentes. Les résultats montrés dans la figure 2.16 valident à nouveau la tendance dégagée précédemment pour le flux de France 2.

Pour les deux chaînes de télévision traitées et pour toutes les quantités de flux considérées, nous obtenons les meilleurs résultats de détection des répétitions d'inter-programmes et de découpage pour un rayon maximal $Rmax$ des *clusters* petit mais non nul.

Pour la suite de nos expérimentations, nous pouvons fixer les paramètres du *clustering*. Le rayon maximal initial est fixé à $Rmax_0 = 0$, le pas d'incrémentation est fixé à $\Delta_{Rmax} = 0,005$, et le critère d'arrêt est fixé au rayon maximal petit mais non nul $Rmax_{stop} = 0,01$.

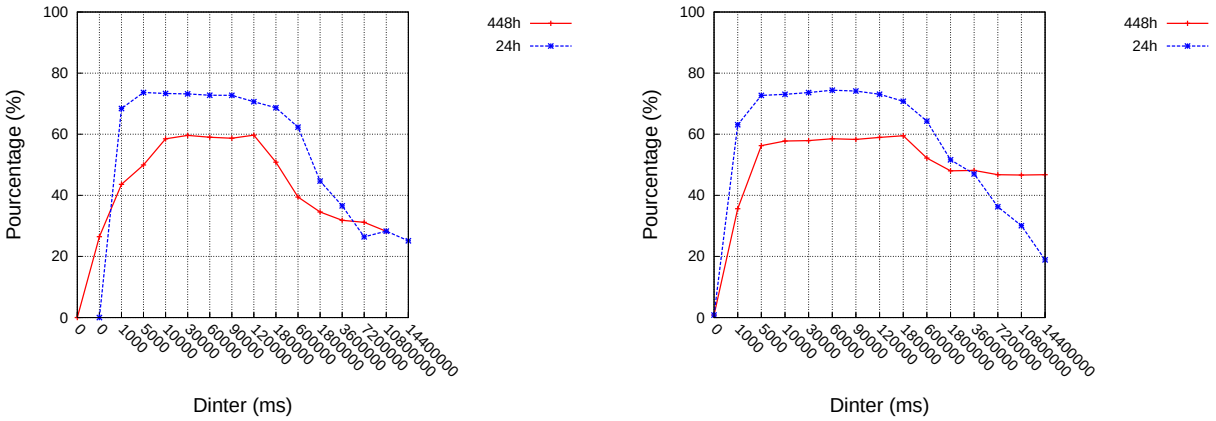



(a) Selon la F-mesure $F_{plan}$ (b) Selon la F-mesure $F_{2s}$

FIGURE 2.16 – Évaluation de la détection des répétitions d'inter-programmes et du découpage pour chaque ensemble de *clusters* créé pour chaque valeur du rayon maximal $Rmax$ sur TF1 pour différentes quantité de flux.

Cette configuration est optimale pour France 2 et pour TF1 et pour 24 heures de flux ou pour une semaine de flux. Cette étape de notre niveau de délinéarisation est donc, pour la suite, réalisée de manière automatique et générique.

#### 2.5.3.4 Configuration des paramètres de la construction des répétitions




(a) Selon la F-mesure $F_{plan}$ (b) Selon la F-mesure $F_{2s}$

FIGURE 2.17 – Évaluation de la détection des répétitions d'inter-programmes et du découpage en fonction du paramètre $D_{inter}$ de construction des répétitions sur France 2 pour différentes quantités de flux.

Lors des expériences précédentes, nous avons choisi de manière intuitive les paramètres de l'étape de construction des répétitions à partir des *clusters*. Nous avons alors réalisé des expériences supplémentaires pour déterminer empiriquement ces paramètres.

La construction des répétitions dépend principalement de 3 paramètres :

1) $D_{inter}$ : la distance temporelle inter-*cluster* maximale ;
2) $SC_{min}$ : la similarité minimum entre deux *clusters* ;
3) $D_{max}$ : la distance temporelle minimum entre deux images clés d'un même *cluster*.

Nous avons fait varier de manière indépendante chacun de ces paramètres à la fois sur 448 heures et sur 24 heures sur France 2. Nous avons alors évalué la méthode de détection des répétitions d'inter-programmes selon la F-mesure $F_{plan}$ et le découpage selon la F-mesure $F_{2s}$.

La figure 2.17 illustre les résultats de la variation du paramètre $D_{inter}$. Lorsque le $D_{inter}$ est trop petit, celui-ci empêche la méthode de trouver des similarités entre les *clusters*. $D_{inter}$ représente en effet un seuil sur les distances temporelles entre les images clés de deux *clusters* entrelacés. Lorsque le $D_{inter}$ est trop grand, celui ci entraîne la création d'occurrences de répétitions aberrantes dont les images clés sont des images clés trop espacées. Le choix de $D_{inter}$ peut donc s'effectuer dans l'intervalle compris environ entre 10000 ms et 120000 ms d'après la figure 2.17. Cet intervalle est valable pour les deux quantités de flux considérées (24 heures et 448 heures). Nous avons, de plus, expérimentalement vérifié que cet intervalle est aussi valable sur la chaîne TF1.

Des courbes similaires ont été obtenues pour les autres paramètres. Ces courbes montrent des zones pour lesquels les résultats sont les meilleurs pour toutes les quantités de flux considérées et pour les deux chaînes étudiées. Nous fixons donc les paramètres de la construction des répétitions à partir de ces courbes. Les paramètres gardés sont les suivants :

$$D_{inter} = 30000\ ms,\ SC_{min} = 0,95 \text{ et } D_{max} = 8000\ ms.$$

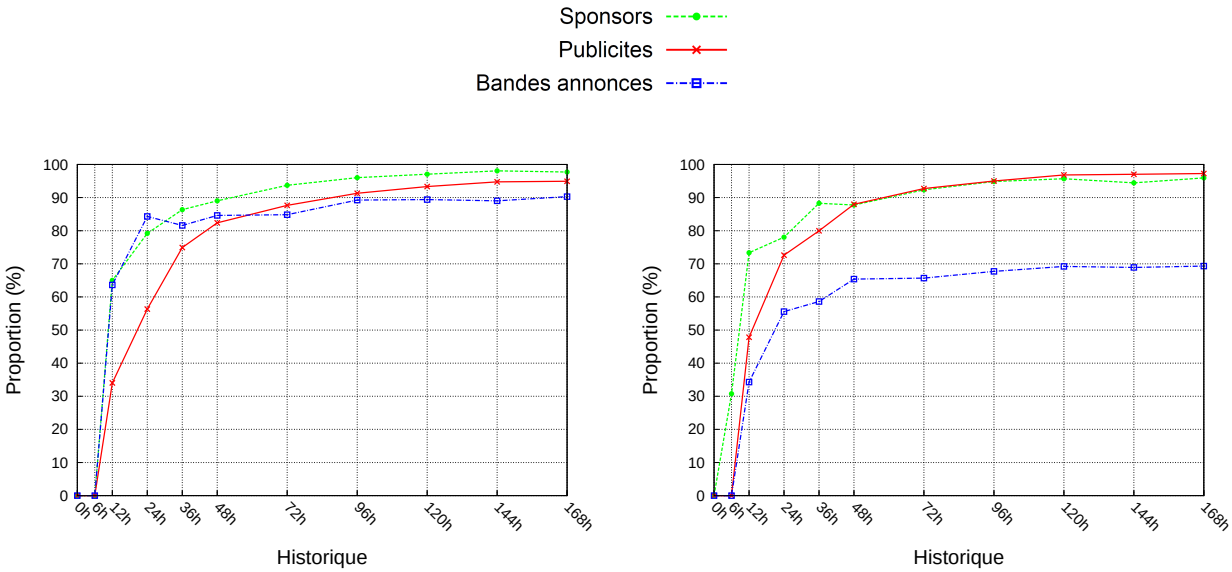
### 2.5.4 Expériences 2 : étude de la continuité

La continuité de ce niveau de délinéarisation est sa capacité à traiter un flux télévisuel en continu. Elle implique la gestion d'un nombre indéterminé d'opérations répétées pour chaque nouvelle portion de flux à analyser. Il est important que les temps de calcul requis par ces opérations soient stables. Ceci est nécessaire pour faire face aux flux ininterrompus.

Lors de l'analyse périodique d'un flux portion par portion, nous détectons les répétitions contenues dans un historique actualisé par chaque nouvelle portion. Les temps de calcul de ce niveau de délinéarisation dépendent donc de la durée de l'historique et de la durée des portions de flux qui mettent à jour l'historique.

#### 2.5.4.1 Choix de la durée de l'historique

Les répétitions sont détectées à travers un historique qui représente une partie du flux télévisuel passé. Si cet historique est grand, le système devient capable de détecter des répétitions dont les occurrences sont très espacées dans le temps. Nous nous sommes donc intéressés à la durée de cet historique. Nous souhaitons plus particulièrement savoir si les inter-programmes sont répétés de manière concentrée ou de manière espacée.




(a) Pour France 2 (b) Pour TF1

FIGURE 2.18 – Proportion des inter-programmes rediffusés par catégorie dans un historique variable.

Pour déterminer la durée de l'historique, nous nous basons sur une étude de la proportion des inter-programmes répétés. Cette proportion est le nombre d'inter-programmes rediffusés sur le nombre total d'inter-programmes pour une durée donnée. La figure 2.18 donne les proportions obtenues par catégorie d'inter-prorgammes sur les chaînes France 2 et TF1 en fonction de la durée. Ces résultats ont été calculés en utilisant la vérité terrain.

Les courbes de la figure 2.18 valident notre hypothèse de base concernant cette thèse : les inter-programmes sont répétés majoritairement dans le flux. En effet, plus de 95 % des publicités et des parrainages sont rediffusés dans la semaine. Les bandes annonces sont aussi rediffusées à environ 90 % sur France 2 et à environ 70 % sur TF1 sur une semaine. Nous rappelons, de plus, que les publicités, les parrainages et les bandes annonces représentent la majeure partie des inter-programmes sur France 2 et sur TF1 (cf. les statistiques de notre corpus de l'annexe A).

Ces courbes nous permettent également de noter une stabilité à partir de 96 heures sur les deux chaînes. Cette stabilité implique qu'une durée d'historique supérieure à 96 heures ne permet pas d'augmenter radicalement la proportion d'inter-programmes rediffusés. Par conséquent, une durée d'historique supérieure à 96 heures ne permet pas de découvrir beaucoup plus de nouveaux inter-programmes rediffusés. Cela ne permet donc pas d'obtenir *a priori* un meilleur découpage grâce aux inter-programmes répétés. Il n'est donc pas utile de conserver un historique de plusieurs semaines. À partir de ces résultats, nous choisissons la taille de notre historique. Bien que nous n'ayons pas besoin d'un historique supérieur à 96 heures, nous utilisons un historique de 7 jours. Ces 7 jours permettent

de bien caractériser la répartition des occurrences des répétitions sur une semaine. Cette information nous sert dans le chapitre 3 suivant pour la classification des segments.

#### 2.5.4.2 Choix de la durée des portions de flux à traiter périodiquement

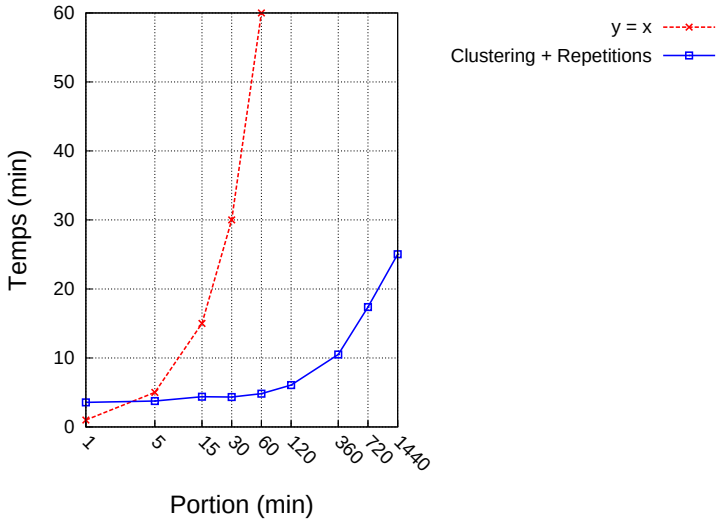



FIGURE 2.19 – Analyse des temps de calcul du premier niveau de délinéarisation en fonction de la taille des portions de flux à traiter pour un historique de 7 jours. L'échelle est logarithmique pour plus de visibilité.

Nous rappelons qu'un flux est traité périodiquement, c'est à dire portion par portion. Pour que notre solution fonctionne en continu, les temps de calculs nécessaires au traitement d'une portion doivent rester inférieur à la durée de cette portion. Nous évaluons, pour cela, les temps de calcul des opérations du *clustering* et de construction des répétitions pour une portion. Ces deux opérations sont les plus coûteuses. Les opérations restantes sont la description du flux et le découpage à partir des répétitions. La description du flux est effectuée en temps réel par une machine dédiée. Le découpage est une opération dont la complexité est négligeable par rapport au *clustering* et à la construction des répétitions.

Grâce à l'expérience précédente, nous utilisons un historique de 7 jours. Nous analysons, alors, dans cette expérience la taille possible des portions de flux à traiter sur cet historique. Lorsque la complexité temporelle pour traiter une portion de flux reste inférieure à la durée de cette portion, cela signifie que la solution peut traiter sur la même machine, de façon continue, des portions les unes après les autres.

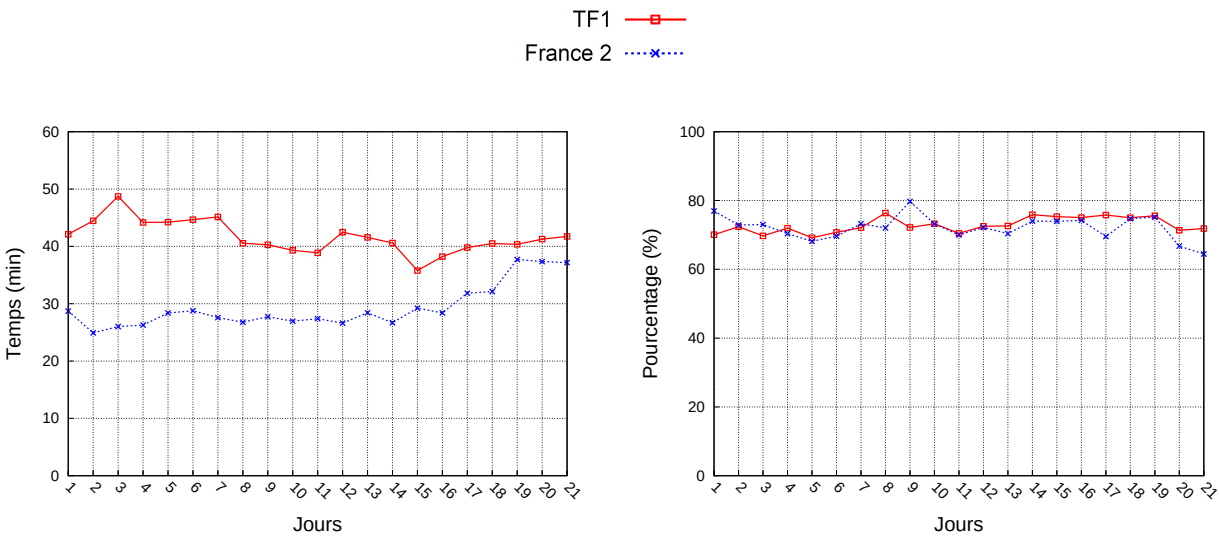
Nous faisons donc varier la taille de la portion de flux à traiter. Pour chaque taille, nous ajoutons cette portion à l'historique suivant notre méthode incrémentale présentée dans la section 2.3.5. La figure 2.19 montre les résultats.

Pour une portion de flux supérieure à 5 minutes, l'expérience montre que les temps de calcul sont considérablement plus courts que le temps total de la portion traitée. Cela signifie que ce niveau de délinéarisation peut facilement traiter en continu toute portion de

plus de 5 minutes pour un historique de 7 jours. Dans notre configuration, nous choisissons une portion de flux de 24 heures.

Dans le cadre d'un service de télévision à la demande, nous pouvons envisager d'appliquer ce premier niveau de délinéarisation tous les jours à minuit. Ce mode de fonctionnement permet d'extraire chaque jour tous les programmes TV de la veille. D'après les résultats, nous pouvons aussi imaginer appliquer le premier niveau de délinéarisation toutes les 10 minutes. Ce dernier mode de fonctionnement permet alors d'extraire un programme quelques minutes après sa fin de diffusion.

#### 2.5.4.3 Stabilité des traitements




(a) Évaluation des temps de calcul

(b) Évaluation du découpage (selon la F-mesure $F_{2s}$)

FIGURE 2.20 – Stabilité de notre système sur 3 semaines.

Les expériences précédentes nous ont permis de connaître les temps de calcul de ce premier niveau de délinéarisation. Le dernier point important de la continuité est la stabilité. Nous devons, en effet, nous assurer que les temps de calcul restent relativement constants malgré les flux TV ininterrompus. Pour estimer cette stabilité, nous avons appliqué ce premier niveau de délinéarisation sur 3 semaines à la fois sur France 2 et sur TF1. La figure 2.20 présente les résultats. Pour appliquer ce premier niveau de délinéarisation nous avons utilisé un historique de 7 jours initialisé avec la première semaine de notre corpus. Ensuite, nous avons continuellement mis à jour cet historique en ajoutant de manière incrémentale chaque jour des 3 semaines restantes de notre corpus.

Les résultats montrent que ce premier niveau de délinéarisation est globalement stable en temps de calcul sur les 3 semaines en considérant France 2 et TF1. L'évaluation du découpage selon la mesure $F_{2s}$ de notre protocole est aussi stable. L'écart entre le temps

maximum et le temps minimum de calcul est de seulement une dizaine de minutes. Les temps de calcul nécessaires sur France 2 sont plus courts que ceux nécessaires sur TF1. Cela est seulement dû au fait que nous avons exceptionnellement bénéficié d'une machine plus performante pour France 2. En ce qui concerne les différences de l'évaluation du découpage entre France 2 et TF1, une analyse détaillée est proposée dans la section 2.5.5 suivante.

### 2.5.5 Expériences 3 : étude de l'efficacité

Les expériences 1 et 2 précédentes permettent de fixer les différents paramètres de ce premier niveau de délinéarisation. Les expériences de cette section visent maintenant à approfondir les résultats obtenus avec les paramètres choisis. Pour cela, nous étudions plus précisément l'efficacité sur une semaine de flux TV à la fois sur France 2 et sur TF1 en utilisant un historique de 7 jours.

L'efficacité est la vérification que ce premier niveau de délinéarisation fournit des résultats attendus. Elle s'évalue grâce aux mesures d'évaluation que nous avons définies dans notre protocole d'évaluation de la section 2.5.2. Nous analysons donc l'efficacité de la détection des répétitions d'inter-programmes puis l'efficacité du découpage à partir des répétitions.

#### 2.5.5.1 Analyse de la détection des répétitions

Nous avons, tout d'abord, évalué la détection des répétitions suivant la F-mesure $F_{plan}$ sur la semaine de TF1 et la semaine de France 2. Nous avons utilisé les paramètres déterminés dans la section 2.5.3. Les résultats sont présentés dans le tableau 2.2. Les résultats de $F_{plan}$ sont cohérents avec ceux présentés précédemment dans les courbes de la section 2.5.3. Nous allons, à présent, les approfondir.

| | France 2 | TF1 |
|---|---|---|
| $F_{plan}$ | 65,28 % | 69,61 % |
| Nombre de répétitions détectées | 551 | 713 |
| Nombre total d'occurrences des répétitions | 2695 | 3675 |
| Nombre d'occurrences maximum dans une répétition | 29 | 26 |
| Type de la répétition la plus fréquente | Publicité | Publicité |

Tableau 2.2 – Évaluation de la détection des répétitions d'inter-programmes sur une semaine.

Le tableau 2.2 donne également, pour la semaine de France 2 et la semaine de TF1, le nombre de répétitions détectées, le nombre total d'occurrences de celles-ci, le nombre d'occurrences maximum dans une répétition et le type de celle-ci. Nous remarquons alors que TF1 contient plus de répétitions de manière générale. D'après les statistiques calculées en annexe A sur le corpus, la chaîne TF1 diffuse en réalité plus d'inter-programmes. Le flux de la chaîne TF1 contient donc naturellement plus de répétitions.

Nous avons ensuite évalué la précision $P_{rep}$ des répétitions détectées. Cette précision n'est pas la précision $P_{plan}$ des répétitions d'inter-programmes. $P_{rep}$ teste si toutes les répétitions détectées sont bien des répétitions. Nous avons choisi les 100 répétitions contenant le plus d'occurrences et 100 autres répétitions sélectionnées aléatoirement. Toutes les répétitions vérifiées à la main étaient de réelles répétitions. Nous estimons donc la précision $P_{rep}$ à environ 100 %. Toutes les répétitions détectées ne sont malheureusement pas des inter-programmes. Elles peuvent contenir des programmes entiers, des génériques ou des morceaux répétés de reportages. $P_{plan}$ vaut ainsi seulement $50,04$ % sur France 2 et $56,54$ % sur TF1.

Le rappel $R_{plan}$ est de $93,89$ % sur France 2 et de $90,52$ % sur TF1. Nous avons aussi calculé le rappel $R_{plan}$ par catégorie d'inter-programmes. Ces rappels sont présentés dans le tableau 2.3. Presque au moins 94 % des publicités et des bandes annonces répétées sont bien détectées par notre méthode de détection des répétitions. D'après l'annexe A, les publicités et les bandes annonces constituent environ 90 % des inter-programmes. L'expérience de la section 2.5.4.1 montre de plus que les publicités et les bandes annonces sont répétées en moyenne à plus de 95 % sur une semaine pour France 2 et TF1. Par conséquent, nous pouvons en déduire que nous détectons par notre méthode de détection des répétitions au moins 80 % $\approx$ 94 % $\times$ 90 % $\times$ 95 % de tous les inter-programmes simplement grâce aux répétitions.

| | France 2 | TF1 |
|---|---|---|
| Publicités | 95,19 % | 94,03 % |
| Bande-Annonces | 99,58 % | 93,94 % |
| Parrainages | 82,31 % | 68,30 % |

Tableau 2.3 – Rappel $R_{plan}$ par catégorie d'inter-programme sur une semaine.

Le tableau 2.3 montre aussi que les parrainages sont détectés avec un peu plus de difficultés. Ce phénomène est plus accentué sur TF1. Il provient de la nature des parrainages. En effet, un parrainage est majoritairement constitué d'un seul plan affichant un simple logo de l'entreprise marraine. Ce plan est par conséquent décrit par une unique image clé. La rediffusion d'un tel parrainage génère ainsi un seul *cluster* similaire à aucun autre *cluster*. Il n'est donc pas sélectionné lors de la création des groupes de *clusters* similaires.

De manière plus générale, les occurrences des répétitions proviennent au minimum de 2 *clusters* car elles sont construites à partir de groupes de *clusters* similaires. Les *clusters* similaires à aucun autre *cluster* ne génèrent donc pas d'occurrences. Pour remédier à cela, nous avons essayé d'étendre ces *clusters* appelés *clusters* « solitaires » dans la section 2.3.4.2. Cette extension particulière fonctionne comme la méthode d'extension proposée dans la section 2.3.4.3.

Le fait d'utiliser les *clusters* solitaires permet de détecter de nombreuses autres répétitions. La comparaison du nombre de répétitions détectées avec ou sans les *clusters* solitaires est donnée par le tableau 2.4. Les *clusters* solitaires permettent de détecter environ deux fois plus de répétitions. Nous avons mesuré manuellement, selon le même procédé expliqué plus haut, $P_{rep}$ sur ces nouvelles répétitions. Nous estimons toujours $P_{rep}$ à en-

| | France 2 | TF1 |
|---|---|---|
| Utilisation des *clusters* similaires seulement | 551 | 713 |
| Utilisation des *clusters* solitaires | 1064 | 1371 |

Tableau 2.4 – Nombre de répétitions obtenues sur une semaine avec ou sans utilisation des *clusters* solitaires.

viron 100 %. Nous avons aussi évalué la détection des répétitions d'inter-programmes en utilisant les *clusters* solitaires. Les résultats sont donnés dans le tableau 2.5. Le rappel global $R_{plan}$ est de $98,58$ % sur France 2 et de $96,64$ % sur TF1. Les résultats sont effectivement améliorés avec l'utilisation des *clusters* solitaires. Les répétitions constituées d'un seul plan sont mieux détectées. Il reste cependant des plans répétés qui ne sont pas détectés. Ces plans répétés non détectés sont principalement des transitions progressives. En effet, les frontières des plans au niveau des transitions progressives ne sont pas toujours bien détectées. Par conséquent, les images clés au sein de ces plans peuvent être décalées. Lorsque les images clés sont décalées, les *clusters* s'apparient mal avec d'autres *clusters* et les extensions échouent plus facilement.

| | France 2 | TF1 |
|---|---|---|
| Publicités | 98,62 % | 97,48 % |
| Bande-Annonces | 99,79 % | 96,33 % |
| Parrainages | 95,30 % | 88,69 % |

Tableau 2.5 – Rappel $R_{plan}$ avec les *clusters* solitaires.

L'augmentation du rappel annoncée s'associe aussi malheureusement à une diminution de la précision $P_{plan}$. En effet, $P_{plan}$ vaut alors $47,72$ % sur France 2 et $50,80$ % sur TF1. La F-mesure $F_{plan}$ de cette nouvelle méthode de détection des répétitions est donnée dans le tableau 2.6. Les écarts avec la F-mesure $F_{plan}$ du tableau 2.2 ne sont pas grands car la diminution de la précision est compensée par l'augmentation du rappel. Nous choisissons néanmoins de conserver notre méthode initiale de détection des répétitions qui n'emploie pas les *clusters* solitaires. C'est la méthode qui assure le meilleur compromis selon la F-mesure $F_{plan}$. Nous favorisons de plus la précision afin de ne pas sur-découper le flux avec le doublement du nombre de répétitions comme nous le montrons dans le tableau 2.7 de la section 2.5.5.2 suivante.

#### 2.5.5.2 Analyse du découpage à partir des répétitions

Le tableau 2.7 expose les résultats de l'évaluation selon la F-mesure $F_{2s}$ et le rappel $R_{2s}$ du découpage d'une semaine de flux à partir des répétitions. Ces résultats de $F_{2s}$ sont cohérents avec ceux des expériences précédentes. Ils contiennent également le nombre total de segments délimités et ils s'accompagnent, de plus, des résultats de l'évaluation du

| | France 2 | TF1 |
|---|---|---|
| $F_{plan}$ | 64,31 % | 66,59 % |

TABLEAU 2.6 – Évaluation de la détection des répétitions d'inter-programmes sur une semaine selon $F_{plan}$ avec les *clusters* solitaires.

découpage à partir des répétitions générées avec les *clusters* solitaires. Les *clusters* solitaires engendrent bien un sur-découpage du flux. Le sur-découpage se traduit par la création de segments trop nombreux. Bien que les bornes des programmes et des inter-programmes ont ainsi plus de chance d'être détectées, les segments issus d'un sur-découpage s'éloignent de la structure inhérente à la télévision. Ainsi, les segments sont plus difficiles à classer par la suite. Un exemple de sur-découpage évident est l'utilisation de la segmentation en plans.

| | France 2 | TF1 |
|---|---|---|
| Nombre de segments | 3764 | 4705 |
| $F_{2s}$ | 77,14 % | 80,54 % |
| $R_{2s}$ | 80,43 % | 76,77 % |

(a) Utilisation des *clusters* similaires seulement

| | France 2 | TF1 |
|---|---|---|
| Nombre de segments | 5549 | 6830 |
| $F_{2s}$ | 71,52 % | 77,99 % |
| $R_{2s}$ | 91,45 % | 87,69 % |

(b) Utilisation des *clusters* solitaires

TABLEAU 2.7 – Évaluation du découpage à partir des répétitions sur une semaine sur France 2 et sur TF1.

À la lecture des résultats du tableau 2.7, il apparaît que toutes les bornes des inter-programmes et des programmes ne sont pas parfaitement détectées et que certaines bornes supplémentaires sont indiquées. Les bornes supplémentaires sont dues aux répétitions qui ne sont pas des inter-programmes comme les génériques ou des morceaux répétés de reportages. Les segments issus de ces répétitions sont classés dans le chapitre 3 suivant. Pour les bornes qui ne sont pas détectées, celles-ci sont en partie dues aux inter-programmes non répétés diffusés à la suite. Mais ces bornes non détectées proviennent aussi des inter-programmes non répétés diffusés juste avant ou après un programme comme cela est illustré par la figure 2.21. Ces inter-programmes non répétés sont souvent des bandes annonces du type « *dans un instant ...* » qui indiquent les programmes proches à venir. Ce sont

des annonces précises à un horaire particulier. Elles ne sont pas répétées. Ces annonces possèdent néanmoins des caractéristiques très particulières comme la présence du texte « *dans un instant* ». Nous pouvons donc envisager dans nos travaux futures une technique spécifique pour les détecter.

Nous proposons d'évaluer le découpage à 20 secondes près, ce qui correspond à la durée moyenne d'un inter-programme selon notre annexe A. L'évaluation à 20 secondes près permet d'estimer l'impact des inter-programmes non répétés non détectés. Le tableau 2.8 donne les résultats suivant la F-mesure $F_{20s}$ et le rappel $R_{20s}$. Selon le tableau, plus de 95 % des bornes de programmes ou d'inter-programmes sont détectées à un inter-programme près. Par conséquent, notre méthode sera dans beaucoup de cas précise à un inter-programme près. Nous mesurons précisément l'impact de cette imprécision dans le chapitre 4.

| | France 2 | TF1 |
|---|---|---|
| $F_{20s}$ | 87,99 % | 94,91 % |
| $R_{20s}$ | 98,95 % | 95,52 % |

TABLEAU 2.8 – Évaluation du découpage à partir des répétitions sur une semaine selon $F_{20s}$.

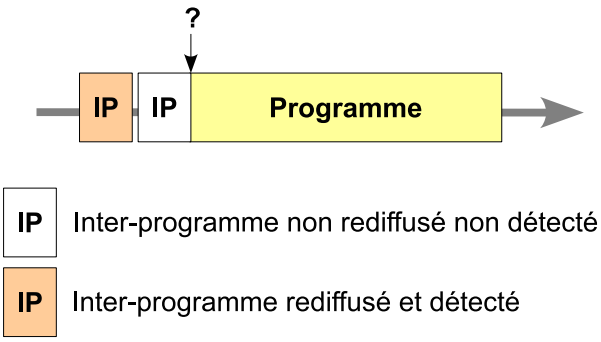



FIGURE 2.21 – Exemple de borne non détectée.

Nous comparons de plus ces résultats avec trois autres évaluations dans le tableau 2.9. La première comparaison évalue $F_{2s}$ la nuit de 1h du matin à 8h du matin. La nuit, les programmes TV sont parfois non séparés par des inter-programmes. Nous ne pouvons donc détecter toutes les bornes des programmes la nuit. Les résultats sont donc moins fiables. La seconde comparaison évalue à l'inverse $F_{2s}$ pendant la journée seulement, de 8h du matin à 1h dans la nuit. Les résultats sont, par opposition avec la nuit, plus précis. Enfin, notre troisième comparaison mesure les performances d'un découpage qui utilise les bornes des plans à la place des bornes des segments issus des répétitions. Les plans permettent d'obtenir pratiquement toutes les bornes. Le rappel $R_{2s}$ est donc proche de 100 %. Ce-

pendant les plans sur-découpent les flux télévisuels. La précision est donc excessivement basse et les résultats sont en dessous de 13 %.

| | France 2 | TF1 |
|---|---|---|
| La nuit | 60,91 % | 61,31 % |
| La journée | 80,65 % | 82,40 % |
| À partir des plans sur le jour entier | 10,12 % | 12,09 % |

TABLEAU 2.9 – Évaluation du découpage à partir des répétitions la nuit et la journée sur une semaine selon $F_{2s}$ et évaluation du découpage à partir des plans sur une semaine selon $F_{2s}$.

### 2.5.6 Synthèse

Les résultats obtenus par notre protocole d'évaluation nous apportent de précieuses informations relatives à ce premier niveau de délinéarisation.

Nous avons déterminé des paramètres pour ce premier niveau de délinéarisation qui ne dépendent ni de la chaîne ni de la quantité de flux à traiter. Ces paramètres peuvent donc être fixés. Nos méthodes de détection des répétitions et de découpage du flux à partir de ces répétitions sont donc complètement automatiques et génériques à ce stade. Nous avons également vérifié que ces méthodes peuvent s'appliquer sur des flux TV ininterrompus. Dans notre contexte, nous avons choisi d'utiliser un historique de 7 jours mis à jour par des traitements périodiques toutes les 24 heures.

Les expériences nous ont permis de valider nos hypothèses de bases. Nous pouvons utiliser les inter-programmes répétés pour découper le flux en segments. Les inter-programmes répétés peuvent être détectés avec de bonnes performances grâce à notre méthode de détection des répétitions basée sur un micro-*clustering* des images clés. Cependant, nous ne pouvons pas détecter les inter-programmes non répétés placés par exemple juste devant ou après un programme. Ce premier niveau de délinéarisation délimite donc les bornes des programmes et des inter-programmes à une borne d'inter-programme près seulement. Cette imprécision ne provient pas de notre méthode de détection des répétitions mais de l'utilisation des répétitions.

L'ensemble des segments que nous obtenons sont des inter-programmes répétés, des inter-programmes non répétés situés entre deux inter-programmes répétés, des parties de programmes non répétées et des parties de programmes répétées. Nous nous penchons, dans le chapitre 3 suivant, sur le tri de ces segments.

## 2.6 Exploitation de la détection des répétitions pour la détection de bandes annonces

Une bande annonce est un inter-programme particulier qui comme son nom l'indique annonce des programmes prochainement diffusés. De nombreuses bandes annonces

contiennent directement des morceaux des programmes qu'elles annoncent afin d'intéresser le téléspectateur. Cette particularité fait que certaines bandes annonces peuvent se déduire de l'analyse des répétitions. C'est le sujet de cette section. L'intérêt de détecter certaines bandes annonces est alors de faciliter la classification ultérieure des segments issus des répétitions.

La méthode que nous proposons pour la détection de certaines bandes annonces est issue de nos dernières recherches, nous ne l'avons pas incluse dans notre système de délinéarisation des flux TV.

### 2.6.1 Méthode de détection de bandes annonces

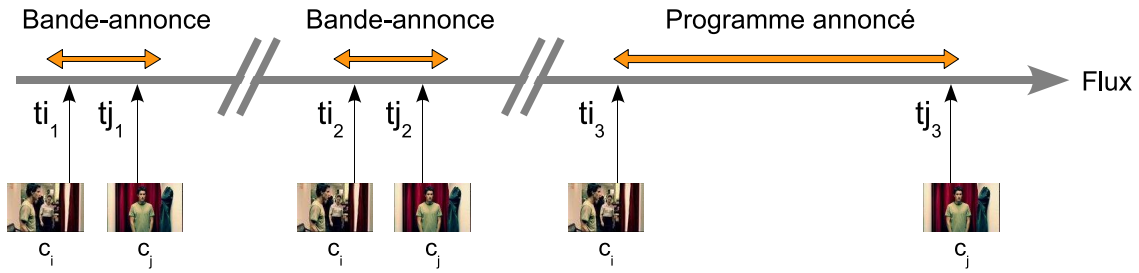



FIGURE 2.22 – Exemple de détection d'une bande annonce. Deux *clusters* $c_i$ et $c_j$ sont similaires sans les dernières images clés. Les dernières images clés des *clusters* $c_i$ et $c_j$ sont aussi beaucoup plus espacées que les autres images.

Lorsqu'une bande annonce reprend des images d'un programme pour l'annoncer, la bande annonce conserve, dans beaucoup de cas, les images dans leur ordre chronologique. Il en découle que les images clés d'une bande annonce peuvent réapparaitre dans le même ordre dans le programme annoncé. Ces images clés rediffusés dans le même ordre définissent des occurrences de répétitions. Cependant, l'écart entre les images clés est beaucoup plus important dans le programme que dans l'occurrence de la bande annonce comme cela est illustré dans la figure 2.22.

Lors de la phase de détection des répétitions, nous recherchons donc des *clusters* d'images clés tels que les premières images clés appartiennent à des occurrences d'une répétition et que la dernière image clé appartienne à un programme long. Pour tester si les premières images clés appartiennent à des occurrences d'une répétition, nous retirons la dernière image clé de chaque *cluster* et nous testons si les *clusters* sont similaires, au sens de la similarité définie dans la section 2.3.4.1. Nous utilisons donc la méthode usuelle sans prendre en compte la dernière image clé des *clusters*. Nous testons ensuite si les dernières images clés des *clusters* appartiennent à un programme long. C'est le cas si les images clés sont beaucoup plus espacées.

Pour pouvoir détecter les images clés communes entre les bandes annonces et les programmes annoncés, nous recherchons des images presque identiques. Nous autorisons en fait plus de variations entre les descripteurs $D_C$ car les bandes annonces contiennent souvent des incrustations textuelles. Pour appliquer cette technique, nous augmentons donc

le rayon maximal *Rmax* du *clustering*.

Les nouvelles répétitions que nous obtenons ne permettent pas de découper le flux avec de bons résultats car le rayon *Rmax* est trop élevé. Pour remédier à cela, nous conservons les répétitions calculées suivant les paramètres choisis dans les expériences de la section 2.5. Ces répétitions sont alors croisées avec les répétitions calculées pour les bandes annonces avec un rayon *Rmax* plus grand. Les répétitions initiales correspondent alors avec des répétitions calculées pour les bandes annonces. Lorsqu'une répétition calculée pour les bandes annonces est définie comme bande annonce alors sa correspondance est aussi définie comme bande annonce. Cette solution possède l'inconvénient de nécessiter deux *clustering*. Cela augmente les temps de calcul.

### 2.6.2 Expérimentation de la détection de bandes annonces

Nous ne proposons qu'une étude préliminaire de la détection de bandes annonces. Pour cela, nous avons limité notre corpus aux semaines n°1 et n°2 sur la chaîne France 2.

| Bandes annonces (BA) | Semaine n°1 | Semaine n°2 |
|---|---|---|
| BA détectées | 85 | 92 |
| BA dans la vérité terrain | 411 | 429 |

Tableau 2.10 – Détection de bandes annonces par la méthode d'analyse du *clustering*.

Les résultats de la détection sont présentés dans le tableau 2.10. Sur un total de 840 segments de bande annonce dans la vérité terrain, nous en avons détectées 177. Nous n'avons pas détecté de segments de bande annonce qui ne soient pas une vraie bande annonce. Cette méthode permet donc d'obtenir une précision de 100 % pour un rappel d'environ 21 %. Nous avons donc bien été capable d'extraire des bandes annonces à partir de l'exploitation de la détection des répétitions. L'intérêt de cette méthode est, suivant nos résultats préliminaires, qu'elle est très spécifique à certaines bandes annonces. Elle peut apporter ainsi des informations cruciales pour la classification.

# Chapitre 3

# Classification des segments

## Sommaire



Ce chapitre traite du second niveau de délinéarisation. Pour situer ce niveau, nous rappelons le fonctionnement général de notre approche de délinéarisation. Notre approche analyse des portions de flux. Ces portions sont découpées en segments à partir de la détection des répétitions. Les segments sont des occurrences de répétitions ou bien des morceaux de flux entre deux occurrences identifiées. Ils sont issus de répétitions détectées dans les portions et leur historique comme cela est illustré dans la figure 3.1 et expliqué dans la section 2.3.5. Au final, le but de ce chapitre est de proposer une méthode de classification des segments ainsi formés.

La classification des segments vise à différencier les occurrences des répétitions d'inter-programmes des autres occurrences de répétitions. Elle vise également à identifier les segments de programmes et les segments d'inter-programmes qui ne sont pas répétés. Elle permet même de détecter des segments de publicités, de bandes annonces ou de

parrainages. Cette classification a pour objectif de révéler ainsi la structure sous-jacente d'un flux en programmes et en inter-programmes.

Dans l'optique d'une délinéarisation automatique, notre méthode de classification est soumise aux contraintes de généricité, d'efficacité, d'automaticité et de continuité.

L'organisation de ce chapitre se décline en trois sections. La première section présente les méthodes existantes liées à la classification de segments audiovisuels. La deuxième section présente en détail notre technique de classification de segments. Cette technique est basée sur la programmation logique inductive qui est dans un premier temps expliquée. La troisième section donne les résultats de l'évaluation de notre méthode de classification de segments afin de classer les segments de programme et les segments d'inter-programmes. Cette section explore davantage notre méthode de classification pour classer les inter-programmes suivant leurs différentes catégories telles les publicités, les bandes annonces ou les parrainages.

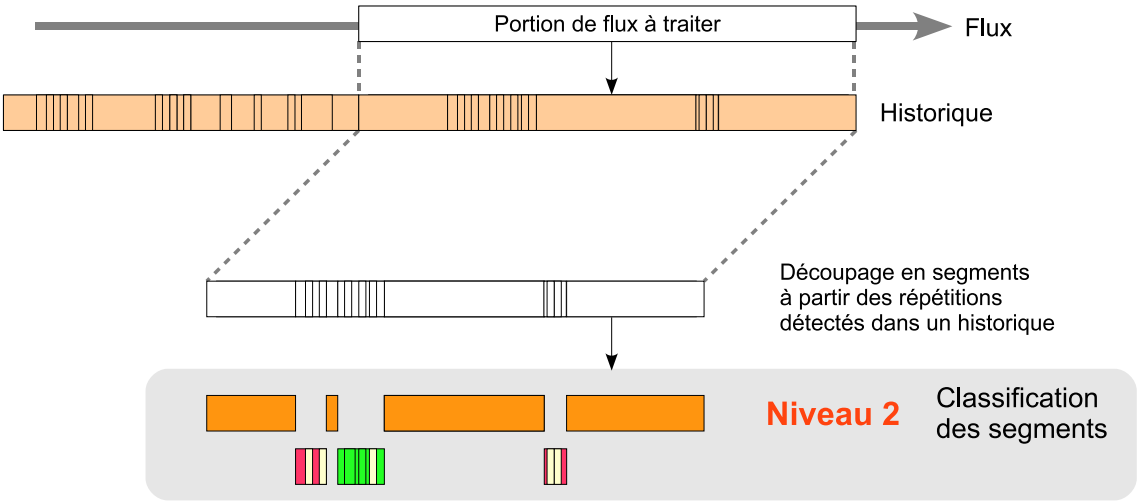



FIGURE 3.1 – Niveau 2 de délinéarisation : classification des segments.

## 3.1 Méthodes existantes pour la classification de segments audiovisuels

Nous nous intéressons principalement à la classification en genre des segments audiovisuels et, plus précisément, à la classification des segments d'un flux TV dans les genres « programmes » ou « inter-programmes ». Cette dernière tâche est réalisée naturellement par les téléspectateurs. Bien qu'il puisse y avoir quelques ambiguïtés visuelles entre certains programmes et certains inter-programmes, le contexte de diffusion des contenus aide les téléspectateurs. Ainsi, si un téléspectateur visionne une publicité au cours d'une émission d'analyse des publicités[1], il sait si cette publicité appartient ou non à l'émission.

1. Par exemple, l'émission hebdomadaire « Culture Pub » diffusée par la chaîne française M6 entre les années 1987 et 2005.

Le téléspectateur se base sur les jingles ou les annonces du présentateur. Il peut également reconnaître simplement une publicité habituelle. Les deux classes « programmes » et « inter-programmes » sont donc *a priori* des genres de diffusions bien définis.

Les caractéristiques telles la présence ou l'absence de logo, les images monochromes et les silences environnants, etc. décrivent les contextes dans lesquels les segments apparaissent. Néanmoins, nous avons déjà vu dans le chapitre 1 que les méthodes les plus efficaces pour la détection des inter-programmes n'emploient pas ces dernières caractéristiques. Celles-ci sont généralement trop spécifiques aux chaînes à traiter. Pour les mêmes raisons, il est difficile d'identifier si un segment appartient au genre « inter-programme » uniquement à partir de ces caractéristiques. La propriété de répétition des inter-programmes est généralement préférée à ces caractéristiques. Cependant, toutes les répétitions ne sont pas des inter-programmes. Pour détecter les inter-programmes parmi des occurrences de répétitions, les méthodes de détection des répétitions [CBF06, ZZZY08] définissent empiriquement des seuils sur la durée des occurrences des répétitions obtenues. Nous montrerons dans les expériences de la section 3.3 que ces seuils sur la durée ne suffisent pas à différencier un inter-programme d'une partie répétée d'un programme.

De manière plus générale, les principales techniques de classification automatique en genre de segments audiovisuels reposent sur des caractéristiques multimodales [BC08, TV07, MM08]. Ces caractéristiques multimodales sont souvent textuelles, auditives et/ou visuelles. Parmi les caractéristiques visuelles, les couleurs, les visages, les contours, les textures ou les mouvements sont souvent choisis. Pour modéliser les informations temporelles de ces caractéristiques dans les vidéos, les modèles de Markov cachés (HMM – *Hidden Markov Model*) sont particulièrement adaptés [DAW00, LDJ03]. Certaines méthodes [YSS02] apprennent des règles logiques de classification de type « Si ... alors ... ». Ces règles peuvent être construites à partir d'un arbre de décision appris grâce à l'algorithme C4.5 [Mit97]. Zhou *et al.* [ZDK02] utilisent de telles règles pour structurer des vidéos de basket. L'ensemble de ces approches reste cependant limité à un nombre fini de genres très particuliers de programmes ou d'inter-programmes. Parmi les genres considérés, nous retrouvons généralement la publicité, les journaux, les dessins-animés et le sport. Comme nous l'avons expliqué dans le chapitre 1, les publicités peuvent en effet posséder des caractéristiques très spécifiques. Les journaux sont aussi des émissions très particulières avec des enchaînements de plans présentateurs et de reportages. Les dessins animés possèdent des propriétés de couleurs et de formes très fortes. Enfin, les évènements sportifs se déroulent généralement autour d'un environnement statique comme un terrain de tennis ou de foot.

Contrairement à ces quelques genres très particuliers, le genre « programme » est trop hétérogène [RMX+02] pour être décrit simplement par un ensemble de caractéristiques multimodales. Ce sont des dessins animés, des séries comiques très courtes, des documentaires, des films, des séries classiques, des jeux de type télé réalité, des magazines, des concerts, du théâtre, des journaux, etc. Médiamétrie [2] [Pol07] et *The moving image genre form guide* [THL98] définissent plus d'une centaine de genres différents de programmes. Cette hétérogénéité fait que les contenus de programmes peuvent prendre pratiquement toutes les formes visuelles et auditives dont parfois celles des inter-programmes. Il existe, ainsi, des programmes qui contiennent des fragments ressemblant à des inter-programmes.

2. Médiamétrie est une société indépendante et reconnue, chargée de la mesure d'audience des médias audiovisuels français.

Une émission musicale peut, en effet, diffuser de nouveaux clips qui réapparaissent lors des publicités à propos des nouveaux albums. Nous pouvons trouver aussi des émissions sur les publicités ou quelques films dans le domaine de la publicité[3]. Les exemples des clips musicaux et des publicités sont rares mais ils existent. De manière inverse, certains inter-programmes contiennent des fragments de programmes. C'est le cas des bandes annonces qui présentent généralement des parties des programmes à venir. La présence de fragments de programmes dans les bandes annonces est un problème fréquent non négligeable. Les bandes annonces peuvent représenter jusqu'à 25 % (cf. annexe A) de l'ensemble des inter-programmes. Ce problème a d'ailleurs été soulevé par Naturel [Nat07].

La classification en programmes et en inter-programmes requiert donc des caractéristiques autres que des caractéristiques multimodales seules. Les autres caractéristiques recherchées peuvent être des caractéristiques relationnelles. Quelques travaux de classification de plans vidéo utilisent des règles représentant des relations entre les caractéristiques multimodales. Snoek et Worring [SW05b] définissent des relations temporelles entre les segments issus de l'audio, de la video ou du texte. Les relations temporelles décrivent si un segment A précède, chevauche ou appartient à un autre segment B. Ces relations sont introduites dans quelques systèmes d'apprentissage (C4.5 ou SVM – *Support Vector Machine*) pour classer des plans vidéo en différents évènements. Les auteurs montrent que ces relations temporelles améliorent l'utilisation des caractéristiques multimodales. Dorado *et al.* [DCI04] utilisent la logique floue pour décrire des caractéristiques audiovisuelles sous la forme de caractéristiques logiques. Ces caractéristiques logiques sont ensuite liées à des concepts sémantiques (présence d'une personne, reportage, etc. ). Pour cela des règles d'associations sont apprises automatiquement. Elles permettent de construire des règles logiques simples pour classer les plans des journaux télévisés en plans « présentateur » et en plans « reportages ». Suivant le même objectif, Carrive [Car00] définit d'abord une hiérarchie de concepts sémantiques. Ces concepts déterminent par exemple la présence d'un visage en gros plan, la présence de parole ou la présence de sous-titres sur l'image. Il établit ensuite des relations temporelles entre les différents segments vidéo et audio. Ces concepts et ces relations temporelles lui permettent de définir manuellement des règles logiques pour classer les plans vidéo. Un plan interview est un segment de parole qui chevauche un segment d'un visage en gros plan. Ces dernières méthodes classent des segments de plans à partir de relations parfois logiques et parfois temporelles entre les caractéristiques audiovisuelles. Leurs applications traitent cependant de la structuration **intra**-programme. Elles ne sont pas directement adaptées à notre sujet.

La classification de segments audiovisuels ne se limite pas à la classification en genre des segments. Par exemple, dans le domaine de l'indexation, des méthodes [GPR05, SCN+05] classent les segments suivant ce qu'ils contiennent, par exemple des paysages, des voitures, des humains, de la météo, etc. Ces méthodes définissent pour cela des concepts sémantiques à partir de caractéristiques multimodales. Cependant, ces concepts permettent seulement de décrire les contenus des segments audiovisuels.

Par rapport à ces travaux, nous proposons, tout d'abord, dans notre approche d'utiliser d'autres caractéristiques comme les propriétés des distributions des occurrences des répétitions. Nous ajoutons, ensuite, à ces propriétés des dépendances « contextuelles » et « relationnelles ». Toutes ces nouvelles caractéristiques sont modélisées et utilisées à tra-

3. Nous pouvons citer le film « *99 francs* » réalisé par Jan Kounen en 2007.

vers des règles logiques apprises automatiquement. Au final, nous appliquons ces règles pour la classification en genre des segments. Notre méthode est expliquée en détail dans la section suivante.

## 3.2 Notre méthode de classification

Nous cherchons à classer des segments audiovisuels en segments de programme et en segments d'inter-programme. Nous supposons pour cela que les segments des deux classes peuvent être classés principalement grâce à leur contexte, sans se baser sur leur caractéristiques visuelles ou auditives. Nous souhaitons donc utiliser les propriétés des segments voisins et des segments liés par des relations quelconques. Pour cela, nous nous basons sur les informations apportées par les répétitions que nous avons préalablement détectées. Nous définissons ainsi des informations « contextuelles » et « relationnelles ».

Nous pouvons associer aux segments découpés des propriétés issues des répétitions. Ces propriétés peuvent être le nombre d'occurrences des répétitions, l'écart moyen entre les occurrences de chaque répétition ou d'autres propriétés caractérisant les distributions des occurrences des répétitions. L'étude des occurrences des répétitions permet de définir des règles pour distinguer certains segments. Par exemple, un segment « répété », c'est à dire issu d'une occurrence d'une répétition, est un inter-programme et, plus précisément, une publicité si la répétition de laquelle il est issu contient beaucoup d'occurrences et si ces occurrences s'étalent tout au long des journées. Un autre exemple de règle est la classification d'un segment en programme lorsque le segment est répété tous les jours presque à la même heure. Cela correspond, en effet, aux génériques des programmes. Des règles peuvent aussi être définies sur des segments non répétés grâce à leur contexte. Ainsi, les segments courts entourés de deux segments préalablement classés comme des inter-programmes sont aussi des inter-programmes. De plus, suivant la connaissance *a priori* dont nous disposons sur les chaînes de télévision que nous souhaitons traiter, nous pouvons établir d'autres règles. Par exemple, nous savons que pour les chaînes européennes il n'y aura pas plus de 12 minutes de publicités par heure glissante [LPE07].

Nous pouvons donc déjà établir plusieurs règles simples de classification des segments à partir de leurs répétitions, de leur contexte et de leurs relations. Cependant, la construction manuelle de ces règles de classification n'est pas évidente. Il peut exister des règles non intuitives mais très utiles qui prennent en compte les caractéristiques ou les classes connues des segments voisins. De plus, les règles sont établies pour chaque nouvelle chaîne à traiter. Notre stratégie consiste donc à apprendre automatiquement ces règles de classification. L'outil que nous choisissons d'utiliser pour cela est la programmation logique inductive (PLI ou ILP – *Inductive Logic Programming*).

Dans la section suivante, nous présentons, plus précisément, notre méthode globale. Nous définissons, ensuite, des notions de bases et les règles de classification. Enfin, nous détaillons chaque module qui compose notre méthode.

### 3.2.1 Présentation générale

Notre méthode de classification automatique repose sur un apprentissage supervisé de règles de classification. Nous utilisons pour cela un ensemble d'apprentissage. Cet ensemble d'apprentissage est un découpage du flux en segments dans lequel chaque segment a été

manuellement classé. Dans notre cas, les classes sont les genres télévisuels des segments. Nous utilisons principalement la classe des programmes et la classe des inter-programmes mais d'autres classes peuvent être utilisées. À partir de l'ensemble d'apprentissage, des règles de classification sont apprises. Ces règles de classification sont de la forme suivante : « Si le segment $\mathtt{S_A}$ est ... alors $\mathtt{S_A}$ est un ... ». Ces règles de classification apprises sont, ensuite, appliquées automatiquement sur les segments du flux continu. Notre méthode se compose donc d'une partie « hors-ligne » d'apprentissage des règles et d'une partie « en-ligne » d'application des règles. La classification est dite en-ligne et automatique car elle s'effectue en continu pour chaque portion du flux analysé. L'apprentissage est dit hors-ligne par opposition. Il ne s'effectue qu'une seule fois pour configurer notre méthode.

En réalité, notre méthode permet de découvrir la structure du flux. Les règles de classification expriment les agencements des segments entre eux. Elles montrent, par exemple, dans quels contextes apparaissent les segments de programme et les segments d'inter-programme. Ces règles de classification traduisent donc la structure sous-jacente du flux. L'existence de la structure sous-jacente provient de la nature même des flux. L'objectif de notre méthode se concentre alors sur l'apprentissage d'une structure pérenne. La pérennité assure en effet que les règles de classification apprises restent valables longtemps et qu'elles ne sont pas limitées à l'ensemble d'apprentissage. Les travaux menés par Poli [Pol07] montrent la stabilité des horaires de diffusion des programmes sur le long terme. Cette stabilité dévoile une structure pérenne au niveau des programmes. Nous nous basons donc sur ces résultats pour rechercher une structure pérenne au niveau des programmes et des inter-programmes.

Nous utilisons la programmation logique inductive (PLI) pour réaliser l'apprentissage automatique des règles de classification et, par extension, de notre structure. Le principal avantage de la PLI est l'utilisation de la logique des prédicats [Ver01]. Cette logique permet de représenter des relations complexes et variées. La PLI est ainsi souvent utilisée dans des problèmes de modélisation de structures comme par exemple pour des molécules chimiques [KSS95], pour le traitement du langage [Cla03] ou pour les rythmes cardio-vasculaires [Fro05].

Le fonctionnement global de notre méthode de classification est illustré dans le schéma général de la figure 3.2. Il repose sur trois modules :

1) **le module de description logique.** Il décrit l'ensemble d'apprentissage dans une représentation logique constituée de prédicats. Il décrit également le flux TV, découpé automatiquement en segments, dans le même langage de représentation logique ;

2) **le module d'apprentissage.** Il produit un ensemble de règles de classification à partir de la lecture des prédicats de l'ensemble d'apprentissage ;

3) **le module de classification.** Il utilise les règles de classification apprises et les prédicats issus des segments du flux pour classer les segments.

Nous remarquons dans la figure 3.2 la possibilité de rajouter manuellement des règles spécifiques à partir de connaissances *a priori* sur le flux. Ces règles supplémentaires permettent de compléter l'ensemble des règles apprises automatiquement pour faire face à des situations particulières. Elles ne sont cependant pas obligatoires.

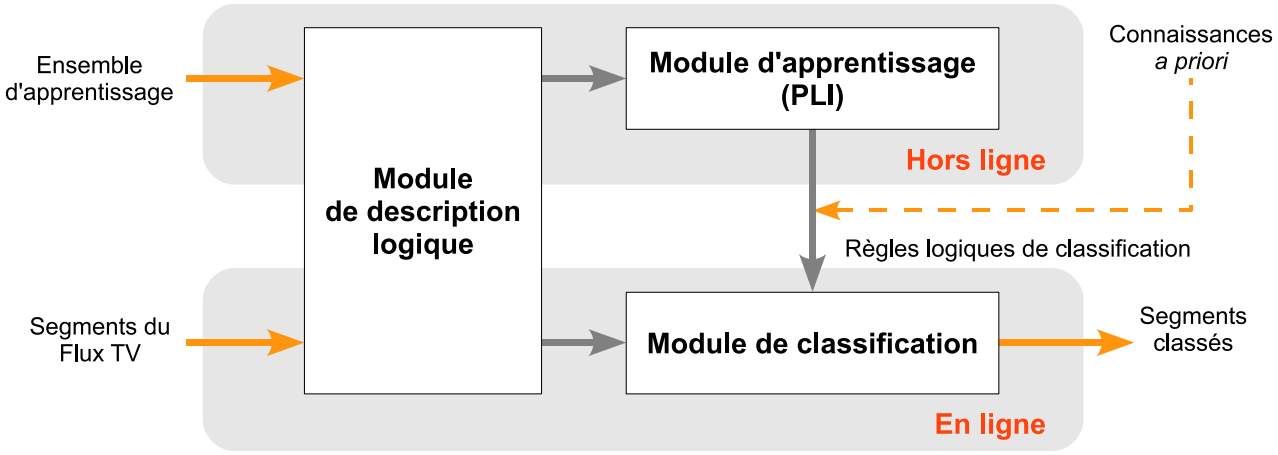



FIGURE 3.2 – Schéma général de notre méthode de classification automatique des segments du flux.

### 3.2.2 Notions de base

Notre méthode repose sur l'utilisation de la logique des prédicats. Le lecteur trouvera dans [Ver01] les définitions générales d'un prédicat, d'une variable, d'un littéral, d'une formule, d'un quantificateur $(\exists, \forall)$, d'une disjonction $(\vee)$, d'une conjonction $(\wedge)$, d'une négation $(\neg)$, etc. Dans notre approche, nous utilisons des prédicats particuliers appelés prédicats de segment. Ces prédicats de segment permettent d'exprimer les règles de classification.

#### 3.2.2.1 Les prédicats de segment

Les prédicats de segment représentent des propriétés vraies ou fausses pour un ou plusieurs segments. Un exemple de prédicat de segment simple est le prédicat qui informe que le segment $\mathtt{S_A}$ dure entre $27,5$ secondes et $32,5$ secondes :

$$\mathtt{Duree}(\mathtt{S_A}, [27,5\ \mathtt{secondes},\ 32,5\ \mathtt{secondes}[).$$

De manière plus précise, les prédicats de segment que nous manipulons sont des fonctions propositionnelles d'un ensemble fini de symboles et de variables dans l'ensemble binaire $\{0,1\}$ ou {VRAI, FAUX}. Les variables de ces prédicats sont des segments. Les symboles et les variables sont les arguments des prédicats de segment. Chaque prédicat de segment possède un nom. Les noms et les symboles servent à représenter les significations des propriétés des prédicats. Dans l'exemple, « $[27,5\ \mathtt{secondes}, 32,5\ \mathtt{secondes}[$ » est le symbole qui représente l'intervalle de temps compris entre $27,5$ secondes et $32,5$ secondes. Nous utilisons un nombre fini de symboles différents.

L'instanciation d'un prédicat de segment est la détermination de la valeur « VRAI » ou « FAUX » du prédicat en fonction de ses arguments. Les prédicats de segment peuvent alors se définir de deux manières : en extension ou en intension. Un prédicat de segment défini en extension est décrit par l'ensemble des instanciations possibles de ce prédicat

pour lesquelles le prédicat est « VRAI ». Un prédicat de segment défini en intension est décrit par une combinaison de quantificateurs, de disjonctions, de conjonctions et/ou de négations d'autres prédicats.

Un exemple de prédicat de segment $\mathtt{F_1}$ défini en extension pour un ensemble de segments $\mathtt{S_A}$, $\mathtt{S_B}$, $\mathtt{S_C}$ et $\mathtt{S_D}$ est :

$$\mathtt{F_1}(\mathtt{S_A}).$$
$$\mathtt{F_1}(\mathtt{S_D}).$$

Un exemple de prédicat de segment $\mathtt{F_2}$ défini en intension par 2 variables $\mathtt{X}$ et $\mathtt{Y}$ est :

$$\mathtt{F_2}(\mathtt{X},\mathtt{Y}) \leftarrow \mathtt{F_1}(\mathtt{X}) \wedge \mathtt{F_1}(\mathtt{Y}), \neg(\mathtt{X} = \mathtt{Y}).$$

#### 3.2.2.2 Les règles de classification

Nous définissons un prédicat de classe comme un prédicat de segment à un seul argument. Ce prédicat de classe détermine l'appartenance d'un segment $\mathtt{S_A}$ à un genre télévisuel. Par exemple, les prédicats de classe $\mathtt{Programme(S_A)}$ (noté $\mathtt{P(S_A)}$) et $\mathtt{Inter\text{-}programme(S_A)}$ (noté $\mathtt{IP(S_A)}$) expriment le fait qu'un segment $\mathtt{S_A}$ appartient au genre des programmes ou au genre des inter-programmes.

Une règle de classification est une règle de la forme « Si ... alors ... ». Dans la logique des prédicats, les règles s'expriment sous la forme d'implications logiques. Les règles de classification sont, au final, des prédicats de classe définis en intension.

L'application d'une règle de classification sur un segment $\mathtt{S_A}$ consiste à tester la valeur du prédicat de classe de la règle avec le segment $\mathtt{S_A}$ comme argument. Si la valeur est « VRAI » le segment est dit classé par la règle dans la classe de la règle et la règle est dite vraie sur le segment $\mathtt{S_A}$.

Un exemple de règle de classification est la règle qui signifie qu'un segment est un inter-programme si le segment se répète plus de 16 fois et que les occurrences des répétitions s'étalent sur 7 jours. Cette règle s'écrit sous la forme suivante :

$$\underbrace{\mathtt{Inter\text{-}Programme(X)}}_{\text{Prédicat de classe de tête}} \leftarrow \underbrace{\mathtt{NbRepetition}(\mathtt{X}, [16,\ \infty[), \mathtt{NombreDeJours}(\mathtt{X}, 7)}_{\text{Prédicats de corps}}.$$

« $[16,\ \infty[$ » et « $7$ » sont des symboles. « $\mathtt{NbRepetition}$ » et « $\mathtt{NombreDeJours}$ » sont des prédicats de segment. La règle de classification possède un prédicat de classe de tête et des prédicats de corps. Le prédicat de classe de tête représente la classe de la règle de classification. Les prédicats de corps correspondent aux prédicats utilisés dans la définition en intension de la règle de classification. Un prédicat de classe étant un prédicat, il peut apparaître parmi des prédicats de corps.

### 3.2.3 Module de description logique

L'ensemble d'apprentissage et les segments du flux sont tous deux des ensembles de segments. La description logique sert à décrire ces segments en entrée selon une même représentation afin d'apprendre des règles sur l'ensemble d'apprentissage qui pourront être par la suite appliquées sur les segments du flux.

*% Définitions des durées des segments*
Duree($S_A$, [27,5 secondes, 32,5 secondes[).
Duree($S_B$, [7,5 secondes, 12,5 secondes[).
Duree($S_C$, [12,5 secondes, 17,5 secondes[).
Duree($S_D$, [27,5 secondes, 32,5 secondes[).

*% Définitions du nombre de répétition des segments*
NbRepetition($S_A$, 3).
NbRepetition($S_B$, 1).
NbRepetition($S_C$, 1).
NbRepetition($S_D$, 5).

*% Définitions du voisinage*
Position($S_A$, $S_B$, 1).
Position($S_B$, $S_C$, 1).
Position($S_C$, $S_D$, 1).
Position(X, Y, N) ← Position(X, Z, 1), Position(Z, Y, N), N = M + 1.

FIGURE 3.3 – Exemple de description logique.

La représentation utilisée est un ensemble de prédicats de segments définis en extension ou en intension. Un exemple de description logique est donné dans la figure 3.3. Cette description comporte un ensemble d'instanciations des prédicats de segment `Duree, NbRepetition et Position` définis en extension et le prédicat de segment `Position` défini en intension.

Nous présentons et listons à présents tous les types de prédicats de segment de notre approche employés pour la description logique. Nous séparons les prédicats en trois types : les prédicats simples, les prédicats relationnels, les prédicats contextuels.

En plus de ces types, le module de description utilise également les prédicats de classe. Les prédicats de classe sont particuliers. Ils sont définis en extension dans l'ensemble d'apprentissage pour l'apprentissage. Ils sont ensuite définis en intension à partir de règles de classification. Pour les segments issus d'un découpage du flux, les instanciations des prédicats de classe sont découvertes progressivement par application des règles de classification qui fournissent les genres des segments.

#### 3.2.3.1 Prédicats simples

Les prédicats simples représentent des caractéristiques propres d'un segment. Certaines caractéristiques décrivent les distributions des occurrences des répétitions. Nous rappelons que les répétitions sont détectées en plaçant la portion de flux analysée dans un historique.

Pour le calcul des caractéristiques, nous considérons alors toujours les occurrences de chaque répétition par rapport à l'historique entier. Nous ne limitons pas les occurrences de chaque répétition à la portion de flux sur laquelle la classification est effectuée.

Les prédicats simples sont tous définis en extension. Ils sont décrit dans la suite de cette section.

***La durée***

La durée peut varier de quelques millisecondes à plusieurs heures pour un segment de programme. Pour représenter la durée dans un formalisme logique, nous utilisons plusieurs intervalles. Les intervalles choisis sont d'abord étroits pour représenter les petits segments puis plus larges pour les segments longs. Pour un intervalle $\mathtt{I}$ et un segment $\mathtt{S_A}$, nous obtenons le prédicat simple défini en extension :

$$\mathtt{Duree}(\mathtt{S_A},\ \mathtt{I}) \equiv \text{ « La durée de } \mathtt{S_A} \text{ appartient à } \mathtt{I}. \text{ »}$$

Nous utilisons les intervalles suivants :

$[\mathtt{0\ secondes},\ \mathtt{2,5\ secondes}[$, $[\mathtt{2,5\ secondes},\ \mathtt{7,5\ secondes}[$,
$[\mathtt{7,5\ secondes},\ \mathtt{12,5\ secondes}[$, $[\mathtt{12,5\ secondes},\ \mathtt{17,5\ secondes}[$,
$[\mathtt{17,5\ secondes},\ \mathtt{22,5\ secondes}[$, $[\mathtt{22,5\ secondes},\ \mathtt{27,5\ secondes}[$,
$[\mathtt{27,5\ secondes},\ \mathtt{32,5\ secondes}[$, $[\mathtt{32,5\ secondes},\ \mathtt{37,5\ secondes}[$,
$[\mathtt{37,5\ secondes},\ \mathtt{42,5secondes}[$, $[\mathtt{42,5\ secondes},\ \mathtt{47,5\ secondes}[$,
$[\mathtt{42,5\ secondes},\ \mathtt{75\ secondes}\ (\mathtt{1,25\ minutes})[$, $[\mathtt{1,25\ minutes},\ \mathtt{1,75\ minutes}[$,
$[\mathtt{1,75\ minutes},\ \mathtt{2,75\ minutes}[$, $[\mathtt{2,75\ minutes},\ \mathtt{7,5\ minutes}[$ et $[\mathtt{7,5\ minutes},\ \infty[$.

Ces intervalles sont majoritairement centrés sur des multiples de 5 secondes. Ils ont été choisis suite à une étude empirique de la distribution des durées des segments. Cette étude a été, de plus, combinée avec notre connaissance *a priori* des flux TV.

***Le nombre d'occurrences de la répétition***

Si le segment est répété, c'est-à-dire issu d'une occurrence d'une répétition, ce nombre est le nombre d'occurrences de la répétition calculée dans l'historique entier, comme expliqué plus haut. Si le segment n'est pas issu d'une occurrence d'une répétition, ce nombre vaut 1. Pour un intervalle $\mathtt{I}$ et un segment $\mathtt{S_A}$, nous obtenons le prédicat simple défini en extension :

$$\mathtt{NbRepetition}(\mathtt{S_A},\ \mathtt{I}) \equiv \begin{array}{l}\text{« Le nombre d'occurrences de la répétition de } \mathtt{S_A} \\ \text{appartient à } \mathtt{I}. \text{ »}\end{array}$$

Nous utilisons les intervalles et singletons suivants :

$\{\mathtt{1}\}$, $\{\mathtt{2}\}$, $\{\mathtt{3}\}$, $\{\mathtt{4}\}$, $\{\mathtt{5}\}$, $\{\mathtt{6}\}$, $\{\mathtt{7}\}$, $[\mathtt{8},\ \mathtt{15}]$ et $[\mathtt{16},\ \infty[$.

***La fréquence des plans***

La fréquence des plans est la moyenne par minute du nombre de changements de plan sur l'ensemble du segment. Pour un intervalle $\mathtt{I}$ et un segment $\mathtt{S_A}$, nous obtenons le prédicat

simple défini en extension :

$$\texttt{PlansParMinute}(\texttt{S}_\texttt{A},\ \texttt{I}) \equiv \begin{array}{l}\text{« La fréquence des plans en une minute de}\\ \texttt{S}_\texttt{A} \text{ appartient à } \texttt{I}.\text{ »}\end{array}$$

Nous utilisons les intervalles suivants :

$[0,\ 25[$, $[25,\ 35[$, $[35,\ 55[$ et $[55,\ \infty[$.

Pour établir ces intervalles, nous avons calculé la fréquence des plans par minute d'un grand nombre de segments de programmes et d'inter-programmes.

### *Le nombre de jours*

Les occurrences de la répétition d'un segment répété peuvent se répartir sur plusieurs jours dans l'historique. Pour un intervalle I et un segment $\texttt{S}_\texttt{A}$, nous obtenons le prédicat simple défini en extension :

$$\texttt{NombreDeJours}(\texttt{S}_\texttt{A},\ \texttt{I}) \equiv \begin{array}{l}\text{« Le nombre de jours entre la première oc-}\\ \text{currence et la dernière occurrence de la ré-}\\ \text{pétition associée à } \texttt{S}_\texttt{A} \text{ appartient à } \texttt{I}.\text{ »}\end{array}$$

Nous utilisons les intervalles et singletons suivants :

$\{1\}$, $\{2\}$, $\{3\}$, $\{4\}$, $\{5\}$, $\{6\}$, $\{7\}$, $[8,\ \infty[$.

### *Le nombre de jours consécutifs*

Les occurrences de la répétition d'un segment répété peuvent aussi se répartir sur des jours consécutifs dans l'historique. Pour un intervalle I et un segment $\texttt{S}_\texttt{A}$, nous obtenons le prédicat simple défini en extension :

$$\texttt{NombreDeJoursConsecutifs}(\texttt{S}_\texttt{A},\ \texttt{I}) \equiv \begin{array}{l}\text{« Le plus grand nombre de jours consécu-}\\ \text{tifs contenant au moins une occurrence de}\\ \text{la répétition associée à } \texttt{S}_\texttt{A} \text{ appartient à } \texttt{I}.\text{ »}\end{array}$$

Nous utilisons les mêmes intervalles et singletons que pour le prédicat NombreDeJours.

### *Les jours de début de semaine*

Les occurrences de la répétition d'un segment répété peuvent se répartir sur des jours qui peuvent être uniquement des jours de début de semaine : lundi, mardi, mercredi, jeudi et vendredi. Pour un segment $\texttt{S}_\texttt{A}$, nous obtenons le prédicat simple défini en extension :

$$\texttt{JoursDeSemaine}(\texttt{S}_\texttt{A}) \equiv \begin{array}{l}\text{« Les occurrences de la répétition associée}\\ \text{à } \texttt{S}_\texttt{A} \text{ se répartissent uniquement sur des}\\ \text{jours de début de semaine. »}\end{array}$$

***La diffusion une seule fois par jour***

Les occurrences de la répétition d'un segment répété peuvent se répartir sur un nombre de jours égal exactement au nombre d'occurrences de la répétition. En d'autres termes, le contenu répété de la répétition peut être diffusé une seule fois par jour. Pour un segment $\mathtt{S_A}$, nous obtenons le prédicat simple défini en extension :

$$\mathtt{RepetitionEgalJours}(\mathtt{S_A}) \equiv \begin{array}{l}\text{« Les occurrences de la répétition associée} \\ \text{à } \mathtt{S_A} \text{ se répartissent sur un nombre de jours} \\ \text{égal exactement au nombre d'occurrences} \\ \text{de la répétition. »}\end{array}$$

***La localisation***

La borne inférieure d'une occurrence de la répétition d'un segment répété apparaît temporellement dans une heure d'une journée. Cette heure est notée $h_j$.

La localisation est la moyenne des heures $h_j$ des occurrences de la répétition d'un segment répété. Pour un intervalle `I` et un segment $\mathtt{S_A}$, nous obtenons le prédicat simple défini en extension :

$$\mathtt{Localisation}(\mathtt{S_A},\ \mathtt{I}) \equiv \text{« La localisation de } \mathtt{S_A} \text{ appartient à } \mathtt{I}. \text{ »}$$

Nous utilisons les intervalles suivants :

[`6h`, `9h30`[, [`9h30`, `13h`[, [`13h`, `16h`[, [`16h`, `20h`[ et [`20h`, `6h`[.

***L'écart moyen de localisation***

Les heures $h_j$ des occurrences de la répétition d'un segment répété s'étalent entre une heure $h_{min}$ et une heure $h_{max}$. L'écart moyen de localisation est donc le rapport :

$$\frac{|h_{min} - h_{max}|}{\text{Nombre d'occurrences de la repetition } - 1}$$

Pour un intervalle `I` et un segment $\mathtt{S_A}$, nous obtenons le prédicat simple défini en extension :

$$\mathtt{EcartLocalisation}(\mathtt{S_A},\ \mathtt{I}) \equiv \begin{array}{l}\text{« L'écart moyen de localisation de } \mathtt{S_A} \\ \text{appartient à } \mathtt{I}. \text{ »}\end{array}$$

Nous utilisons les intervalles suivants :

{`0`}, ]`0`, `10 minutes`[, [`10`, `30 minutes`[, [`30`, `45 minutes`[,
[`45`, `75 minutes`[, [`75`, `180 minutes`[ et [`180 minutes`, `24 heures`].

#### 3.2.3.2 Prédicats relationnels

Les prédicats relationnels sont des prédicats qui définissent des liens entre les segments. Ils sont définis à la fois en extension et en intension. Avant de présenter les prédicats relationnels, nous expliquons la notion de voisinage.

### *Définition du voisinage*

L'ensemble d'apprentissage issu de la vérité terrain peut se représenter par une suite de segments. De manière similaire, le découpage du flux TV fournit une suite de segments. Ces suites de segments sont illustrées dans la figure 3.4. Un segment quelconque $\mathtt{S}_0$ dans une suite de segments est entouré de segments dits « voisins ». La position entre un segment $\mathtt{S}_a$ et un segment voisin $\mathtt{S}_b$ est le nombre de segments existant après $\mathtt{S}_a$ avant d'atteindre le segment $\mathtt{S}_b$. Dans la figure 3.4, le segment $\mathtt{S}_n$ est à une position $n$ du segment $\mathtt{S}_0$. La position peut être négative.

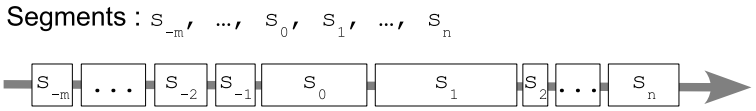



FIGURE 3.4 – Segments voisins du flux. Le segment $\mathtt{S}_n$ est à une position $n$ du segment $\mathtt{S}_0$.

### *La position*

Le prédicat de position définit les relations de voisinages qui existent entre les segments comme cela est illustré dans la figure 3.5. Toutes les positions ne peuvent pas être exprimées en extension. Cela entraîne trop de possibilités. Par conséquent, seuls les segments adjacents sont indiqués. Pour deux segments $\mathtt{S}_A$ et $\mathtt{S}_B$, nous obtenons ainsi le prédicat relationnel défini en extension :

$$\mathtt{Position}(\mathtt{S}_A, \mathtt{S}_B, 1) \equiv \text{ « Le segment } \mathtt{S}_B \text{ est à une position 1 de } \mathtt{S}_A. \text{ »}$$

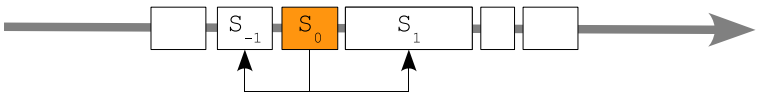



FIGURE 3.5 – Relation de voisinage.

Pour considérer l'ensemble des positions possibles entre les segments, nous utilisons aussi le prédicat relationnel défini en intension :

$$\mathtt{Position}(\mathtt{X}, \mathtt{Y}, \mathtt{N}) \leftarrow \mathtt{Position}(\mathtt{X}, \mathtt{Z}, 1), \mathtt{Position}(\mathtt{Z}, \mathtt{Y}, \mathtt{M}), \mathtt{N} = \mathtt{M} + 1.$$
$$\mathtt{Position}(\mathtt{X}, \mathtt{Y}, -\mathtt{N}) \leftarrow \mathtt{Position}(\mathtt{Y}, \mathtt{X}, \mathtt{N})$$

### *L'occurrence*

Nous utilisons aussi la relation qui lie toutes les occurrences d'une répétition. Cette relation indique si deux segments répétés sont issus d'occurrences d'une même répétition.

De la même manière que pour la position, nous ne définissons pas toutes les possibilités en extension. Nous définissons le prédicat en extension puis en intension. Pour deux segments $\mathtt{S_A}$ et $\mathtt{S_B}$, nous obtenons le prédicat relationnel défini en extension :

$$\mathtt{MemeRepetition}(\mathtt{S_A}, \mathtt{S_B}) \equiv \begin{array}{l}\text{« Les segments } \mathtt{S_B} \text{ et } \mathtt{S_A} \text{ appartiennent à} \\ \text{une même répétition. »}\end{array}$$

Nous utilisons aussi le prédicat relationnel défini en intension :

$$\mathtt{MemeRepetition}(\mathtt{X}, \mathtt{Y}) \leftarrow \mathtt{MemeRepetition}(\mathtt{Y}, \mathtt{X}).$$
$$\mathtt{MemeRepetition}(\mathtt{X}, \mathtt{Y}) \leftarrow \mathtt{MemeRepetition}(\mathtt{X}, \mathtt{Z}), \mathtt{MemeRepetition}(\mathtt{Z}, \mathtt{Y}).$$

Cette relation permet de décrire, en plus, un segment répété à partir des segments voisins des occurrences de la répétition du segment répété. Cela est illustré dans la figure 3.6 où les segments $\mathtt{O_{k-1}}$ et $\mathtt{O_{l+1}}$ sont reliés au segment $\mathtt{S_0}$ en tant que voisins d'occurrences $\mathtt{O_k}$ et $\mathtt{O_l}$ d'une même répétition.

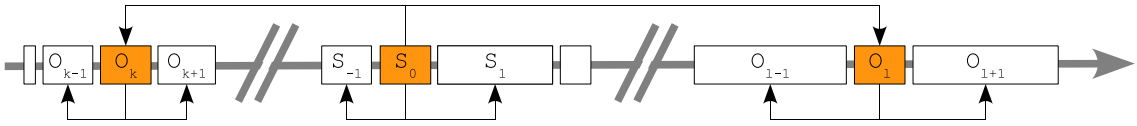



FIGURE 3.6 – Relation d'occurrence et de voisinage.

#### 3.2.3.3 Prédicats contextuels

Les prédicats contextuels sont des prédicats qui utilisent les propriétés des segments voisins ou liés aux occurrences de la répétition à laquelle le segment appartient.

***La densité de répétitions***

Nous savons que les inter-programmes sont généralement diffusés par groupe. Sachant que les inter-programmes se répètent en majorité, les groupes d'inter-programmes forment des régions de segments répétés. Si un segment court se trouve à l'intérieur d'une région de segments répétés, il est fort probable que ce soit un inter-programme.

Pour utiliser cette propriété, nous définissons la densité de répétitions. La densité de répétitions autour d'un segment est le nombre de segments qui sont des occurrences de répétitions dans une fenêtre de temps d'une certaine taille centrée sur ce segment. Un segment appartient à une région de forte densité si la densité autour du segment est supérieure à un certain seuil. En pratique, le seuil est fixé à $2,5$ segments répétés en moins de 5 minutes. De plus, un segment appartient aussi à une région de densité forte si ce segment est issu d'une occurrence d'une répétition qui contient au moins une occurrence qui appartient à une région de densité forte. Pour un segment $\mathtt{S_A}$, nous obtenons ainsi le prédicat contextuel défini en extension :

$$\mathtt{HauteDensite}(\mathtt{S_A}) \equiv \begin{array}{l}\text{« Le segment } \mathtt{S_A} \text{ appartient à une région} \\ \text{de densité forte. »}\end{array}$$

***Le contexte des répétitions***

Il peut être intéressant d'utiliser le fait que toutes les occurrences d'une répétition dans l'historique sont toujours suivies ou précédées du même type de segments. Par exemple, une répétition dont les occurrences sont toujours suivies d'un segment long peut être une répétition d'un parrainage ou d'un générique. Une répétition dont les occurrences sont toujours suivies d'une occurrence d'une autre répétition sera plus probablement une répétition d'un inter-programme.

Pour modéliser le contexte des répétitions, nous utilisons les moyennes des caractéristiques des segments précédant ou suivant toutes les occurrences d'une répétition. Ces segments particuliers sont décrits par la figure 3.7. Pour le segment $\mathtt{S_k}$, la moyenne des durées des segments suivant les occurrences de la répétition liée au segment est :

$$\frac{\mathtt{Duree}(\mathtt{S_{k+1}}) + \mathtt{Duree}(\mathtt{S_{l+1}})}{2}$$

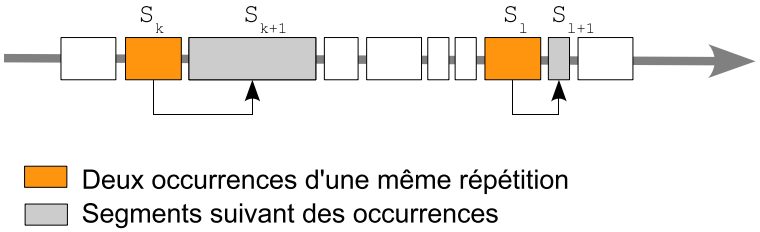



FIGURE 3.7 – Segments suivant des occurrences d'une même répétition.

Nous utilisons seulement les moyennes des durées, des nombres de jours et des nombres d'occurrences des répétitions des segments suivant et précédant les occurrences. Par exemple, pour un segment $\mathtt{S_A}$ et un intervalle `I`, nous obtenons le prédicat contextuel défini en extension :

$\mathtt{MoyenneDureeOccurrencesPrecedentes}(\mathtt{S_A}, \mathtt{I}) \equiv$ « La moyenne des durées des segments précédant les occurrences de la répétition liée au segment $\mathtt{S_A}$ appartient à `I`. »

Pour décrire ces moyennes, nous utilisons les intervalles des prédicats simples : `Duree`, `NbRepetition` et `NombreDeJours`. Ces intervalles ont été définis dans la section 3.2.3.1.

### 3.2.4 Module d'apprentissage

L'apprentissage des règles logiques de classification s'effectue à partir d'un ensemble d'apprentissage. Cet ensemble d'apprentissage est décrit par le module de description logique. Il fournit les prédicats de segment. L'ensemble d'apprentissage fournit en plus des exemples de segments classés. Ces exemples sont des prédicats de classe définis en extension. Le principe du module d'apprentissage est de généraliser ces exemples d'instanciation selon leurs descriptions apportées par le module de description. L'objectif est donc de généraliser des prédicats de classe définis en extension en des prédicats de classe définis

en intension. Nous obtenons des règles de classification. Le principe est résumé dans la figure 3.8.

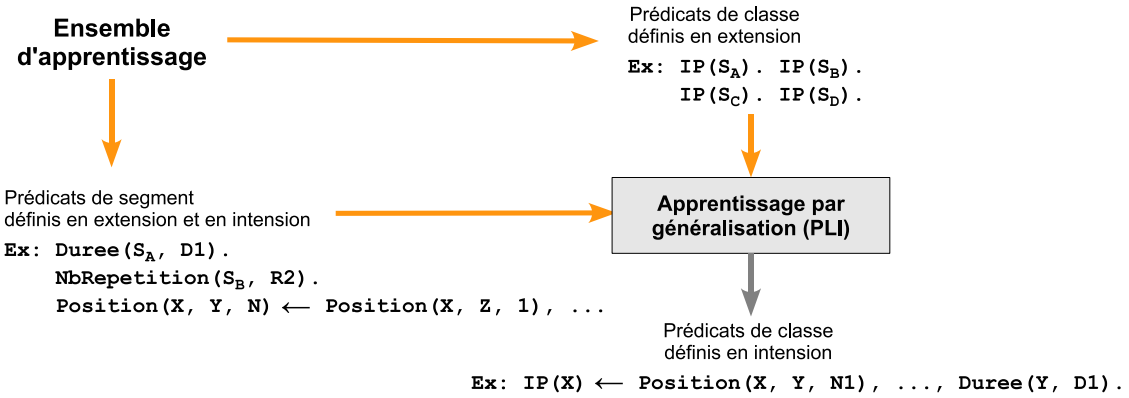



FIGURE 3.8 – Apprentissage par généralisation de prédicats de classe définis en extension.

L'algorithme d'apprentissage est une généralisation de règles logiques à partir de faits logiques constitués de prédicats de classe définis en extension et de prédicats de segment. Si certaines propriétés sont vraies pour certains exemples selon certaines caractéristiques, elles peuvent l'être pour d'autres exemples encore, voire pour tous les exemples. Ce raisonnement est une « induction » et il est réalisé par la programmation logique inductive (PLI). Un ensemble de règles est donc induit de cette manière.

Nous présentons, en premier, notre problème d'apprentissage de la classification des segments. Nous donnons, ensuite, quelques généralités sur la PLI et nous détaillons l'algorithme *Aleph* [Sri07] utilisé pour la généralisation des règles. Enfin, nous proposons une variante de notre module d'apprentissage. Cette variante réalise, en plus, une étape de validation des règles construites. Elle permet alors de trier les règles de classification suivant un niveau de pertinence.

#### 3.2.4.1 L'apprentissage de la classification

Nous nous plaçons dans le cadre d'un problème d'apprentissage supervisé [Mit97, CM02]. L'objectif est de modéliser une fonction $f$ à partir d'un ensemble d'observations. Dans le cadre de la classification des segments, la fonction à modéliser est la fonction $f_C$ qui détermine si un segment appartient ou n'appartient pas à la classe $C$. La fonction $f_C$ est à valeurs dans l'ensemble $\{0,1\}$ : $f_C(x) = 1$ si $x$ est un segment de la classe $C$ et $f_C(x) = 0$ sinon. Les observations sont alors des exemples issus des segments manuellement classés dans l'ensemble d'apprentissage. L'ensemble des exemples est communément noté $\mathcal{E}$. Ces exemples s'accompagnent de l'ensemble $\mathcal{P}_{description}$ des prédicats de segment de l'ensemble d'apprentissage.

Théoriquement, pour une fonction $f$ quelconque, il n'y a pas de certitude à ce que les observations connues puissent représenter les observations futures. Un algorithme d'apprentissage nécessite donc des connaissances *a priori* sur la fonction à modéliser. Ces connaissances se traduisent par la définition d'un ensemble d'hypothèses $\mathcal{H}$ auquel la

fonction $f$ est supposée appartenir. L'apprentissage consiste alors à parcourir $\mathcal{H}$ afin de déterminer l'hypothèse $h$ qui modélise le mieux $f$.

Des fonctions relativement simples à modéliser sont les fonctions indicatrices, c'est à dire à valeurs seulement dans $\{0, 1\}$. L'ensemble des exemples $\mathcal{E}$ se divise en deux parties disjointes : les exemples positifs $\mathcal{E}_+$ et les exemples négatifs $\mathcal{E}_-$. Les exemples positifs sont des exemples $x$ pour lesquels $f(x) = 1$. Ils sont dits couverts par $f$. Les exemples négatifs sont des exemples $x$ pour lesquels $f(x) = 0$. Pour une hypothèse $h$ donnée, l'ensemble des exemples tels que $h(x) = 1$ est appelé la couverture de $h$. Pour deux hypothèses $h_1$ et $h_2$, nous notons $h_1 \prec h_2$ si la couverture de $h_1$ est incluse dans la couverture de $h_2$. L'hypothèse $h_2$ est alors dite « plus générale que » $h_1$. Cette relation définit une relation d'ordre dans l'ensemble des hypothèses $\mathcal{H}$. Elle permet de parcourir $\mathcal{H}$ de manière ordonnée.

Le module d'apprentissage fonctionne uniquement pour une classe $C$ spécifique. Il modélise $f_C$ à partir des exemples positifs et négatifs. Ce module peut être utilisé autant de fois qu'il y a de classes à traiter par le système. Nous proposons ainsi d'apprendre des règles de classification à la fois pour la classe des programmes et pour la classe des inter-prorgammes comme cela est illustré dans la figure 3.9. Notre méthode n'est cependant pas limitée à l'apprentissage de la classification en seulement deux classes. Nous montrerons dans la section 3.3.5 comment étendre notre méthode pour la classification des inter-programmes en trois classes : les publicités, les bandes annonces et les parrainages.

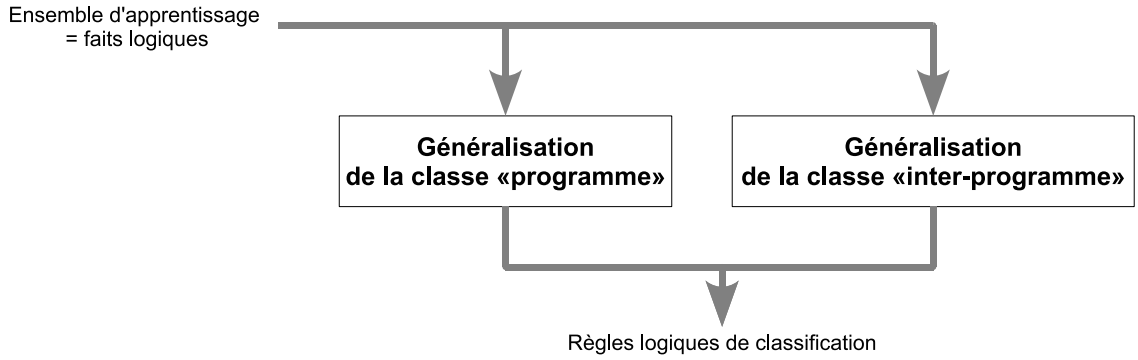



FIGURE 3.9 – Apprentissage de règles logiques à la fois pour la classe des programmes et pour la classe des inter-programmes.

#### 3.2.4.2 Généralités sur la programmation logique inductive

La PLI est une technique d'apprentissage supervisé. Une présentation théorique précise de la PLI est proposée dans [Cla03] et dans les travaux fondateurs de la PLI [Mug91, LD94, MR94].

L'hypothèse de base de la PLI est de supposer que la fonction $f$ à apprendre peut s'exprimer sous la forme d'un ensemble de règles d'implication « Si $x$ ... alors $f(x) = 1$ ». Une méthode pour apprendre des règles d'implication est de construire un arbre de décision puis de transformer chaque décision en une règle logique. Ces règles logiques restent cependant limitées. La PLI permet alors d'étendre ces règles à la logique des prédicats. Le

mode de fonctionnement de l'algorithme est incrémental. Les règles sont apprises une par une et l'ensemble final des règles constitue la modélisation de la fonction $f$.

La PLI se limite généralement à l'apprentissage de règles logiques exprimables sous la forme de clauses de Horn qui sont un sous-ensemble de la logique des prédicats. Une clause de Horn est une formule logique composée d'une disjonction finie de prédicats dont toutes les variables sont quantifiées universellement et qui possède zéro ou un seul prédicat non précédé d'une négation. Une clause de Horn s'écrit des différentes manières suivantes :

$$\mathtt{A} \vee \neg \mathtt{B_1} \vee \ldots \vee \neg \mathtt{B_n}$$
$$\mathtt{A} \leftarrow \mathtt{B_1}, \ldots, \mathtt{B_n}$$
« Si $\mathtt{B_1}$, et ..., et $\mathtt{B_n}$ alors $\mathtt{A}$. »

Pour établir ces clauses de Horn, les différents algorithmes de la PLI reposent sur des opérateurs de construction des règles. Ces opérateurs peuvent être utilisés de manière descendante ou ascendante. De manière descendante, les opérateurs singularisent. Par exemple, la clause de Horn vide est par nature « VRAI » pour tous les exemples positifs et négatifs de l'ensemble des exemples $\mathcal{E}$. Cette clause vide est singularisée en étant modifiée pour qu'elle ne couvre pas les exemples négatifs. À l'inverse, de manière ascendante, les opérateurs universalisent. Par exemple, un exemple positif est une clause de Horn constituée d'un seul prédicat qui est un prédicat de classe. La clause formée par l'exemple positif est universalisée en étant modifiée pour qu'elle couvre d'autres exemples positifs. Les clauses sont modifiées en utilisant les opérateurs de construction des règles comme les opérateurs suivants :

- enlever/ajouter un prédicat ;
- remplacer un symbole en variable ou une variable en symbole ;
- remplacer une variable par une autre différente ou deux variables différentes par une seule.

Grâce à la logique des prédicats, les règles généralisées pour modéliser la fonction $f$ sont explicites et compréhensibles par un humain. Elles peuvent être également écrites manuellement par un humain. Cela offre la possibilité de rajouter des hypothèses *a priori* fournies par un expert. Malheureusement, la logique des prédicats est aussi contraignante. Elle rend complexe la manipulation des données numériques qui doivent, le plus souvent, être catégorisées.

#### 3.2.4.3 Aleph

Dans nos travaux, nous utilisons le système de PLI *Aleph* réalisé par Srinivasan [Sri07]. D'autres systèmes de PLI existent. Les plus connus sont Foil [Qui90], Progol [Mug95] et Claudien [DRD97]. La dernière modification du système de PLI date de l'année 2007 ce qui fait d'*Aleph* un système de PLI récent.

L'algorithme de PLI d'*Aleph* est présenté dans l'algorithme 3. Pour généraliser une règle de classification, *Aleph* se base sur le *Mode Directed Inverse Entailment*, MDIE [Mug95]. *Aleph* construit donc une règle dite la plus spécifique à partir des exemples et la généralise. Il utilise pour cela un mixte des approches descendante et ascendante. Les étapes de construction de la règle de classification la plus spécifique et de généralisation de cette règle peuvent être contrôlées par des « biais ». Ces biais orientent l'algorithme vers la

fonction $f$ à apprendre. Un exemple de biais simple est le fait d'imposer à toutes les règles d'utiliser au moins un prédicat spécifique donné. Nous détaillons, dans les paragraphes suivants, la construction de la règle de classification la plus spécifique, la généralisation de cette règle de classification, l'évaluation des règles de classification et l'utilisation des biais.

**Aleph**

▷ **Entrées:**
$\mathcal{E}_+$ : Ensemble d'exemples positifs, prédicats de classe ;
$\mathcal{E}_-$ : Ensemble d'exemples négatifs, prédicats de classe ;
$\mathcal{P}_{description}$ : Ensemble de prédicats de segment ;

◁ **Sorties:**
$h_f$ : Ensemble de règles de classification ;

**Début**
1: **pour** $e_i \in \mathcal{E}_+$ **faire**
2: Construire la règle de classification la plus spécifique à partir d'$e_i$ et de $\mathcal{P}_{description}$ ;
3: Trouver des règles de classification plus générales que la règle de classification la plus spécifique ;
4: Ajouter à $h_f$ la règle de classification possédant le meilleur score d'évaluation suivant $\mathcal{E}_+$ et $\mathcal{E}_-$. Cette règle est plus générale que la règle de classification la plus spécifique ;
5: **fin pour**
**Fin**

Algorithme 3 – Généralisation de règles logiques de classification suivant *Aleph*.

### *Construction de la règle de classification la plus spécifique*

La méthode de construction de la règle de classification la plus spécifique est présentée dans [Mug95]. Cette étape est appelée « saturation ». Elle permet de borner l'espace de recherche des règles. En pratique, cette méthode consiste à ajouter à un exemple positif, $e_i \in \mathcal{E}_+$, des instanciations de prédicats de segment de l'ensemble des prédicats de segment $\mathcal{P}_{description}$. La règle de classification la plus spécifique obtenue peut être infinie. Elle est donc limitée par des biais appropriés qui contrôlent, par exemple, le choix du nombre de prédicats de segment à ajouter.

Nous illustrons cela par un exemple. L'exemple positif, $e_i$, se traduit par l'instanciation d'une classe : $\texttt{IP}(\texttt{S}_\texttt{A})$. Des instanciations de prédicats de segment sont ajoutées à cet exemple. Nous obtenons la règle de classification :

$$\texttt{IP}(\texttt{S}_\texttt{A}) \leftarrow \texttt{NbRepetition}(\texttt{S}_\texttt{A}, 3), \texttt{Position}(\texttt{S}_\texttt{A}, \texttt{S}_\texttt{N}, 15), \texttt{MemeRepetition}(\texttt{S}_\texttt{B}, \texttt{S}_\texttt{N}), \ldots$$
$$\ldots, \texttt{Position}(\texttt{S}_\texttt{A}, \texttt{S}_\texttt{ZZ}, 32), \texttt{Duree}(\texttt{S}_\texttt{ZZ}, [7, \texttt{5secondes}, 12, \texttt{5secondes}[).$$

***Généralisation de la règle de classification la plus spécifique***

Cette étape est appelée « réduction » de la règle de classification la plus spécifique. Elle utilise des opérateurs de construction des règles aussi appelés opérateurs de raffinement (*refinement operators*). La règle de classification la plus spécifique est construite à partir d'instanciations de prédicats de segment. Le principe des opérateurs de généralisation est de sélectionner des sous-ensembles des instanciations utilisées dans la règle de classification la plus spécifique puis de remplacer les instanciations par des variables. Nous obtenons ainsi des prédicats décrits en intension.

Dans l'exemple précédent, « $\texttt{IP}(\texttt{S}_\texttt{A}) \leftarrow \texttt{NbRepetition}(\texttt{S}_\texttt{A}, 3)$. » est un sous ensemble de la règle de classification la plus spécifique. « $\texttt{IP}(\texttt{X}) \leftarrow \texttt{NbRepetition}(\texttt{X}, 3)$. » est une règle potentielle.

Le processus de recherche des règles potentielles utilise des arbres pour représenter les hiérarchies entre les sous-ensembles d'instanciations de la règle de classification la plus spécifique. Ces arbres permettent de parcourir efficacement l'ensemble des règles potentielles. Certains biais permettent alors d'élaguer certaines branches de l'arbre de recherche pour plus d'efficacité.

***Évaluation des règles de classification construites***

Cette étape permet de calculer le score d'évaluation d'une règle potentielle par rapport à l'ensemble d'apprentissage. Pour déterminer le score d'évaluation d'une règle, la couverture de cette règle est déterminée. Nous obtenons le nombre $N_P$ des exemples positifs couverts par la règle et le nombre $N_N$ des exemples négatifs également couverts. Une bonne règle, selon le score d'évaluation, est une règle qui maximise $N_P$ et minimise $N_N$. Le score d'évaluation *Eval* que nous utilisons pour une règle potentielle R est le suivant :

$$Eval(R) = \frac{N_P}{N_P + N_N}.$$

***Biais***

Les biais sont des informations *a priori* utilisées pour optimiser la recherche de la fonction $f$ à apprendre. *Aleph* dispose d'un moyen flexible de définition des biais. *Aleph* est, en effet, implémenté entièrement dans le langage logique Prolog [SS94] qui est un langage interprété [4]. Les biais sont donc écrits dans le langage prolog suivant les consignes d'*Aleph*. Ils sont ensuite directement compris par l'algorithme sans avoir à modifier l'algorithme lui même.

La PLI différencie trois types de biais que nous utilisons : les biais syntaxiques, les biais sémantiques et les biais de recherche.

1) Les biais syntaxiques spécifient la forme des règles de classification construites. Ils permettent de limiter le nombre de prédicats dans les règles de classification, de limiter le nombre de variables utilisées ou de vérifier qu'il n'y a pas de variables inutiles.

4. L'interpréteur Prolog utilisé est *YAP* – Yet Another Prolog, `http://www.dcc.fc.up.pt/~vsc/Yap/`.

2) Les biais sémantiques imposent des contraintes sur la nature des prédicats et de leurs arguments. Les biais sémantiques déclarent explicitement quels prédicats sont autorisés pour la construction des règles de classification. Ils indiquent aussi quels arguments peuvent être utilisés en entrée, en sortie ou comme symbole. Un argument en entrée peut être un argument du prédicat de tête de la règle de classification. Sinon, un argument en entrée dans un prédicat `F_i` est un argument qui apparait en sortie dans un prédicat précédent `F_i` dans la règle de classification. Un argument symbole est un argument fixé à partir de la description fournit par l'ensemble d'apprentissage.

3) Les biais de recherche restreignent le parcours des arbres lors de la recherche de généralisation. Ces biais permettent d'élaguer certaines branches. Ces branches peuvent contenir des prédicats qui ne sont pas utiles. Par exemple, à partir d'un prédicat `Position` nous élaguons toutes les sous branches qui partent d'un autre prédicat `Position`. De cette manière nous assurons que les règles ne contiennent pas deux prédicats `Position` à la suite. Ce biais de recherche permet d'assurer une certaine homogénéité dans les règles recherchées. Un autre biais de recherche est une limite sur le nombre de nœuds à évaluer pendant la recherche. Nous la fixons à 30000. Ce biais permet de contrôler la complexité temporelle de l'apprentissage. Les biais de recherche que nous utilisons sont les suivants :

   – nous interdisons l'utilisations du prédicat de classe de tête dans le corps de la règle. Nous utilisons ce biais pour éviter les tautologies dans la récursivité ;

   – nous interdisons l'utilisation consécutive du même prédicat `Position` ou `Occurrence`. Pour obtenir des règles biens formées, nous imposons l'utilisation de prédicats de segment simples après chaque prédicat de voisinage ;

   – les positions que nous avons définies peuvent se répéter. La position $D$ de $\mathtt{S_A}$ à $\mathtt{S_B}$ est aussi la position $-D$ de $\mathtt{S_B}$ à $\mathtt{S_A}$. Nous interdisons la répétition d'une position.

#### 3.2.4.4 La validation des règles logiques

Toutes les règles de classification construites par la PLI ne possèdent pas toutes la même pertinence. Certaines règles sont très générales. Elles sont vraies sur un très grand nombre de segments de l'ensemble d'apprentissage sans aucune erreur. D'autres règles sont vraies sur un très grand nombre de segments mais avec quelques ou beaucoup d'erreurs. Il existe aussi des règles très spécifiques qui sont vraies pour peu de segments mais sans aucune erreur.

Nous avons donc introduit une étape de tri des règles suivant leur pertinence. Pour cela, l'ensemble d'apprentissage est divisé en deux parties. Cela est résumé dans la figure 3.10. Une partie est utilisée pour la généralisation de règles et une autre partie est utilisée pour la validation des règles apprises. La validation estime un niveau de pertinence pour

chaque règle. Toutes ces informations supplémentaires sont importantes pour le processus de classification.

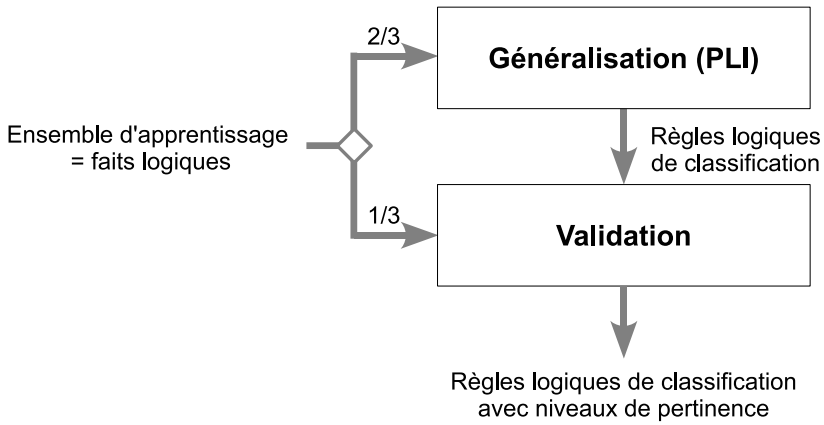



FIGURE 3.10 – Généralisation et validation de règles logiques à partir de faits. L'ensemble d'apprentissage est séparé en deux parties. La première partie représente 2/3 de l'ensemble d'apprentissage.

Pour évaluer une règle, nous favorisons la notion de précision au détriment du rappel sur les segments appartenant à la classe apprise. En effet, il nous semble important qu'une règle classe correctement les segments. Ces segments nouvellement classés peuvent aussi servir pour classer d'autres segments grâce aux relations et aux contextes définis entre les segments.

Le niveau de pertinence est donc proportionnel à la précision des règles. Plus une règle est précise et plus son niveau de pertinence est fort. Nous notons 0 le niveau de pertinence le plus faible. Pour calculer un niveau de pertinence, nous appliquons la règle sur les segments de l'ensemble de validation et nous calculons la couverture de cette règle. Les exemples couverts par la règle sont les segments qui ont été classés comme appartenant à la classe apprise.

Nous fixons trois seuils sur la précision. Le niveau de pertinence numéro 3, le plus haut, correspond aux règles pour lesquelles la précision est de 100 %. Le niveau de pertinence numéro 2 correspond aux règles pour lesquelles la précision est supérieure à 90 %. Le niveau de pertinence numéro 1 correspond aux règles pour lesquelles la précision est supérieure à 75 %. Les autres règles ont un niveau de pertinence égal à 0.

### 3.2.5 Module de classification

Le dernier module réalise la classification dite « en-ligne » des segments de la portion du flux à délinéariser. Ce module utilise les règles de classification apprises par la PLI ou fournies par la connaissance *a priori* d'un expert. Ces règles sont toutes des prédicats de classe définis en intension. L'application des règles dépend des prédicats de segment fournis par le module de description des segments du flux. Le principe est résumé dans la figure 3.11.

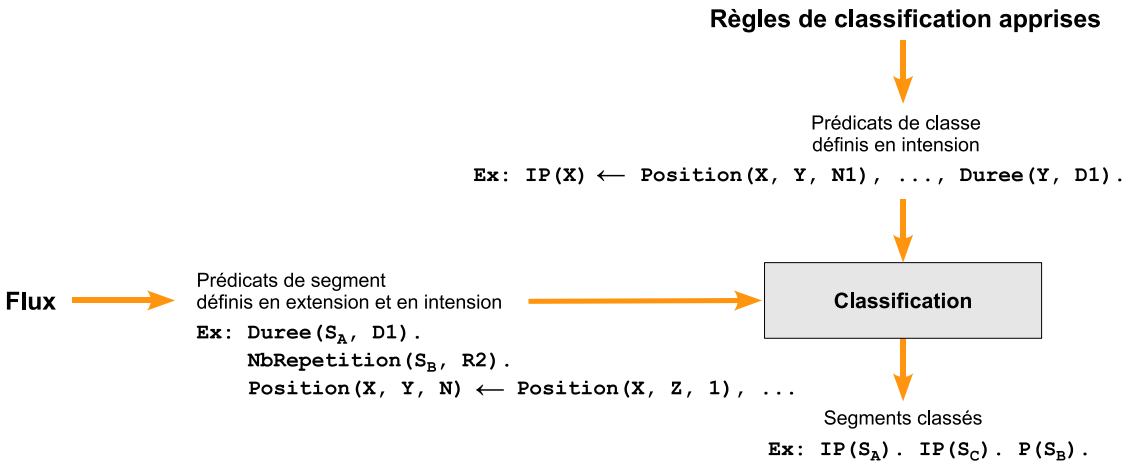



FIGURE 3.11 – Classification par application de prédicats de classe définis en intension.

Les règles fournies peuvent correspondre à plusieurs classes et il peut y avoir des ambiguités lors de leur application. Selon une variante du module d'apprentissage, les règles peuvent être triées par niveaux de pertinence. Nous distinguons, de plus, les règles apprises automatiquement et les règles *a priori* fournies par un expert. Par conséquent, l'application de toutes ces règles sur tous les segments et la décision finale de la classe de chaque segment requièrent un algorithme spécifique. Avant d'expliquer cet algorithme, nous organisons les règles en une hiérarchie de plusieurs types.

#### 3.2.5.1 Hiérarchie des règles de classification

Les règles sont composées de prédicats simples, contextuels, relationnels et/ou de classe. Nous commençons par distinguer deux types de règles :

- **Les règles mono-classe ($1_C$) :** une seule classe est utilisée dans la description de ces règles. Tous les prédicats de classe de ces règles sont de la même classe qui est la classe du prédicat de tête. Par exemple [5] :

$$\texttt{IP}(\texttt{S}_\texttt{A}) \leftarrow \texttt{Position}(\texttt{S}_\texttt{A}, \texttt{S}_\texttt{B}, 1), \texttt{IP}(\texttt{S}_\texttt{B}), \texttt{Position}(\texttt{S}_\texttt{A}, \texttt{S}_\texttt{C}, -1), \texttt{IP}(\texttt{S}_\texttt{C}).$$

- **Les règles multi-classes ($N_C$) :** plusieurs classes sont utilisées dans la description de ces règles. Elles contiennent au moins un prédicat de classe différent de la classe du prédicat de tête. Par exemple [5] :

$$\begin{aligned}\texttt{P}(\texttt{S}_\texttt{A}) \leftarrow &\texttt{Position}(\texttt{S}_\texttt{A}, \texttt{S}_\texttt{B}, 3), \texttt{IP}(\texttt{S}_\texttt{B}), \texttt{MemeRepetition}(\texttt{S}_\texttt{A}, \texttt{S}_\texttt{C}),\\ &\texttt{Position}(\texttt{S}_\texttt{C}, \texttt{S}_\texttt{D}, 1), \texttt{P}(\texttt{S}_\texttt{C}).\end{aligned}$$

Nous distinguons ensuite deux sous-types de règles mono-classe et multi-classes :

5. L'exemple fourni n'est pas un exemple de règle réelle apprise. Il sert simplement à illustrer les différents types de règles.

– **Les règles non récursives :** ces règles ne contiennent pas de prédicat de classe identique à la classe du prédicat de tête. Par exemple [5] :

$$\texttt{IP}(\texttt{S}_\texttt{A}) \leftarrow \texttt{Position}(\texttt{S}_\texttt{A}, \texttt{S}_\texttt{B}, 1), \texttt{P}(\texttt{S}_\texttt{B}), \texttt{NbRepetition}(\texttt{S}_\texttt{A}, 7).$$

– **Les règles récursives :** ces règles contiennent au moins un prédicat de classe identique à la classe du prédicat de tête. Par exemple [5] :

$$\texttt{IP}(\texttt{S}_\texttt{A}) \leftarrow \texttt{MemeRepetition}(\texttt{S}_\texttt{A}, \texttt{S}_\texttt{B}), \texttt{IP}(\texttt{S}_\texttt{B}).$$

Les règles sont ainsi organisées selon la hiérarchie proposée dans la figure 3.12.

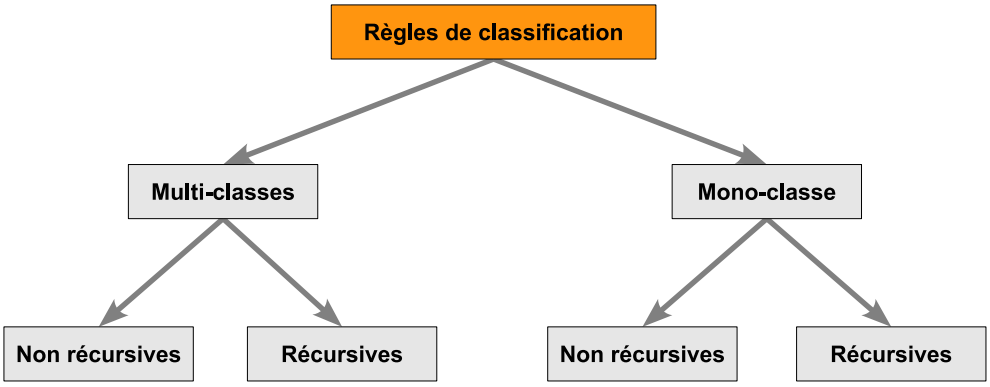



FIGURE 3.12 – Hiérarchie des règles de classification.

Pour un ensemble $E$ de règles de classification, nous notons :

– $E_{\{C,1_C,i\}} \equiv$ « L'ensemble des règles mono-classe ($1_C$) de $E$ possédant un niveau de pertinence égal à $i$ et dont la classe du prédicat de tête est $C$. »

– $E_{\{C,N_C,i\}} \equiv$ « L'ensemble des règles multi-classes ($N_C$) de $E$ possédant un niveau de pertinence égal à $i$ et dont la classe du prédicat de tête est $C$. »

L'application des règles pour la classification nécessite de prendre en compte la classe des règles de classification, le niveau de pertinence des règles et, de plus, les types des règles. Cela requiert des précautions. Les règles récursives, mono-classe et multi-classes sont, en effet, très particulières. Une règle récursive peut engendrer de nombreuses fausses classifications si elle est appliquée sur des segments préalablement classés avec des règles possédant un niveau de pertinence bas. De la même manière, une règle mono-classe ou multi-classes dépend des classifications précédentes. C'est, par ailleurs, pour les mêmes raisons, que nous avons privilégié la précision au détriment du rappel lors du calcul de la pertinence des règles.

#### 3.2.5.2 Algorithme d'application des règles

L'algorithme d'application des règles se base sur les niveaux de pertinence. Les règles du plus haut niveau de pertinence sont appliquées en premier. Ensuite, les règles des

niveaux inférieurs sont appliquées. Les niveaux de pertinence sont donc traités de manière décroissante. Les niveaux de pertinence ne sont utilisés qu'une seule fois chacun. Cela est nécessaire pour que des règles d'un niveau de pertinence supérieur ne s'appuient pas sur les résultats de classification de règles d'un niveau de pertinence inférieur. Les segments classés par des règles d'un niveau de pertinence, $i$, ne peuvent donc pas servir de support à des règles d'un niveau de pertinence supérieur, $i_+ > i$. Cela risque de détériorer, en effet, les résultats. Par contre, ils servent de support aux règles de niveaux de pertinence inférieurs ou égaux, $i_- \leq i$.

Au final, l'algorithme d'application des règles est détaillé dans les algorithmes 4 et 5. Il se déroule globalement en deux étapes :

1) pour un niveau de pertinence donné, les règles mono-classe sont appliquées pour chaque classe. Les classes qui couvrent théoriquement le plus de segments sont traitées en premier ;
2) les règles multi-classes sont appliquées pour chaque classe tant qu'elles peuvent s'appliquer.

Les règles mono-classe ou multi-classes des différentes classes sont appliquées dans un ordre précis. Les règles non récursives puis récursives sont appliquées. Les règles récursives sont appliquées tant que leur application permet de classer de nouveaux segments qui, potentiellement, peuvent servir pour d'autres règles récursives. De la même manière, les règles multi-classes sont exécutées tant que leur application permet de classer de nouveaux segments. Les règles récursives et multi-classes sont donc appliquées tant qu'elles sont effectives. Plus précisément, une règle appliquée est dite effective si la règle est vraie pour au moins un segment et que cette règle ne s'est pas déjà appliquée sur ce même segment.

Il reste à introduire l'application des règles de classification *a prori*. Ces règles *a prori* sont aussi de plusieurs types mono-classe, multi-classes, récursives et non récursives. Le même algorithme est utilisé. Les règles *a priori* sont finalement traitées comme des règles normales. Nous fixons leur niveau de pertinence à un niveau maximum, $n_{apriori}$, supérieur à tous les autres niveaux. Elles s'imposent ainsi sur les règles automatiquement apprises.

Une variante de l'algorithme d'application des règles peut utiliser des niveaux de pertinence minimum. Ces niveaux de pertinence minimum interdisent d'appliquer des règles d'un niveau de pertinence inférieur à ces niveaux. Nous pouvons alors définir un niveau de pertinence minimum $np$ pour toutes les règles. Nous pouvons aussi choisir un niveau de pertinence minimum $np_{rec}$ pour les règles récursives et un niveau de pertinence minimum $np_{multi}$ pour les règles multi-classes. Ces trois niveaux de pertinence minimums $np$, $np_{rec}$ et $np_{multi}$ permettent alors d'optimiser les performances globales du module de classification en minimisant les fausses classification de segment.

#### 3.2.5.3 Décision de la classe de chaque segment

À la suite de l'application de plusieurs règles de classification, les segments peuvent être classés en une ou plusieurs classes différentes. En effet, les ensembles des segments couverts par les différentes règles ne sont pas disjoints. La détermination de la classe d'un segment à l'issue de l'application de toutes les règles requiert par conséquent une décision. Cette détermination de la classe d'un segment est aussi nécessaire pendant l'algorithme d'application des règles pour pouvoir utiliser les règles de niveaux de pertinence inférieurs ou les règles récursives, mono-classe et multi-classes. La décision de la classe de chaque

**Application des règles**

▷ **Entrées:**
$np_{max}$ : Niveau de pertinence maximum ;
$E$ : Ensemble de règles (apprises et *a priori*) ;
$L_{Segments}$ : Liste des segments à classer ;

◁ **Sorties:**
$L_{Segments}$ : Liste des segments classés ;

**Début**

**pour** $i = np_{max} > 0$ à 0 **faire**
  **répéter**
    **pour** chaque classe $C$ **faire**
      *// règles mono-classe*
      ApplicationDétaillée($E_{\{C,1_C,i\}}$, $L_{Segments}$) ;
    **fin pour**
    **répéter**
      **pour** chaque classe $C$ **faire**
        *// règles multi-classes*
        ApplicationDétaillée($E_{\{C,N_C,i\}}$, $L_{Segments}$) ;
      **fin pour**
    **jusqu'à** Aucune règle appliquée de $E$ n'est effective
  **jusqu'à** Aucune règle appliquée de $E$ n'est effective
**fin pour**

**Fin**

Algorithme 4 – Application des règles de classification. Les règles mono-classes sont appliquées en premier.

**ApplicationDétaillée**

▷ **Entrées:**
$E$ : Ensemble de règles ;
$L_{Segments}$ : Liste des segments à classer ;

◁ **Sorties:**
$L_{Segments}$ : Liste des segments éventuellement classés ;

**Début**
1: Appliquer à $L_{Segments}$ les règles **non récursives** de $E$ ;
2: **répéter**
3: Appliquer à $L_{Segments}$ les règles **récursives** de $E$ ;
4: **jusqu'à** Aucune règle appliquée de $E$ n'est effective

**Fin**

Algorithme 5 – Application détaillée des règles de classification. Les règles non récursives sont appliquées en premier.

segment peut donc être effectuée plusieurs fois en plus de la décision finale de la classe de chaque segment.

Notre méthode de décision de la classe de chaque segment repose sur un système de votes. Chaque règle de classification dont le prédicat de classe est vrai pour un segment vote pour une classe spécifique pour un segment particulier. Le vote possède alors une valeur dépendante du niveau de pertinence de la règle. Les règles *a priori* sont des règles absolues. La valeur de leur vote est donc infinie. Pour les règles normales, nous avons évalué empiriquement que nous obtenons de meilleurs résultats de classification lorsque la valeur d'un vote est $2^i$ pour une règle de niveau de pertinence $i$. Ainsi, une règle de niveau de pertinence $i$ possède 2 fois plus de poids qu'une règle de niveau de pertinence $i-1$.

Chaque segment peut posséder des votes attribués à chaque classe possible. Nous proposons deux modes de décision :

1) « **Le vote unanime** ». La classe du segment est la seule classe qui a reçu des votes. Si plusieurs classes ont reçu des votes, le segment n'est pas classé.

2) « **Le vote majoritaire** ». La classe du segment est la classe qui a reçu strictement le plus de votes. Si plusieurs classes sont à égalité, le segment n'est pas classé.

Lorsqu'une classe unique est déterminée pour un segment, celle-ci peut être soit fixée pour la suite de l'algorithme d'application des règles, soit laissée ouverte à des modifications ultérieures en fonction des prochaines règles appliquées et des prochains votes. Les expériences de la section 3.3 montrent quelle est la configuration la plus performante.

## 3.3 Résultats

Nous avons évalué notre méthode de classification des segments audiovisuels basée sur la programmation logique inductive en fonction des quatre contraintes clés de généricité, d'efficacité, d'automaticité, et de continuité. Nous rappelons que ces contraintes sont nécessaires pour la construction d'un système global de délinéarisation automatique.

La généricité et l'efficacité de notre méthode sont étudiées en réalisant des expériences sur deux chaînes : TF1 et France 2. Les résultats de ces expériences sont analysés dans la suite de cette section pour montrer les performances de notre classification des segments.

La continuité et l'automaticité de notre méthode sont assurées par le système de découpage. Ce dernier fournit des segments à partir d'une analyse du flux en portions de 24 heures. Pour ce niveau de délinéarisation, nous appliquons simplement un ensemble de règles de classification sur les segments découpés de chaque portion. L'ordre d'application des règles est bien défini. Il ne dépend pas de paramètres cruciaux. L'application des règles est donc automatique. Cet ensemble de règles requiert néanmoins une étape d'apprentissage. Cet apprentissage est nécessaire pour la configuration du système de délinéarisation sur chaque chaîne de télévision. Il est supervisé et il s'effectue sur une semaine de flux TV précisément annotée comme dans l'annexe B. Cependant, notre système reste complètement automatique lors de son utilisation. Pour la continuité, nous appliquons en plus notre méthode sur tous les jours de notre ensemble de test et nous vérifions que les résultats sont stables et cohérents.

Pour présenter les résultats, nous détaillons en premier le contexte expérimental, puis le protocole d'évaluation et enfin les expériences. Nous classons dans un premier temps les segments en segments de programme et en segments d'inter-programme puis, dans un second temps, nous classons les segments en segments de programme long et en segments qui ne sont pas des programmes longs (les programmes courts et les inter-programmes). Nous proposons aussi une classification des segments d'inter-programmes en trois catégories : les publicités, les bandes annonces et les parrainages.

### 3.3.1 Contexte expérimental

Nous gardons le contexte expérimental utilisé pour le découpage en segments à partir des répétitions du flux. De plus, nous utilisons directement les résultats du découpage obtenus par la méthode du chapitre 2 précédent. Nous donnons dans le tableau 3.1 le nombre total de segments découpé sur France 2 et sur TF1 pour les 4 semaines. Nous indiquons également dans ce même tableau, le nombre de segments de programme et le nombre de segments d'inter-programme à classer.

Le découpage utilise un historique d'une semaine que nous conservons en entier pour prendre en compte les propriétés de répétitions des segments. Nous analysons cependant le flux par portions de 24 heures selon le même procédé utilisé pour le découpage. Pour chaque jour analysé, nous considérons donc les segments issus des répétitions calculées sur une semaine avec les paramètres fixés dans la section 2.5.

Pour les deux chaînes France 2 et TF1, nous utilisons plus précisément la première semaine du corpus de l'annexe A pour réaliser l'apprentissage. Les 3 autres semaines constituent notre ensemble de test. La deuxième semaine est une semaine transitoire car les répétitions sont calculées sur un historique qui empiète sur les répétitions de la se-

| | Nombre de segments total | Programmes | Inter-programmes |
|---|---|---|---|
| Semaine n°1 | 3764 | 1316 | 2448 |
| Semaine n°2 | 3811 | 1198 | 2613 |
| Semaine n°3 | 3917 | 1210 | 2707 |
| Semaine n°4 | 3846 | 1218 | 2628 |

(a) France 2

| | Nombre de segments total | Programmes | Inter-programmes |
|---|---|---|---|
| Semaine n°1 | 4705 | 931 | 3774 |
| Semaine n°2 | 5044 | 1022 | 4022 |
| Semaine n°3 | 5330 | 1057 | 4273 |
| Semaine n°4 | 5397 | 1084 | 4313 |

(b) TF1

TABLEAU 3.1 – Segments résultant du découpage effectué à partir des répétitions.

maine utilisée pour l'apprentissage. Les deux dernières semaines du corpus sont de réelles semaines de test sans aucun lien avec l'apprentissage. La figure 3.13 illustre cette utilisation des semaines du corpus.

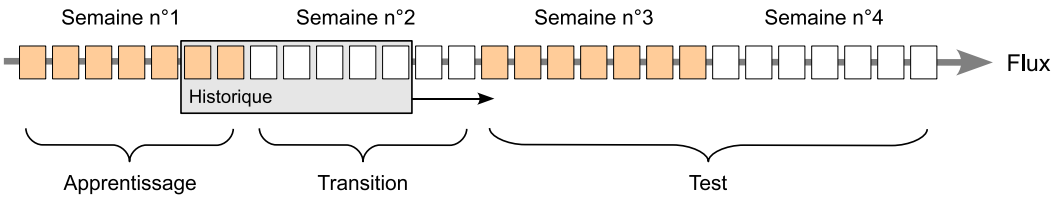



FIGURE 3.13 – Utilisation des semaines du corpus pour l'apprentissage et le test de la classification de portions de 24 heures avec prise en compte d'un historique d'une semaine.

### 3.3.2 Protocole d'évaluation

Nous évaluons la classe attribuée à chaque segment par notre méthode. Pour évaluer la classe attribuée à un segment, nous la comparons avec la classe réelle du segment. Cette classe réelle est obtenue par mise en correspondance des segments découpés automatiquement et des segments de la vérité terrain établis manuellement. Un segment découpé est mis en correspondance avec le segment de la vérité terrain qui le chevauche le plus.

Dans un système de classification, des matrices de confusion sont utilisées pour l'évaluation. Chaque ligne d'une matrice de confusion correspond à l'ensemble des segments

qui appartiennent à une seule classe réelle. La répartition de ces segments dans chaque colonne indique dans quelle classe les segments ont été classés par notre méthode. Une classification de bonne qualité maximise la diagonale principale de la matrice pour laquelle la classe de la colonne correspond à la classe de la ligne. Dans notre système, nous utilisons une colonne supplémentaire $\emptyset$ pour représenter les segments qui n'ont pas été classés.

Un exemple de matrice de confusion pour 2 classes A et B est donné dans le tableau 3.2. Une bonne classification maximise $m_1$ et $m_5$ par rapport à $m_2$, $m_3$, $m_4$ et $m_6$.

| | A | B | $\emptyset$ |
|---|---|---|---|
| A | $m_1$ | $m_2$ | $m_3$ |
| B | $m_4$ | $m_5$ | $m_6$ |

Tableau 3.2 – Exemple de matrice de confusion pour 2 classes A et B.

Dans le cas particulier des matrices de confusion à 2 classes, un cœfficient peut être calculé pour représenter la corrélation entre les classes réelles et les classes attribuées par notre méthode. Il fournit une mesure globale de la qualité de la classification. Il est lié aux cœfficients de corrélation de Pearson. Ce cœfficient noté $Phi$ est obtenu par la formule suivante :

$$Phi = \frac{m_1 m_5 - (m_2 + m_3)(m_4 + m_6)}{\sqrt{(m_1 + m_2 + m_3)(m_1 + m_4 + m_6)(m_5 + m_2 + m_3)(m_5 + m_4 + m_6)}} \in [-1, 1].$$

### 3.3.3 Expériences 1 : classification des segments en segments de programme et en segments d'inter-programme

Cet ensemble d'expériences concerne la classification de segments en segments de programme et en segments d'inter-programme. Les premières expériences étudient les performances des différentes configurations de notre méthode de classification. Les résultats nous permettent en particulier de choisir le mode de décision pour la classification et de configurer l'ordre d'application des règles apprises en fonction de leurs niveaux de pertinence. Ensuite, les dernières expériences servent à analyser les performances de la classification plus en détail.

Lors de la configuration de notre méthode, nous avons noté trop de paramètres interdépendants à prendre en compte. Nous ne présentons pas d'expériences croisées. Par conséquent, l'ordre de réalisation des expériences est important.

#### 3.3.3.1 Choix du mode de décision de la classe de chaque segment

L'attribution d'une classe à un segment peut s'effectuer par différents modes de décision. Nous comparons dans cette expérience, les différents résultats suivant le mode de décision. Nous utilisons alors toutes les règles apprises par notre système et validées suivant les 4 différents niveaux de pertinence.

Les différents modes de décision expliqués dans la section 3.2.5.3 sont notés selon les termes suivants :

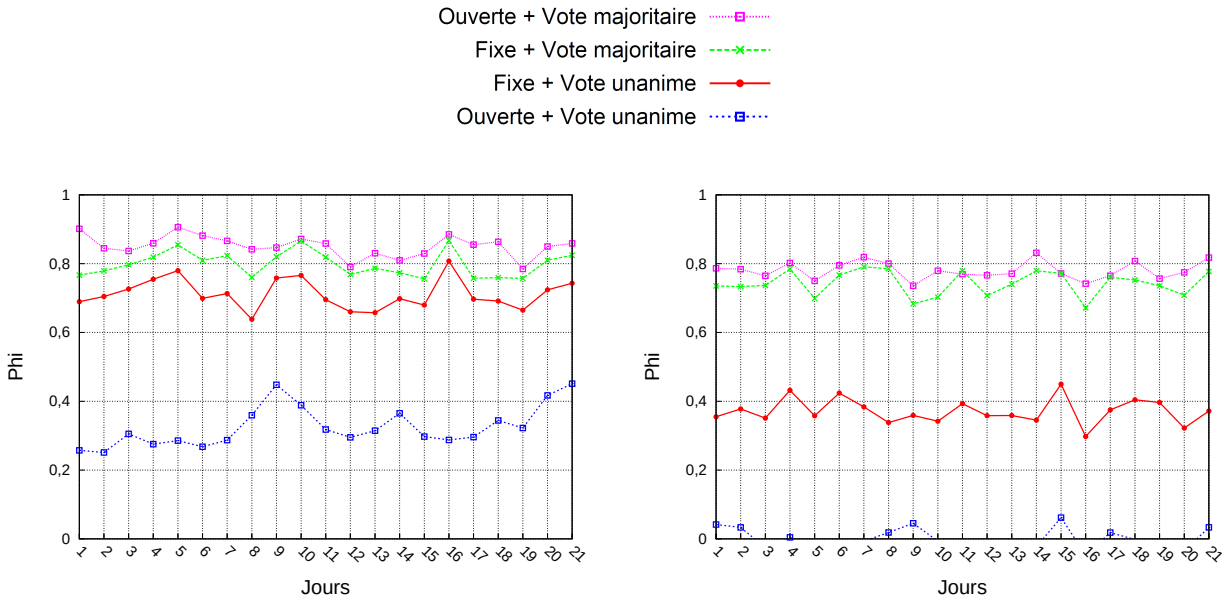
– « Fixe + Vote unanime » représente le vote unanime dans lequel la classe de tout segment classé à un moment donné est fixée pour le reste de l'algorithme d'application des règles ;

– « Fixe + Vote majoritaire » représente le vote majoritaire dans lequel la classe de tout segment classé à un moment donné est fixée pour le reste de l'algorithme d'application des règles ;

– « Ouverte + Vote unanime » représente le vote unanime dans lequel la classe de tout segment classé est laissée ouverte à d'autres votes pour le reste de l'algorithme d'application des règles ;

– « Ouverte + Vote majoritaire » représente le vote majoritaire dans lequel la classe de tout segment classé à un moment donné est laissée ouverte à d'autres votes pour le reste de l'algorithme d'application des règles.

La figure 3.14 montre les résultats obtenus sur les deux chaînes étudiées. Pour chacun des modes de décision, nous avons calculé le cœfficient Phi sur chaque jour des semaines de test. Une configuration apparait globalement meilleure que les autres : « Ouverte + Vote majoritaire ». Par contre, le vote unanime apparaît, de manière générale, comme une mauvaise décision. En particulier, lorsque la classe est laissée ouverte à modifications, les règles d'un bas niveau de pertinence votent, comme nous pouvions nous y attendre, de manière incorrecte pour de nombreux segments empêchant ainsi le vote unanime.




(a) France 2 (b) TF1

FIGURE 3.14 – Performances de Phi suivant le mode de décision de la classe de chaque segment.

#### 3.3.3.2 Utilisation des règles *a priori*

Notre système possède la propriété de pouvoir intégrer facilement de nouvelles règles *a priori* fournies par un expert. Dans nos expériences, nous considérons les trois règles *a priori* suivantes :

$$\begin{aligned} \texttt{P}(\texttt{S}_\texttt{A}) &\leftarrow \texttt{Duree}(\texttt{S}_\texttt{A}, [2,75\ \texttt{minutes},\ 7,5\ \texttt{minutes}[). \\ \texttt{P}(\texttt{S}_\texttt{A}) &\leftarrow \texttt{Duree}(\texttt{S}_\texttt{A}, [7,5\ \texttt{minutes},\ \infty[). \\ \texttt{IP}(\texttt{S}_\texttt{A}) &\leftarrow \texttt{NbRepetition}(\texttt{S}_\texttt{A}, [16,\ \infty[). \end{aligned}$$

Ces règles sont simples. En réalité, elles ont été déjà apprises par notre système d'apprentissage. Le fait de déclarer ces règles comme des règles *a priori* permet seulement de modifier leurs niveaux de pertinence attribués automatiquement. Ces règles décrivent les faits que les segments de plus de 2 min 45 s sont des segments de programmes et que les segments issus d'occurrences de répétitions de plus de 16 occurrences sont des segments d'inter-programmes. Ce sont des règles dont nous sommes sûrs étant donnée la structure du flux des chaînes de France 2 et de TF1.

La figure 3.15 montre que ces quelques règles *a priori* permettent d'améliorer légèrement les résultats.


(a) France 2 (b) TF1

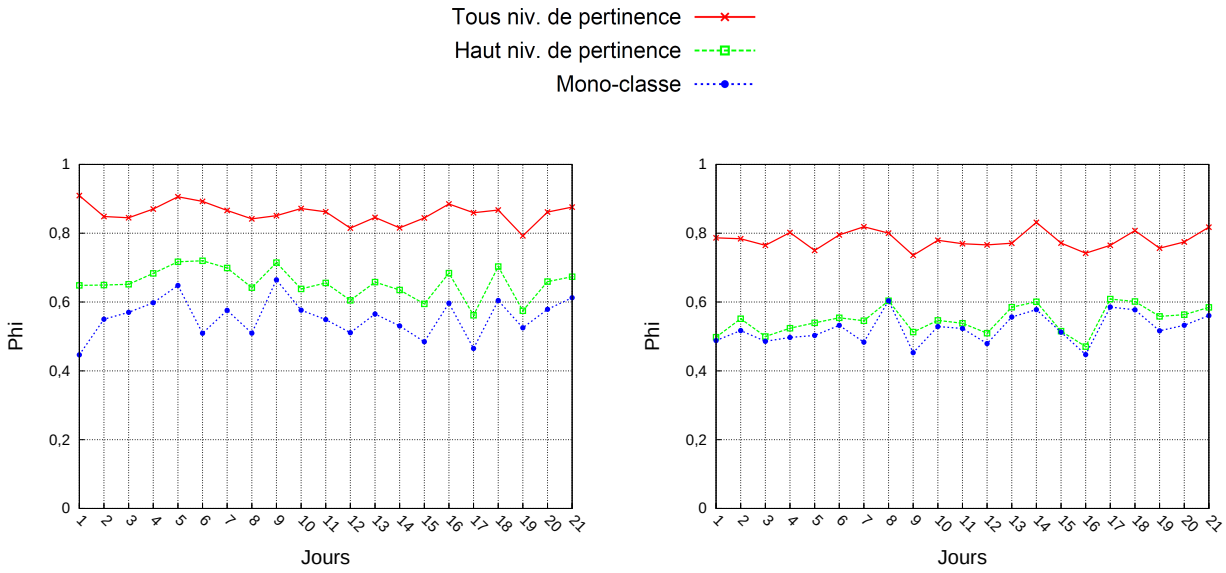
FIGURE 3.15 – Performances de Phi avec ou sans l'utilisation de règles *a priori*.

#### 3.3.3.3 Utilisation des niveaux de pertinence des règles

Nous avons défini plusieurs niveaux de pertinence des règles de classification. Ces niveaux présentés dans la section 3.2.4.4 peuvent être contrôlés lors de l'algorithme d'appli-

cation des règles détaillé dans la section 3.2.5. Nous avons déterminé 3 types de configuration dans l'utilisation de ces niveaux de pertinence. Dans la première configuration, notée « Mono-classe », nous utilisons seulement les règles de type mono-classe d'un haut niveau de pertinence. Pour cela, nous interdisons tous les niveaux de pertinence des règles multi-classes. Dans la seconde configuration, « Haut niv. de pertinence », nous utilisons à la fois les règles multi-classes et les règles mono-classe d'un haut niveau de pertinence seulement. Enfin dans la troisième configuration, « Tous niv. de pertinence » , nous utilisons toutes les règles suivant tous les niveaux de pertinence.

La figure 3.16 montre les résultats obtenus sur chaque jour des semaines de test des deux chaînes étudiées. Les résultats montrent qu'il est plus intéressant d'utiliser toutes les règles mono-classe et multi-classes suivant tous les niveaux de pertinence lors de l'application de notre algorithme de classification. Les performances des règles d'un haut niveau de pertinence permettent seulement de classer les segments avec une précision meilleure mais un rappel moins élevé. Elles ne sont pas adaptées à la classification de tous les segments.




(a) France 2 (b) TF1

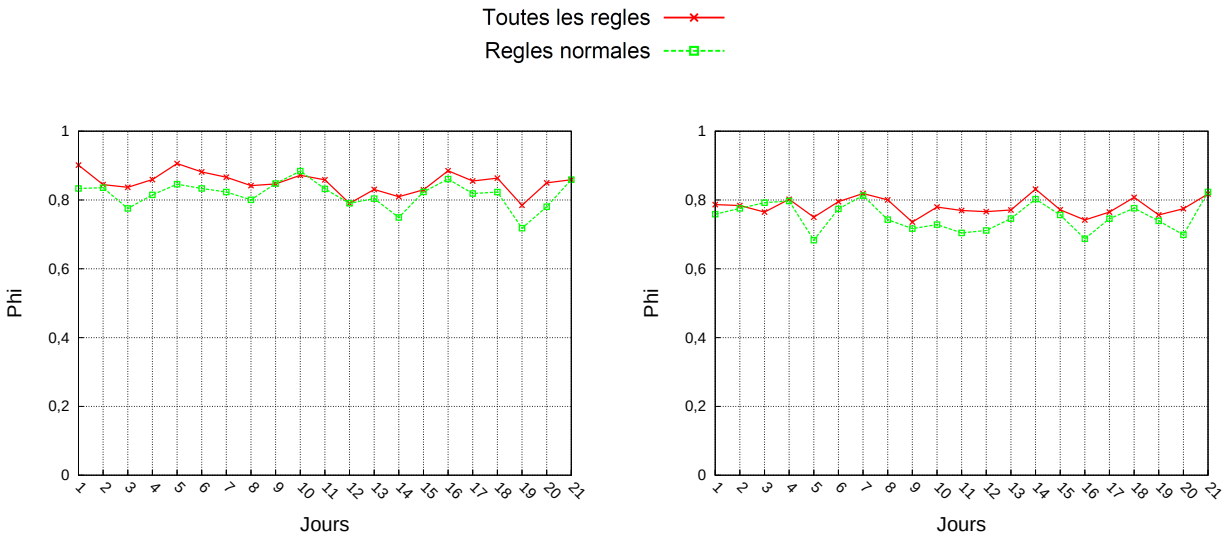
FIGURE 3.16 – Performances de Phi suivant trois principales configurations d'utilisation des niveaux de pertinence.

#### 3.3.3.4 Variation du nombre de règles apprises

Nous avons détaillé dans la section 3.2.4 le système *Aleph* [Sri07] de programmation logique inductive que nous avons utilisé. Nous avons configuré ce système suivant un certains nombres de biais et de paramètres que nous avons déterminé de manière empirique. Ces diverses configurations permettent en réalité de contrôler le nombre de règles apprises par la programmation logique inductive. Nous avons isolé deux configurations principales

d'utilisation d'*Aleph*. La différence entre les deux solutions est le nombre de règles apprises. La configuration « Règles normales » utilise les règles fournies normalement par *Aleph*. La configuration « Toutes les règles » utilise, en plus, les règles crées par *Aleph* qui sont rejetées car une règle meilleure suivant le score d'évaluation a été trouvée.

La figure 3.17 montre les résultats suivant le nombre de règles apprises. Il apparait que la configuration utilisant un nombre plus important de règles fournit de meilleurs résultats. Nous conservons donc cette configuration. Le fait d'utiliser un nombre plus important de règles joue en notre faveur car nous obtenons également plus de règles d'un haut niveau de pertinence. Nous devons cependant faire attention à ne pas réaliser un sur-apprentissage.




(a) France 2 (b) TF1

FIGURE 3.17 – Performances de Phi suivant le nombre de règles apprises.

Les règles de classification que nous utilisons se répartissent en 4 niveaux de pertinence et en 2 types : mono-classe et multi-classes. Nous avons calculé le nombre de règles de classification apprises pour chaque type et chaque niveau de pertinence. Nous présentons ces nombres dans le tableau 3.3 et le tableau 3.4. Un nombre similaire de règles de classification a été appris sur les deux chaînes. Le tableau montre que plus de 40 % des règles conservent un niveau de pertinence élevé. Nous pourrions donc envisager un plus grand nombre de niveaux de pertinence pour mieux différencier les règles.

#### 3.3.3.5 Analyse des résultats de classification

Les précédentes expériences nous ont permis de configurer notre méthode de classification des segments en segments d'inter-programme et en segments de programme. Nous analysons à présent les résultats de manière plus approfondie.

Afin de mettre en perspective les résultats obtenus, nous avons sélectionné 3 méthodes simples de classification des segments. Ces méthodes sont les suivantes :

| Classe | type de règles<br>Pertinence | Niv. 0<br>basse | Niv. 1<br>moyenne | Niv. 2<br>haute | Niv. 3<br>très haute |
|---|---|---|---|---|---|
| Inter-programme | mono-classe | 7 | 7 | 28 | 33 |
| | multi-classes | 10 | 2 | 20 | 27 |
| Programme | mono-classe | 8 | 3 | 17 | 28 |
| | multi-classes | 13 | 1 | 9 | 4 |

Tableau 3.3 – Nombres de règles par niveau de pertinence sur France 2.

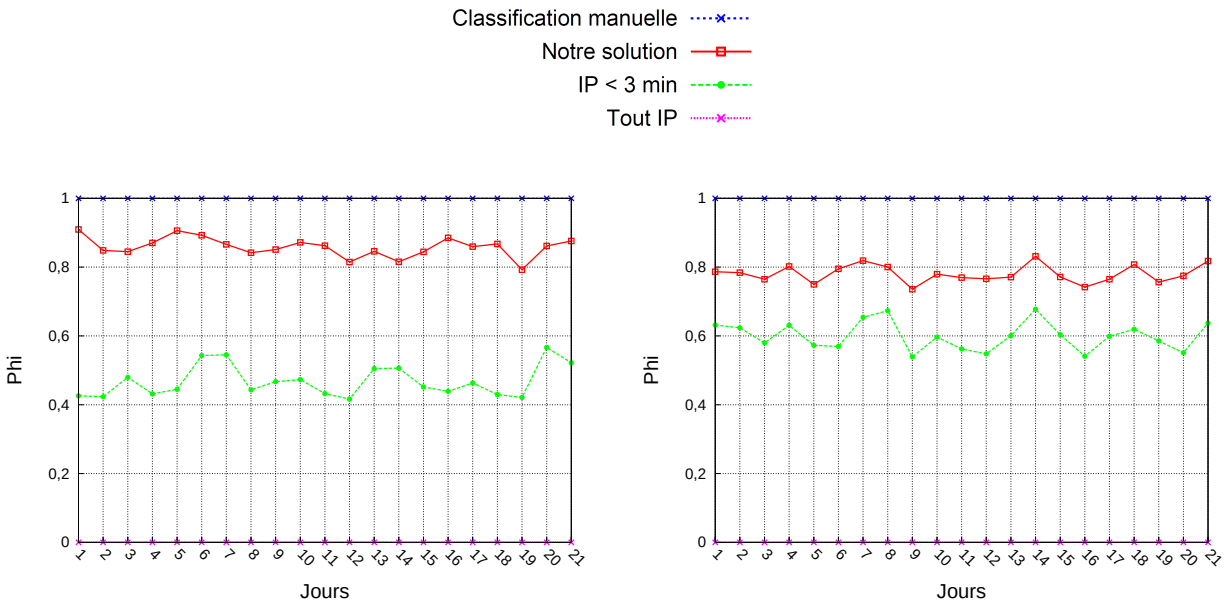
| Classe | type de règles<br>Pertinence | Niv. 0<br>basse | Niv. 1<br>moyenne | Niv. 2<br>haute | Niv. 3<br>très haute |
|---|---|---|---|---|---|
| Inter-programme | mono-classe | 12 | 8 | 26 | 17 |
| | multi-classes | 5 | 8 | 35 | 71 |
| Programme | mono-classe | 12 | 5 | 17 | 11 |
| | multi-classes | 12 | 2 | 4 | 2 |

Tableau 3.4 – Nombres de règles par niveau de pertinence sur TF1.

1) « Classification manuelle » classe chacun des segments suivant leur classe dans la vérité terrain. La classification obtenue est donc manuelle. Elle représente la classification idéale à atteindre par notre méthode, compte-tenu du découpage du flux réalisé ;
2) « Tout IP » classe tous les segments comme des inter-programmes. Il est important que notre méthode soit largement supérieure à cette solution naïve ;
3) « IP $<$ 3 min » utilise un simple seuil sur la durée des segments. Un segment est alors classé comme un segment d'inter-programme si sa durée est inférieure à 3 minutes. Cette solution est le type de solution souvent utilisée par défaut dans les méthodes existantes présentées dans le chapitre 1 [CBF06, ZZZY08].

La figure 3.18 montre les résultats. Comme il était prévisible, les résultats de la « Classification manuelle » sont maximaux et les résultats de « Tout IP » sont à zéro. Les résultats de la méthode « IP $<$ 3 min » restent relativement élevés mais notre solution est bien nettement supérieure. La méthode « IP $<$ 3 min » bénéficie de ces résultats car elle classe correctement au moins tous les inter-programmes. Tous les inter-programmes possèdent en effet une durée inférieure à 3 minutes. Au final, notre solution obtient des résultats performants par rapport aux classifications de référence. Les résultats restent bien stables pour chaque journée de notre ensemble de test. Notre solution vérifie donc notre contrainte clé de continuité.

Comme ces résultats l'indiquent aussi, notre solution effectue quelques erreurs. Pour mesurer l'efficacité globale de notre solution, nous présentons ainsi les matrices de confu-




(a) France 2 (b) TF1

FIGURE 3.18 – Comparaison des performances de Phi de la classification des segments.

sion dans le tableau 3.5. Ces matrices de confusion correspondent aux résultats de classification de l'ensemble des jours de test.

| | IP | P | ∅ |
|---|---|---|---|
| Inter-programme (IP) | 7682 | 240 | 26 |
| Programme (P) | 434 | 3190 | 2 |

(a) France 2

| | IP | P | ∅ |
|---|---|---|---|
| Inter-programme (IP) | 12517 | 91 | 0 |
| Programme (P) | 979 | 2183 | 1 |

(b) TF1

TABLEAU 3.5 – Matrices de confusion de notre classification des segments en segments d'inter-programme et en segments de programme sur les 3 semaines de test.

À partir des résultats présentés dans le tableau 3.5, nous pouvons calculer des mesures de précision et de rappel pour la classification des segments en segments d'inter-programme. À titre indicatif, nous donnons également les mesures de précision et de rappel en termes de plans correctement classés et d'images correctement classées. Tous ces résultats sont donnés dans le tableau 3.6.

Lors de l'analyse visuelle des résultats, nous avons identifié trois types d'erreurs. La première erreur courante est un segment d'inter-programme qui précède ou suit un segment de programme. Ces segments que nous appelons segments de « frontière » sont majoritaire-

| Inter-programmes | Précision | Rappel |
|---|---|---|
| Segments | 94,65 % | 96,65 % |
| Plans | 94,97 % | 93,54 % |
| Images | 89,79 % | 96,54 % |

(a) France 2

| Inter-programmes | Précision | Rappel |
|---|---|---|
| Segments | 92,75 % | 99,28 % |
| Plans | 91,50 % | 95,71 % |
| Images | 88,81 % | 98,40 % |

(b) TF1

TABLEAU 3.6 – Performances en termes de précision et de rappel de la classification des segments en inter-programmes sur les 3 semaines de test.

ment des parrainages. Ils sont incorrectement classés comme segments de programme. La deuxième erreur fréquente est, à l'inverse, un segment court de générique de programme classé en tant qu'inter-programme. Enfin, la troisième erreur courante est un segment de programme court classé en tant que segment d'inter-programme. Ces types d'erreurs partagent un point commun. Elles concernent, en effet, toutes des segments situés aux limites de zones de segments de la même classe. Ce sont des segments de séparation.

Pour mesurer la part de chaque type d'erreurs, nous proposons dans le tableau 3.7 des matrices de confusion détaillées. Ces matrices montrent quelles classes ont été attribuées aux différents types de segments de séparation. Les erreurs que nous avons notées visuellement sont confirmées. Les segments de frontière représentent environ la moitié des erreurs de classification des programmes. Les génériques et les programmes courts représentent environ les trois quarts des erreurs de classification des inter-programmes. Notre solution rencontre donc des difficultés pour classer les segments de séparation, c'est à dire les parrainages, les génériques et les programmes courts. Nous verrons dans le chapitre 4 qu'elles en sont les conséquences sur le processus de délinéarisation.

Une des difficultés pour classer les segments de séparation est que les distributions des occurrences des répétitions des segments de séparation sont souvent similaires. Par exemple, un parrainage est souvent associé à un programme spécifique. Il est donc rediffusé de la même manière que le générique de ce programme. Cela rend les génériques et les parrainages difficiles à séparer à partir de nos descripteurs. Une solution envisageable serait d'ajouter des descripteurs supplémentaires comme la détection de textes dans les images.

Une autre difficulté pour classer ces segments de séparation provient en partie de notre configuration du découpage du flux en segments. Nous avons, en effet, choisi de ne pas utiliser les *clusters* solitaires au cours des expériences de la section 2.5.5. Nous rappelons que ces *clusters* solitaires généraient un sur-découpage du flux. L'inconvénient de ce choix était alors que les parrainages étaient moins bien détectés. Pour la méthode de classification, la réduction de la détection des parrainages fournit moins d'exemples fiables à partir desquels généraliser de bonnes règles de classification.

#### 3.3.3.6 Comparaison avec un classifieur supervisé binaire classique

Pour compléter l'évaluation de notre méthode de classification, nous avons comparé les résultats de notre solution avec les résultats obtenus par un classifieur supervisé binaire classique. Les machines à vecteurs de supports (SVM – *support vector machine*) sont

| | | IP | P |
|---|---|---|---|
| IP | Segments de frontière | 1418 | 110 |
| P | Génériques | 138 | 472 |
| | Programmes courts | 188 | 451 |

(a) France 2

| | | IP | P |
|---|---|---|---|
| IP | Segments de frontière | 1906 | 54 |
| P | Génériques | 403 | 239 |
| | Programmes courts | 358 | 421 |

(b) TF1

TABLEAU 3.7 – Matrices de confusion détaillées suivant trois types de segments particuliers sources d'erreurs.

de bons exemples de classifieurs classiques. Ils ont, en effet, montré leur performances dans de nombreux problèmes de classification. Nous avons utilisé l'implémentation de *libSVM* [CL01] et nous avons configuré notre SVM avec un noyau gaussien.

L'avantage des SVM par rapport à la programmation logique inductive est le fait qu'ils produisent une réponse simple. Il n'y a pas d'algorithme d'utilisation de règles de classification. Les SVM utilisent, de plus, des descripteurs numériques. Il n'y a donc pas besoin de catégoriser les données.

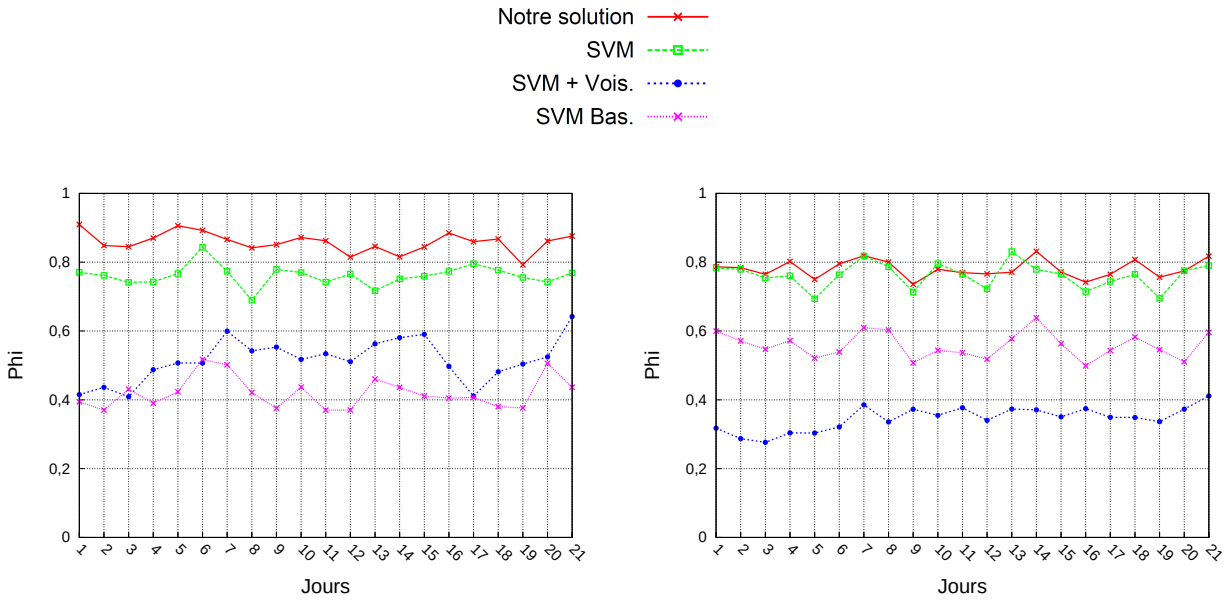
Nous avons sélectionné trois façon d'utiliser les SVM :

1) « SVM Bas. » utilise seulement les descripteurs de durée, de nombre d'occurrences d'une répétition et de la fréquence des plans pour chaque segment. Des vecteurs de 3 dimensions sont donnés en entrée d'un SVM ;

2) « SVM » utilise simplement tous les descripteurs des prédicats simples dont, en particulier, les descripteurs des distributions des occurrences des répétitions. Des vecteurs de 21 dimensions sont donnés en entrée d'un SVM ;

3) « SVM + Vois. » utilise tous les descripteurs des prédicats simples et tous les descripteurs des prédicats simples des segments précédant et suivant chaque segment. Des vecteurs de 63 dimensions sont donnés en entrée d'un SVM.

Les résultats sont présentés dans la figure 3.19. Les trois façons d'utiliser les SVM nous informent de l'importance des descripteurs utilisés. La différence entre « SVM » et « SVM Bas. » montre que les descripteurs caractérisant les distributions des occurrences des répétitions apportent de précieuses informations pour la classification. Les mauvais résultats de « SVM + Vois. » traduisent la difficulté de modéliser des informations de voisinages avec les SVM. De manière générale, notre solution reste meilleure que les SVM en dépit des difficultés à intégrer ces informations sur TF1. Elle profite de la programmation logique inductive qui modélise les informations contextuelles et relationnelles que nous n'avons pas pu intégrer aux SVM. Ces derniers résultats valident nos hypothèses pour la classification des segments à partir des propriétés relatives aux distributions des occurrences des répétitions et des informations contextuelles et relationnelles entre les segments.




(a) France 2 (b) TF1

FIGURE 3.19 – Comparaison des performances de Phi de la classification des segments avec un SVM.

### 3.3.4 Expériences 2 : classification des segments en segments de programme long et en segments qui ne sont pas des programmes longs

L'objectif des expériences de cette section est simplement de fournir les résultats de la classification en segments de programme long et en segments qui ne sont pas des programmes longs (les programmes courts et les inter-programmes). Nous utilisons les résultats de cette classification dans le chapitre 4 suivant.

Nous rappelons qu'un programme long est un programme qui possède une durée supérieure à 5 minutes comme nous l'avons défini en introduction. L'intérêt de classer les segments en segments de programme long est d'extraire uniquement les parties de programmes longs. Ces parties peuvent ensuite être regroupées ou réunifiées en programmes longs entiers. Cette classification est ainsi importante pour l'extraction des programmes TV du dernier niveau de délinéarisation.

De plus, les segments courts sont indirectement détectés par cette classification. Les segments courts sont, en effet, les segments qui sont classés en segments de programme par la classification de la section 3.3.3 précédente mais qui ne sont pas classés en tant que programme long par cette classsification. Comme les programmes courts ne sont pas séparés par des coupures d'inter-programmes à cause de leur petite durée, les segments détectés comme programme court représentent des programmes courts extraits en entiers.

Pour cette classification, nous conservons la même configuration que pour la précédente

classification des segments en inter-programmes et en programmes (programmes courts et programmes longs). Nous affichons les performances globales de la classification dans le tableau 3.8. Ces performances ont été calculées sur les journées de 8h du matin à 1h le lendemain matin. C'est, en effet, sur cette période de la journée que l'extraction des programmes est évaluée.

| | PL | A | ∅ |
|---|---|---|---|
| Programme long (PL) | 1479 | 152 | 0 |
| Autre (A) | 241 | 6775 | 0 |

(a) France 2

| | PL | A | ∅ |
|---|---|---|---|
| Programme long (PL) | 1070 | 385 | 0 |
| Autre (A) | 121 | 11827 | 0 |

(b) TF1

Tableau 3.8 – Matrices de confusion de notre classification des segments en segments de programme long sur les journées des jours des 3 semaines de test.

Comme dans la section 3.3.3 précédente, la majorité des erreurs sont dues à des segments de séparation. L'impact de ses performances dans le processus de délinéarisation est évalué dans le chapitre 4 suivant.

### 3.3.5 Expériences 3 : classification des segments en catégories d'inter-programmes

Notre approche a été conçue dans un premier temps pour classer les segments de programme et les segments d'inter-programme. Néanmoins, les inter-programmes se divisent en plusieurs catégories. Les publicités, les bandes annonces et les parrainages sont les principales catégories d'inter-programmes. Nous avons donc cherché à classer les segments en ces différentes catégories en conservant notre approche.

La classification des inter-programmes apporte une description plus complète du flux TV. Elle n'est pas sans applications. Selon notre étude des méthodes existantes du chapitre 1, nous avons déjà vu que de nombreux travaux se sont intéressés à la publicité. Les publicités constituent en effet un point crucial dans le système économique des chaînes de télévision. En effectuant une classification des inter-programmes en catégories, nous proposons ainsi une méthode originale de détection de la publicité. En plus des publicités, nous proposons de détecter les parrainages. Comme ces derniers portent des messages commerciaux, ils rejoignent les publicités en ce qui concerne les applications. Ainsi, la pige des parrainages est tout aussi importante que la pige des publicités. Cependant les techniques actuelles de détection de publicités ne proposent pas de système de détection des parrainages. Enfin, la classification des inter-programmes détecte les bandes annonces. Les bandes annonces extraites peuvent être associées aux programmes qu'elles annoncent. Ces bandes annonces fournissent alors des aperçus pertinents déjà construits qui peuvent alimenter des catalogues de programmes TV dans un service de télévision à la demande.

La classification des inter-programmes en différentes catégories est une tâche annexe dans la délinéarisation automatique des flux TV. Elle ne contribue pas directement à l'extraction des programmes ou à la délimitation des inter-programmes.

Nous présentons rapidement notre méthode de classification des segments en catégories d'inter-programmes, puis nous donnons quelques résultats.

#### 3.3.5.1 Méthode de classification des segments en catégories d'inter-programmes

Notre méthode de classification des segments en catégories d'inter-programmes est la même que la méthode précédente de classification des segments. La seule différence est que nous manipulons plus de deux classes. Le système de vote présenté dans la section 3.2.5.3 reste valide pour plus de deux classes. Il permet bien de choisir une catégorie d'inter-programme pour chaque segment à classer. Au final, l'algorithme d'application des règles de classification s'exécute normalement.

Nous réalisons, malgré tout, un changement dans la méthode de classification. Nous ajoutons, en effet, une propriété logique pour la création des règles de classification. La propriété que nous ajoutons est la caractérisation possible d'un segment en tant que bande annonce à partir de l'analyse du *clustering* utilisé lors de la détection des répétitions. Nous avons déjà expliqué cette exploitation possible des répétitions pour la détection de bande annonce dans la section 2.6.

Cette propriété logique spécifique à certaines bandes annonces est ajoutée au module de description logique. Pour un segment $\mathtt{S_A}$, nous obtenons ainsi le prédicat simple supplémentaire défini en extension :

$$\mathtt{EstBandeAnnonce}(\mathtt{S_A}) \equiv \begin{array}{l}\text{« Le segment } \mathtt{S_A} \text{ est une occurrence d'une}\\ \text{bande annonce détectée grâce à l'analyse}\\ \text{du } \textit{clustering} \text{ utilisé lors de la détection}\\ \text{des répétitions. »}\end{array}$$

Ce nouveau prédicat permet d'apprendre des règles spécifiques aux bandes annonces avec un niveau de pertinence élevé. Par la suite, d'autres règles relationnelles peuvent s'appuyer sans risques sur les résultats de classification fournis par ces règles fiables.

#### 3.3.5.2 Résultats

Nous ne proposons qu'une étude préliminaire de la classification des segments en catégories d'inter-programmes. Nous avons limité notre corpus aux semaines n°1 et n°2 sur la chaîne France 2. Nous avons utilisé la première semaine pour l'apprentissage et la validation. Nous avons ensuite testé notre méthode sur la deuxième semaine de test.

Les résultats sont présentés dans la matrice de confusion du tableau 3.9. Ces résultats correspondent aux mesures de précision et de rappel données dans le tableau 3.10.

Les performances de notre méthode sont mitigées. Cette dernière souffre de la difficulté à classer les segments de séparation. Ainsi aucun parrainage n'est correctement classé. Les parrainages se répartissent presque équitablement entre les publicités et les programmes. La propriété spécifique à certaines bandes annonces issues de la détection des répétitions permet néanmoins de détecter les bandes annonces avec une bonne précision. Nous notons qu'il y a moins de segments de bande annonce que de bandes annonces dans la vérité terrain. Cela provient principalement des bandes annonces « dans un instant ... » non découpées par les répétitions, car celles-ci ne se répètent pas.

| | Pub | Bande annonce | Parrainage | Autres IP | Programme | Programme court |
|---|---|---|---|---|---|---|
| Pub | 1744 | 4 | 5 | 19 | 38 | 2 |
| Bande annonce | 131 | 194 | 5 | 17 | 32 | 23 |
| Parrainage | 36 | 6 | 0 | 0 | 39 | 10 |
| Autres IP | 48 | 0 | 2 | 197 | 24 | 11 |
| Programme | 34 | 4 | 7 | 10 | 983 | 5 |
| Programme court | 54 | 2 | 0 | 2 | 83 | 63 |

TABLEAU 3.9 – Matrice de confusion de la classification des segments en différentes catégories d'inter-programmes.

| | Précision | Rappel |
|---|---|---|
| Pub | 85,20 % | 96,25 % |
| Bande annonce | 92,38 % | 47,43 % |
| Parrainage | 0 % | 0 % |

TABLEAU 3.10 – Mesures de précision et de rappel de la classification des segments en différentes catégories d'inter-programmes.

### 3.3.6 Synthèse

Nous avons proposé une méthode de classification de segments audiovisuels. Cette méthode repose sur les propriétés de distribution des répétitions des segments. Elle emploie également des informations relationnelles et contextuelles. Ces informations sont apprises à partir d'un ensemble d'apprentissage. Elles traduisent l'agencement des programmes et des inter-programmes dans un flux TV, c'est-à-dire la structure sous-jacente du flux.

Nous avons évalué à travers des expériences les performances de notre méthode de classification. Ces expériences ont confirmé l'intérêt des informations contextuelles et relationnelles. Notre méthode fournit ainsi de bons résultats généraux pour la classification des inter-programmes et des programmes.

# Chapitre 4

# Extraction des programmes TV

## Sommaire



Nous présentons enfin le dernier niveau de délinéarisation de notre système. Dans ce niveau, les programmes TV sont extraits de chaque portion analysée du flux continu. Les programmes TV ainsi extraits peuvent ensuite être archivés ou bien conservés dans des catalogues pour alimenter automatiquement des nouveaux services de télévision à la demande.

Dans ce chapitre, nous traitons seulement de l'extraction des programmes. Nous reprenons dans la figure 4.1 le fonctionnement par portion de notre système. Nous montrons ainsi le contexte de l'extraction des programmes. Selon les niveaux de délinéarisation précédents, le flux est analysé périodiquement par portions de 24 heures. Chaque portion est découpée en une suite de segments classés en tant que segments de programme ou d'inter-programme.

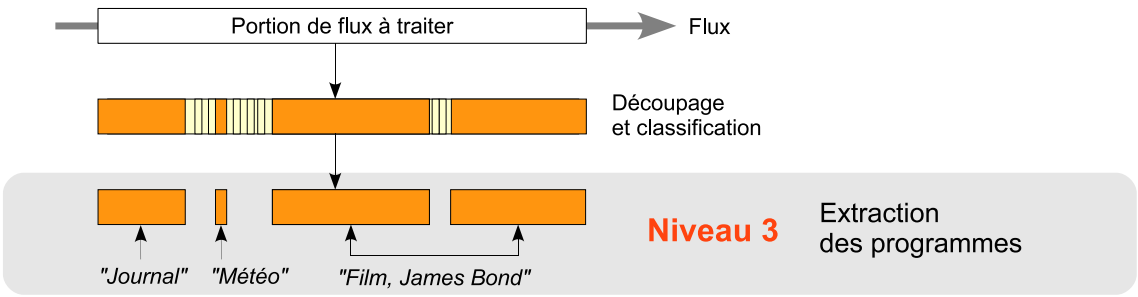



FIGURE 4.1 – Contexte du niveau de délinéarisation pour l'extraction des programmes TV.

L'extraction des programmes repose principalement sur la résolution de deux problèmes :

1) **la réunification des programmes.** Un programme TV peut, en effet, être diffusé en plusieurs parties correspondant à plusieurs segments de programme. L'extraction du programme *entier* dépend donc de la reconstruction de ce programme à partir des segments qui le composent : c'est la réunification. Si le programme TV est constitué d'un seul segment de programme, il est important de distinguer ce segment des autres segments de programme adjacents ;

2) **l'étiquetage des programmes.** Les programmes TV sont couramment identifiés par des titres dans les guides de programmes. Ces titres permettent une indexation des programmes pour les services de télévision à la demande. Pour que l'extraction automatique soit complète il est important de déterminer ces titres : c'est l'étiquetage.

Lorsque les métadonnées du flux télévisuel sont disponibles, celles-ci fournissent des titres et des horaires approximatifs de certains des programmes diffusés. Ce sont de précieuses informations pour la réunification et l'étiquetage. Cependant, l'analyse des métadonnées effectuée dans le chapitre 1 montre que les métadonnées sont imprécises et souvent incomplètes. Selon leur disponibilité, leur précision et leur complétude, les métadonnées ne permettent l'extraction que d'une partie des programmes. Tous les programmes ne peuvent pas être extraits en utilisant les métadonnées.

Nous proposons donc, dans ce chapitre, d'abord une méthode d'extraction automatique de certains programmes à partir des métadonnées, puis une méthode d'extraction du reste des programmes basée sur le contenu audiovisuel seulement. La méthode basée sur le contenu audiovisuel n'effectue qu'une réunification automatique. L'étiquetage y devient manuel. Ces deux méthodes sont finalement expérimentées. Les résultats de la première méthode montrent alors les performances globales de notre solution pour la délinéarisation automatique des flux télévisuels.

## 4.1 Extraction des programmes basée sur les métadonnées

Notre système entier de délinéarisation automatique des flux TV repose, dans se phase finale, sur l'utilisation des métadonnées éventuelles qui accompagnent les flux TV. Ces métadonnées (de type EPG ou EIT) permettent d'extraire automatiquement certains programmes. Elles contiennent trois types d'information importante :

1) les titres des programmes TV diffusés,

2) les horaires approximatifs des programmes TV diffusés,

3) l'ordre de diffusion des programmes TV diffusés.

Les métadonnées peuvent s'interpréter comme des suites de segments temporels possédant un titre. Ces segments sont notés « segments de métadonnées » dans le reste de ce chapitre. De manière directe, le problème d'extraction des programmes TV peut alors être vu comme un problème de mise en correspondance. En effet, les segments automatiquement découpés et classés en programme forment des parties de programmes TV comme cela est illustré dans la figure 4.2. Les segments de métadonnées représentent les programmes TV de ces parties de programmes. Par conséquent, les segments automatiquement découpés correspondent temporellement avec les segments de métadonnées.

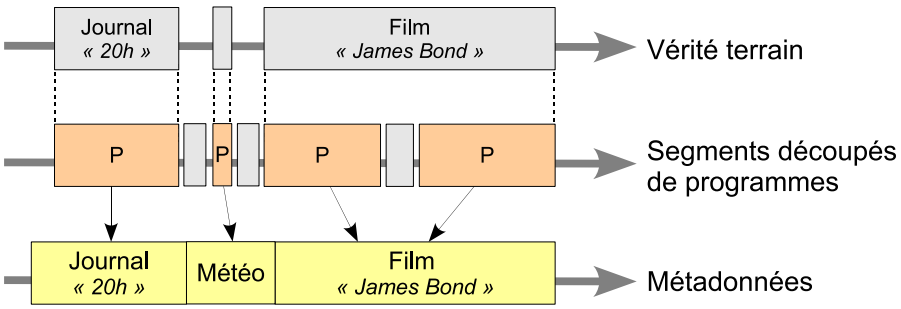



FIGURE 4.2 – Principe de la mise en correspondance des segments découpés de programme avec les métadonnées.

La mise en correspondance résout bien à la fois la problématique de réunification et d'étiquetage. La réunification est effectuée lorsque plusieurs segments découpés correspondent avec un même segment de métadonnées. L'étiquetage a lieu à chaque mise en correspondance. Chaque segment découpé mis en correspondance avec un segment de métadonnées reçoit le titre du segment de métadonnées avec lequel il s'est associé.

L'efficacité de la mise en correspondance dépend malheureusement de la justesse du découpage et de la fiabilité des métadonnées. En pratique, les segments découpés sont souvent sur-segmentés et les métadonnées sont imprécises et incomplètes. Il existe malgré tout un ensemble de programmes qui peuvent s'extraire par cette procédure de mise en correspondance. C'est l'ensemble des programmes longs. Nous présentons alors notre méthode de mise en correspondance locale des segments de programme découpés avec les programmes longs des métadonnées.

### 4.1.1 Les programmes extractibles à partir des métadonnées : les programmes longs

Nous séparons les programmes en deux groupes : les programmes longs et les programmes courts. Les programmes longs sont des programmes qui durent plus de 5 minutes. Ils s'agit de la majeure partie des éléments utiles de la télévision tels les films, les journaux, les documentaires, les magazines, les séries, etc. Les programmes courts sont à l'inverse des programmes de moins de 5 minutes. Ce sont, par exemple, des bulletins météo, des magazines courts d'information pratique, des résumés d'évènements sportifs, des *clips* musicaux, des jeux concours, des résultats de jeux, etc.

L'étude de la qualité des métadonnées présentée dans le chapitre 1 montre que les programmes courts et les programmes longs ne sont pas égaux vis à vis des métadonnées. Les programmes courts annoncés sont en effet souvent plus incomplets que les programmes longs. De plus, les imprécisions indiquées dans les métadonnées ne possèdent pas le même impact sur les programmes courts ou sur les programmes longs. Une erreur d'approximation de 5 minutes dans les horaires des métadonnées a plus de conséquence sur un programme de 2 minutes que sur un programme d'une heure. Dans le cas d'un programme court, l'horaire erroné peut correspondre avec l'horaire du programme court suivant ou précédent. Enfin, nous avons remarqué que l'ordre de diffusion des programmes courts annoncé dans les métadonnées n'est pas toujours respecté.

La figure 4.3 compare une suite de programmes courts diffusés dans le flux avec leurs équivalents annoncés dans les métadonnées. Les flèches épaisses montrent les associations à effectuer par une mise en correspondance. L'exemple montre en particulier pour la météo, que l'élément correspondant à la météo dans les métadonnées est annoncé avec quelques minutes d'avance tel qu'il semble plus correspondre avec le PMU. L'exemple est pris lors d'une soirée sur TF1, après le journal télévisé. La situation n'est donc pas rare. Cela montre à quel point il peut être ambigu de mettre en correspondance des segments de programmes courts avec les métadonnées.

L'imprécision des métadonnées relatives aux programmes courts provient en partie du fait que certains de ces programmes servent à combler des trous de programmation. Si du temps de diffusion reste à définir et que les quotas de diffusion de publicité sont atteints, il arrive que certains programmes courts soient alors rediffusés pour compléter les manques dans le flux. Ces programmes courts particuliers sont donc difficiles à prévoir de façon fiable dans les métadonnées.

Bien que les programmes courts soient difficiles à extraire à partir des métadonnées, les programmes longs restent relativement bien annoncés dans les métadonnées et leurs imprécisions temporelles impactent moins la mise en correspondance. Nous proposons alors une première méthode spécifique pour extraire les programmes longs. Dans la section 4.2.1 suivante nous exposons une solution de secours pour extraire les programmes courts.

### 4.1.2 Mise en correspondance locale des segments découpés avec les programmes longs des métadonnées

Notre méthode d'extraction de programmes basée sur les métadonnées ne s'applique qu'aux programmes longs annoncés dans les métadonnées. Puisque nous ne travaillons que sur les programmes longs, nous choisissons de classer spécifiquement, à partir de la programmation logique inductive, les segments en segments de programmes longs ou non.

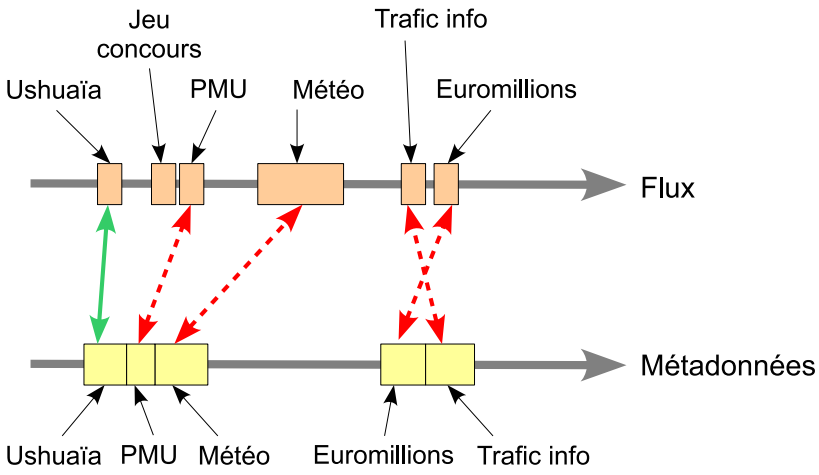



FIGURE 4.3 – Comparaison d'une suite de programmes courts diffusés et leurs équivalents dans les métadonnées. Les flèches épaisses montrent les correspondances à effectuer. Les flèches en pointillés montrent les difficultés.

Nous remplaçons donc la classification en segments de programme et d'inter-programme par une classification en segment de programme long ou non. Les résultats expérimentaux de cette classification sont donnés dans la section 3.3.4. Nous obtenons donc une suite de segments de programmes longs à mettre en correspondance avec les programmes longs des métadonnées.

Nous choisissons une méthode de mise en correspondance locale. Cette technique associe à chaque segment découpé le segment de métadonnées qui lui correspond le mieux. La mise en correspondance est dite locale par opposition à une mise en correspondance globale de type *Dynamic Time Warping* (DTW). La DTW a été employée par Naturel [Nat07] pour l'étiquetage de segments de programmes courts et longs. L'auteur se heurtait de la même manière à un mauvais étiquetage des programmes courts.

La mise en correspondance globale permet à l'ensemble des segments de métadonnées de correspondre au mieux à l'ensemble des segments de programme long découpés automatiquement. Les incomplétudes des métadonnées influent sur la mise en correspondance de l'ensemble des segments découpés. Pour limiter l'influence des incomplétudes des métadonnées, nous avons préféré une mise en correspondance locale. Les expériences de la section 4.3 permettent de comparer les performances de ces deux méthodes de mise en correspondance. Elles confirment notre choix.

Une manière simple d'effectuer la mise en correspondance locale est de projeter les segments de métadonnées sur les segments découpés de programme long. Cela est représenté dans la figure 4.4.

Cette méthode de mise en correspondance est très simple. Tous les segments découpés qui sont mis en correspondance avec le même segment de métadonnées récupèrent le même titre. Ces segments sont alors réunifiés en un seul programme TV qui porte le titre commun à ces segments.

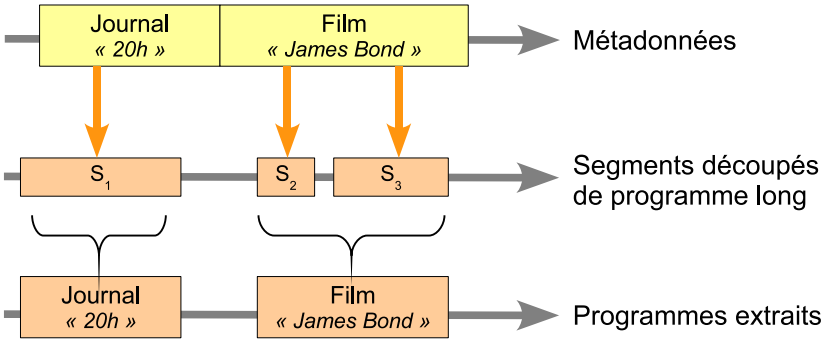



FIGURE 4.4 – Extraction des programmes longs à partir de la projection des métadonnées sur les segments découpés.

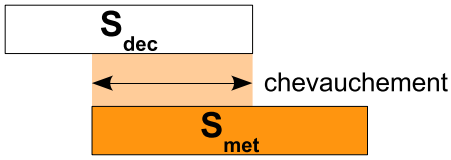



FIGURE 4.5 – Illustration du chevauchement entre deux segments $S_{dec}$ et $S_{met}$.

Pour projeter les métadonnées sur les segments découpés de programme, nous employons une mesure de chevauchement. La durée d'un segment $S$ est notée $|S|$. Le taux de chevauchement $tc_{ext}(S_{dec}, S_{met})$ d'un segment $S_{dec}$ découpé avec un autre segment $S_{met}$ de métadonnées est illustré dans la figure 4.5. Il s'exprime par le rapport :

$$tc_{ext}(S_{dec}, S_{met}) = \frac{|S_{dec} \cap S_{met}|}{|S_{dec}|}$$

Notre méthode d'alignement local fonctionne en deux étapes :

1) la première étape prétraite les segments découpés. Cette étape réunifie deux segments de programme long si ces segments sont séparés par une durée inférieure à 1 seconde. Ce prétraitement réunifie les segments de programme long consécutifs.
2) la deuxième étape sélectionne pour chaque segment découpé de programme long le segment de métadonnées pour lequel le taux de chevauchement $tc_{ext}$ est maximum.

## 4.2 Extraction des programmes basée sur le contenu audiovisuel

La méthode présentée précédemment permet d'extraire automatiquement des programmes longs lorsque les métadonnées sont disponibles. Pour pallier l'incomplétude et

l'indisponibilité des métadonnées, nous proposons une deuxième méthode d'extraction des programmes TV. Cette deuxième méthode extrait tous les programmes TV longs et courts à partir des caractéristiques audiovisuelles seulement. Cependant, elle n'effectue qu'une réunification automatique basée contenu. L'étiquetage automatique de l'extraction n'est pas traité. Les programmes réunifiés sont alors étiquetés manuellement.

Cette deuxième méthode propose seulement une délinéarisation semi-automatique des flux TV. Cette solution est cependant intéressante car l'intervention humaine reste faible et limitée à l'étiquetage des programmes automatiquement délimités.

Nous ne proposons pas de solution à l'étiquetage automatique basé sur l'analyse du contenu audiovisuel. Il existe toutefois quelques travaux qui se sont intéressés précisément à ce problème. En particulier, Jin *et al.* [JH01] analysent des transcriptions de la parole pour construire des titres à des reportages de journaux. Certains travaux [MBD+06] cherchent à inférer des métadonnées à partir du contenu télévisuel. Ces métadonnées inférées servent, ensuite, à indexer le contenu télévisuel afin de faciliter les recherches ultérieures. De manière plus générale, toutes les techniques d'indexation audiovisuelle [SW05a] peuvent s'appliquer sur les programmes réunifiés pour explorer le flux télévisuel à partir de requêtes utilisant des mots clés, des images clés ou des vidéos clés.

Nous présentons en premier notre méthode de réunification basée contenu, puis nous détaillons les caractéristiques audiovisuelles qu'elle utilise.

### 4.2.1 Méthode de réunification basée contenu des programmes TV

Dans notre méthode, nous avons fait le choix de traiter de nouveau les programmes courts et les programmes longs séparément.

#### 4.2.1.1 Extraction des programmes courts

Les programmes courts possèdent la propriété de n'être jamais entrecoupés par des coupures publicitaires. Par conséquent, chaque programme court se constitue d'un seul segment découpé de programme. Les programmes courts sont donc extraits directement à l'issue de la classification des segments. De manière plus précise, les programmes courts sont les segments qui sont classés, par la programmation logique inductive, en tant que programmes, lors de la classification des segments en programmes et en inter-programmes, mais qui ne sont pas classés en tant que programmes longs, lors de la classification des segments en programmes longs et en segments qui ne sont pas des programmes longs.

#### 4.2.1.2 Extraction des programmes longs

Pour réunifier les programmes longs, nous nous basons sur la classification des segments en programmes longs. Nous cherchons en effet à réunifier les parties des programmes longs. Nous n'avons donc pas besoin de traiter les segments de programmes courts. Cette classification est effectuée grâce à la programmation logique inductive. Les résultats expérimentaux de cette classification sont donnés dans la section 3.3.4.

La réunification automatique basée contenu consiste à regrouper ensemble tous les segments de programme long découpés qui appartiennent à un même programme TV. Cette opération est illustrée dans la figure 4.6. Dans cet exemple, les segments A et B sont à séparer alors que les segments B et C sont à réunifier. Un groupe de segments

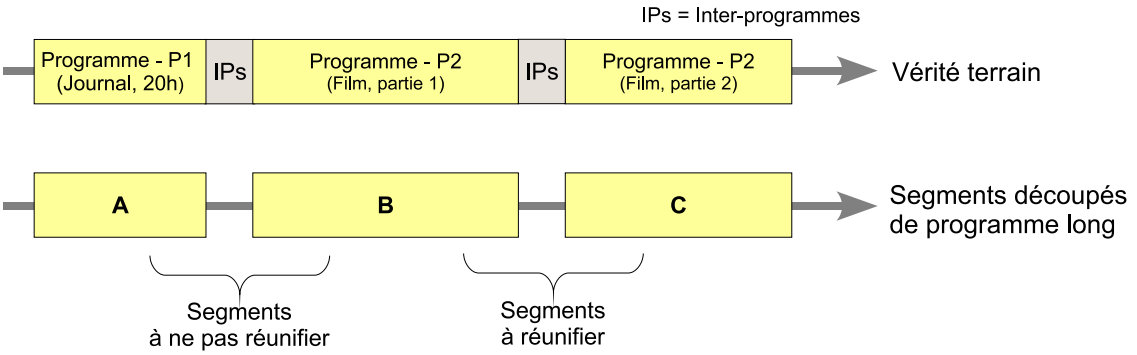



FIGURE 4.6 – Illustration de la réunification basée contenu des segments de programme long.

qui appartiennent à un même programme TV définissent au final un seul segment de programme TV dont la borne inférieure est la plus petite borne inférieure du groupe de segments et dont la borne supérieure est la plus grande borne supérieure du groupe de segments. À l'issue de la réunification, nous obtenons un ensemble de programmes longs. Il ne manque plus qu'une annotation de ces programmes pour obtenir une extraction complète.

Le problème de la réunification basée contenu ressemble au problème de construction des scènes [YYL96, Han02] à partir des segments de plans. Cependant, ces deux problèmes restent bien distincts. Les segments de programme sont dans un premier temps plus complexes à comparer que deux simples plans. De plus, un programme TV se répartit sur seulement quelques parties de programme alors qu'une scène peut se composer de plusieurs centaines de plans. Nous ne nous sommes donc pas inspiré des techniques de construction des scènes.

Les groupes de segments qui forment un seul et même programme sont en réalité des groupes de segments consécutifs. Il n'est donc pas nécessaire de comparer tous les segments entre eux afin de les regrouper en programmes TV. En effet, les différentes parties des programmes TV ne s'entrecroisent pas pour un souci évident de cohérence dans la diffusion. Nous proposons donc de comparer chaque couple de segments consécutifs. Pour chacun de ces couples, nous cherchons une réponse binaire : soit les deux segments du couple appartiennent à un même programme TV, soit les deux segments du couple n'appartiennent pas à un même programme TV.

Pour réaliser ce choix binaire, nous utilisons un classifieur supervisé binaire classique. Celui-ci détermine si deux segments consécutifs appartiennent ou non à un même programme TV à partir de caractéristiques audiovisuelles et d'un ensemble d'exemples de segments à réunifier ou à ne pas réunifier. Le classifieur considère ainsi deux classes : « à réunifier » et « à ne pas réunifier ». Cela est illustré dans la figure 4.7. Nous avons sélectionné une machine à vecteurs de supports (SVM – *support vector machine*) comme classifieur. Nous n'avons pas utilisé la programmation logique inductive car nous n'avons pas dans ce contexte besoin de modéliser de relations complexes entre les segments consécutifs. Nous souhaitons simplement utiliser des caractéristiques numériques et non logiques.

Les SVM sont des techniques bien connues de classification supervisée binaire. Ils sont performants dans de nombreux problèmes de classification. Ils traitent des caractéristiques numériques. Ils possèdent peu de paramètres à configurer et ils prennent facilement en compte des vecteurs de caractéristiques de grande dimension. Nous avons utilisé l'implémentation de *libSVM* [CL01] et nous avons configuré notre SVM avec un noyau gaussien.

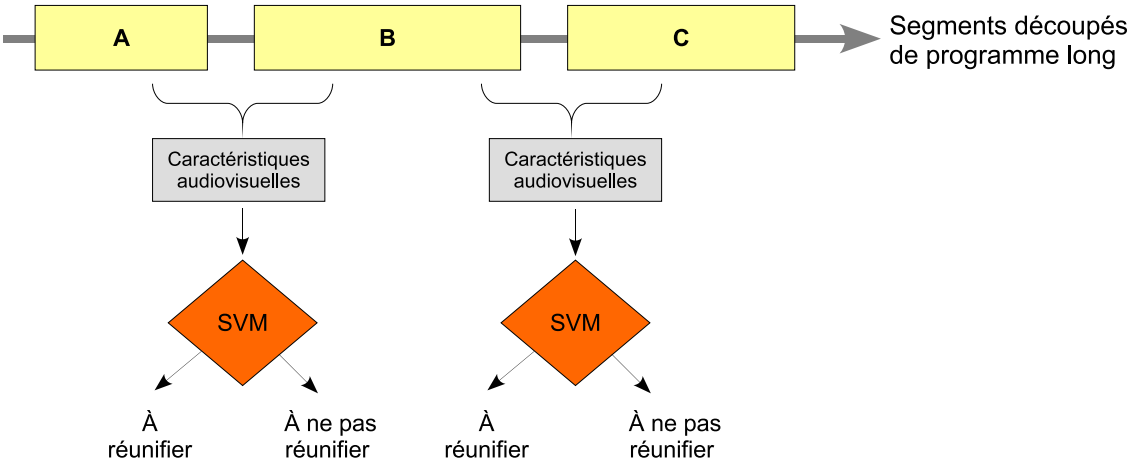



FIGURE 4.7 – Schéma de notre solution de réunification basée contenu utilisant un classifieur SVM.

### 4.2.2 Caractéristiques audiovisuelles utilisées pour la réunification basée contenu

Notre méthode de réunification basée sur le contenu audiovisuel repose sur le choix d'un certain nombre de caractéristiques audiovisuelles qui traduisent l'appartenance de deux segments à un même programme. Notre méthode repose sur deux points :

1) un même programme TV reste homogène selon certaines caractéristiques tout au long de sa durée ;
2) les caractéristiques de deux programmes consécutifs différents sont différentes.

Pour analyser l'homogénéité d'un programme TV tout au long de sa durée, chaque couple de segments consécutifs est décrit par un ensemble de caractéristiques audiovisuelles que nous répartissons en quatre groupes : les caractéristiques de plans, les caractéristiques de couleurs, les caractéristiques de visages et les caractéristiques de répétitions. Ces caractéristiques produisent un vecteur de caractéristiques multidimensionnel. Ce vecteur est donné en entrée du classifieur SVM pour chaque couple de segments.

#### 4.2.2.1 Caractéristiques de plans

Les changements de plans suivent généralement la narration des programmes TV. Ainsi, les changements de plans moyens de deux parties d'un même film partagent des

similitudes. Ils participent au style du réalisateur et ils dépendent du programme. Les changements de plans d'un film d'action sont plus rapides que ceux d'une interview. Les changements de plans d'un jeu télévisé sont réguliers et suivent les dialogues entre le présentateur et les participants. Ces changements de plans caractérisent la structure des programmes TV.

Pour caractériser ces changements de plans, nous utilisons la même segmentation en plans que celle utilisée dans la section 2.3. Nous décrivons chaque segment à l'aide des 5 caractéristiques suivantes :

- la fréquence des changements de plans durs ;
- la fréquence des transitions progressives ;
- la durée du plus long plan ;
- la durée moyenne des plans ;
- l'écart-type de la durée des plans.

Pour un couple de segments de programmes longs, nous utilisons alors les distances euclidiennes entre chacune de ces 5 caractéristiques de plans. Cela nous permet de générer un premier vecteur de caractéristiques à 5 dimensions à donner en entrée du classifieur SVM.

#### 4.2.2.2 Caractéristiques de couleurs

Les distributions moyennes des couleurs particularisent les programmes TV. Elles sont fondamentales à notre perception des programmes. Les couleurs d'un match de tennis ne seront en effet pas les mêmes que celles d'un match de football. Les couleurs d'un dessin animé se distingueront aussi de celles d'une série.

Nous décrivons les distributions moyennes des couleurs à l'aide d'histogrammes de couleurs dans le format RVB (Rouge Vert Bleu). Nous quantifions chaque couleur RVB sur 5 niveaux pour obtenir un histogramme final à $125 = 5 \times 5 \times 5$ dimensions. Nous utilisons plus précisément 2 histogrammes de couleurs pour chaque segment :

- l'histogramme des couleurs moyennes : cet histogramme représente les histogrammes cumulés des couleurs de toutes les images clés du segment. L'histogramme final est normé par la durée du segment ;
- l'histogramme des couleurs communes : cet histogramme représente les couleurs communes à toutes les images clés du segment. C'est l'histogramme d'intersection des histogrammes des images clés du segment.

Les images clés sont calculées selon la méthode décrite dans la section 2.3.

Pour un couple de segments de programmes longs, nous utilisons un panel de plusieurs distances entre les histogrammes des couleurs moyennes et communes des deux segments pour couvrir de nombreuses différences et ou similitudes. En particulier, nous utilisons la distance de corrélation de Pearson entre les distributions représentées par les histogrammes. Nous utilisons aussi la distance de Bhattacharyya. Cela conduit à ajouter 4 dimensions au vecteur d'entrée du classifieur SVM pour chaque couple de segments.

#### 4.2.2.3 Caractéristiques de visages

Les visages des personnages ou des personnes filmées forment notre troisième groupe de caractéristiques. Nous n'effectuons pas de reconnaissance des visages. Nous utilisons seulement la détection des visages. La forme et la position générale des visages révèlent les gros plans, les dialogues et les regroupements d'individu.

Nous avons employé l'algorithme de détection des visages de Garcia *et al.* [GD04]. Cet algorithme repose sur un réseau de neurones convolutionnel. De la même manière que pour les histogrammes de couleurs, nous appliquons la détection des visages sur toutes les images clés des segments de programme. Nous décrivons un segment grâce aux 5 caractéristiques suivantes :

- le nombre total de visages détectés divisé par la durée du segment ;
- la moyenne du nombre de visages détectés par image clé ;
- l'écart type du nombre de visages détectés par image clé ;
- la taille du plus gros visage détecté ;
- les coordonnées de la position moyenne du plus gros visage détecté sur chaque image clé.

Pour chaque couple de segments de programmes longs, nous utilisons alors les distances euclidiennes entre chacune de ces 5 caractéristiques de visages. Cela ajoute encore 5 dimensions au vecteur de caractéristiques donné en entrée du classifieur SVM.

#### 4.2.2.4 Caractéristiques de répétitions

Les dernières caractéristiques que nous définissons utilisent les répétitions communes entre les segments consécutifs de programmes longs. L'idée principale est de détecter des images clés similaires entre les deux segments de chaque couple de segments de programmes longs. Par exemple, le plan présentateur d'un magazine d'information se répète dans les différentes parties de ce magazine. De la même manière, une série ou un dessin animé peut contenir quelques plans répétés quasiment à l'identique. Par contre, deux segments de programmes différents ne partagent généralement pas de parties communes.

Afin de détecter les images clés similaires entre les deux segments, nous calculons des signatures binaires pour chaque image clé. Ces signatures binaires sont calculées suivant le même processus que celui utilisé pour les descripteurs simples de la section 2.3. Les signatures des images clés sont donc des signatures sur 64 bits qui peuvent être efficacement comparées en utilisant une distance de *Hamming*. Deux images clés sont alors dites similaires si leur distance de *Hamming* est inférieure à un seuil fixé empiriquement.

Nous calculons ensuite de manière exhaustive des groupes d'images clés similaires à partir des images clés des deux segments. Un groupe d'images clés similaires est dit « commun aux deux segments » s'il contient au moins une image de chacun des segments. Un couple de segments peut enfin se décrire par une caractéristique de répétition qui est le nombre de groupes d'images clés similaires « communs aux deux segments » sur le nombre de groupes d'images clés similaires total construits avec les deux segments. Cela ajoute une dimension au vecteur de caractéristiques donné en entrée du classifieur SVM.

Basé sur le même principe, nous utilisons aussi deux autres caractéristiques de répétition. La première utilise la couleur dominante pour déterminer la similarité entre deux

images clés. La deuxième emploie le nombre de visages détectés. Ces deux dernières caractéristiques de répétitions grossières s'avèrent bien représenter les points communs entre deux segments de programmes longs.

## 4.3 Résultats

Les méthodes d'extraction des programmes fournissent notre dernière étape de la délinéarisation des flux de télévision. Nous évaluons dans cette section nos deux méthodes d'extraction. La première méthode se base sur les métadonnées. Elle conduit directement à une délinéarisation complètement automatique des programmes longs. Son expérimentation évalue alors les performances globales de notre système entier présenté dans cette thèse. La deuxième méthode n'utilise que le contenu audiovisuel et s'arrête à la réunification des programmes. Elle produit simplement une aide à un étiquetage manuel nécessaire.

Nous rappelons que notre système est soumis aux quatre contraintes clés de généricité, d'efficacité, d'automaticité et de continuité. Nous avons montré comment ces contraintes sont respectées par les niveaux de délinéarisation de découpage et de classification. Nous étudions donc ces contraintes pour ce dernier niveau de délinéarisation. Selon la présentation de notre méthode, l'extraction des programmes à partir des métadonnées est un processus simple qui se réalise très rapidement. Celui-ci n'impacte pas l'automaticité de notre système. De la même manière, la capacité à analyser périodiquement des portions de flux n'est pas remise en cause. Nous n'évaluons donc pas non plus la continuité. Nous évaluons seulement la généricité et l'efficacité. Pour évaluer la généricité, nous appliquons notre méthode de nouveau aux deux chaînes TF1 et France 2. Pour évaluer l'efficacité, nous définissons un protocole d'évaluation que nous appliquons sur notre corpus présenté en annexe A.

Nous conservons la même organisation pour nos évaluations que dans les chapitres précédents. Nous présentons en premier le contexte expérimental avec le corpus utilisé et le protocole d'évaluation puis nous expliquons les expériences que nous avons effectuées.

### 4.3.1 Contexte expérimental

Les expériences réalisées pour l'extraction des programmes TV disposent exactement du même contexte expérimental que celui utilisé dans les expérimentations des chapitres précédents. Nous utilisons le même corpus et le même environnement d'expérimentation.

Nos méthodes d'extraction reposent sur le découpage du flux en segments et sur la classification préalable des segments en segments de programme longs et en segments qui ne sont pas des programmes longs. La première semaine du corpus est donc utilisée pour réaliser l'apprentissage nécessaire à la classification des segments. Nous nous basons plus précisément sur les résultats obtenus dans la section 3.3.4.

Pour évaluer notre solution nous utilisons un ensemble de test composé des semaines n° 2, n° 3 et n° 4 de notre corpus à la fois sur TF1 et sur France 2. La deuxième semaine est une semaine de transition. Nous rappelons, en effet, que notre système emploie un historique d'une semaine pour le découpage. La classification des segments de la deuxième semaine utilise ainsi des informations qui ont également servi lors de l'apprentissage de la classification.

Nous avons montré dans le chapitre 1 que les métadonnées de type EIT étaient plus complètes et plus précises que celles de type EPG. Malheureusement, pour des raisons techniques indépendantes de notre volonté, nous n'avons pas pu obtenir les métadonnées EIT sur tous les jours de notre corpus. Les principales raisons sont dues à des pannes, à des mises à jour et à la maintenance de la plateforme d'acquisition. Nous avons cependant réussi à récupérer la majorité des EPG. Par conséquent, nous avons mis en place une procédure de fusion des métadonnées EIT et EPG. Cette procédure utilise les métadonnées EIT lorsqu'elles sont disponibles. Puis, elle remplace les métadonnées EIT manquantes par les métadonnées EPG équivalentes. Malgré cela, nous n'avons pas pu récupérer de métadonnées pour un jour parmi les 21 jours qui constituent nos 3 semaines de test. Nous évaluons donc nos résultats sur 20 jours seulement.

### 4.3.2 Protocole d'évaluation

Nous cherchons à évaluer l'extraction des programmes TV. Ces programmes TV se définissent par une borne inférieure et une borne supérieure. Les programmes TV sont extraits correctement si leurs bornes inférieures et leurs bornes supérieures sont bien délimitées et si ces programmes sont étiquetés avec un titre correct. Nous utilisons donc deux types d'information pour l'évaluation.

La première information est le nombre de programmes extraits correctement. Un programme extrait correctement est un segment qui obtient un titre qui existe dans la vérité terrain. Ce nombre de programmes extraits correctement est comparé au nombre de programmes à extraire dans la vérité terrain et au nombre total de programmes extraits par notre système. Cela nous permet de calculer obtenons des mesures de précision et de rappel.

La deuxième information utile est l'imprécision temporelle moyenne des programmes extraits correctement. L'imprécision temporelle est représentée dans la figure 4.8. L'imprécision temporelle de la borne inférieure est la différence entre la borne inférieure du programme automatiquement extrait et la borne inférieure du programme correspondant dans la vérité terrain. L'imprécision temporelle de la borne supérieure se définit de la même manière. Pour exprimer l'imprécision temporelle moyenne des programmes extraits, nous calculons simplement la moyenne des imprécisions temporelles des bornes inférieures et supérieures pour chaque programme.

Nous n'évaluons pas la précision et le rappel en terme de plans correctement étiquetés. Ces évaluations sont en effet difficile à interpréter car elles apparaissent relativement élevées. Ces valeurs sont élevées car le pourcentage de plans mal étiquetés est faible par rapport au nombre de plans des programmes longs. Par conséquent, les résultats élevés de précision et de rappel ne reflètent pas bien l'imprécision temporelle des étiquetages.

### 4.3.3 Expériences 1 : extraction des programmes longs basée sur les métadonnées

Nous présentons en parallèle les résultats de notre évaluation sur TF1 et sur France 2. Selon notre protocole d'évaluation, nous mesurons en premier le nombre de programmes extraits correctement puis, l'imprécision temporelle moyenne des programmes extraits. Enfin, nous analysons les résultats de notre méthode de délinéarisation automatique. Les résultats du chapitre 2 montrent que nous obtenons un découpage beaucoup moins précis la

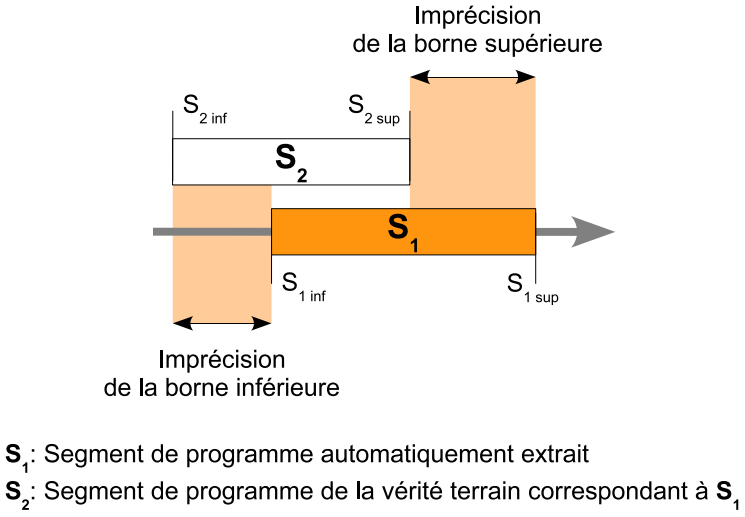



FIGURE 4.8 – Imprécision temporelle de la borne inférieure et de la borne supérieure.

nuit en raison de la diminution du nombre d'inter-programmes. De plus, les métadonnées sont moins complètes la nuit suivant notre étude du chapitre 1. Par conséquent, nous n'utilisons pas la nuit pour l'évaluation de l'extraction des programmes. Les résultats suivants sont donc calculés sur des journées débutant à 8h du matin et se terminant à 1h le lendemain matin. Les programmes de la nuit sont généralement des programmes à très faible audience ou des rediffusions. Il n'est donc pas grave de ne pas extraire ces programmes. Nous préférons ainsi nous concentrer sur les programmes de la journée qui représentent les programmes les plus pertinents pour un service de télévision à la demande.

#### 4.3.3.1 Nombre de programmes longs extraits correctement

Le tableau 4.1 présente le nombre de programmes longs extraits correctement pour chaque chaîne (France 2 et TF1). Ce tableau donne aussi les nombres de programmes longs délimités dans la vérité terrain et le nombre de programmes longs annoncés dans les métadonnées.

| | VT | Métadonnées | | Notre solution | |
|---|---|---|---|---|---|
| | # | # | # ∈ VT | # | # ∈ VT |
| France 2 | 272 | 284 | 269 | 276 | 268 |
| TF1 | 297 | 326 | 295 | 309 | 289 |

TABLEAU 4.1 – Nombre (#) de programmes longs dans la vérité terrain (VT), annoncés dans les métadonnées et extraits automatiquement.

Nous remarquons en premier que le nombre de programmes longs annoncés dans les mé-

tadonnées ne correspond pas au nombre de programmes longs réellement diffusés. Nous notons un nombre supérieur de programmes annoncés. Le tableau 4.1 donne aussi le nombre de programmes longs annoncés dans les métadonnées effectivement diffusés. Il s'avère donc que, pour les métadonnées que nous avons récupérées et fusionnées, il y a plus de programmes longs annoncés que de diffusés et il y a en plus des programmes longs diffusés non annoncés. Ces données complètent l'évaluation des métadonnées que nous avons réalisée dans le chapitre 1.

Le tableau 4.1 montre le nombre de programmes longs extraits automatiquement. Ce nombre est toujours inférieur au nombre de programmes longs annoncés dans les métadonnées. Nous extrayons en effet les programmes longs à partir des métadonnées. Nous ne pouvons donc pas extraire plus de programmes longs qu'il y en a d'annoncés.

Au final, notre solution permet d'extraire, sur France 2 dans la journée, des programmes longs de la vérité terrain avec un rappel de 98,5 % et une précision de 97,1 %. Notre solution extrait, sur TF1 dans la journée, des programmes longs de la vérité terrain avec un rappel de 97,3 % et une précision de 93,5 %. Ces résultats montrent que notre solution est très performante pour l'étiquetage automatique des segments. Cependant, il reste difficile d'obtenir une extraction exhaustive de tous les programmes longs à cause de la qualité des métadonnées.

#### 4.3.3.2 Imprécision temporelle moyenne des programmes extraits

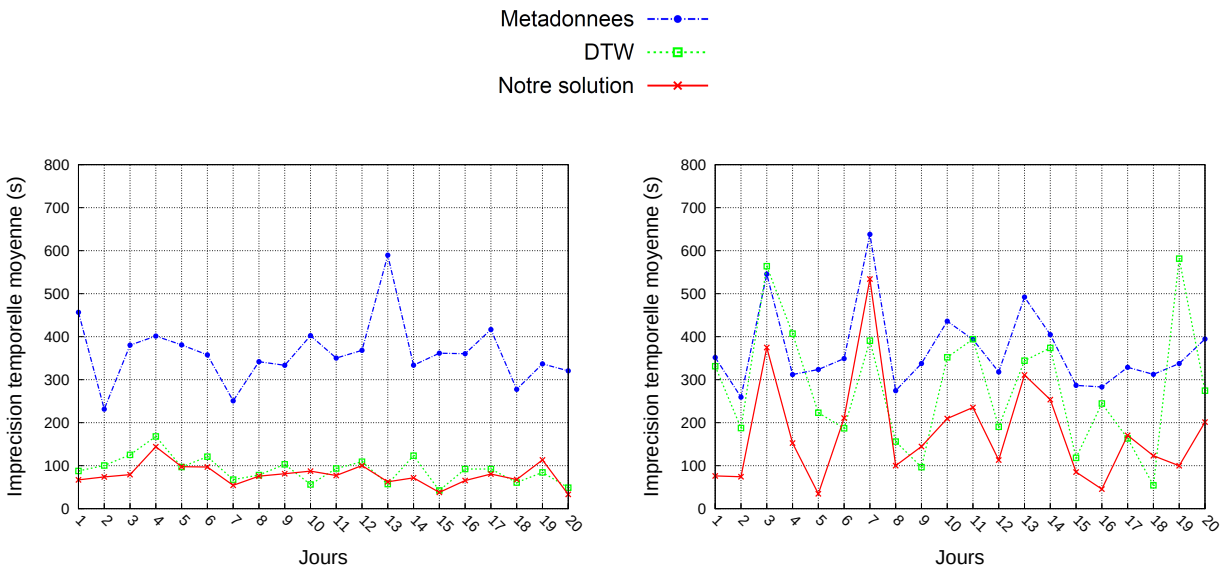
Nous avons évalué l'imprécision temporelle moyenne des programmes longs extraits automatiquement selon deux points de vue à la fois sur TF1 et sur France 2. Nous avons d'abord évalué l'imprécision temporelle moyenne pour chaque journée de nos semaines de test. Nous avons, ensuite, réparti manuellement les programmes en 6 catégories pour lesquelles nous avons calculé individuellement l'imprécision temporelle moyenne.

##### *Évaluation par journée*

L'intérêt de l'évaluation de l'imprécision temporelle moyenne pour chaque journée de notre ensemble de test est de vérifier notre contrainte de continuité. Nous observons alors si notre système se comporte de manière relativement stable lors de la délinéarisation en continu du flux. Les résultats sont montrés dans la figure 4.9. Les résultats de notre solution sont donnés par les courbes basses continues. Ces courbes montrent que l'imprécision temporelle moyenne n'augmente pas continuellement avec le temps. L'imprécision temporelle moyenne reste donc relativement stable.

Afin de mettre en perspective l'imprécision temporelle moyenne calculée, nous avons calculé deux autres imprécisions temporelles moyennes de référence.

La première imprécision temporelle moyenne de référence est l'imprécision temporelle moyenne des segments de métadonnées. Elle est calculée pour chaque segment de métadonnées qui a été utilisé pour extraire correctement un programme long. Cette imprécision temporelle moyenne est représentée par des courbes hautes en gros pointillés intitulées « Métadonnées ». Elle représente le maximum à ne pas dépasser. En effet, si notre imprécision temporelle moyenne est au dessus, cela signifie que les résultats de notre système sont moins performants que des résultats obtenus en utilisant uniquement les segments de métadonnées. Sur France 2, l'imprécision temporelle moyenne de notre solution est large-




(a) France 2 (b) TF1

FIGURE 4.9 – Imprécision temporelle des bornes supérieures et inférieures des programmes longs extraits automatiquement et annoncés dans les métadonnées.

ment inférieure à celle des métadonnées. Sur TF1, l'imprécision temporelle moyenne de notre solution est très proche de celle des métadonnées. Nos résultats sont meilleurs sur France 2 que sur TF1. La cause de la différence est la qualité inférieure des métadonnées collectées pour TF1.

La deuxième imprécision temporelle moyenne de référence est l'imprécision temporelle moyenne des programmes extraits par une mise en correspondance basée sur la DTW. Cette méthode est expliquée dans la section 4.1.2. Cette imprécision temporelle moyenne est représentée par des courbes en petits pointillés intitulées « DTW ». Elle permet de comparer la mise en correspondance locale avec la mise en correspondance globale. Dans la figure 4.9, l'imprécision temporelle moyenne de notre solution est bien inférieure à celle de la DTW. Cela confirme notre choix. La mise en correspondance locale est bien moins influencée par les incomplétudes des métadonnées. Nous pouvons noter toutefois, que l'imprécision temporelle moyenne de la DTW est parfois inférieure à celle de notre solution. Dans ces cas, les métadonnées étaient en réalité décalées sur une partie de la journée. La mise en correspondance globale a permis alors de recaler les segments alors que la mise en correspondance locale n'a pas détecté et corrigé ce décalage. Cela montre qu'une approche mixte pourrait être pertinente.

### *Évaluation par catégorie*

Nous évaluons à présent l'imprécision temporelle moyenne des programmes correctement extraits selon les 6 catégories considérées, à savoir : les journaux, les jeux, les

émissions de TV réalité, les films, les séries et les émissions sportives. Les résultats sont donnés dans le tableau 4.2 pour France 2 et dans le tableau 4.3 pour TF1.

Comme dans l'évaluation par journée, les résultats sont moins imprécis sur France 2 que sur TF1. Cela est dû en grande partie à la qualité supérieure des métadonnées sur France 2.

L'imprécision temporelle des journaux provient majoritairement de la mauvaise classification des nombreux programmes courts qui entourent les journaux. Dès qu'un programme court est classé en programme long, il se transforme en partie potentielle de programme long. Comme les métadonnées ne sont pas très précises, ces programmes courts deviennent des parties des journaux.

Les résultats sur France 2 montrent de très bon résultats pour les films (et les programmes de soirée). Cela nous autorise à envisager un système qui récupère uniquement les programmes longs extraits automatiquement en soirée pour les ajouter à un catalogue de programmes disponibles automatiquement pour la télévision à la demande.

Sur France 2 et sur TF1, les programmes sportifs sont les plus mal annoncés. Les programmes sportifs sont des enregistrements de courses ou de match souvent en direct qui sont annoncés dans les métadonnées avant d'avoir eu lieu. Leurs durées et leurs horaires sont donc très difficiles à prévoir.

| | Imprécision temporelle moyenne | |
|---|---|---|
| | Borne inférieure | Borne supérieure |
| Journaux | 49 s | 2 m 53 s |
| Jeux | 7 s | 3 m 26 s |
| TV réalité | – | – |
| Films | 13 s | 9 s |
| Séries | 20 s | 57 s |
| Sport | 1 m 46 s | 1 m 47 s |

Tableau 4.2 – Évaluation des imprécisions temporelles moyennes par catégorie sur France 2.

| | Imprécision temporelle moyenne | |
|---|---|---|
| | Borne inférieure | Borne supérieure |
| Journaux | 1 m 1 s | 7 m 25 s |
| Jeux | 23 s | 2 m 15 s |
| TV réalité | 1 m 48 s | 1 m 12 s |
| Films | 7 m 15 s | 10 m 35 s |
| Séries | 1 m 41 s | 1 m 12 s |
| Sport | 19 m 23 s | 29 m 20 s |

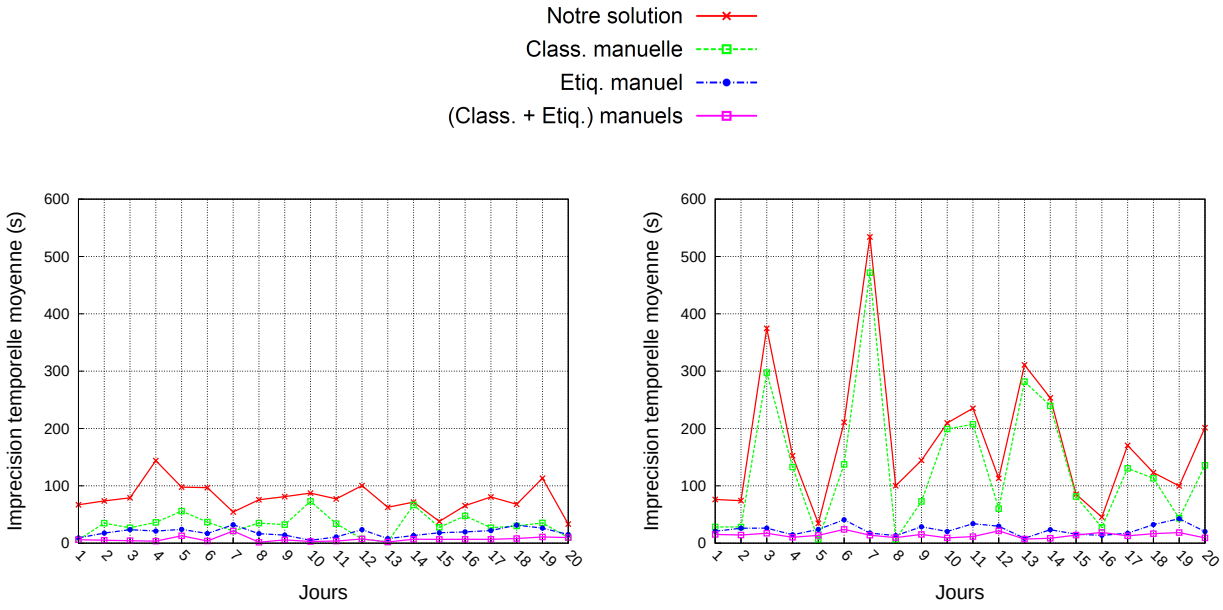
Tableau 4.3 – Évaluation des imprécisions temporelles moyennes par catégorie sur TF1.

#### 4.3.3.3 Analyse des résultats de notre solution de délinéarisation automatique

L'extraction des programmes à partir des métadonnées constitue la dernière étape de notre solution de délinéarisation automatique des flux. Les résultats présentés dans la

section 4.3.3.2 précédente montrent une imprécision temporelle moyenne finale de l'ordre de plusieurs minutes. Cette imprécision temporelle moyenne est meilleure que celle fournie par les métadonnées. Cependant, elle ne permet pas d'offrir un service de télévision à la demande complètement automatique.

Afin de comprendre l'imprécision temporelle moyenne, nous avons étudié la part de chaque niveau de délinéarisation. Les résultats sont donnés dans les courbes de la figure 4.10.




(a) France 2 (b) TF1

FIGURE 4.10 – Imprecision temporelle des bornes supérieures et inférieures des programmes longs extraits automatiquement. Comparaison des résultats avec une classification manuelle et/ou un étiquetage manuel.

Le premier niveau de délinéarisation est le niveau de découpage en segments. Pour mesurer la part de ce niveau sur l'imprécision temporelle moyenne finale des programmes extraits, nous avons classé et étiqueté chaque segment découpé grâce à la vérité terrain. Cela remplace une classification et un étiquetage manuels. Les résultats sont représentés par les courbes basses continues intitulées « (Class. + Etiq.) manuels ». Ces courbes montrent des imprécisions temporelles moyennes de l'ordre de quelques secondes. Cette imprécision temporelle provient du découpage précis à seulement un inter-programme près expliqué dans le chapitre 2.

Le second niveau de délinéarisation est le niveau de classification des segments. Nous avons réalisé un étiquetage « manuel » grâce à la vérité terrain des segments classés par notre méthode de classification automatique. Les résultats obtenus sont donnés par les courbes en pointillés avec des ronds intitulées « Etiq. manuel ». Les différences entre les

courbes « Etiq. manuel » et « (Class. + Etiq.) manuels » reflètent la part de nos erreurs de classification de ce niveau sur l'imprécision temporelle moyenne finale. Les courbes étant très proches, elles traduisent les bonnes performances de notre classification automatique. L'écart mesuré entre les courbes correspond aux segments de séparation de la section 3.3 qui sont mal classés.

Le troisième et dernier niveau de délinéarisation est le niveau d'extraction des programmes. La part de ce niveau sur l'imprécision temporelle moyenne finale est donnée par l'évaluation de l'étiquetage automatique de segments de programme issus d'une classification « manuelle » grâce à la vérité terrain. Cette évaluation est représentée par les courbes en pointillés avec des carrés intitulées « Class. manuel ». La comparaison entre les courbes « Class. manuel » et « (Class. + Etiq.) manuels » montre les erreurs simplement dues à l'utilisation des métadonnées pour réaliser l'étiquetage. Ces erreurs sont clairement visibles sur TF1. Cela peut révéler une mauvaise utilisation des métadonnées. Dans notre cas, cela confirme la mauvaise qualité des métadonnées de TF1 que nous avons récupérées.

La source principale des imprécisions temporelles moyennes importantes est l'utilisation des métadonnées pour l'étiquetage. Une solution est de ne plus utiliser les métadonnées pour effectuer une réunification et un étiquetage. C'est une piste que nous avons commencé à explorer à travers notre méthode d'extraction des programmes basée sur le contenu audiovisuel.

### 4.3.4 Expériences 2 : extraction des programmes basée sur le contenu audiovisuel

L'extraction des programmes basée sur le contenu audiovisuel consiste en une réunification automatique basée contenu des segments de programme long. Nous évaluons donc seulement l'extraction des programmes longs selon cette technique.

Pour les expériences, nous avons seulement utilisé la chaîne TF1 de notre corpus. TF1 est la chaîne pour laquelle nous avons récolté des métadonnées de moins bonne qualité. Nous montrons ainsi dans quelle mesure notre technique permet de pallier les défauts des métadonnées. De plus, la particularité de cette chaîne par rapport à la chaîne France 2 est de comporter de nombreuses coupures publicitaires à l'intérieur de ses programmes longs diffusés. Ces coupures publicitaires divisent les programmes de la chaîne. Elles rendent pertinente notre solution de réunification. En ce qui concerne la chaîne France 2, nous avons noté trop peu de segments à réunifier.

Avant d'évaluer l'extraction des programmes longs en utilisant la réunification basée contenu, nous avons évalué la technique de réunification basée contenu.

#### 4.3.4.1 Évaluation de la réunification basée contenu

Nous rappelons que la technique de réunification basée contenu analyse les segments par couple de segments consécutifs. Pour chaque couple, elle détermine si les segments appartiennent ou non à un même programme TV. La réunification repose sur l'apprentissage supervisé d'un classifieur SVM. Nous utilisons une semaine d'apprentissage et une semaine de test. À titre indicatif, nous avons mesuré le nombre moyen de segments découpés par programme TV. Ce nombre est de $1,85$ pour la première semaine d'apprentissage et de $1,87$ pour la semaine de test. Au total, la semaine d'apprentissage comporte 192 couples

de segments de programmes consécutifs dont 89 couples de segments sont à réunifier. La semaine de test contient 181 couples de segments dont 85 à réunifier.

L'évaluation de la réunification s'exprime par une matrice de confusion donnée dans le tableau 4.4. Cette matrice de confusion donne les nombres de couples de segments réunifiés correctement par rapport aux couples de segments à ou à ne pas réunifier.

| | Couples de segments réunifiés | Couples de segments non réunifiés |
|---|---|---|
| Couples de segments à réunifier | 82 | 3 |
| Couples de segments à ne pas réunifier | 5 | 91 |

Tableau 4.4 – Matrice de confusion de la réunification basée contenu des couples de segments de programme.

Ainsi, selon le tableau 4.4, seulement 5 couples de segments ont été incorrectement réunifiés et 3 couples n'ont pas été réunifiés sur notre semaine de test. Afin de mieux comprendre ces résultats, nous évaluons les programmes longs ainsi extraits.

#### 4.3.4.2 Évaluation de l'extraction des programmes longs réunifiés

Nous reprenons le protocole d'évaluation initial afin de mesurer les performances de notre système pour l'extraction des programmes longs lorsque les métadonnées sont indisponibles.

Les résultats de l'extraction sont présentés dans le tableau 4.5. Tous les programmes ont été répartis dans les 6 catégories suivantes : les journaux, les jeux, les émissions de TV réalité, les films, les séries et le sport. La catégorie « Films » contient plus précisément tous les émissions de soirée. Sur toute la semaine nous avons compté 77 programmes longs à extraire. Ces programmes sont diffusés dans la période entre 12h00 et 24h00.

| | Nombre de programmes extraits correctement | Nombre de programmes dans la vérité terrain | Borne inférieure | Borne supérieure |
|---|---|---|---|---|
| Journaux | 13 | 14 | 15 s | 27 s |
| Jeux | 13 | 15 | 19 s | 10 s |
| TV réalité | 8 | 10 | 58 s | 3 s |
| Films | 14 | 18 | 8 s | 23 s |
| Séries | 13 | 20 | 30 s | 23 s |
| Sport | 0 | 0 | – | – |
| Tous | 61 | 77 | 23,6 s | 18,6 s |

Tableau 4.5 – Évaluation des programmes longs réunifiés.

Les 8 erreurs réalisées sur la réunification basée contenu de couples de segments ont engendré des erreurs d'extraction sur 16 programmes. Sur la semaine de test, notre méthode de réunification basée contenu a permis d'extraire correctement 79 % des programmes longs pour lesquels la moyenne de l'imprécision temporelle est inférieure à 23,6 secondes pour la borne inférieure et supérieure.

Les programmes les mieux extraits sont les journaux. Les journaux ne sont en fait jamais séparés par de la publicité. La difficulté consiste à laisser les journaux seuls, c'est à

dire à ne pas les fusionner avec la météo qui les entoure. Les seconds types de programmes bien extraits sont les jeux télévisés et les émissions de TV réalité. Ces émissions possèdent en effet des caractéristiques très spécifiques qui sont les marques des émissions. Ensuite, nous extrayons les films et les séries. Ces deux types se sont avérés plus difficiles à extraire. Nos caractéristiques ne permettent pas, en effet, de caractériser la différence entre deux séries américaines consécutives qui souvent partagent de nombreuses similarités.

Les imprécisions temporelles moyennes calculées pour les 6 catégories et présentées dans le tableau 4.5 sont de l'ordre de quelques secondes. Ils rejoignent les résultats de la courbe « Etiq. manuel » présentée dans la figure 4.10. Ils sont encourageants. Ils montrent qu'il est possible d'obtenir une délimitation complètement automatique des programmes à défaut de pouvoir les étiqueter proprement. Notre méthode de réunification basée contenu permet ainsi de ne plus dépendre des métadonnées pour la délimitation des bornes des programmes. Les métadonnées restent cependant une source précieuse d'informations en particulier pour l'étiquetage des noms des programmes. Une solution hybride est à envisager pour combiner les métadonnées avec notre méthode de réunification basée sur le contenu audiovisuel. Dans cette solution, la phase d'utilisation des métadonnées est à retarder au maximum afin de ne pas détériorer prématurément les résultats.

### 4.3.5 Comparaison avec les résultats des méthodes existantes

Les systèmes existants de délinéarisation automatique des flux TV sont, comme notre solution, des systèmes complexes. Ils sont difficilement reproductibles. Par conséquent, nous ne proposons pas une comparaison expérimentale. De plus, ces systèmes utilisent des ensembles de test différents. Ces ensembles sont établis à partir de flux télévisuels variés et provenant de divers pays. Chaque flux TV possède des caractéristiques spécifiques. Il est donc non concluant de comparer directement les résultats divulgués sur ces systèmes.

Malgré ces difficultés, nous avons identifié deux systèmes analogues au notre parmi les méthodes existantes présentées dans le chapitre 1. Ces deux systèmes sont issus des travaux de Naturel [Nat07] et de Poli [Pol07]. Ils délinéarisent un flux de France 2. Bien que nous n'ayons pas pu reproduire ces systèmes, nous proposons une comparaison théorique. Cette comparaison analyse dans quelle mesure chaque système vérifie nos quatre contraintes clés de généricité, d'efficacité, d'automaticité, et de continuité.

Les travaux de Poli [Pol07] et de Naturel [Nat07] sont antérieurs aux débuts de nos travaux. Nous nous sommes ainsi inspirés des idées validées et des pistes proposées dans ces travaux. En particulier, notre système repose sur la détection d'inter-programmes par leur propriété de répétition à l'instar de Naturel [Nat07] et nous apprenons une structure suivant l'exemple de Poli [Pol07]. Au final, notre solution s'insère comme une suite à ces travaux avec lesquels nous comparons nos contributions.

#### *La généricité*

Mis à part le problème de la qualité de nos métadonnées sur TF1, notre système se comporte de manière similaire à la fois sur TF1 et sur France 2. Nous avons montré dans l'annexe A les différences significatives entre ces deux chaînes. La généricité est donc vérifiée pour au moins ces deux chaînes.

Ce résultat est intéressant. Il permet d'appuyer l'utilisation des répétitions pour la

détection des inter-programmes et pour la délinéarisation. En effet, nous validons ainsi la méthode de détection des inter-programmes par les répétitions pour une chaîne différente de celle utilisée par Naturel [Nat07]. Cependant, pour la délinéarisation, nous ne vérifions au final notre approche que sur deux chaînes. En comparaison, Poli [Pol07] a évalué sa méthode sur les 6 grandes chaînes de la télévision française. Toutefois, Poli [Pol07] a pu bénéficier des outils et services de l'Institut national de l'audiovisuel chargé notamment de l'archivage des flux TV français. Dans notre situation, nous avons dû gérer l'acquisition en continu de notre corpus constitué de plusieurs semaines de flux TV.

***L'efficacité***

Notre solution permet de délimiter à la fois les programmes et les inter-programmes. Les inter-programmes sont aussi détectés par Naturel [Nat07] mais pas par Poli [Pol07]. Pour les inter-programmes, nous avons mesuré, sur France 2 et sur 21 jours, une F-mesure par rapport aux images de $F_{img} = 93,04$ % dans la section 3.3. Naturel [Nat07] obtient sur 20 jours une F-mesure calculée aux niveaux des images de $F_{img} = 91,2$ % de classification des inter-programmes. Bien que les mêmes périodes de flux sur France 2 n'ont pas été utilisées, nos résultats apparaissent très positifs. De plus, nous rappelons que notre méthode permet de détecter tous les inter-programmes répétés dans un flux. La méthode de Naturel [Nat07] reconnaît seulement les inter-programmes référencés au préalable.

Par rapport aux programmes, l'imprécision temporelle moyenne sur France 2 que nous avons évaluée est du même ordre de grandeur que la précision temporelle mesurée par Poli [Pol07] sur la même chaîne. Néanmoins nous ne mesurons pas la nuit alors que celle-ci est mesurée dans les travaux de Poli [Pol07]. Ces derniers travaux apparaissent donc plus précis. Nous n'avons pas pu estimer l'imprécision temporelle obtenue par Naturel [Nat07]. Les résultats fournis sont seulement donnés en terme de précision et de rappel des images ou émissions correctement étiquetées.

La méthode de Poli [Pol07] détecte les bornes des programmes suivant deux étapes. Il prédit d'abord des horaires approximatifs, puis il aligne ces horaires sur le flux grâce à la détection de transitions à partir des plans, des silences, des images monochromes et des logos. Nous n'avons pas utilisé ces dernières caractéristiques que nous considérons trop spécifiques aux chaînes à analyser. Dans le cas de la délinéarisation d'une chaîne spécifique, nous pouvons envisager d'ajouter ces traitements afin d'améliorer notre imprécision temporelle.

***L'automaticité***

Notre système dépend comme les systèmes de Poli [Pol07] et de Naturel [Nat07] d'un apprentissage supervisé. Un apprentissage est réalisé pour la configuration de chaque méthode qui reste automatique lors de leur fonctionnement. L'automaticité des méthodes provient de l'apprentissage utilisé. Elle est liée à la quantité de données nécessaires pour l'apprentissage et la durée de validité de l'apprentissage acquis.

À propos de la quantité de données, notre apprentissage demande la construction d'une vérité terrain d'au moins une semaine de flux pour chaque chaîne à traiter. La taille de cet ensemble d'apprentissage est 7 fois plus importante que la taille de l'ensemble d'apprentissage utilisé par Naturel [Nat07] dans son ensemble de vidéos de référence (EVR).

Cependant, la taille de notre ensemble d'apprentissage est négligeable par rapport à celle utilisée par Poli [Pol07]. Poli emploie plusieurs années de vérité terrain. Dans chaque solution la même précision de la vérité terrain peut être employée.

En ce qui concerne la durée de validité, nous n'avons pas noté de dégradation dans les résultats sur nos 3 semaines de test. Ces résultats sont encourageants, mais des expériences complémentaires sont indispensables pour vérifier la durée de validité sur une plus longue durée. En comparaison, Naturel [Nat07] montre déjà une chute de 10 % dans ses résultats au bout d'une période de test équivalente. L'apprentissage utilisé par Naturel [Nat07] consiste à reconnaître des morceaux de programmes ou d'inter-programmes ayant été diffusé un jour passé. L'apprentissage utilise donc les données de manière brute. Il est normal que ces données perdent en valeur informative avec le temps. Pour remédier à cela, Naturel [Nat07] propose de mettre à jour son ensemble de référence. Cependant, la technique de mise à jour ne permet pas de conserver la même qualité que l'apprentissage d'origine.

Notre apprentissage possède une durée de validité plus importante que Naturel [Nat07] grâce à l'apprentissage d'une structure entre les programmes et les inter-programmes. Contrairement à Naturel, nous n'utilisons pas seulement le fait qu'un morceau précis du flux soit simplement un programme ou un inter-programme. Nous utilisons, en plus, le fait que généralement les morceaux d'inter-programmes sont reliés aux morceaux de programme d'une certaine manière. Cette idée de structure provient des travaux de Poli [Pol07] qui a montré la stabilité des flux TV sur plusieurs années. Poli [Pol07] apprend, en effet, la structure des guides de programmes diffusés chaque jour. Début janvier 2009, la chaîne France 2 a supprimé la publicité pendant les soirées. Cette décision a bouleversé l'organisation habituelle des programmes. Les programmes de soirée ont été diffusés plus tôt. Ce changement a théoriquement eu un impact important pour la méthode de Poli [Pol07] qui nécessite alors de reconstituer un ensemble d'apprentissage conséquent. Face à ce changement, notre solution apparait plus flexible. La suppression de la publicité n'empêche pas notre méthode de détecter les bornes des programmes à partir des répétitions. Nous avons, en effet, noté que les bandes annonces et les parrainages ont été conservés entre les programmes. Bien que les horaires de diffusion des programmes changent, l'agencement des inter-programmes et des programmes reste inchangé. Néanmoins, nous n'avons pas pu tester l'efficacité de nos règles de classification sur des données de 2009. Dans le cas où cet agencement change complètement, notre système ne requiert q'une semaine de données d'apprentissage. Il est donc rapidement opérationnel.

#### *La continuité*

Nous avons expliqué dans le chapitre 2 que notre solution utilisait un historique glissant périodiquement dans le flux TV. Nous avons étudié la taille nécessaire de l'historique et nous avons montré que notre système pouvait ainsi analyser des flux TV ininterrompus.

La gestion de la continuité que nous avons ajouté au système de *clustering* ne constitue pas une contribution par rapport aux autres méthodes de délinéarisation. Cette contrainte de continuité est, en effet, également vérifiée par les systèmes de Poli [Pol07] et de Naturel [Nat07].

### 4.3.6 Synthèse

Pour que la délinéarisation automatique des flux TV fournisse des résultats utilisables par les téléspectateurs à travers des services de télévision à la demande, une méthode performante d'extraction des programmes est requise.

Les métadonnées fournissent en apparence des données très précieuses pour l'extraction automatique des programmes TV. Nous avons essayé d'utiliser ces métadonnées. Lorsque ces métadonnées sont relativement précises, elles permettent grâce à notre méthode d'extraire des programmes avec une petite imprécision temporelle. En particulier, les films de soirée sur France 2 peuvent être ajoutés de manière complètement automatique à un catalogue. Les résultats obtenus ne sont cependant pas convaincants pour la chaîne TF1. Accusant la qualité des métadonnées, nous avons imaginé une solution de réunification des programmes sans l'aide des métadonnées. Cette solution a montré des résultats très encourageants. Cependant, les programmes réunifiés ne sont pas étiquetés.

# Conclusion générale

*Le monde n'a peut être pas de sens,*
*mais il a des structures, et tout est là.*

Jean-Claude Clari, romancier, 1973.

Nous avons présenté le produit de trois années de travaux de thèse concentrés sur la conception d'un système automatique d'analyse des flux télévisuels. Les flux télévisuels sont des sources d'éléments divers et variés d'actualités, d'informations culturelles et de divertissements. Tous ces éléments sont, cependant, réunis en un unique continuum qui ne possède *a priori* pas de bornes pour un système automatique. Pour pouvoir séparer ces éléments réunis, notre système automatique repose sur la détection des répétitions et l'apprentissage de règles logiques de classification. Cela nous permet de retrouver la structure inhérente à la télévision. Les différents éléments sont alors délimités par cette structure. Nous effectuons ainsi une délinéarisation des flux télévisuels.

Nous résumons, à présent, les contributions apportées par ces travaux de thèse puis nous proposons quelques perspectives.

## Synthèse et contributions

Notre système analyse le flux TV portion par portion. Chaque portion est analysée suivant trois niveaux de délinéarisation. Nous rappelons le fonctionnement de notre système de délinéarisation en trois niveaux de délinéarisation dans la figure 4.11. Le point de départ de notre système est la détection des rediffusions des inter-programmes. Ces rediffusions d'inter-programmes sont détectées à travers la détection de toutes les répétitions de la portion analysée placée dans un historique de flux TV. Ces répétitions forment des segments dans le flux qui sont ensuite classés par des règles logiques. Ces règles sont apprises par la programmation logique inductive qui modélise les contextes et les relations entre les segments du flux. Une fois classés, les segments détectés de programmes longs sont réunifiés puis étiquetés grâce aux métadonnées lorsque celles-ci sont disponibles. Les segments détectés de programmes longs sont seulement réunifiés grâce au contenu audiovisuel des segments lorsque les métadonnées ne sont pas disponibles.

Notre contribution principale repose sur trois contributions élémentaires apportées lors de la réalisation des trois niveaux de délinéarisation. La première de ces contributions est la

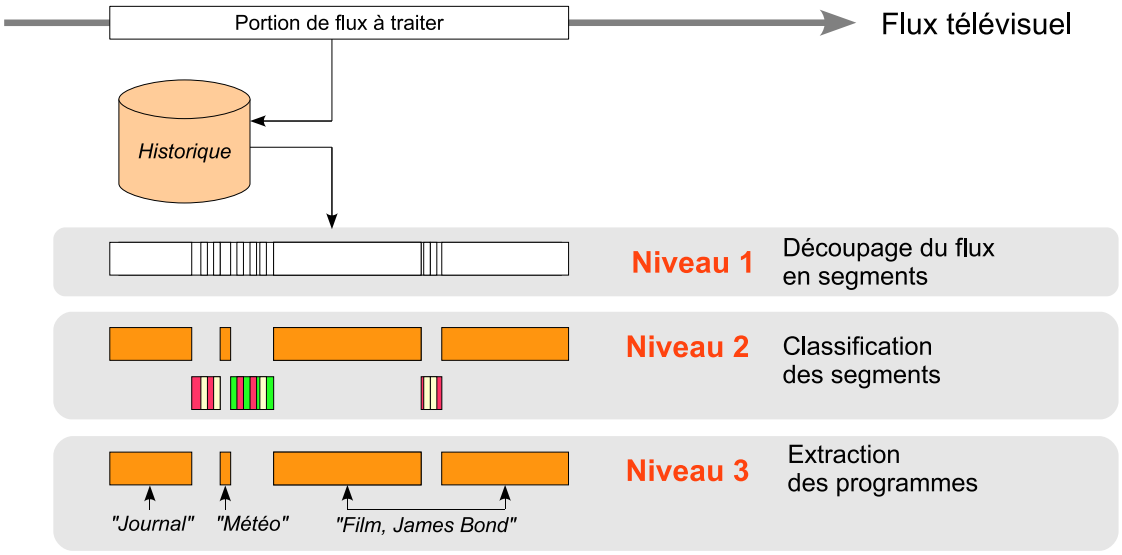



FIGURE 4.11 – Fonctionnement de notre système de délinéarisation en trois niveaux de délinéarisation.

conception d'un système de détection des répétitions qui utilise le micro-*clustering*. Cette technique permet de détecter, de façon complètement non supervisée, des répétitions possédant un nombre inconnu d'occurrences quasiment identiques. La seconde contribution élémentaire est la classification des segments audiovisuels par la programmation logique inductive. Cette méthode de classification diffère des méthodes traditionnelles. Elle utilise des caractéristiques de répétitions des segments. Elle exploite et modélise les relations entre les segments et elle apprend des règles de classification qui révèlent la structure contenue dans l'ensemble d'apprentissage. La dernière contribution élémentaire est la réunification basée contenu des segments de programme en programmes TV entiers. Cette solution permet notamment d'extraire les programmes TV à partir de leurs caractéristiques audiovisuelles lorsque les métadonnées ne sont pas disponibles.

Nous avons conçu notre système afin qu'il respecte quatre contraintes clés : la généricité, l'efficacité, l'automaticité et la continuité. Ces contraintes ont été vérifiées, dans une certaine mesure, pour chaque niveau de délinéarisation à travers des explications et des expériences. La généricité de notre système réside dans sa capacité à traiter tous les flux TV contenant des inter-programmes répétés. Elle a été vérifiée sur deux chaînes françaises de télévision. L'efficacité de notre système a été montrée à travers les résultats de nombreuses expériences. Elle a aussi été confrontée à certains résultats issus de méthodes existantes. Plus précisément, les résultats montrent que notre système est capable de délinéariser les flux avec de bonnes performances. L'automaticité du système a été contrôlée. Notre système est configuré une fois par chaîne à analyser puis il traite automatiquement cette chaîne. Enfin, notre système analyse de manière stable chaque journée de trois semaines de flux TV ininterrompus. Cela valide la dernière contrainte de continuité bien que nous ayons fait le choix de ne pas traiter les flux de la nuit.

L'ensemble des contributions apportées par cette thèse a été soumis à la communauté scientifique internationale à travers plusieurs publications et brevets. La liste de ceux-ci est donnée à la fin de ce document dans l'annexe E. Pour montrer les résultats de ces travaux, nous avons de plus développé deux applications logicielles. Ces applications sont détaillées dans l'annexe C de ce document.

## Perspectives

Les résultats des travaux effectués dans cette thèse ouvrent de nouvelles pistes de travail.

Les premières perspectives concernent des améliorations possibles des travaux de cette thèse. Par exemple, dans la section 2.5.5.1, nous avons choisi pour des raisons de performances globales de ne pas utiliser les répétitions basées sur l'extension des *clusters* solitaires. Mais nous avons limité ainsi notre capacité à détecter les parrainages et d'autres inter-programmes. Une perspective simple serait d'utiliser absolument toutes les répétitions détectées et de nous baser alors sur la classification des segments pour « nettoyer » le supplément de segments ainsi obtenus. Une autre amélioration de nos travaux, est la poursuite de nos recherches pour la classification des segments d'inter-programmes en différentes catégories. De la même manière, nous souhaitons étendre notre technique de détection de bandes-annonces à partir des répétitions.

Nous avons de plus identifié un certain nombre de difficultés auxquelles il est intéressant de s'attaquer. Une difficulté importante rencontrée par notre solution est l'utilisation des métadonnées. Ces métadonnées sont en effet imprécises et incomplètes. Les résultats présentés dans le chapitre 4 montrent au final que la technique de réunification basée sur le contenu audiovisuel obtient des résultats moins imprécis temporellement que la méthode basée sur les métadonnées. On peut donc imaginer un système d'extraction des programmes qui combine à la fois la réunification basée sur le contenu et la mise en correspondance avec les métadonnées pour obtenir une extraction avec étiquetage des programmes qui soit moins imprécise temporellement. Il est très peu probable que les métadonnées s'améliorent avec le temps. Pour les raisons économiques déjà évoquées, ce n'est pas dans l'intérêt des chaînes de télévision. À cause de leur incomplétude et de leur imprécision, les métadonnées ne peuvent réellement être utilisées que pour l'étiquetage. Un défi important pour la délinéarisation automatique est finalement l'obtention d'une délinéarisation exclusivement basée sur les signaux audio et vidéo. L'objectif est alors d'obtenir la structure sous-jacente des flux télévisuels sans étiquetage et de réaliser l'étiquetage grâce à des techniques de reconnaissance optique de caractères (OCR) et de reconnaissance de la parole.

Le problème des génériques et des segments incorrectement classés aux frontières des programmes est une autre difficulté importante. Il parait pertinent de classer proprement les génériques et les segments aux frontières des programmes afin de délimiter plus précisément les bornes des programmes. Dans le même genre de difficulté, nous notons aussi le problème des inter-programmes non répétés qui précèdent les programmes. Ces inter-programmes spécifiques empêchent de trouver quelques bornes précisément. Une piste de recherche pour remédier à ces deux problèmes peut être dans l'ajout de caractéristiques textuelles. Ces génériques et ces inter-programmes spécifiques affichent en effet des textes particuliers qui peuvent être utilisés grâce à des techniques de détection de textes ou

d'OCR.

Nous venons de proposer des caractéristiques textuelles pour aider à la résolution de difficultés posées par les génériques et des inter-programmes spécifiques. De manière plus générale, d'autres caractéristiques peuvent compléter et améliorer les résultats. En particulier, il peut être intéressant d'utiliser le signal audio et les répétitions audio. Une autre perspective de recherche est aussi l'utilisation des caractéristiques audiovisuelles pour la classification. Nous avons montré que la programmation logique inductive pouvait classer des segments audiovisuels avec de bonnes performances en utilisant des caractéristiques de répétition des segments et des informations contextuelles, relationnelles. Cependant les techniques traditionnelles utilisant des caractéristiques audiovisuelles pour la classification ne sont évidemment pas à rejeter. La combinaison de ces caractéristiques peut donc augmenter les performances de la classification.

Dans un contexte industriel, notre solution de délinéarisation n'est pas suffisante. Bien que nous ayons vérifié dans une certaine mesure nos quatre contraintes clés de généricité, d'efficacité, d'automaticité et de continuité, les résultats de la délinéarisation ne sont pas suffisamment complets pour pouvoir être utilisés directement, sur n'importe quelle chaîne, 24 heures sur 24, 7 jours sur 7, dans des services réels de télévision à la demande par exemple.

Pour notre système, nous avons intentionnellement choisi une approche générique adaptée à un maximum de chaînes TV. Cependant, nous avons aussi vu qu'une méthode spécifique à une chaîne TV particulière pouvait obtenir de meilleurs résultats pour cette chaîne. Une perspective applicative dans un but industriel, est donc de concevoir une plateforme de délinéarisation. Cette plateforme combinerait des traitements spécifiques efficaces avec des outils génériques. La plateforme serait ainsi flexible et capable de délinéariser de nombreux flux dont certains avec une excellente précision.

Pour finir, il nous paraît important d'étendre encore la généricité de notre méthode. Nous souhaitons ainsi étudier la délinéarisation des chaînes étrangères ou thématiques et analyser les résultats obtenus. Notre méthode peut s'adapter aux différentes chaînes. Par exemple, sur les chaînes thématiques, la rediffusion des programmes longs peut devenir une information aussi importante que la rediffusion des inter-programmes. En effet, une chaîne thématique comme un chaîne de cinéma rediffuse souvent ses programmes longs. La détection de la rediffusion des programmes longs est relativement facile à détecter. Elle fournit directement les bornes des programmes rediffusés sans nécessiter de réunification.

# Annexes

Sommaire

Annexe A

# Le corpus télévisuel

Toutes les expériences réalisées dans cette thèse reposent sur un seul et même corpus télévisuel. Afin d'alléger la lecture de ce document, ce corpus est décrit une unique fois dans cette annexe.

Le corpus télévisuel est constitué de deux flux télévisuels réels enregistrés en continu sur une période d'un mois pendant l'été 2007. Le premier flux est issu de la première grande chaîne nationale française « TF1 ». Le second flux est issu de la deuxième grande chaîne nationale française « France 2 ». Ces deux flux contiennent exactement 4 semaines d'éléments de télévision, soit 1344 heures au total.

Lors des expériences, les flux sont divisés en plusieurs parties. Nous utilisons plus précisément 4 parties qui correspondent aux 4 semaines enregistrées. Ces semaines sont identifiées suivant les numéros donnés dans le tableau 6.

| Numéro de la semaine | Description | Chaînes |
|---|---|---|
| n° 1 | Semaine du 09 Juillet au 15 Juillet 2007 | TF1 et France 2 |
| n° 2 | Semaine du 16 Juillet au 22 Juillet 2007 | TF1 et France 2 |
| n° 3 | Semaine du 23 Juillet au 29 Juillet 2007 | TF1 et France 2 |
| n° 4 | Semaine du 30 Juillet au 05 Août 2007 | TF1 et France 2 |

Tableau 6 – Présentation du corpus télévisuel.

Nous avons sélectionné deux chaînes de télévision de manière à vérifier la généricité de notre système. Nous avons alors choisi TF1 et France 2 pour leur nature différente. TF1 est une chaîne privée. Son financement dépend principalement des recettes publicitaires. TF1 contient donc une part importante de messages commerciaux qui se retrouvent dans les inter-programmes. France 2 est à l'inverse une chaîne publique financée en premier par la taxe de la redevance audiovisuelle. Elle contient donc moins de publicités dans ses inter-programmes.

Pour mieux caractériser ces deux chaînes de télévision, nous avons calculé quelques caractéristiques statistiques sur l'ensemble du corpus. Celles-ci sont résumées dans le tableau 7. Les proportions par catégorie d'inter-programmes ne sont calculées que pour les 3 catégories principales : les publicités, les bandes annonces et les parrainages. Au total, les publicités, les bandes annonces et les parrainages forment à eux seuls plus de 95 % de tous les inter-programmes à la fois sur France 2 et sur TF1.

Les données du tableau 7 montrent les différences entre les deux chaînes. Il y a pratiquement 1,5 fois plus d'éléments d'inter-programmes sur TF1 que sur France 2. La proportion d'éléments publicitaires est aussi plus importante sur TF1. Mais France 2 compense ce manque de publicité en diffusant plus de bandes annonces.

| Caractéristiques | TF1 | France 2 |
|---|---|---|
| Durée totale des programmes courts | 1 j. 1 h. 25 m. 23 s. | 20 h. 25 m. 43 s. |
| Durée totale des inter-programmes (IP) | 3 j. 9 h. 54 m. 0 s. | 2 j. 8 h. 20 m. 12 s. |
| Durée moyenne d'un IP | 18,6 s | 19,3 s |
| Proportion des programmes courts dans le flux | 3,78 % | 3,04 % |
| Proportion des IP dans le flux | 12,19 % | 8,38 % |
| Proportion des publicités dans les IP | 70,73 % | 62,99 % |
| Proportion des bandes annonces dans les IP | 18,06 % | 27,10 % |
| Proportion des parrainages dans les IP | 6,55 % | 5,55 % |

Tableau 7 – Caractéristiques de TF1 et de France 2 sur 4 semaines.

D'un point de vue technique, le corpus télévisuel est enregistré par fichiers de 10 minutes. Ces fichiers courts sont compressés puis organisés par chaînes, par journées et par heures. Cette configuration permet d'optimiser l'accès aléatoire à un instant quelconque du large corpus télévisuel.

Annexe B

# La vérité terrain

Les résultats de notre système sont évalués vis à vis d'une vérité terrain. Cette vérité terrain correspond à ce que devrait produire idéalement notre système. Elle a été réalisée à la main. La vérité terrain permet de comparer les résultats obtenus de manière automatique avec des résultats de référence. Il en découle des mesures de performance calculées automatiquement.

La vérité terrain est un élément clé dans notre protocole d'évaluation. Nous décrivons en premier la vérité terrain utilisée puis nous proposons une mesure de la qualité de notre vérité terrain.

## Description de la vérité terrain

La vérité terrain a été construite sur le corpus télévisuel décrit dans l'annexe A. Pour les deux chaînes TF1 et France 2 et pour les 4 semaines du corpus, tous les programmes et les inter-programmes ont été délimités à la main. Plus précisément, nous avons détectés à la main les bornes de début et de fin de toutes les parties de programmes et de tous les inter-programmes. Ces éléments ainsi délimités ont été annotés par un nom et un type. Le nom résume le contenu des éléments télévisuels. Le type est choisi parmi les 6 catégories suivantes :

1) les programmes longs,

2) les programmes courts,

3) les publicités,

4) les bandes annonces,

5) les parrainages,

6) les autres inter-programmes.

Au total, nous avons délimité 20031 bornes inférieures de programmes et d'inter-programmes sur les 4 semaines de TF1 et 13615 bornes inférieures de programmes et d'inter-programmes sur les 4 semaines de France 2.

| Caractéristiques | TF1 | France 2 |
|---|---|---|
| Nombre d'inter-programmes (IP) répétés | 545 | 348 |
| Nombre total d'occurrences d'IP répétés | 3884 | 2704 |
| Nombre d'occurrences maximum d'un IP répété | 38 | 34 |
| Type de l'IP répété le plus fréquent | Parrainage | Parrainage |

Tableau 8 – Caractéristiques des inter-programmes répétés sur une semaine sur France 2 et sur TF1.

De plus, les semaines n° 1 de TF1 et de France 2 ont été analysées de manière plus approfondie. Tous les inter-programmes ont été rassemblés manuellement en groupes d'inter-programmes quasi identiques. Pour cela, nous avons utilisé les noms donnés aux contenus des inter-programmes télévisuels et nous avons comparé deux à deux les éléments de noms voisins. Ce regroupement des inter-programmes identiques, et donc répétés, détermine les occurrences des répétitions des inter-programmes. Il crée ainsi une vérité terrain spécifique pour l'évaluation de la détection des répétitions d'inter-programmes. Nous donnons dans le tableau 8 des caractéristiques des inter-programmes répétés sur les semaines n° 1 de TF1 et de France 2.

Lors de l'analyse approfondie des semaines n° 1 de TF1 et de France 2, nous avons aussi recensé manuellement les programmes longs rediffusés et les programmes courts rediffusés. Cette opération nous permet de calculer la proportion de flux non rediffusé dans le flux sur une semaine de TF1 et sur une semaine de France 2. Les résultats sont donnés dans le tableau 9.

| | TF1 | France 2 |
|---|---|---|
| Proportion de flux non rediffusé dans le flux | 79,73 % | 83,90 % |

Tableau 9 – Proportion de flux non rediffusé dans le flux sur une semaine de TF1 et sur une semaine de France 2.

Pour construire la vérité terrain, nous avons employé un logiciel conçu spécifiquement pour cela. Ce logiciel minimise le nombre d'interactions à effectuer par un utilisateur pour délimiter un nouvel élément et l'annoter. La figure 12 présente l'interface de ce logiciel. Tous les éléments de l'interface sont directement accessibles. Il n'y a pas de boite de dialogue. L'utilisateur navigue dans un premier temps dans le flux à travers les contrôles disponibles. Puis, il délimite en un seul « clic » les différentes bornes. Il peut alors entrer les annotations. Nous avons défini les spécifications de ce logiciel qui a été implémenté par une société tierce.

### Qualité de la vérité terrain

La vérité terrain est très coûteuse à obtenir. Dans le cadre de la télévision, elle requiert de regarder en accéléré tout le contenu du corpus télévisuel. C'est un travail long, répétitif et fastidieux. À cause de ces difficultés, la vérité terrain n'est pas parfaite. Elle comporte quelques erreurs. Lorsque ces erreurs sont très importantes, elles produisent des résultats aberrants au cours des évaluations. Les erreurs très importantes peuvent donc être identifiées. Malheureusement, il peut persister d'autres erreurs mineures difficiles à détecter.

Afin d'estimer la qualité de la vérité terrain sur le corpus entier, nous avons étudié la qualité de la vérité terrain sur une semaine du corpus. Pour cela, nous avons comparé deux vérités terrain réalisées par deux personnes différentes (A et B) sur une même semaine de TF1. Les deux personnes ont travaillé suivant les mêmes consignes et les mêmes conditions. Nous avons ensuite comparé les bornes des éléments de programmes longs et

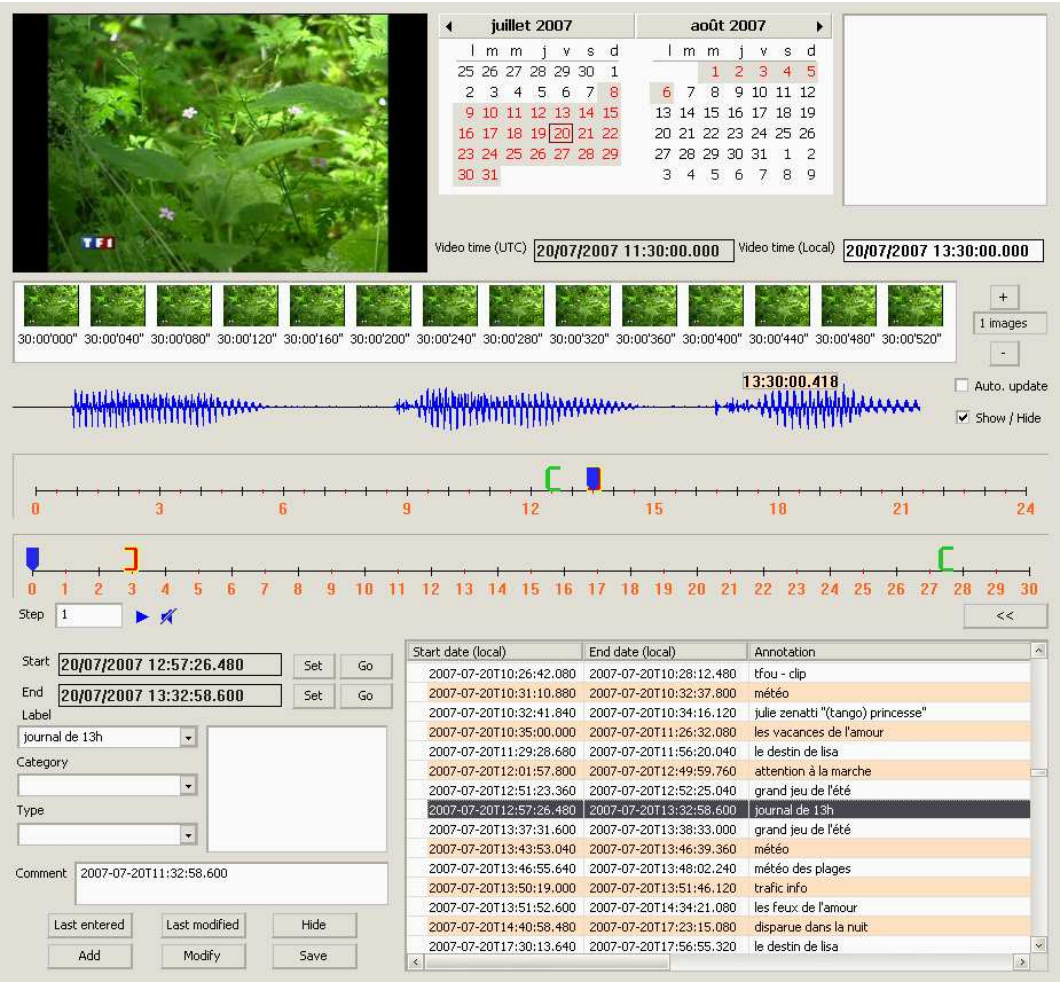



FIGURE 12 – Interface du logiciel de création de la vérité terrrain.

de programmes courts. Les différences entre les deux vérités terrains ont alors permis de déterminer les erreurs effectuées par chacune des personnes.

Le tableau 10 montre les erreurs détectées sur la semaine étudiée. Parmi ces erreurs, une erreur était due à l'oubli d'un programme court. Les autres erreurs étaient dues à l'oubli d'une coupure très courte d'inter-programmes. Nous avons compté 420 programmes courts et programmes longs. Au final, nous pouvons estimer que la vérité terrain comporte très peu d'erreurs.

Nous avons également évalué les écarts entre les bornes inférieures établies par la personne A et les bornes inférieures établies par la personne B. Idéalement, ces écarts sont nuls. En moyenne, nous avons mesuré un écart maximum de 2,64 secondes et un écart moyen d'environ 0,23 seconde (soit environ 6 images) entre les bornes inférieures délimitées par les personnes A et B. Nous pouvons en conclure que notre vérité terrain est seulement précise à la seconde près. Ces écarts ne nuisent pas à l'intégrité des programmes

| | Nombre d'erreurs |
|---|---|
| Personne A | 3 |
| Personne B | 1 |

Tableau 10 – Erreurs réalisées par deux personnes différentes lors de la construction d'une même semaine de vérité terrain dans les mêmes conditions.

délimités par les personnes A et B. Ils proviennent souvent du fait que les débuts et fins des programmes peuvent être fondus avec l'habillage des chaînes de télévision. Ainsi, des jingles propres aux chaînes de télévision ou des images monochromes peuvent être faussement inclus dans les programmes. Ces écarts peuvent aussi provenir des parrainages et des logos diffusés avant et après les programmes. Les parrainages qui dépendent des chaînes de télévision n'appartiennent pas aux programmes alors que les logos des producteurs des programmes y appartiennent. Ces deux éléments peuvent se ressembler. Il devient donc ambigü de les différencier lors du choix des bornes. Toutes ces ambigüités peuvent se résoudre en suivant des protocoles très stricts lors de la délinéarisation manuelle. Néanmoins, ces protocoles ralentissent l'efficacité de la construction de la vérité terrain. Nous n'avons pas pu nous les imposer.

Les erreurs et les écarts révélés dans cette petite étude montrent bien la difficulté d'établir des délinéarisations manuelles précises. Détecter manuellement une borne d'un programme est une tâche ennuyeuse qui conduit à ces erreurs humaines.

Annexe C

# Réalisations logicielles

Les travaux de cette thèse ont donné lieu à l'implémentation de deux démonstrations logicielles. Ces démonstrations permettent de visionner d'une manière confortable les résultats obtenus automatiquement par notre système. Elles ont été utiles pour illustrer nos résultats au sein du groupe Orange - France Télécom mais aussi au près de la communauté scientifique internationale. Elles ont été développées sous Visual Studio 2008 en C++ avec la bibliothèque Microsoft Foundation Class (MFC).

La première démonstration offre la possibilité d'explorer les ensembles de répétitions détectées. La deuxième démonstration sert à naviguer à travers les programmes TV automatiquement délinéarisés.

## Démonstration des résultats de la détection des répétitions

Les répétitions constituent la base de notre système. Il nous est très important de bien les détecter. Afin d'étudier ces répétitions d'une manière agréable, nous avons développé l'interface présentée dans la figure 13. À gauche, toutes les séquences répétées sont listées. Lorsqu'une séquence répétée est sélectionnée, toutes ses occurrences sont affichées à droite. Pour chacune des occurrences, un marqueur est ajouté en bas à droite afin d'indiquer la position de l'occurrence dans la journée et dans le calendrier. Dans cet exemple, la séquence répétée sélectionnée correspond au message de santé publique destiné aux enfants « Évite de grignoter dans la journée ». Nous voyons, grâce à l'interface, que cette séquence répétée est rediffusée 22 fois et que ces diffusions apparaissent généralement le matin entre 6h et 11h.

Afin de vérifier visuellement les répétitions, nous avons ajouté le moyen de lire en parallèle les différentes occurrences d'une séquence répétée. Un exemple de lecture en parallèle est donné dans la figure 14. Les occurrences sélectionnées sont affichées côte à côte et les images de toutes les occurrences sont lues simultanément. Cette fonctionnalité nous est très utile pour analyser avec précision les bornes des répétitions détectées.

## Démonstration des résultats de la délinéarisation

Les résultats finaux de notre système sont des segments de programmes et d'interprogrammes. Les bornes de ces segments sont conservés dans des fichiers XML. Nous avons donc implémenté une interface pour naviguer de manière ergonomique dans les listes XML des segments délinéarisés. Cette interface est montrée dans la figure 15.

L'interface montre le résultat de la délinéarisation d'un flux télévisuel. Elle se divise en 3 parties. La première partie montre les segments en orange automatiquement délinéarisés par notre méthode. La deuxième partie présente en vert la vérité terrain établie pour le flux. Enfin, la troisième partie donne en bleu les informations de métadonnées lorsqu'elles sont disponibles. Chacune des parties est composée d'un repère temporel qui représente les heures de diffusion du flux vidéo considéré. Il est naturellement possible de naviguer dans

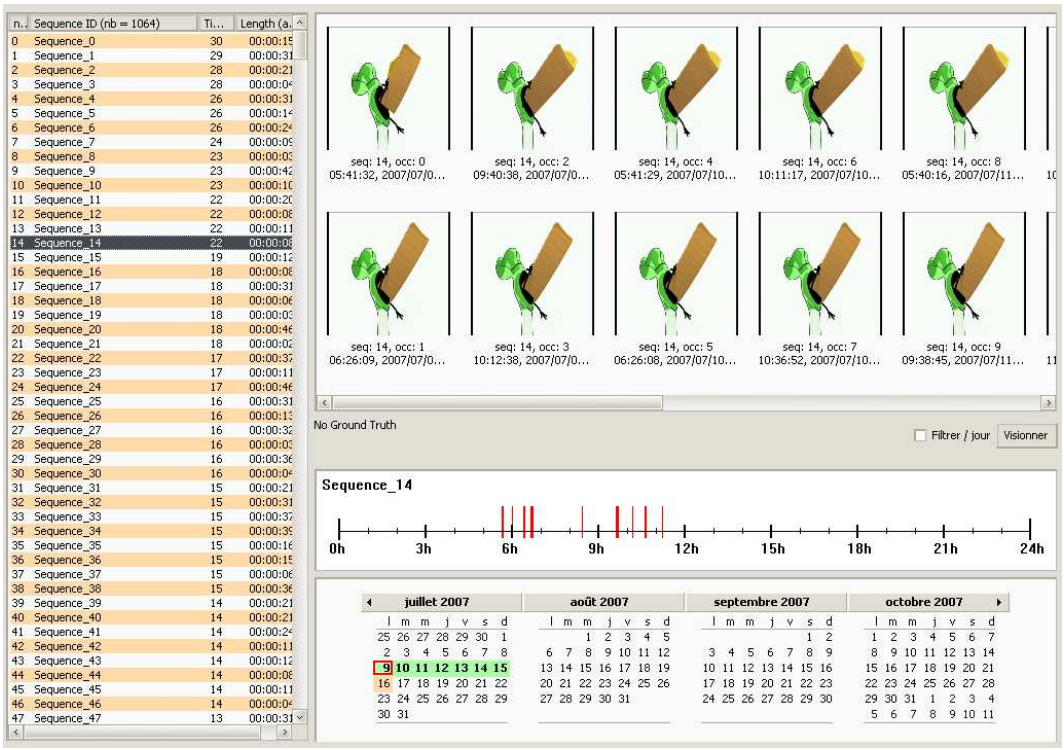



FIGURE 13 – Interface de l'explorateur des répétitions détectées.


FIGURE 14 – Interface du lecteur en parallèle des répétitions télévisuelle.

le temps grâce aux contrôles disponibles en bas à droite de l'interface. Ces trois parties sont toujours synchronisées dans le temps.

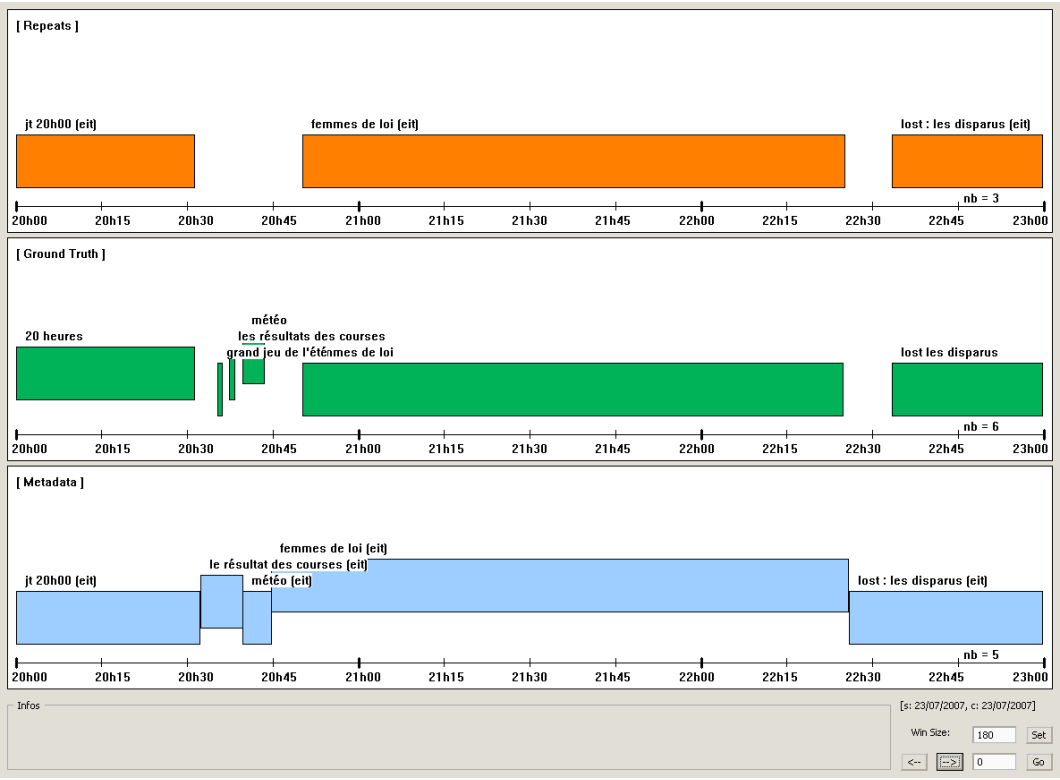



FIGURE 15 – Interface de la démonstration des résultats de la délinéarisation automatique.

L'exemple de la figure 15 illustre un bon fonctionnement de notre système. Les métadonnées disponibles sont décalées dans le temps par rapport à la vérité terrain. Notre méthode parvient bien à délimiter des segments de programmes qui correspondent visuellement à ceux de la vérité terrain.

Annexe D

# Description détaillée de l'algorithme du micro-*clustering*

Nous décrivons plus précisément, dans cette annexe, l'algorithme de micro-*clustering* utilisé pour la détection des images clés répétées. Cet algorithme est l'algorithme de micro-*clustering* présenté dans la thèse de Berrani [Ber04]. C'est une amélioration de l'algorithme BIRCH (*Balanced Iterative Reducing and Clustering using Hierarchies* [ZRL96]) qui construit des petits *clusters*. Nous présentons d'abord les micro-*clusters* puis l'algorithme de micro-*clustering*.

## Représentation des micro-*clusters*

Avant de détailler l'algorithme de micro-*clustering*, nous présentons les *clusters* utilisés. Ces *clusters* sont, dans notre contexte, des groupes d'images clés. Ils peuvent s'agrandir et fusionner avec d'autres *clusters*.

Nous considérons un ensemble d'images clés décrites par $d$ caractéristiques. Ces images clés sont à regrouper dans un espace euclidien à $d$ dimensions. Les *clusters* construits par l'algorithme sont représentés par des « CF-vecteurs »(*Clustering Feature Vector*). Un CF-vecteur est initialement un triplet $CF = (p, \vec{c}, r)$, où $p$ est le nombre d'images clés $\vec{i}_1, \ldots, \vec{i}_p$ du *cluster*, $\vec{c}$ est le centre de gravité et $r$ est le rayon effectif de l'hypersphère englobante (sphère de dimension $d$) centrée sur $\vec{c}$. Ici, $r$ est la distance entre le centre et la plus lointaine image clé parmi les images clés du *cluster*.

$$\vec{c} = \frac{\sum_{k=1}^{p} \vec{i}_k}{p}$$
$$r = \max_{1 \leq k \leq p} \|\vec{i}_k - \vec{c}\|$$

Lorsque deux CF-vecteurs $CF_1 = (p_1, \vec{c}_1, r_1)$ et $CF_2 = (p_2, \vec{c}_2, r_2)$ fusionnent, le rayon effectif $\tilde{r}$ à l'issue de la fusion est légèrement surestimé. Le CF-vecteur résultant est $CF = CF_1 + CF_2$. La figure 16 illustre la fusion de deux micro-*clusters*.

$$CF = (p_1 + p_2, \vec{c} = \frac{p_1\vec{c}_1 + p_2\vec{c}_2}{p_1 + p_2}, \tilde{r})$$
$$\tilde{r} = \max(\|\vec{c}_1 - \vec{c}\| + r_1, \|\vec{c}_2 - \vec{c}\| + r_2)$$

Finalement, un CF-vecteur devient un triplet $CF = (p, \vec{c}, \tilde{r})$. $p$ et $\vec{c}$ sont inchangés mais $r$ devient $\tilde{r}$, le rayon d'une hypersphère englobante de l'ensemble des images clés mais qui n'est pas forcément minimale. Un autre rayon utilisé est le rayon moyen des *clusters*. L'inconvénient du rayon moyen est que si un *cluster* contient beaucoup d'images clés rapprochées, une image clé un peu trop éloignée impacte peu sur le rayon moyen et risque d'être absorbée au détriment de la compacité. L'avantage de surestimer le rayon est dans un premier temps d'éviter le calcul des distances entre toutes les images clés du

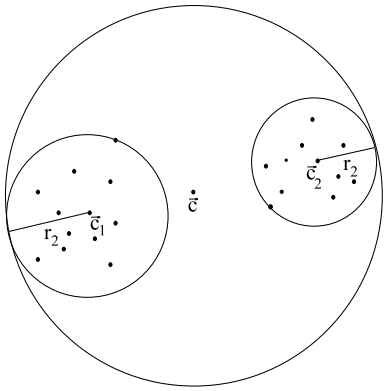



FIGURE 16 – Fusion de deux *clusters* (figure extraite de la thèse de Berrani [Ber04]).

*cluster* et de son centre de gravité. Cependant, la surestimation aide en plus à la compacité des *clusters*. Pour un même rayon $r$, les images clés du *cluster* sont plus proches si le rayon est surestimé.

### Description de l'algorithme de micro-*clustering*

L'algorithme en entier est détaillé par les algorithmes 6, 7 et 8. Nous adoptons les notations suivantes :

$A \leftarrow B$ : affecter $B$ à la variable $A$
$L \leftarrow L, X$ : ajouter l'élément $X$ à la fin de la liste $L$
$Taille(L)$ : retourne le nombre d'éléments de la liste $L$
$L[i]$ : retourne le $i^e$ élément de la liste $L$
$CF.p$ : retourne le nombre d'images clés du CF-vecteur $CF$
$CF.\vec{c}$ : retourne le centre de gravité du CF-vecteur $CF$
$CF.r$ : retourne le rayon du CF-vecteur $CF$

La performance de l'algorithme repose principalement dans le choix des paramètres qui contrôlent le rayon maximal, noté $Rmax$, des *clusters*. Ces paramètres sont $Rmax_0$, le rayon maximal initial des *clusters*, $\Delta_{Rmax}$, le pas d'incrémentation de ce rayon maximal, et $Rmax_{stop}$, le rayon maximal qui constitue le critère d'arrêt. Au lieu de limiter le rayon maximal $Rmax$ par $Rmax_{stop}$, nous pouvons aussi limiter le nombre de passes ou le nombre maximum de *clusters* créés. L'algorithme peut, par exemple, se simplifier à l'algorithme 9 dans lequel une seule passe est effectuée. Ce dernier algorithme dépend finalement seulement du choix du rayon maximal $Rmax$ des *clusters*.

Nous remarquons qu'aucune structure d'indexation multidimensionnelle n'est utilisée lors de l'insertion des *clusters* contrairement à l'algorithme BIRCH [ZRL96] original. Une liste simplement chaînée suffit. Le choix d'une organisation séquentielle plus coûteuse s'est imposée à cause du phénomène de « la malédiction de la dimension » (*dimensionality curse*). Les raisons précises à cela sont multiples et clairement expliquées dans [Ber04].

**micro-clustering**

▷ **Entrées:**
$L_0$ : liste de vecteurs ;
$Rmax_0$ : valeur initiale du rayon maximal Rmax ;
$\Delta_{Rmax}$ : valeur à ajouter pour incrémenter le rayon maximal Rmax ;
$Rmax_{stop}$ : critère d'arrêt du rayon maximal Rmax ;

◁ **Sorties:**
$L_{clusters}$ : liste de CF-vecteurs des micro-clusters ;

**Début**
1: $L_{clusters} \leftarrow \emptyset$ ;
2: **pour** $i = 1$ à $Taille(L_0)$ **faire**
3: $CF_{tmp} \leftarrow$ nouveau CF-vecteur de rayon nul à partir de $L_0[i]$ ;
4: $L_{clusters} \leftarrow L_{clusters}, CF_{tmp}$ ;
5: **fin pour**
6: $Rmax \leftarrow Rmax_0$ ;
7: **tant que** $Rmax \leq Rmax_{stop}$ **faire**
8: $L_{clusters} \leftarrow$ micro-clustering-simple$(L_{clusters}, Rmax)$ ;
9: $Rmax \leftarrow Rmax + \Delta_{Rmax}$ ;
10: **fin tant que**

**Fin**

Algorithme 6 – Micro-*clustering* de vecteurs avec variation de $Rmax$

**micro-clustering-simple**

▷ **Entrées:**
$L_0$ : liste de CF-vecteurs ;
$Rmax$ : valeur du rayon maximal Rmax ;

◁ **Sorties:**
$L_{clusters}$ : liste de CF-vecteurs des micro-clusters ;

**Début**
1: $L_{clusters} \leftarrow \emptyset$ ;
2: $L_{outliers} \leftarrow \emptyset$ ;
3: **pour** $i = 1$ à $Taille(L_0)$ **faire**
4: **si** $L_0[i].p < 2$ **alors**
5: $L_{outliers} \leftarrow L_{outliers}, L_0[i]$ ;
6: **sinon**
7: inserer$(L_0[i], L_{clusters}, Rmax)$ ;
8: **fin si**
9: **fin pour**
10: **pour** $j = 1$ à $Taille(L_{outliers})$ **faire**
11: inserer$(L_{outliers}[j], L_{clusters}, Rmax)$ ;
12: **fin pour**

**Fin**

Algorithme 7 – Micro-*clustering* simple de CF-vecteurs

**inserer**

▷ **Entrées:**
$CF$ : CF-vecteur à insérer ;
$L$ : liste de CF-vecteurs dans laquelle effectuer l'insertion ;
$Rmax$ : valeur du rayon maximal Rmax ;

◁ **Sorties:**
$L$ : liste de CF-vecteurs modifiée ;

**Début**
1: $i \leftarrow$ indice du CF-vecteur de $L$ le plus proche de $CF$ ;
2: $CF_{tmp} \leftarrow L[j] + CF$ ; *// Fusion*
3: **si** $CF_{tmp}.r > Rmax$ **alors**
4: $\quad L \leftarrow L, CF$ ;
5: **sinon**
6: $\quad L[j] \leftarrow CF_{tmp}$ ;
7: **fin si**

**Fin**

Algorithme 8 – Insertion d'un CF-vecteur

**micro-clustering sans variation de Rmax**

▷ **Entrées:**
$L_0$ : liste de vecteurs ;
$Rmax$ : valeur du rayon maximal Rmax ;

◁ **Sorties:**
$L_{clusters}$ : liste de CF-vecteurs des micro-clusters ;

**Début**
1: $L_{clusters} \leftarrow \emptyset$ ;
2: **pour** $i = 1$ à $Taille(L_0)$ **faire**
3: $\quad CF_{tmp} \leftarrow$ nouveau CF-vecteur de rayon nul à partir de $L_0[i]$ ;
4: $\quad L_{clusters} \leftarrow L_{clusters}, CF_{tmp}$ ;
5: **fin pour**
6: $L_{clusters} \leftarrow$ micro-clustering-simple$(L_{clusters}, Rmax)$ ;

**Fin**

Algorithme 9 – Micro-*clustering* de vecteurs sans variation de $Rmax$

Annexe E

# Références de l'auteur

Cette thèse a été écrite en s'appuyant sur les travaux suivants soumis à la communauté scientifique spécialisée nationale et internationale.

## Revues scientifiques

G. Manson et S.-A. Berrani. Automatic TV Broadcast Structuring. *International Journal of Digital Multimedia Broadcasting, special issue on Video Analysis, Abstraction and Retrieval : Techniques and Applications*, vol. 2010, Article ID 153160, 16 pages, 2010.

S.-A. Berrani, G. Manson et P. Lechat. A non supervised approach for repeated sequence detection in TV broadcast streams. *Signal Processing : Image Communication, special issue on Semantic Analysis for Interactive Multimedia Services*, 23(7) :525-537, 2008.

## Brevets

G. Manson et S.-A. Berrani. Procédé de fusion de segments de programmes audiovisuels. Brevet déposé par Orange Labs – France Télécom R&D le 6 février 2009.

S.-A. Berrani et G. Manson. Procédé de classification automatique de segments composant un flux TV. Brevet déposé par Orange Labs – France Télécom R&D le 15 février 2008.

## Congrès internationaux

G. Manson et S.-A. Berrani. Content-Based Video Segment Reunification for TV Program Extraction. *Proceedings of the IEEE International Symposium on Multimedia*, San Diego, Californie, USA, décembre 2009.

G. Manson et S.-A. Berrani. Repetition Density-Based Approach For TV Program Extraction. *Proceedings of the IEEE International Workshop on Image Analysis for Multimedia Interactive Services*, Londres, Royaume-Uni, mai 2009.

G. Manson et S.-A. Berrani. TV Broadcast Macro-Segmentation using the Repetition Property of Inter-Programs. *Proceedings of the IASTED International Conference on Signal Processing : Pattern Recognition and Applications*, Innsbruck, Autriche, février 2009.

G. Manson et S.-A. Berrani. An Inductive Logic Programming-Based Approach for TV Stream Segment Classification. *Proceedings of the IEEE International Symposium on Multimedia*, Berkeley, Californie, USA, décembre 2008.

S.-A. Berrani, P. Lechat et G. Manson. TV broadcast macro-segmentation : Metadata-based vs content-based approaches. *Proceedings of the ACM International Conference on Image and Video Retrieval*, Amsterdam, Pays-Bas, juillet 2007.

### Congrès nationaux

G. Manson et S.-A. Berrani. Structuration automatique de flux télévisés. *Actes du colloques GRETSI sur le traitement du signal et des images*, Dijon, France, septembre 2009.

### Démonstrations

G. Manson, X. Naturel et S.-A. Berrani. Automatic Program Extraction From TV Streams. *Proceedings of the ACM European Interactive TV Conference*, Louvain, Belgique, juin 2009.

G. Manson, X. Naturel et S.-A. Berrani. Online Macro-segmentation of Television Streams. *Proceedings of the ACM International Multimedia Modeling Conference*, Sophia-Antipolis, France, janvier 2009.

G. Manson et S.-A. Berrani. A TV Stream Macro-segmentation System. *Proceedings of the IEEE International Symposium on Multimedia*, Berkeley, Californie, USA, décembre 2008.

# Bibliographie


[AFAT04] A. Albiol, M.J. Fulla, A. Albiol, et L. Torres. Detection of TV commercials. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 541–544, Montréal, Québec, Canada, mai 2004.

[Aok05] H. Aoki. *Image and Video Retrieval*, volume 3568, chapitre High-Speed Dialog Detection for Automatic Segmentation of Recorded TV Program, pages 49–58. Lecture Notes in Computer Science, Springer Berlin / Heidelberg, août 2005.

[Bas88] M. Basseville. Detecting changes in signals and systems – A survey. *Automatica Journal*, 24(3) :309–326, mai 1988.

[BC08] D. Brezeale et D. J. Cook. Automatic video classification : A survey of the literature. *IEEE Transactions on Systems, Man and Cybernetics*, 38(3) :416–430, mai 2008.

[Ber04] S.-A. Berrani. *Recherche approximative de plus proches voisins avec contrôle probabiliste de la précision ; application à la recherche d'images par le contenu.* Thèse de doctorat, Université de Rennes 1, février 2004.

[BL06] S.-A. Berrani et P. Lechat. Structuring a digital data stream. (WO2008065297), Brevet déposé par France Telecom, $1^{er}$ décembre 2006.

[BS02] J. Brassil et H. Schulzrinne. Structuring internet media streams with cueing protocols. *IEEE/ACM Transactions on Networking*, 10(4) :466–476, août 2002.

[Car00] J. Carrive. *Classification de séquences audiovisuelles.* Thèse de doctorat, Laboratoire d'informatique de Paris 6, septembre 2000.

[CBF06] M. Covell, S. Baluja, et M. Fink. Advertisement detection and replacement using acoustic and visual repetition. *Proceedings of the IEEE International Workshop on Multimedia Signal Processing*, pages 461–466, Victoria, BC, Canada, octobre 2006.

[CL01] C.-C. Chang et C.-J. Lin. *LIBSVM : a Library for Support Vector Machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[Cla03] V. Claveau. *Acquisition automatique de lexiques sémantiques pour la recherche d'information.* Thèse de doctorat, Université de Rennes 1, 2003.

[CM02] A. Cornuéjols et L. Miclet. *Apprentissage artificiel.* Collection technique et scientifique des télécommunications, Eyrolles, décembre 2002.

[CN05] S.S. Cheung et T.P. Nguyen. Mining arbitrary-length repeated patterns in television broadcast. *Proceedings of the IEEE International Conference on Image Processing*, pages 181–185, Gênes, Italie, septembre 2005.

[CS04] B. Coskun et B. Sankur. Robust video hash extraction. *Proceedings of the IEEE Signal Processing and Communications Applications Conference*, pages 292–295, Vienne, Autriche, septembre 2004.

[DAW00] Nevenka Dimitrova, Lalitha Agnihotri, et Gang Wei. Video classification based on hmm using text and faces. *Proceedings of the European Signal Processing Conference*, pages 1373–1376, Tampere, Finlande, septembre 2000.

[DCH04] P. Duygulu, M.-Y. Chen, et A. Hauptmann. Comparison and combination of two novel commercial detection methods. *Proceedings of the IEEE International Conference on Multimedia and Expo*, pages 1267–1270, Taipei, Taiwan, juin 2004.

[DCI04] A. Dorado, J. Calic, et E. Izquierdo. A rule-based video annotation system. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(5) :622–633, mai 2004.

[DJN+02] N. Dimitrova, S. Jeannin, J. Nesvadba, T. McGee, L. Agnihotri, et G. Mekenkamp. Real time commercial detection using MPEG features. *Proceedings of the International Conference On Information Processing and Management of Uncertainty in knowledge-based systems*, pages 1–6, Annecy, France, juillet 2002.

[DL09] I. Döhring et R. Lienhart. Mining tv broadcasts for recurring video sequences. *Proceedings of the ACM International Conference on Image and Video Retrieval*, Santorin, Grèce, juillet 2009.

[DRD97] L. De Raedt et L. Dehaspe. Clausal discovery. *Machine Learning Journal*, 26(2-3) :99–146, 1997.

[DWZ+06] L.-Y. Duan, J. Wang, Y. Zheng, J.S. Jin, H. Lu, et C. Xu. Segmentation, categorization, and identification of commercial from tv streams using multimodal analysis. *Proceedings of the ACM international conference on Multimedia*, pages 201–210, Santa Barbara, Californie, USA, octobre 2006.

[EKSJ08] E. El-Khoury, C. Sénac, et P. Joly. Unsupervised TV program boundaries detection based on audiovisual features. *Proceedings of the IEEE International Conference on Visual Information Engineering*, pages 498–503, Xi'an, Chine, juillet 2008.

[ETS97a] European Telecommunication Standard Institute. Digital video broadcasting (DVB) ; Specification for service information (SI) in DVB systems. Standard 300 468 Édition 2, janvier 1997.

[ETS97b] European Telecommunication Standard Institute. Enhanced teletext specification. Standard 300 706 Édition 1, mai 1997.

[ETS98] European Telecommunication Standard Institute. Television systems ; specification of the domestic video programme delivery control system (PDC). Standard 300 231 Édition 2, avril 1998.

[ETS09] European Telecommunication Standard Institute. TV-Anytime specifications. Standard 102 822 Series, 2003 – 2009.

[FBGS04] B. Fauvet, P. Bouthemy, P. Gros, et F. Spindler. *Image and Video Retrieval*, volume 3115, chapitre A Geometrical Key-Frame Selection Method Exploiting

Dominant Motion Estimation in Video, pages 419–427. Lecture Notes in Computer Science, Springer Berlin / Heidelberg, juin 2004.

[FC03] J. T. Foote et M. L. Cooper. Media segmentation using self-similarity decomposition. *Proceedings of the Conference on Storage and retrieval for media databases*, pages 167–175, Santa Clara, CA, USA, janvier 2003.

[Fro05] E. Fromont. *Apprentissage multisource par programmation logique inductive : application à la caractérisation d'arythmies cardiaques.* Thèse de doctorat, Université de Rennes 1, décembre 2005.

[GD04] C. Garcia et M. Delakis. Convolutional face finder : A neural architecture for fast and robust face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11) :1408 – 1423, novembre 2004.

[GFT98] B. Günsel, A. Mufit Ferman, et A. Murat Tekalp. Temporal video segmentation using unsupervised clustering and semantic object tracking. *Journal of Electronic Imaging*, 7(3) :592–604, juillet 1998.

[GPR05] M. Guironnet, D. Pellerin, et M. Rombaut. Video classification based on low-level feature fusion model. *Proceedings of the European Signal Processing Conference*, Antalya, Turquie, septembre 2005.

[GS06] J.M. Gauch et A. Shivadas. Finding and identifying unknown commercials using repeated video sequence detection. *Journal of Computer Vision and Image Understanding*, 103(1) :80–88, juillet 2006.

[Hai05] S. Haidar. *Comparaison des Documents Audiovisuels par Matrice de Similarité.* Thèse de doctorat, Université Toulouse III - Paul Sabatier, septembre 2005.

[Han02] A. Hanjalic. Shot-boundary detection : unraveled and resolved ? *IEEE Transactions on Circuits and Systems for Video Technology,*, 12(2) :90–105, février 2002.

[Her06] C. Herley. Argos : Automatically extracting repeating objects from multimedia streams. *IEEE Transactions on Multimedia*, 8(1) :115–129, février 2006.

[Hin71] D.V. Hinkley. Inference about the change-point from cumulative sum tests. *Biometrika Journal*, 58(3) :509–523, décembre 1971.

[HK02] J. Haitsma et T. Kalker. A highly robust audio fingerprinting system. *Proceedings of the International Symposium on Music Information Retrieval*, pages 107–115, Paris, France, octobre 2002.

[HLZ05] X.-S. Hua, L. Lu, et H.-J. Zhang. Robust learning-based TV commercial detection. *Proceedings of the IEEE International Conference on Multimedia and Expo*, pages 149–152, Amsterdam, Pays-Bas, juillet 2005.

[HWJ94] A. Hampapur, T. Weymouth, et R. Jain. Digital video segmentation. *Proceedings of the ACM International Conference on Multimedia*, pages 357–364, San Francisco, CA, USA, octobre 1994.

[JH01] R. Jin et A.G. Hauptmann. Automatic title generation for spoken broadcast news. *Proceedings of the International Conference on Human Language Technology Research*, pages 1–3, San Diego, Californie, USA, mars 2001.

[Jol05] A. Joly. *Recherche par similarité statistique dans une grande base de signatures locales pour l'identification rapide d'extraits vidéo.* Thèse de doctorat, Université de La Rochelle, février 2005.

[KC01] I. Koprinska et S. Carrato. Temporal video segmentation a survey. *Signal Processing : Image Communication Journal*, 16(5) :477–500, janvier 2001.

[KCO+01] S. Kurtz, J. V. Choudhuri, E. Ohlebusch, C. Schleiermacher, J. Stoye, et R. Giegerich. Reputer : the manifold applications of repeat analysis on a genomic scale. *Nucleic Acids Research Journal*, 29(22) :4633–4642, novembre 2001.

[KSS95] R. D. King, M. J. E. Sternberg, et A. Srinivasan. Relating chemical activity to structure : An examination of ilp successes. *New Generation Computing Journal*, 13(3/4) :411–433, décembre 1995.

[KSY07] Y. Kawai, H. Sumiyoshi, et N. Yagi. Automated production of TV program trailer using electronic program guide. *Proceedings of the ACM international conference on Image and Video Retrieval*, pages 49–56, Amsterdam, Pays-Bas, juillet 2007.

[KY98] J. R. Kender et B.-L. Yeo. Video scene segmentation via continuous video coherence. *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 367–373, Santa Barbara, CA, USA, juin 1998.

[LD94] N. Lavrac et S. Dzeroski. *Inductive Logic Programming : Techniques and Applications.* Ellis Horwood, 1994.

[LDJ03] C. Lu, M.S. Drew, et Au J. An automatic video classification system based on a combination of hmm and video summarization. *International Journal of Smart Engineering System Design*, 5(1) :33–45, janvier–mars 2003.

[LGS06] Z. Liu, D.C. Gibbon, et B. Shahraray. Multimedia content acquisition and processing in the miracle system. *Proceedings of the IEEE Consumer Communications and Networking Conference*, pages 272–276, Las Vegas, Nevada, USA, janvier 2006.

[LHC99] C.-C. Liu, J.-L. Hsu, et A. L. P. Chen. Efficient theme and non-trivial repeating pattern discovering in music databases. *Proceedings of the International Conference on Data Engineering*, pages 14–21, Sydney, Australie, mars 1999.

[LKE97] R. Lienhart, C. Kuhmunch, et W. Effelsberg. On the detection and recognition of television commercials. *Proceedings of the IEEE International Conference on Multimedia Computing and Systems*, pages 509–516, Ottawa, Ontario, Canada, juin 1997.

[LKYH05] H. Lee, J.-G. Kim, S.-J. Yang, et J. Hong. Personalized tv services based on tv-anytime for personal digital recorder. *IEEE Transactions on Consumer Electronics*, 51(3) :885–892, août 2005.

[LLXT05] L. Liang, H. Lu, X. Xue, et Y.-P. Tan. Program segmentation for TV videos. *Proceedings of the IEEE International Symposium on Circuits and Systems*, pages 1549–1552, Kobe, Japon, mai 2005.

[LPE07] Le parlement européen et le conseil de l'union européenne. Directive 2007/65/ce modifiant la directive 89/552/cee visant à la coordination de certaines dispositions législatives, réglementaires et administratives des États

membres relatives à l'exercice d'activités de radiodiffusion télévisuelle, 11 décembre 2007.

[LZJ02] L. Lu, H.-J. Zhang, et H. Jiang. Content analysis for audio classification and segmentation. *IEEE Transactions on Speech and Audio Processing*, 10(7) :504–516, octobre 2002.

[LZZJ07] Y. Li, D. Zhang, X. Zhou, et J.S. Jin. A confidence based recognition system for tv commercial extraction. *Proceedings of the Conference on Australasian Database*, pages 57–64, Gold Coast, Australie, décembre 2007.

[MAEA05] A. Metwally, D. Agrawal, et A. El Abbadi. Duplicate detection in click streams. *Proceedings of the International conference on World Wide Web*, pages 12–21, Chiba, Japon, mai 2005.

[MBD+06] A. Messina, L. Boch, G. Dimino, W. Bailer P. Schallauer, W. Allasia, M. Groppo, M. Vigilante, et R. Basili. Creating rich metadata in the TV broadcast archives environment : The prestospace project. *Proceedings of the International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution*, pages 193–200, Leeds, Royaume-Uni, décembre 2006.

[Mit97] T. Mitchell. *Machine learning*. Computer Science Series, McGraw Hill, 1997.

[MM08] A. Messina et M. Montagnuolo. Multimedia genre characterisation with fuzzy embedding classifiers. *Proceedings of the Ambi-Sys workshop on Ambient media delivery and interactive television*, pages 1–8, Québec City, Canada, février 2008.

[MN03] H. Meinedo et J. Neto. Audio segmentation, classification and clustering in a broadcast news task. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 5–8, Hong Kong, Chine, avril 2003.

[MR94] S.H. Muggleton et L. De Raedt. Inductive logic programming : Theory and methods. *Journal of Logic Programming*, 19/20 :629–679, 1994.

[Mug91] S. Muggleton. Inductive logic programming. *New Generation Computing Journal*, 8(4) :295–318, 1991.

[Mug95] S. Muggleton. Inverse entailment and PROGOL. *New Generation Computing*, 13(3-4) :245–286, 1995.

[Nat07] X. Naturel. *Structuration automatique de flux vidéos de télévision.* Thèse de doctorat, Université de Rennes 1, avril 2007.

[OKH02] J. Oostveen, T. Kalker, et J. Haitsma. Feature extraction and a database strategy for video fingerprinting. *Proceedings of the International Conference on Recent Advances in Visual Information Systems*, pages 117–128, Hsin Chu, Taiwan, mars 2002.

[Pag54] E.S. Page. Continuous inspection schemes. *Biometrika Journal*, 41(1-2) :100–115, juin 1954.

[PFGM04] L. Perez-Freire et C. Garcia-Mateo. A multimedia approach for audio segmentation in tv broadcast news. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 369–372, Montréal, Québec, Canada, mai 2004.

[PGGM04] K.M. Pua, J.M. Gauch, S.E. Gauch, et J.Z. Miadowicz. Real time repeated video sequence identification. *Journal of Computer Vision and Image Understanding*, 93(3) :310–327, mars 2004.

[Pol07] J.-P. Poli. *Structuration automatique de flux télévisuels.* Thèse de doctorat, Université Paul Cézanne d'Aix-marseille 3, mai 2007.

[Pou09] S. Poullot. *Scalable Content-Based Video Copy Detection for Stream Monitoring and Video Mining.* Thèse de doctorat, Conservatoire National des Arts et Métiers, Paris, janvier 2009.

[Qui90] J. R. Quinlan. Learning logical definitions from relations. *Machine Learning Journal*, 5(3) :239–266, août 1990.

[RGL+04] M. Rovira, J. Gonzalez, A. Lopez, J. Mas, A. Puig, J. Fabregat, et G. Fernandez. Indextv : a MPEG-7 based personalized recommendation system for digital TV. *Proceedings of the IEEE International Conference on Multimedia and Expo*, pages 823–826, Taipei, Taiwan, juin 2004.

[RHM99] Y. Rui, T. S. Huang, et S. Mehrotra. Constructing table-of-content for videos. *Multimedia Systems Journal*, 7(5) :359–368, septembre 1999.

[RMX+02] M. Roach, J. Mason, L-Q. Xu, , et F. W. M. Stentiford. Recent trends in video analysis : a taxonomy of video classification problems. *Proceedings of the IASTED International Conference on Internet and Multimedia Systems and Applications*, Hawaii, USA, août 2002.

[RS05] Z. Rasheed et M. Shah. Detection and representation of scene in videos. *IEEE Transactions on Multimedia*, 7(6) :1097–1105, décembre 2005.

[SC00] H. Sundaram et S.-F. Chang. Audio scene segmentation using multiple features, models and time scales. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 2441–2444, Istanbul, Turquie, juin 2000.

[SCN+05] J. R. Smith, M. Campbell, M. Naphade, A. Natsev, et J. Tesic. Learning and classification of semantic concepts in broadcast video. *Proceedings of the International Conference on Intelligence Analysis*, pages 1–6, McLean, Virginie, USA, mai 2005.

[SG07] A. Shivadas et J.M. Gauch. Real-time commercial recognition using color moments and hashing. *Proceedings of the Canadian Conference on Computer and Robot Vision*, pages 465–472, Montréal, Québec, mai 2007.

[SM04] B. Satterwhite et O. Marques. Automatic detection of tv commercials. *IEEE Potentials Journal*, 23(2) :9–12, avril-mai 2004.

[SMOM02] D.A. Sadlier, S. Marlow, N. O'Connor, et N. Murphy. Automatic tv advertisement detection from mpeg bitstream. *Journal of the Pattern Recognition Society*, 35(12) :2719–2726, janvier 2002.

[Sri07] A. Srinivasan. Aleph : A learning engine for proposing hypotheses. Software available at `http://web2.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/aleph.pl`, 2007.

[SS94] L. Shapiro et E. Y. Sterling. *The Art of PROLOG : Advanced Programming Techniques.* MIT Press, avril 1994.

[SW05a] C. G.M. Snoek et M. Worring. Multimodal video indexing a review of the state of the art. *Multimedia Tools and Applications*, 25(1) :5–35, janvier 2005.

[SW05b] C.G.M. Snoek et M. Worring. Multimedia event-based video indexing using time intervals. *IEEE Transactions on Multimedia*, 7(4) :638–647, août 2005.

[TBA08] B. Tahayna, M. Belkhatir, et S. Alhashmi. Automatic indexing and recognition of re-broadcasted video programs through visual and temporal features. *Proceedings of the IEEE International Conference on Multimedia and Expo*, pages 861–864, Hanovre, Allemagne, juin 2008.

[THL98] B. Taves, J. Hoffman, et K. Lund. The moving image genre form guide. `http://www.loc.gov/rr/mopic/migintro.html`, 1998.

[Toz04] E.P.J. Tozer. *Broadcast engineer's reference book*. Elsevier, 2004.

[TV07] B. T. Truong et S. Venkatesh. Video abstraction : A systematic review and classification. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 3(1) :1–37, février 2007.

[Ver01] D. Vernant. *Introduction à la logique standard*. Champs Université, Flammarion, 2001.

[WDL+08] J. Wang, L. Duan, Q. Liu, H. Lu, et J.S. Jin. A multimodal scheme for program segmentation and representation in broadcast video streams. *IEEE Transactions on Multimedia*, 10(3) :393–408, avril 2008.

[WLH00] Y. Wang, Z. Liu, et J.-C. Huang. Multimedia content analysis-using both audio and visual clues. *IEEE Signal Processing Magazine*, 17(6) :12–36, novembre 2000.

[WLQ+09] X. Wang, X. Li, Y. Qian, Y. Yang, et S. Lin. *Opportunities and Challenges for Next-Generation Applied Intelligence*, volume 214, chapitre Break-Segment Detection and Recognition in Broadcasting Video/Audio Based on C/S Architecture, pages 45–51. Studies in Computational Intelligence, Springer Berlin / Heidelberg, mai 2009.

[WLY05] P. Wang, Z.-Q. Liu, et S.-Q. Yang. A probabilistic template-based approach to discovering repetitive patterns in broadcast videos. *Proceedings of the ACM international conference on Multimedia*, pages 407–410, Singapour, novembre 2005.

[Wol96] W. Wolf. Key frame selection by motion analysis. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1228–1231, Atlanta, Georgie, USA, mai 1996.

[YMWL08] J. Yuan, J. Meng, Y. Wu, et J. Luo. Mining recurring events through forest growing. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(11) :1597–1607, novembre 2008.

[YSS02] Y. Yuan, Q-B Song, et J-Y Shen. Automatic video classification using decision tree method. *Proceedings of the International conference on machine learning and cybernetics*, pages 1153–1157, Pékin, Chine, novembre 2002.

[YTX07] X.-F. Yang, Q. Tian, et P. Xue. Efficient short video repeat identification with application to news video structure analysis. *IEEE Transactions on Multimedia*, 9(3) :600–609, avril 2007.

[YYL96] M. Yeung, B.-L. Yeo, et B. Liu. Extracting story units from long programs for video browsing and navigation. *Proceedings of the IEEE International Conference on Multimedia Computing and Systems*, pages 296–305, Hiroshima, Japon, juin 1996.

[ZDK02] Wensheng Zhou, Son Dao, et C. C. Jay Kuo. On-line knowledge- and rule-based video classification system for video indexing and dissemination. *Information Systems Journal*, 27(8) :559–586, décembre 2002.

[ZJK01] T. Zhang et C.-C. Jay Kuo. Audio content analysis for online audiovisual data segmentation and classification. *IEEE Transactions on Speech and Audio Processing*, 9(4) :441–457, mai 2001.

[ZKS93] H. J. Zhang, A. Kankanhalli, et S. W. Smoliar. Automatic partitioning of full-motion video. *Multimedia Systems Journal*, 1(1) :10–28, juin 1993.

[ZLSW95] H. J. Zhang, C. Y. Low, S. W. Smoliar, et J. H. Wu. Video parsing, retrieval and browsing : an integrated and content-based solution. *Proceedings of the ACM international conference on Multimedia*, pages 15–24, San Francisco, Californie, USA, novembre 1995.

[ZRL96] T. Zhang, R. Ramakrishnan, et M. Livny. Birch : An efficient data clustering method for very large databases. *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 103–114, Montréal, Canada, juin 1996.

[ZZZY08] Z. Zeng, S. Zhang, H. Zheng, et W. Yang. Program segmentation in a television stream using acoustic cues. *Proceedings of the International Conference on Audio, Language and Image Processing*, pages 748–752, Shanghai, Chine, juillet 2008.

# Liste des figures

# Liste des tableaux

# Liste des algorithmes