THALES

Research & Technology

Rapport / Document Technique

RAPPORT FINAL
THALES R&T - OUTILEX

SUIVI DES ÉVOLUTIONS

	- le :21/09/06	A le : 07/11/06	B le :	C le :	D le :
Etabli par Signature	B. Goujon	B. Goujon			
Approuvé par Signature					

Indices de révision	Modifications
A	
B	
C	
D	

TABLE DES MATIÈRES

1. INTRODUCTION

Dans le cadre du projet RNTL Outilex, Thales Research & Technology (TRT) a développé un démonstrateur de filtrage audio s'appuyant sur la plate-forme Outilex. Ce document décrit ce démonstrateur, basé sur un composant d'apprentissage de pondération. La première partie du rapport présente le composant, qui permet de générer des graphes pondérés exploitables par la plate-forme Outilex. La deuxième partie du rapport présente le démonstrateur de filtrage thématique de documents audio, qui s'appuie sur les pondérations pour rechercher les éléments phonétiquement pertinents d'un thème. Pour chaque partie, nous décrivons notre approche, l'architecture de l'élément décrit et la procédure d'installation.

2. CONTEXTE DE DÉPART

Alors que l'extraction d'informations textuelles peut se faire en utilisant un outil tel que les moteurs de recherche sur Internet, l'extraction d'informations issues d'un flux audio est plus délicate, car cela nécessite l'utilisation d'un format de représentation des données audio intermédiaire. Dans la plupart des cas, le filtrage thématique, qui consiste à identifier des documents dont le contenu est en relation avec un thème particulier, s'appuie sur une transcription complète du flux audio en texte. L'une des limites de cette approche est que le taux d'erreurs de la transcription peut être élevé, surtout s'il y a des mots inconnus dans le flux audio.

Dans le cadre d'un précédent projet, Thales Research & Technology (TRT) a développé un outil de filtrage de données audio s'appuyant sur le système de reconnaissance de la parole Sphinx 3 du CMU (Carnegie Mellon University), et sur l'environnement de gestion des machines à états finis FSM. La particularité de l'approche suivie consiste à ne pas s'appuyer sur une transcription complète des données audio, mais à rester au niveau phonétique. L'intérêt est de faciliter la gestion des mots inconnus qui ne seront jamais reconnus par une approche s'appuyant sur une transcription en mots mais qui peuvent être repérés via une transcription phonétique.

Dans le contexte de la réalisation d'une plate-forme de traitement linguistique Outilex, nous avons voulu réaliser un nouveau démonstrateur de filtrage de données audio, s'appuyant sur un composant d'apprentissage de pondération de phonèmes et sur la plate-forme Outilex. Le remplacement des FSM par Outilex devrait entre autre nous permettre de faciliter la description d'un thème.

3. LE COMPOSANT

3.1. Présentation

L'objectif du composant est le suivant : à partir d'un thème regroupant des mots ou expressions spécifiques à un domaine, nous générons le graphe Outilex associé à ce thème, qui s'appuie sur des phonèmes et des pondérations. Ce graphe est ensuite appliqué par le démonstrateur pour le filtrage thématique de flux audio.

Chaque suite de phonèmes correspondant à un élément du thème initial (mot ou expression) est pondérée entre 0 et 1. L'apprentissage du poids va permettre d'évaluer la pertinence d'une suite de phonèmes pour caractériser un thème. Ainsi, si l'on s'intéresse aux événements actuels au Liban, il serait logique d'intégrer le

mot « Liban » au thème avec un poids fort, dans une approche « mot ». Or, la suite de phonèmes associée à ce mot est par exemple en français présente dans le mot « Taliban », qui n'a pas de relation directe avec le thème précédent. Partant de cette constatation, nous avons identifié plusieurs cas dans lesquels une suite de phonèmes pourrait être moins pertinente que le mot associé par rapport à un thème :

- une suite de phonèmes correspondant à un élément du thème peut être incluse dans un mot, comme précédemment « Liban » dans « Taliban » ;
- une suite de phonèmes peut être incluse dans une suite de mots, comme « attack » dans « a tactical » ;
- une suite de phonèmes peut correspondre à un homonyme (autre mot ayant un sens différent), comme « Bush » et « bouche » en français.

3.2. Pondération par apprentissage

3.2.1. Pondération à partir de corpus

Nous avons choisi d'effectuer l'apprentissage des pondérations associées aux suites de phonèmes en s'appuyant sur un corpus hors domaine. Ce type d'approche est utilisé pour attribuer par exemple des scores à des candidats termes, dans un objectif d'indexation de textes : dans un premier temps, des candidats termes sont extraits de corpus liés au domaine, et dans un deuxième temps, ces candidats termes sont appliqués sur des corpus hors domaine afin de voir si ils caractérisent ou non le domaine. Ainsi, dans (Patry A., Langlais, Ph 2005), plusieurs métriques sont retenues pour pondérer l'intérêt d'un terme par rapport à un domaine : la fréquence, la longueur, la « log-likelyhood », l'entropie, tf.idf et les probabilités de patron POS. Un premier ensemble de ces métriques permettent de vérifier la pertinence des termes extraits, basés sur plusieurs mots (s'ils peuvent être décomposés, si les mots peuvent être non adjacents ou s'ils sont toujours cooccurrents). Cela ne nous intéresse pas, puisque nous supposons dans notre approche que l'utilisateur connaît les termes (mots ou expressions) qui sont sémantiquement pertinents par rapport au domaine qui l'intéresse. Un deuxième ensemble de métriques permet de pondérer les termes en fonctions de leurs occurrences en corpus. Ainsi, tf.idf, qui est une mesure, proposée par (Salton G., 1989), permet d'attribuer un score d'importance à un terme (unité lexicale) dans un document (unité de contexte). Voici la formule correspondante :

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

Avec:

tf i,j = fréquence d'apparition de i (terme) dans j (document)
df i = nombre de documents du corpus contenant i
N = nombre de documents du corpus

Cette métrique pourrait être assez pertinente pour notre approche, mais présente deux limites. Tout d'abord, notre objectif est de minorer le poids d'un terme très fréquent dans un corpus hors domaine, ce qui constitue une démarche inverse à celle de Salton. De plus, cette mesure nécessite l'utilisation de gros corpus, or nous ne souhaitons pas que la tâche d'apprentissage de pondérations soit complexe à mettre en œuvre, en nécessitant des corpus trop volumineux. D'autres problématiques, détaillées ci-après, nous ont finalement amené à définir notre propre métrique pour l'acquisition des pondérations.

3.2.2. Problématiques

La pondération que nous avons développée a pour objectif de réduire le poids associé à un élément du thème quand la suite de phonèmes qui lui est associée a des occurrences dans des documents non liés au thème.

Cette approche pose plusieurs difficultés :

- Comment être sûr que le corpus utilisé pour apprendre les pondérations est hors thème ?
- Comment gérer les documents qui ont des liens avec des sous-thèmes (par exemple, en rapport avec la cuisine au Liban, quand le thème est « conflit entre le Liban et Israël ») ?
- Quel volume est nécessaire ? quelle taille pour chaque document ? combien de documents ?

Voici les choix que nous avons fait pour résoudre ces difficultés :

- Pour être sûr que le corpus utilisé est hors thème, il suffit de construire le corpus en utilisant soit des documents qui ont été produits à une date très différente (si l'on s'intéresse à la guerre du Golfe vers 1997-1998, on prend des textes actuels), soit des documents associés à des catégories différentes (si l'on s'intéresse à la politique, on prend des textes traitant de sport ou de cuisine).
- Pour ne pas considérer comme peu descriptif un élément du thème qui serait malgré tout dans le corpus (United States peut être dans de nombreux thèmes), nous avons choisi de prendre en compte la représentation en mots : un module va compter le nombre d'occurrences des éléments du thème sous la forme mot (United States par exemple) dans le corpus hors thème. Ainsi, si la suite de phonèmes associée à United States apparaît plusieurs fois dans le corpus hors thème, et si dans la version mot cet élément apparaît autant de fois, alors cette suite de phonèmes est bien associée à « United States ».
- Dans la version qui a développée dans le cadre du projet Outilex, nous avons choisi de ne pas fournir de corpus trop volumineux en entrée, qui serait difficile à construire. De même, nous n'avons pas géré le niveau document, mais uniquement le niveau corpus. Nous avons ainsi utilisé principalement trois corpus contenant entre 7 500 et 58 700 mots.

3.2.3. Formule

Voici la formule de pondération que nous avons utilisé :

$$\text{Poids} = \left(\frac{1}{s + 1 + d} + \frac{s}{s + 1} \right)^2$$

où d = différence entre le nombre d'occurrences de la suite de phonèmes et le nombre d'occurrences du mot dans le corpus,

s = seuil à définir pour obtenir une pondération cohérente : si s = 0, une occurrence (d = 1) fait chuter fortement le poids (poids = 0,25) , si s = 5, une occurrence (d = 1) réduit faiblement le poids (poids = 0,95).

Pour la réalisation du composant, en fonction des mots du thème et des corpus, nous avons fixé ce seuil à 3 (si d = 1, poids = 0,90, si d = 5, poids = 0, 74) .

3.3. Architecture du composant

Voici le schéma qui montre le déroulement des processus du composant :

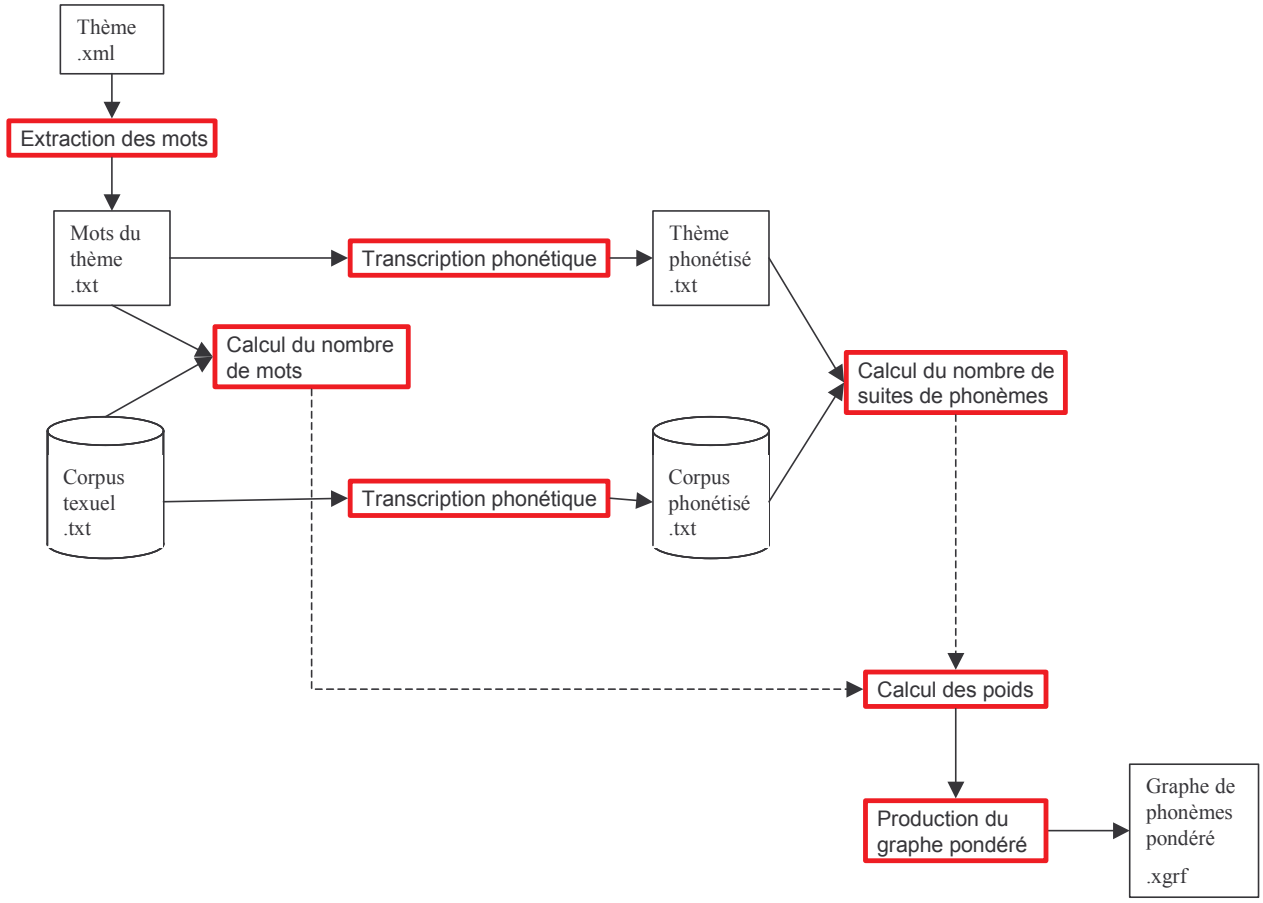

Figure 1 : Architecture fonctionnelle du composant.

3.3.1. Détails sur les entrées / sorties du composant

En entrée, le composant prend deux documents : le thème, au format xml, et le corpus au format .txt.

- **Thème** : le thème contient un ensemble de mots ou expressions, chaque élément étant dans une balise semclass. L'utilisation des minuscules ou majuscules a une importance. L'exemple suivant illustre le thème des attaques contre la coalition lors de la guerre du golfe en 1997.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <profil>
    <type>semantic</type>
  - <filter>
      <title>Attack_against_coalition</title>
      <semclass>Saddam Hussein</semclass>
      <semclass>Hussein</semclass>
      <semclass>Tarek Aziz</semclass>
      <semclass>Tariq Aziz</semclass>
      <semclass>Richard Butler</semclass>
      <semclass>Baghdad</semclass>
      <semclass>Iraq</semclass>
      <semclass>United Nations</semclass>
      <semclass>OPEC</semclass>
      <semclass>Inspector</semclass>
      <semclass>Attack</semclass>
      <semclass>chemical weapons</semclass>
      <semclass>mass destruction</semclass>
      <semclass>Weapons</semclass>
      <semclass>Strike</semclass>
      <semclass>Scud</semclass>
      <semclass>Bill Clinton</semclass>
      <semclass>Clinton</semclass>
      <semclass>New York</semclass>
      <semclass>Washington</semclass>
      <semclass>United States</semclass>
      <semclass>Gulf war</semclass>
      <semclass>war</semclass>
      <semclass>gas</semclass>
    </filter>
</profil>
```

Exemple de thème lié à la guerre du Golfe.

- **Corpus** : le corpus contient plusieurs textes regroupés dans un fichier au format .txt. Ces documents sont de préférence sans rapport avec des sous-thèmes du thème principal, et sont nécessairement dans la même langue que le thème (une phonétisation anglophone de textes français n'aurait aucun intérêt). Ils sont écrits avec des casses classiques (Unicode ANSI). Les corpus utilisés pour les tests ont été créés à partir de textes journalistiques en anglais traitant de tourisme, sport ou divertissement, et à partir de documents autres : une fiction et un ensemble de poèmes.
- **Graphe produit** : le graphe produit est au format .xgrf (Outilex). Il contient pour chaque élément du thème une boîte associée à un sous-graphe et un poids. Chaque sous-graphe, qui correspond à chaque élément du thème, contient la suite de phonèmes.

Voici un extrait du graphe produit.

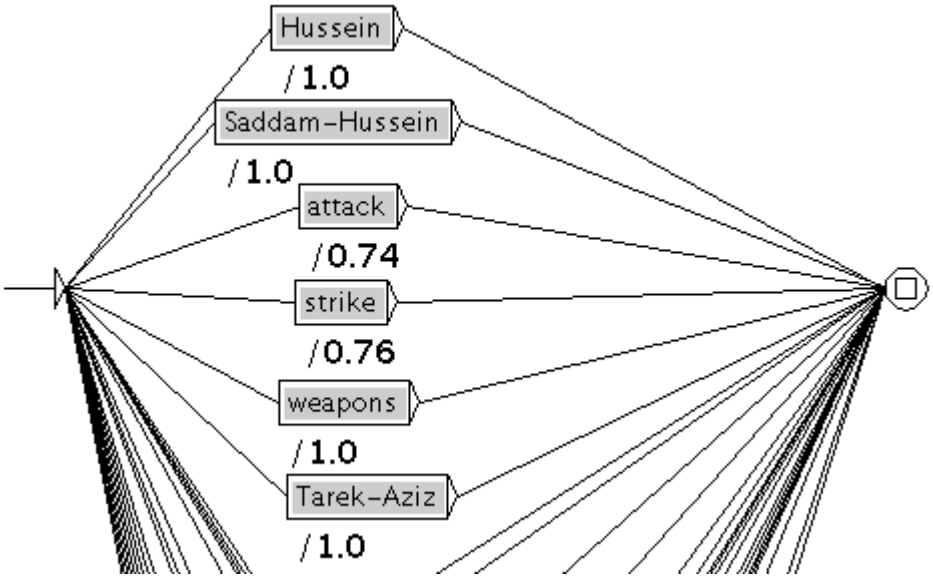

Figure 2 : Extrait du graphe correspondant au thème guerre du golfe

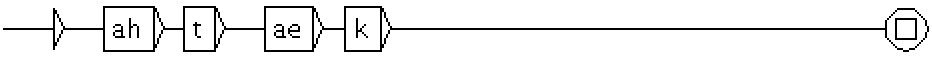

Figure 3 : Contenu du sous-graphe attack.

3.3.2. Détails sur les modules de l'architecture

- **Extraction des mots** : ce module prend en entrée un thème au format xml et fournit en sortie un fichier au format txt contenant les éléments du thème (un par ligne) hors balises xml.
- **Calcul du nombre de mots** : ce module prend en entrée le thème au format texte et le corpus, et fournit en sortie le nombre d'occurrences de chaque mot ou expression du thème dans le corpus. Cela permet de ne pas considérer comme moins important un mot (New York par exemple) qui pourrait être présent dans des textes sans rapport avec le thème (sport par exemple). Pour être comptabilisée, une occurrence ne doit pas être au cœur d'un autre mot (war n'est pas compté dans forward par exemple). La casse n'est pas prise en compte, afin de comptabiliser les occurrences de noms propre écrites entièrement en majuscules, ou avec uniquement les premières lettres en majuscules. Du coup, on risque de repérer des occurrences non pertinentes, comme U.S. qui peut être écrit US, et qui sous cette forme est équivalent à « us ».
- **Transcription phonétique** : ce module prend en entrée un fichier texte (corpus ou thème) et produit en sortie la suite de phonèmes associée. Plusieurs problèmes se présentent ici : la casse n'étant pas gérée pour la transformation en phonèmes, US se retrouve transcrit comme « us » (c'est pourquoi il n'est pas dans le thème). D'autre part, la transcription correspond à une transcription des mots hors contexte, ce qui n'a pas grand-chose à voir avec les transcriptions que l'on peut obtenir dans un flux audio où des mots sont prononcés différemment ou avec des syllabes en moins.

- **Calcul du nombre de suite de phonèmes** : ce module prend en entrée le thème sous sa forme phonétisée et le corpus phonétisé, et récupère en sortie pour chaque élément du thème le nombre d'occurrences de sa suite de phonèmes dans le corpus.
- **Calcul des poids** : ce module prend en entrée pour chaque élément du thème leur nombre d'occurrences en mots dans le corpus et son nombre d'occurrences en suite de phonèmes, et produit un poids entre 0 et 1 représentant l'intérêt de sa suite de phonèmes comme descripteur du thème.
- **Production du graphe pondéré** : ce module prend en entrée les éléments du thème, leurs représentations phonétiques et leurs poids respectifs, et produit en sortie le graphe phonétique pondéré au format Outilex (.xgrf) associé au thème de départ.

3.4. Implémentation du composant

Le composant s'appuie sur un ensemble de classes java organisées en package. Voici les principales classes :

- **main** : - ApprentissagePoids est la classe principale contenant le main qui appelle les différents traitements illustrés sur le schéma précédent ;

  InterfaceComposantOutilex contient l'interface graphique ;

- **utils** : - PerlCode permet de faire appel au code Perl externe ;

  LectureFichier permet de lire des fichiers pour récupérer des informations ;

  MotPondere représente un mot avec ses caractéristiques acquises au cours de l'analyse : suite de phonèmes associée, nombre d'occurrences de la forme mot dans le corpus, nombre d'occurrences de la suite de phonèmes dans le corpus phonétisé, poids associé.

- **graphs** : - Box permet l'écriture d'un élément box dans la structure xml du graphe généré ;

  Graph permet l'écriture d'un graphe au format .xgrf, format utilisé par la plate-forme Outilex.

L'interface permet les actions suivantes :

- Choix du corpus, qui permet de sélectionner un fichier au format txt.
- Choix du thème, qui permet de sélectionner un fichier au format .xml.
- Lancer l'apprentissage des pondérations, qui lance le programme principal.
- Fermer, qui permet de quitter l'application.

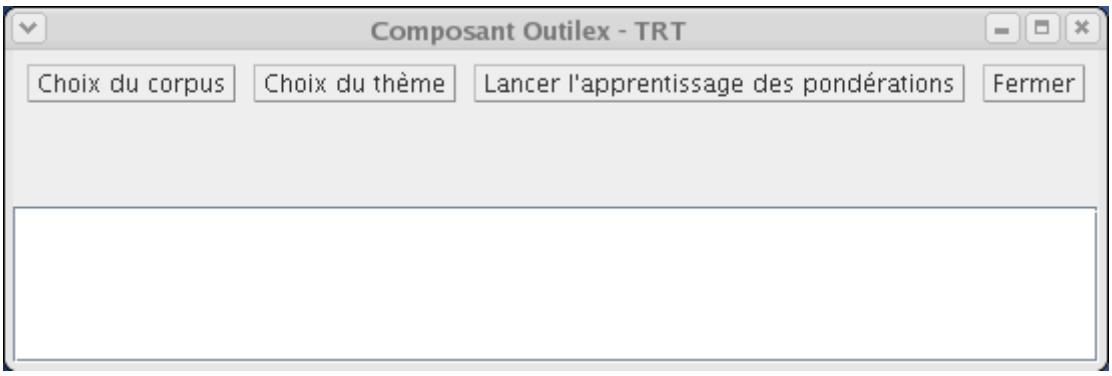

Figure 4 : Interface principale du composant.

3.5. Résultats visés et commentaires

3.5.1. Objectifs d'utilisation du thème pondéré

Le graphe phonétique pondéré obtenu peut être utilisé de plusieurs façons. La façon la plus simple consiste à associer un seuil au graphe phonétique du thème : après pondération, les mots ou suites de mots qui ont un poids inférieur au seuil sont considérés comme non caractéristiques du thème, et ne sont pas pris en compte dans l'étape de recherche dans les données audio.

La façon la plus complexe consiste à composer le résultat de la reconnaissance audio avec le graphe thématique, puis à appliquer un seuil. Lors de l'analyse d'un flux audio, le système de reconnaissance de la parole produit un treillis de phonèmes qui contient des poids correspondant aux probabilités d'avoir chaque suite de phonèmes. La composition du treillis avec le graphe du thème pondéré doit fournir les résultats suivants : le thème est reconnu si le produit entre le poids d'une suite de phonèmes du thème avec la probabilité d'avoir cette suite de phonèmes dans le treillis est supérieure à un seuil. Par exemple, si « Saddam Hussein » a un poids de 1.0 dans le thème et de 0.8 dans le treillis, la probabilité d'avoir identifié le thème est de 0.8. Si par contre, « war » a un poids de 0.67 dans le thème et de 0.69 dans le treillis, alors la probabilité d'avoir identifié le thème est de 0.46.

3.5.2. Premiers résultats de pondération

Nous présentons ici les résultats de la pondération que nous avons obtenus en utilisant trois corpus différents : un premier corpus d'articles de presse hors domaine de 7 500 mots, un deuxième corpus mixte (articles hors domaine et articles du domaine) de14 000 mots, et un troisième corpus étendu, hors domaine et contenant une fiction et des poèmes, de 58 700 mots.

Voici les poids associés aux plus mauvais descripteurs phonétiques, après apprentissage sur ces corpus

	Corpus hors domaine	Corpus mixte	Corpus étendu
attack	0.79	0.76	0.74
war	0.67	0.66	0.59
strike	1.0	1.0	0.76

gas	1.0	1.0	0.79

Voici une interprétation de ces résultats : tout d'abord, les noms propres de ce thème sont de bons descripteurs phonétiques puisqu'ils obtiennent tous des poids de 1. Les éléments qui sont phonétiquement les moins pertinents sont des noms communs courts (5 phonèmes maximum). Comme prévisible, plus le corpus est volumineux, plus le poids a tendance à diminuer, et plus on a la possibilité d'avoir des occurrences de nouveaux éléments (ici strike et gas ne sont phonétiquement présents que dans le corpus étendu).

L'utilisation d'un seuil à 0.7 à cette étape éliminerait uniquement « war » du thème, estimé non pertinent phonétiquement pour caractériser le thème. L'utilisation d'un seuil à 0.8, après apprentissage sur le corpus étendu, éliminerait « attack », « war », « strike » et « gas », considérés comme non pertinents phonétiquement pour caractériser le thème.

3.5.3. Quelques commentaires

Pondération phonétique vs sémantique : le graphe obtenu propose une pondération uniquement phonétique du thème. Or, il semblerait judicieux de coupler cette pondération avec une organisation sémantique des différents éléments du thème. En effet, un thème regroupe des mots ou expressions qui peuvent être regroupés en sous-thèmes : ainsi, le thème lié à la guerre du Golfe contient les sous-thèmes « Etats-Unis », « Iraq » et « conflit ». La combinaison de ces sous-thèmes pourrait par exemple permettre de garder comme descripteurs certains éléments en combinaison avec d'autres si le poids total devient supérieur au seuil défini.

Relations phonétiques entre deux mots : dans notre approche, nous avons souhaité réduire l'intérêt d'un mot décrivant un thème si sa représentation phonétique ne caractérise pas assez le thème. Ainsi, dans l'exemple du thème lié à la guerre du Golfe, le mot « war » se trouve pondéré à 0.67, car la suite de phonèmes qui lui est associée a été retrouvée dans d'autres contextes : « quartet » par exemple. Mais cela n'est pas toujours aussi simple. Ainsi, alors que « award » ou « dwarf » ne sont pas sémantiquement liés à « war », « warrior » ou « wars » sont deux mots qui contiennent la suite de phonèmes associée à « war », tout en étant sémantiquement en relation avec ce mot. Ainsi, un thème devrait contenir l'ensemble des formes dérivées de chaque élément, surtout si ces formes sont phonétiquement liées.

Corpus : le rôle du corpus sur lequel se fait l'apprentissage des poids est très important. Une première solution consistait à utiliser un dictionnaire qui, une fois phonétisé, peut permettre d'identifier le nombre de mots contenant une suite de phonèmes. Mais les dictionnaires contiennent de nombreux mots peu usités, qui ne permettent pas d'avoir une pondération fiable par rapport à l'usage. D'autre part, cette approche ne permet pas de repérer d'éventuelles suites de phonèmes qui se retrouvent dans des suites de mots (exemple « attack » contenu dans « a tactical »). Nous avons finalement opté pour l'utilisation d'un corpus pour apprendre les poids. L'utilisation de corpus phonétisés permet de repérer des suites de phonèmes qui se trouvent incluses dans des mots, ou dans des suites de mots, ou des suites de phonèmes correspondant à des homonymes.

3.6. Processus d'installation du composant

Contenu du livrable : Le livrable contient un répertoire nommé « outilex_trt_component » qui contient l'ensemble du code java (sources et classes) ainsi qu'un fichier README. Il contient aussi un ensemble de scripts Perl dans un sous-répertoire codePerl. Pour les tests, deux corpus de textes et un thème sont fournis.

Installation : Pour installer le composant, il suffit de copier le répertoire « outilex__trt_component » dans un environnement Unix. Le programme se lance de la façon suivante, à partir du répertoire « outilex_trt_component » : `java main.ApprentissagePoids ./`

Paramétrage de l'environnement : Avant de lancer le programme, il est important d'installer et configurer Perl et Java si besoin, et de mettre à jour la variable PATH en suivant l'exemple :

```
setenv PATH /usr/java/jre1.5.0_06/bin:${PATH}
```

Test : Voici le scénario à suivre pour tester le composant : après avoir lancer le programme :

- faire "choix du corpus" puis choisir dans le répertoire /textualData l'un des corpus et valider ;
- faire "choix du thème" puis dans le répertoire /theme choisir le fichier correspondant au thème US-Iraq-conflict.xml et valider ;
- faire "lancer l'apprentissage des pondérations".

Le fichier contenant le graphe phonétique pondéré obtenu se trouve dans le répertoire /theme : US-Iraq-conflict-Words.xgrf. Ce fichier peut être visualisé en utilisant la plate-forme Outilex.

4. LE DEMONSTRATEUR

4.1. Présentation

L'objectif du démonstrateur présenté ici est d'effectuer un filtrage thématique de documents audio. Ce filtrage s'effectue au niveau de la transcription en phonèmes du thème (initialement un ensemble de mots ou expressions) et des données audio. Le thème est transformé en graphe de phonèmes pondéré au format Outilex, afin de valoriser les suite de phonèmes caractérisant un thème et dévaloriser les suites de phonèmes pouvant être présentes dans d'autres contextes non pertinents.

Le deuxième objectif lié à la réalisation de ce démonstrateur est de tester l'intérêt de la plate-forme Outilex, initialement inspirée de la plate-forme Unitex qui est dédiée à l'analyse de textes, pour traiter des documents de type audio, comme le fait la boîte à outils FSM.

Ce démonstrateur s'appuie sur les éléments suivants :

- fichiers audio
- thème XML
- interface
- composant d'apprentissage des pondérations

4.2. Utilisation d'Outilex à la place des FSM

4.2.1. Utilisation précédente des FSM

Les FSM (Finite State Machine Library), développés par AT&T, regroupent des outils et sources pour manipuler des automates pondérés à états finis. Nous présentons ici l'utilisation qui a été faite des FSM lors de la réalisation du précédent démonstrateur de filtrage thématique de documents audio.

La librairie FSM a été utilisée pour générer des automates pondérés à la volée à partir de grammaires formelles représentatives d'un thème. Pour information, dans la représentation graphique des automates, les poids et les étiquettes sont attribués aux arcs de transition, et non aux états.

Le flux audio est transformé en un treillis de phonèmes grâce à un moteur de reconnaissance de la parole (Sphinx). Ce treillis est ensuite transformé en automate pondéré à états finis.

L'étape principale du démonstrateur basé sur les FSM est la composition de l'automate pondéré du thème avec l'automate pondéré du flux audio, qui fournit comme résultat l'intersection la plus probable entre le thème et le flux audio.

4.2.2. Comparaison FSM et Outilex

La plate-forme Outilex qui a été réalisé est adaptée pour le traitement des documents textuels : l'interface permet de sélectionner des grammaires (ou automates), pour les appliquer à des textes. Il n'est par contre pas prévu d'effectuer la composition d'automates (qui consisterait à avoir deux automates en entrée, plutôt qu'un automate et un texte). Cette différence nous a amené à ajouter dans notre démonstrateur Outilex une étape de transformation d'un treillis en texte : au lieu d'avoir plusieurs chemins dans le treillis, nous recopions les chemins (suites de phonèmes) les plus probables (n-best) dans un fichier texte. Au final, on applique donc l'automate du thème sur les n-best au format texte.

4.2.3. La pondération dans la plate-forme Outilex

La plate-forme Outilex permet la pondération des grammaires. Le poids d'un chemin dans un graphe est la somme de tous ses poids. Par défaut, si le poids n'est pas mentionné, sa valeur est 0.

Voici l'exemple fourni par la documentation :

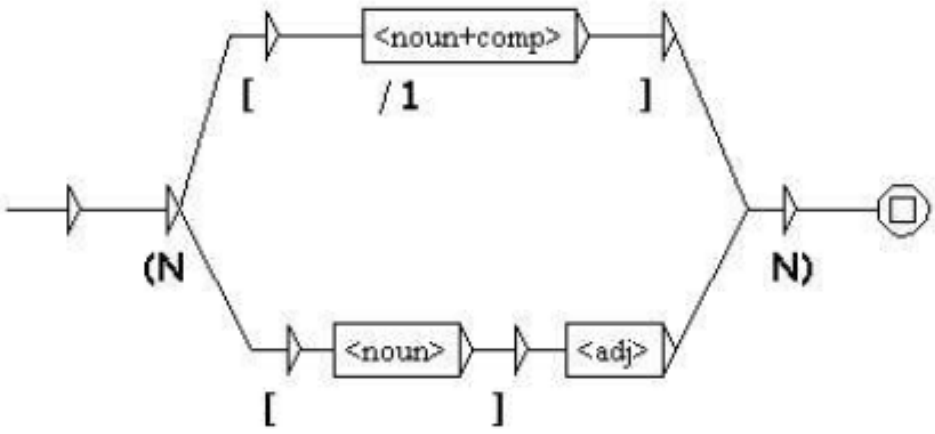

Figure 5 : Graphe pondéré avec Outilex.

Ce graphe permet de repérer en priorité les mots composés par rapport aux mots suivis d'adjectifs, lorsque les deux chemins sont valides, donc de ponctuellement favoriser le plus court chemin.

La pondération n'est pas visible en sortie, et ne peut être récupérée après application du graphe.

4.3. Architecture du démonstrateur

Le démonstrateur s'appuie sur le système de reconnaissance de la parole Sphinx 3, développé à Carnegie Mellon University (CMU).Il s'appuie aussi sur la plate-forme Outilex, et sur le composant d'apprentissage des pondérations décrit précédemment. Le démonstrateur est codé en java, avec des appels de scripts shell et de scripts Perl.

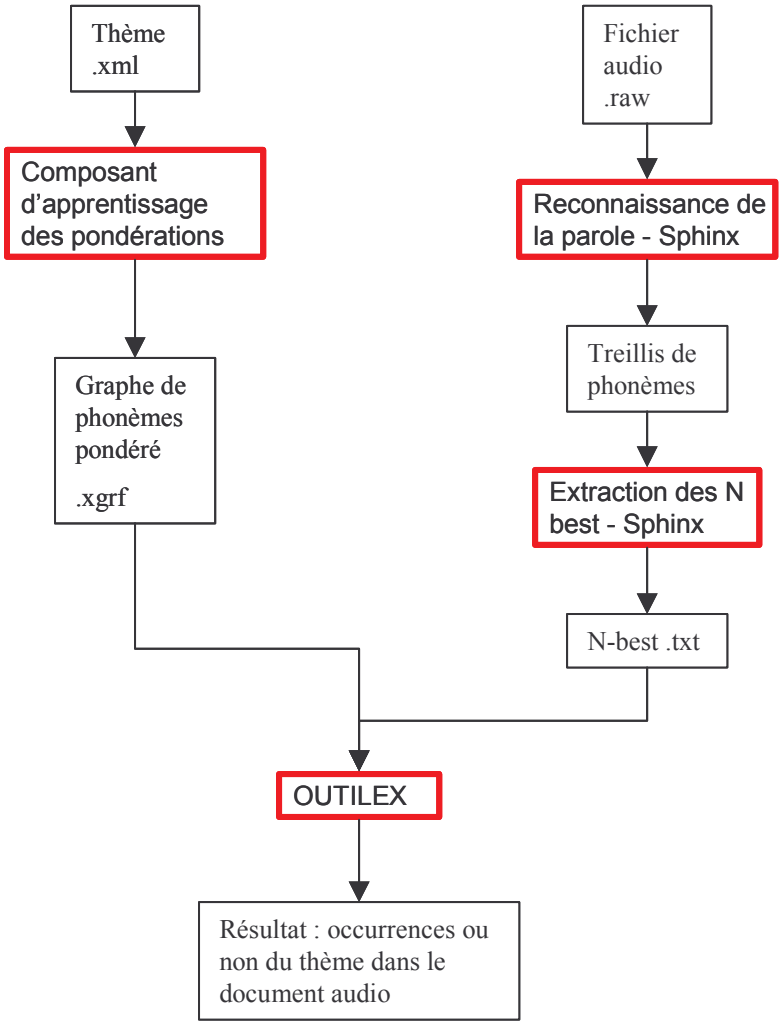

Figure 6 : Architecture fonctionnelle du démonstrateur.

4.3.1. Détails sur les entrées / sorties du démonstrateur

En entrée, le démonstrateur prend deux documents : un thème au format .xml et un fichier audio au format .raw.

- **Thème** : le thème contient un ensemble de mots ou expressions en anglais (pour plus de détails voir la partie décrivant le composant).

- **Fichier audio** : le fichier audio doit contenir de la parole en anglais. Il est de préférence de taille réduite, afin de permettre une analyse rapide. Pour le démonstrateur, nous avons des fichiers entre 240 Ko et 700 Ko. Le démonstrateur offre la possibilité de sélectionner plusieurs fichiers audio en même temps.
- **Résultat** : le résultat est affiché dans la fenêtre du démonstrateur. Si le document audio contient un élément du thème, le message suivant est affiché : « La/les chaine(s) reconnues sont les suivantes : (*suite de phonèmes reconnue*) ». Si le document audio ne contient pas d'élément du thème, le message suivant est affiché : « Le fichier *nomdufichier*.raw ne contient pas le thème recherché. ».

4.3.2. Détails sur les étapes du démonstrateur

- **Composant d'apprentissage des pondérations** : ce composant est décrit précédemment dans ce document.
- **Reconnaissance de la parole – Sphinx** : cette étape s'appuie sur le programme de reconnaissance de la parole Sphinx3_livepretend. Ce programme nécessite en entrée un dictionnaire de phonèmes, un dictionnaire de *filler* (pour marquer les mot inconnu, silences, …), et un modèle de langage phonétique (les mots sont remplacés par des phonèmes). Il utilise aussi un ensemble de paramètres de configuration acoustique (argLivepretend), qui permettent de spécifier l'emplacement des modèles acoustiques de Sphinx3 ainsi que de nombreux arguments comme l'échantillonnage, le nombre de filtres Mel, la normalisation cepstrale, les poids du modèle de langage, … Ce programme produit un treillis de phonèmes (dans generic.nbest.gz ou dans generic.lat en non compressé).
- **Extraction des Nbest – Sphinx** : cette étape s'appuie sur le programme Sphinx3_astar, algorithme qui parse le treillis de phonèmes pour fournir les n phrases (suites de phonèmes) les plus probables, n étant ajustable. Ce programme utilise les mêmes ressources que le programme précédent : dictionnaire de phonèmes, dictionnaire de filler, modèle de langage phonétique et paramètres de configuration acoustique (argAstar). Le résultat (generic.nbest) contient l'ensemble des phrases parsées par le treillis. Le nombre de phrase (correspondant au N) est l'un des paramètres de configuration, défini dans argAstar.
- **Plate-forme Outilex** : la plate-forme Outilex permet d'appliquer une grammaire à un texte. Dans notre démonstrateur, le thème pertinent est transformé en graphe de phonèmes pondéré (= grammaire), et le document audio est transformé en suite de Nbest (= texte). Le résultat est soit vide, si aucun élément du thème n'est présent dans le document audio, soit égal à la suite de phonèmes correspondant à l'élément du thème repéré dans le document audio.

4.4. Implémentation du démonstrateur

Le démonstrateur s'appuie sur une classe principale qui appelle plusieurs scripts shell et le code du composant. Voici l'organisation du code :

- **InterfaceOutilex** est la classe java principale, qui permet de lancer l'interface pour le démonstrateur.
- **Composant_outilex** : Les principales classes du composant sont décrites précédemment.
- **ExecAll.sh** appelle les deux scripts présentés ci-dessous.

- **ExecNbest.sh** lance la reconnaissance de la parole (Sphinx3 livepretend) sur le fichier audio analysé, ainsi que la transformation du treillis obtenu en nbest (Sphinx3 astar).
- **ExecOutilex.sh** lance l'application de la grammaire du thème (.xgrf) sur le texte contenant les nbest de la reconnaissance audio appliquée au fichier audio.

L'interface permet d'effectuer les actions suivantes :

- Apprentissage des pondérations, qui lance l'interface du composant d'apprentissage des pondérations. Lorsque l'apprentissage est fini, le bouton « Fermer » permet de revenir à l'interface du démonstrateur.
- Choix des fichiers audio, qui permet de sélectionner un ou plusieurs fichier(s) audio au format .raw.
- Choix du thème, qui permet de sélectionner le graphe obtenu par l'utilisation du composant d'apprentissage des pondérations (format .xgrf).
- Lancer l'analyse, qui permet d'effectuer les traitements du fichier audio.
- Quitter, pour quitter le démonstrateur.

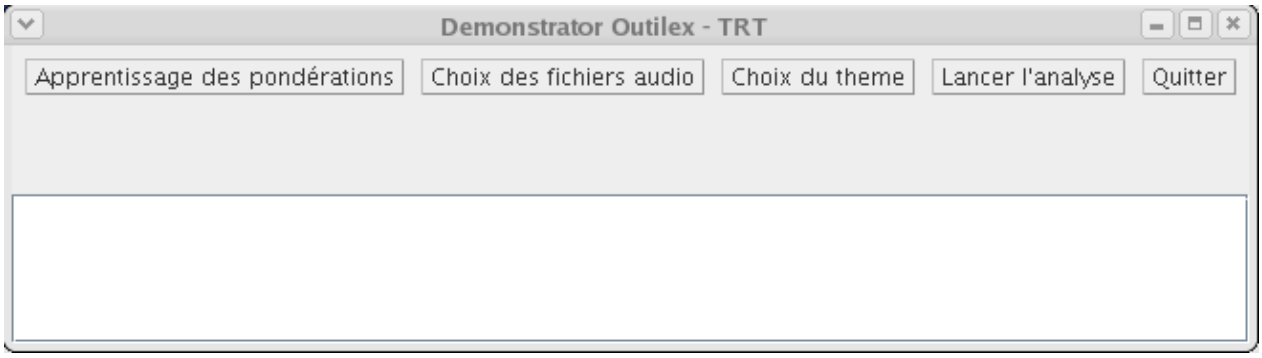

Figure 7 : Interface du démonstrateur Outilex – TRT.

4.5. Résultats visés et commentaires

- **Comparaison FSM-Outilex** : les premiers résultats nous ont montré que la vitesse de traitement du filtrage thématique basé sur Outilex était supérieure à celle de notre démonstrateur basé sur les FSM, ce qui est très intéressant. Cela s'explique aussi par une approche différente pour appliquer le thème à la sortie du système de reconnaissance de la parole. De plus, la description du thème avec le composant d'apprentissage est beaucoup plus simple à effectuer que lors de l'utilisation des FSM, qui nécessitent d'écrire dans un formalisme pointu les éléments du thème.
- **Application des automates pondérés** : le démonstrateur permet l'application d'automates pondérés sur des textes (suites de phonèmes) via la plate-forme Outilex pour extraire les séquences reconnues. Notre objectif concernant l'application des automates pondérés consiste à utiliser un seuil afin de n'appliquer que les chemins dont le poids est supérieur au seuil. Nous avons utilisé avec succès cette spécificité, qui nous permet de ne rechercher par exemple que les suites de phonèmes les plus pertinentes (seuil à 0,7 ou 0,8[1]) par rapport au thème.
- **Transcription de mots et de flux audio** : dans notre démonstrateur, nous utilisons une approche qui consiste à transcrire dans un premier temps des mots d'un thème, pour ensuite rechercher ces

[1] Voir le paragraphe décrivant les résultats obtenus avec le composant.

transcriptions dans des transcriptions de flux audio. Or, les prononciations des mots sont fréquemment variables à l'oral (phonème différent de l'« officiel », phonème absent, « avalé » dans le discours). Par exemple, on recherche Saddam Hussein, qui est associé à la suite de phonèmes « s aa d ah m hh uw s ey n ». Dans l'un des documents audio, où il est question de Saddam Hussein, voici ce que l'on trouve comme suite de phonèmes : « s ae m s eh ey n ». De même Baghdad, phonétisé en « b ae g d ae d », se trouve sous la forme « b aw g d ae t » dans le flux audio (suivi de « to »). Au final, dans un fichier audio contenant Baghdad, Saddam Hussein, United States, strike, seule la suite de phonèmes associée à strike, « s t r ay k » est reconnue. Il serait donc souhaitable d'utiliser un module de phonétisation adapté aux spécificités des flux audio (variation des accents, vitesses), qui puisse générer différentes suites de phonèmes pouvant être attribuées à un élément du thème.

- **Évaluation** : l'un des premiers objectifs de notre travail était de comparer l'intérêt de la plate-forme Outilex par rapport aux FSM dans le cadre du filtrage thématique de données audio. Comme cela a été précisé ci-dessus, nous n'avons pas pu utilisé la plate-forme Outilex dans les mêmes conditions que les FSM, pour notamment composer le graphe du thème avec le treillis de phonèmes issu de la reconnaissance de la parole. Avec la plate-forme Outilex, cette composition est remplacée par l'application d'un graphe thématique sur un texte contenant les suites de phonèmes les plus probables (Nbest). Ces deux approches ne sont pas directement comparables, car avec le treillis on peut repérer une suite de phonèmes se trouvant dans un chemin qui n'est pas parmi les plus probables, ce qui n'est pas possible en se restreignant aux Nbest. D'un autre côté, la restriction aux Nbest permet un gain de temps de traitement important.

4.6. Installation et utilisation du démonstrateur

4.6.1. Procédure d'installation du démonstrateur

Contenu du livrable : Le livrable contient un répertoire nommé « outilex_trt_demonstrator » où se trouve l'ensemble du code (java, perl, scripts shell, Sphinx3) et des données : cinq fichiers audio, un thème au format xml, un dictionnaire de phonèmes. Un sous-répertoire nommé « outilex_trt_component » contient le code associé au composant d'apprentissage de pondération.

Installation : Pour installer le démonstrateur, il suffit de recopier le répertoire du livrable dans un environnement Unix. Le démonstrateur se lance en utilisant le script suivant, à partir du répertoire outilex_trt_demonstrator : `./jl.sh`.

Paramétrage de l'environnement : Avant de lancer le programme, il est important d'installer et configurer Perl et Java si besoin. Le fichier env.sh contient l'ensemble des paramètres (Sphinx et Outilex) à charger. Il suffit de faire « source env.sh » pour charger ces paramètres dans l'environnement.

4.6.2. Scénario de démonstration

Voici le scénario de démonstration :

- Après avoir lancé le programme, cliquer sur le bouton « Apprentissage des pondérations ».
- Dans la fenêtre du premier plan, faire « Choix du corpus » : sélectionner ./textualData/corpusHorsDomaine.txt
- Sélectionner « Choix du thème » et sélectionner ./theme/US-Iraq-conflict.xml.
- Faire « Lancer l'apprentissage des pondérations »

- Fermer la fenêtre.
- Dans la fenêtre principale, faire « Choix des fichiers audio » : sélectionner audioFile04.raw et audioFile06.raw (avec la touche Ctrl).
- Faire « Choix du thème » : sélectionner US-Iraq-conflict-Words.xgrf.
- Sélectionner « Lancer l'analyse ».
- Le résultat suivant apparaît :

Le fichier audioFile04.raw ne contient pas le theme recherché.
Le fichier audioFile06.raw contient le theme recherché. La/les chaine(s) reconnues sont les suivantes : strike

5. CONCLUSION

Thales R&T a réalisé un démonstrateur de filtrage thématique sur des documents audio en s'appuyant sur la plate-forme Outilex. L'utilisation des pondérations gérées par Outilex permet de ne rechercher que les éléments d'un thème dont les suites de phonèmes sont pertinentes par rapport à ce thème. Comparativement au précédent démonstrateur basé sur les FSM, le démonstrateur Outilex est plus rapide, et la description du thème est facilitée. Cependant, comme les deux approches sont différentes, et puisque nous ne disposons pas encore de suffisamment de données audio relatives à un thème précis, il nous est actuellement difficile de proposer une comparaison objective des deux approches.

6. RÉFÉRENCES

Pour plus d'informations sur l'apprentissage de pondérations à partir de corpus :

Alexandre Patry and Philippe Langlais (2005) *"Corpus-Based Terminology Extraction"*, in Proceedings of the 7th International Conference on Terminology and Knowledge Engineering, Copenhagen, Danemark, August 17-18, 2005 , pp. 313—321

Gerard Salton. *Automatic Text Processing*. Addison-Wesley, 1989.

Pour plus d'informations sur les FSM (publications) : http://www.research.att.com/~fsmtools/fsm/ref.html

Pour plus d'informations sur Sphinx : http://cmusphinx.sourceforge.net/html/cmusphinx.php

Pour plus d'informations sur les méthodes classiques de filtrage thématique de documents audio :

Junkawitsch, J., Neubauer, L., Höge, H. et Ruske, G. (1996), A New Keyword Spotting Algorithm with Pre-Calculated Optimal Thresholds. ICSLP, Philadelphia.

Foote, J. (1999), An overview of audio information retrieval. Multimedia Systems 7 (1): pp. 2-10.

Mekhaldi, D., Lalanne, D. et Ingold, R. (2004), Using bi-modal alignment and clustering techniques for documents and speech thematic segmentations. Proceedings of the thirteenth ACM international conference on Information and knowledge management, pp. 69-77.

Nallapati, R. (2003), Semantic language models for topic detection and tracking. Proceedings of the HLT-NAACL 2003 student research workshop, pp. 1-6.