



ÉCOLE POLYTECHNIQUE DE L'UNIVERSITÉ FRANÇOIS RABELAIS DE TOURS

Spécialité Informatique

64, Avenue Jean Portalis

37200 TOURS, FRANCE

Tél. +33 (0)2 47 36 14 14

[www.polytech.univ-tours.fr](http://www.polytech.univ-tours.fr)

## Rapport de Projet de fin d'études 2014

# Clustering interactif d'éléments de contenu(extraits de documents)

### Encadrant

Jean-Yves Ramel

[jean-yves.ramel@univ-tours.fr](mailto:jean-yves.ramel@univ-tours.fr)

### Étudiant

Jing CHEN

[jing.chen-2@etu.univ-tours.fr](mailto:jing.chen-2@etu.univ-tours.fr)

Université François-Rabelais, Tours

DI5 2014 - 2015

Version du 9 mai 2015



# Table des matières

---

<b>1</b>	<b>Remerciements</b>	<b>6</b>
<b>2</b>	<b>Introduction</b>	<b>7</b>
<b>3</b>	<b>Contexte et objectifs du projet</b>	<b>8</b>
3.1	Contexte . . . . .	8
3.1.1	Contexte du sujet . . . . .	8
3.1.2	Contexte du projet . . . . .	8
3.2	Objectifs du projet . . . . .	9
3.3	Bases méthodologiques . . . . .	9
<b>4</b>	<b>Description générale</b>	<b>10</b>
4.1	Environnement du projet . . . . .	10
4.2	Étude des existants . . . . .	10
4.2.1	Retro2014 . . . . .	11
4.2.2	Jeu de données . . . . .	11
4.3	Fonctionnalités et structure générale du système . . . . .	12
4.4	L'architecture générale du système . . . . .	13
4.4.1	Plugin . . . . .	13
4.4.2	RetroCore . . . . .	13
4.4.3	RetroGUI . . . . .	13
4.4.4	RetroUtil . . . . .	14
4.4.5	Diagramme du package . . . . .	14
4.5	Contraintes de développement et d'exploitation . . . . .	15
4.5.1	Cycle de Développement & Design Pattern . . . . .	15
4.5.2	Patron de conception(Design Pattern) . . . . .	15
<b>5</b>	<b>Fouille de données visuelles</b>	<b>17</b>
5.1	Rappel : Analyse en Composantes Principales (ACP) . . . . .	17
5.1.1	Objectifs . . . . .	17
5.1.2	Les étapes d'une ACP . . . . .	18
5.2	Apprentissage automatique . . . . .	19
5.2.1	Systèmes OCR . . . . .	19
5.3	K-Nearest Neighborhoods(KNNs) . . . . .	20
5.4	L'algorithme du KNNs . . . . .	20
5.5	Caractères de l'algorithme KNNs . . . . .	21
5.6	Neuron Network - Deep learning . . . . .	21
5.6.1	Apprentissage en profondeur(Deep learning) . . . . .	21
5.6.2	Modèle probabilistes pour les architectures profondes . . . . .	22
5.7	Forces et Faiblesses de DBNs . . . . .	25



<b>6</b>	<b>Design et Proposition</b>	<b>26</b>
6.1	Proposition de fonctionnalités . . . . .	26
6.1.1	Optimisation visuelle de clusters . . . . .	26
6.1.2	Reconnaissance de clusters . . . . .	27
6.2	Conditions de fonctionnement . . . . .	29
6.3	Cahier de spécification . . . . .	29
<b>7</b>	<b>Travail réalisé- Développement</b>	<b>30</b>
7.1	Mise à jour de RETRO . . . . .	30
7.2	Ajout du module "Analyse Cluster" . . . . .	30
7.2.1	IHM du "Analyse Cluster" . . . . .	31
7.2.2	Ajout de fonctionnalités . . . . .	31
7.2.3	Libraires et classes correspondants . . . . .	33
7.3	Ajout du module "Auto Transcription" . . . . .	33
7.3.1	IHM du "Auto Transcription" . . . . .	33
7.3.2	Ajout de fonctionnalités . . . . .	34
7.3.3	Appliquer la méthode de transcription . . . . .	34
7.3.4	Libraires et classes correspondant . . . . .	35
7.4	Rendus . . . . .	35
<b>8</b>	<b>Gestion de projet</b>	<b>37</b>
8.1	Suivi du projet . . . . .	37
8.2	Planning . . . . .	37
<b>9</b>	<b>Conclusion</b>	<b>39</b>
	<b>Bibliographie</b>	<b>40</b>

# Table des figures

---

3.1	Logo de CESR, RFAI, Awards Google . . . . .	8
4.1	Processus d'extraire caractères . . . . .	10
4.2	Dossiers jeux de données pour Retro2014 . . . . .	11
4.3	Activités Diagramme . . . . .	12
4.4	Architecture du Retro2014 . . . . .	12
4.5	Packages Diagramme . . . . .	14
4.6	Agile cycle . . . . .	15
4.7	MVVM Modèle . . . . .	16
5.1	(a)le nuage des points variables représenté dans l'espace de dim $n$ défini par les individus ;(b)le nuage des points individus représenté dans l'espace de dim $p$ défini par les variables	17
5.2	RBM structure . . . . .	23
5.3	Pré-entraînement par RBM . . . . .	24
5.4	Fine-Tuning par arrière-propagation . . . . .	25
6.1	Maquette d'analyse Cluster . . . . .	26
6.2	Maquette d'Auto Transcription . . . . .	27
6.3	KNNs . . . . .	28
6.4	Design le processus de DBN . . . . .	28
6.5	DBNs design . . . . .	29
7.1	Panel d'Analyse Cluster . . . . .	30
7.2	Main menu . . . . .	31
7.3	Menu flottant . . . . .	31
7.4	Analyse Information du Cluster . . . . .	32
7.5	Nettoyage le panel de l'Analyse Cluster . . . . .	32
7.6	Panel d'Auto Transcription . . . . .	33
7.7	Jeu de modèle pour l'apprentissage . . . . .	34
7.8	Le répertoire du <i>Plugins</i> . . . . .	35
7.9	Le répertoire du <i>XML_Files</i> . . . . .	36
7.10	Le répertoire exécutable . . . . .	36
8.1	Diagramme de Gant prévisionnel . . . . .	37
8.2	Gant réel . . . . .	38

# Remerciements

---

Je tiens à remercier, particulièrement M. Jean-Yves Ramel, mon encadrent, pour son accompagnement tout au long du projet, pour l'aide qu'il m'apporte pendant le déroulement du projet pour le cahier de spécification, ainsi que pour ses nombreuses suggestions et explications et des aides à rédiger des documents. Il a été patient et a su répondre à mes interrogations, notamment sur la structure et des composants de Retro2014.

Je veux remercier aussi mes collègues pour le support et des conseils pendant ce projet.

# Introduction

---

Dans le cadre de notre dernière année en école d'ingénieur en informatique à Polytech'Tours, il nous est demandé de réaliser un Projet de Fin d'Études(PFE). Pour ma part, le sujet sur lequel porte mon PFE est le suivant : "Clustering interactif d'éléments de contenu(extraits de documents)".

Ce projet fait partie du projet PARADIIT(Pattern Redundancy Analysis for Document Image Indexation & Transcription) mis en place par l'équipe des Bibliothèques Virtuelles Humanistes(BVH) du Centre d'Études Supérieures de la Renaissance(CESR) et l'équipe Reconnaissance des Formes et Analyse d'Images(RFAI) du Laboratoire Informatique(LI) de l'université François Rabelais de Tours. PARADIIT a pour objectif la transcription de livres anciens en bibliothèques numériques.

L'objectif du présent PFE est d'améliorer un des "sous-projets" du projet PARADIIT. En effet, le but est l'amélioration la qualité de Clustering et la réalisation de transcription automatique.

Ce document a pour but de décrire le travail effectué pour la réalisation de ce PFE. Afin de vous détailler ce qui a été fait, ce rapport est divisé en 6 parties.

La première est la seconde partie vous permettront de comprendre le contexte, les objectifs du projet ainsi que l'environnement et architecture d'application. La troisième partie explique des théories et méthodes appliquées dans ce projet, Les quatrième et cinquième parties détailleront le travail réalisé grâce à l'explication des différentes tâches du projet. Enfin, la sixième et dernière portera sur le planning.

# Contexte et objectifs du projet

---

## 3.1 Contexte

### 3.1.1 Contexte du sujet

Tous document est généralement constitué de motifs ou éléments de contenu répétitifs qu'il peut être intéressant d'exploiter pour mettre en place des mécanismes d'indexation et de recherche permettant de retrouver rapidement une informations spécifiques dans de grandes masses de documents (CF moteur de recherche GOOGLE). Ce type de mécanismes est assez simple à mettre en place lorsque les documents sont sous forme textuelle (Pages WEB, ASCII) mais devient plus complexe lorsque l'on désire travailler sur des versions numérisées (images).

### 3.1.2 Contexte du projet

**PAEADIIT** Ce projet s'inscrit dans le cadre du projet **PARADIIT** (Pattern Redundancy Analysis for Document Image Indexation & Transcription) dont les membres sont :

- L'équipe des Bibliothèques Virtuelles Humanistes (BVH) du Centre d' Études Supérieures de la Renaissance (CESR).
- L'équipe Reconnaissance des Formes et Analyse d'Images (RFAI) du Laboratoire Informatique (LI) de l'université François Rabelais de Tours.

Le projet **PARADIIT** est une solution de transcription de documents anciens, formée de plusieurs applications développées par l'équipe RFAI. Il permet de convertir des livres anciens en ressources numériques accessibles à tous. Il permet de traiter des images (livres ou documents scannés) comme étant des ensembles de caractères et de trouver la correspondance ASCII de ces caractères.

Les documents à numériser font partie de la bibliothèque du CESR et date set de la Renaissance jusqu'au début du XVIIe siècle. De ce fait, ils présentent des particularités telles que la présence de bruit dans l'image ou encore la présence de caractères anciens propres à l'époque de la Renaissance.

Depuis sa création, en 2004, PARADIIT a été sponsorisé par deux Google Digital Humanities Awards.



FIGURE 3.1 – Logo de CESR, RFAI, Awards Google



## 3.2 Objectifs du projet

L'objectif essentiel du projet est l'amélioration la qualité de Clustering et la réalisation de transcription automatique. Les principales tâches qui devront être effectuées durant ce PFE concernent :

1. Étude de l'existant (architecture, dll et plugins existants)
2. Étude des méthodes de Clustering et fouille visuelle de données utilisables pour comparer et caractériser les éléments sélectionnés
3. Conception et développement des IHM et méthodes interactives permettant la visualisation et l'optimisation des clusters d'éléments de contenu
4. Conception et développement d'une méthode de reconnaissance automatiquement de clusters
5. Développement et test du système
6. Parallèlement, rédaction de cahiers et rapports : document de conception et spécification, cahier de recettes et validation, rapport technique.

## 3.3 Bases méthodologiques

Afin de mener à bien ce projet, voici les outils qui seront utilisés :

- Logiciel de dessiner des interfaces graphiques : Balsamiq Mockups
- Générateur de documentation : SandCastle ( documentation compilation pour gérer les bibliothèques de classes)
- IDE : Visual Studio 2012
- Langage de programmation : C#
- Design Patter : MVVM (Model - View - View Model)
- Framework : .Net 4.5
- Bibliothèques :
  - AForge : bibliothèque de traitement d'image et d'intelligence artificielle.
  - AvalonDock : un contrôleur de l'accueil fenêtre pour WPF
  - Accord.Net : est une extension de AForge.NET, un framework C # populaire pour vision par ordinateur et l'apprentissage. Il offre de nombreuses fonctions d'analyse et de traitement statistique (Site web Accord.Net,[1])
  - DynamicDataDisplay : sert à ajouter de visualisation interactive de l'application Silverlight. Elle permet de créer des dessins linéaires, un diagramme à bulles, des cartes et d'autres complexes de diagramme 2D carte. (Site web DynamicDataDisplay,[2])
- Outil de gestion de projet :
  - Redmaine : gestionnaire de différentes version de source code du projet et gérer le processus de projet PFE.
  - TortoiseSVN : un logiciel de gestion de versions sous forme de Plugin pour Microsoft Windows
  - Git : un site pour partager des open sources et des doncement.

# Description générale

Ce chapitre présente le contexte du projet. Ensuite on présente des existants du projet et l'architecture du logiciel, ainsi des contraintes du développement.

## 4.1 Environnement du projet

Les deux principales applications de PARADIIT permettant la transcription des documents sont : Agora et Retro.

**Agora** : un outil d'analyse d'image qui permet, dans un premier temps, d'extraire les éléments contenus dans des images documents numérisés.

**Retro** : réalisé par M. Jean-Yves Ramel, a pour but de visualiser et d'exploiter les résultats de segmentation fournis par AGORA.

**PFE** : Un autre projet a également été développé dans le cadre d'un PFE pour l'année scolaire 2013/2014 par Samantha GEORGES : le projet PFE « Intégration et interfaçage de logiciels de clustering et de transcription » (GEORGES, 2014). L'application développée lors de ce PFE avait pour but d'intégrer deux logiciels de clustering dans RETRO.

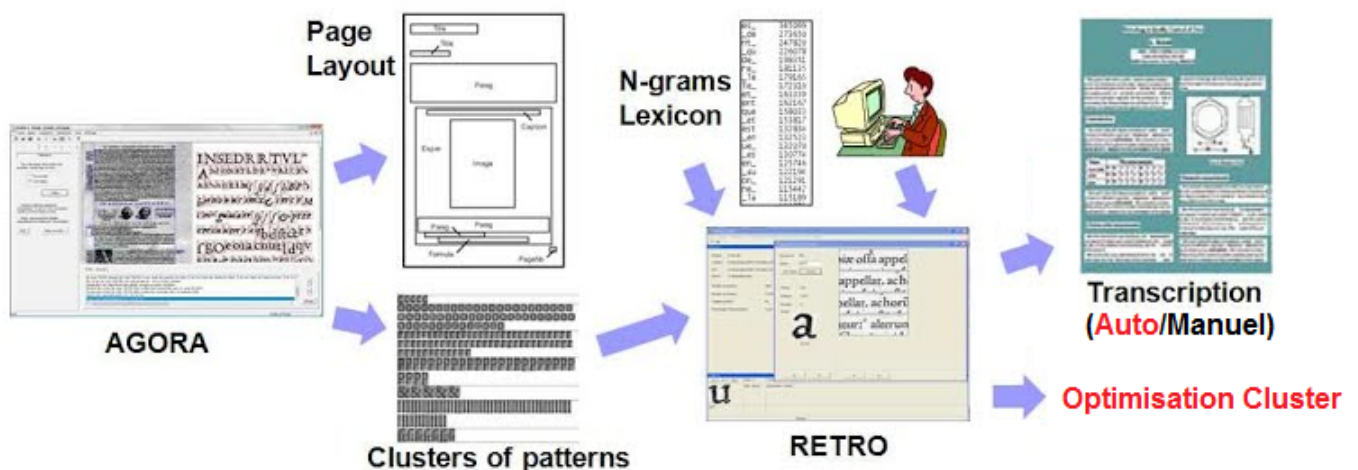


FIGURE 4.1 – Processus d'extraire caractères

Le processus d'extraire caractères est présenté en dessus. Les fonctionnalités à développer sont en rouges.

## 4.2 Étude des existants

Afin de bien maîtriser le sujet du PFE, il faudra d'abord prendre en main le logiciel Retro2014.

### 4.2.1 Retro2014

**Retro2014** est un logiciel a été developpé dans le cadre du PFE par Samantha GERGES qui intègre deux logiciel de clustering dans logiciel Retro et après M. Jean-Yves Ramel fait des modifications sur la structure du projet. Le jeux de données est fournit par M. Ramel. Pour bien comprendre des fonctionnalités, je fais des références sur le site PARADIIT [3].

### 4.2.2 Jeu de données

Retro2014 utilise des résultats de segmentation fournis au préalable par AGORA. Le schéma suivant représente des dossiers constituant des jeux de données pour Retro2014.

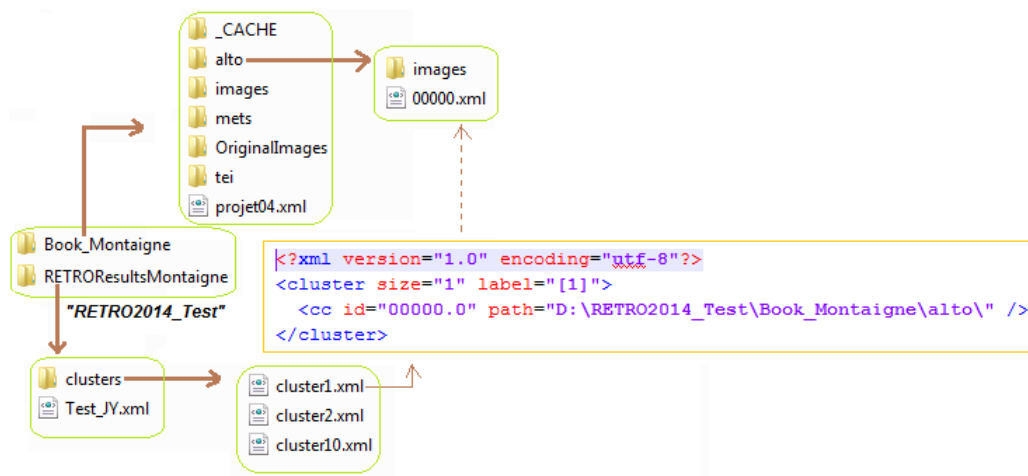


FIGURE 4.2 – Dossiers jeux de données pour Retro2014

un jeu de données du répertoire **RETRO2014\_Test** est donc constitué des deux dossiers :

- **Book\_Montaigne** : dossier contenant le résultat de segmentation par AGORA.
- **RETROResultsMontaigne** : dossier contenant le résultat du projet Retro.

Un jeu de données du répertoire **RETROResultsMontaigne** est constitué des dossiers et des fichiers suivants :

- **clusters** : dossier contenant les fichiers xml du clusters, le fichier de description des informations(nombre de patterns, l'étiquette, ) du cluster.
- **Teste\_JY.xml** : fichier de description du projet Retro.

Un jeu de données du répertoire **Book\_Montaigne** est constitué des dossiers et des fichiers suivants :

- **alto** : dossier contenant la description des fichiers Alto générés par AGORA. Images de tous les composants extraits(bloc, ligne, lettre).
- **images** : dossier contenant les images du projet AGORA avec les noms normalisés.
- **mets** : dossier contenant la description des fichiers Mets générés par AGORA
- **OriginalImages** : dossier contenant l'ensemble des images originales utilisées pour le projet AGORA.
- **tei** : dossier contenant la description des fichiers Tei générés par AGORA.
- **projet04.xml** : fichier de description du projet Retro.

### 4.3 Fonctionnalités et structure générale du système

L'objectif de mon PFE porte surtout sur les fonctionnalités du Clustering et transcription automatique. La description des fonctionnalités concernant ce projet sont présentés par un diagramme d'activités. Le cas d'activités ci-contre montre des scénarios principaux dans Retro2014 : les manipulations de projet Retro, le processus du Clustering, les manipulations de Cluster, les processus de la transcription. Des activités à développer sont en rouge.

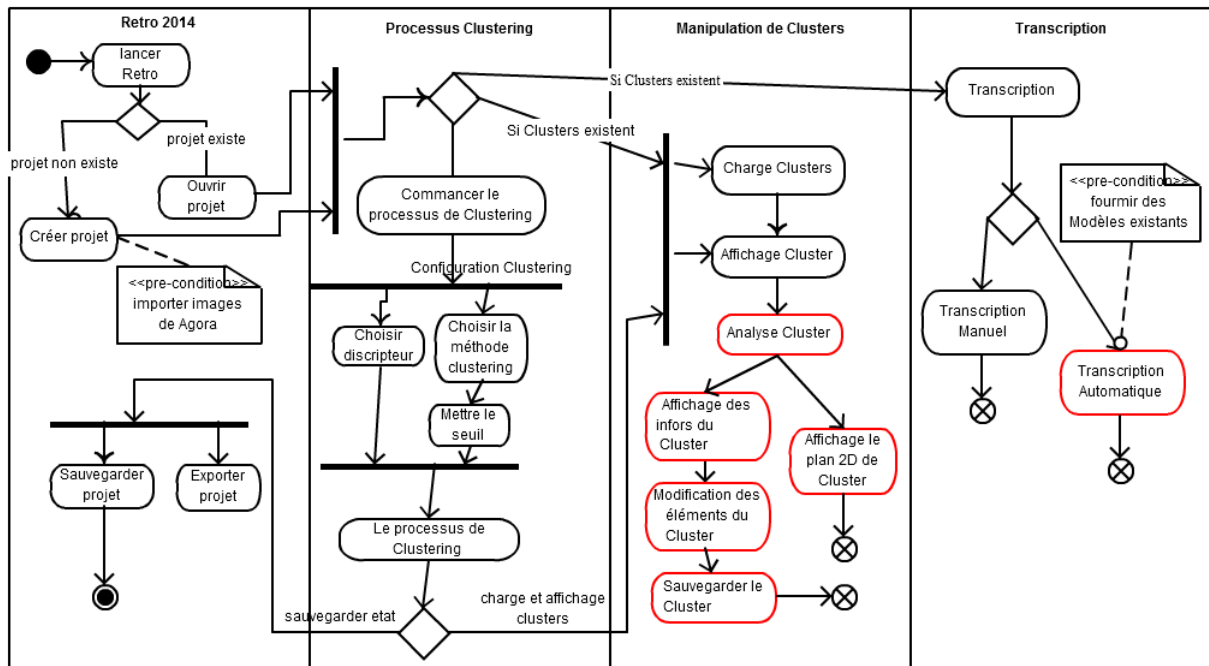


FIGURE 4.3 – Activités Diagramme

Par rapport à la structure générale du système, je présente seulement des composants concernant ce projet. Le diagramme de composants représente des interfaces requis et offerte du composant. L'architecture de Retro2014 déjà existant présente en dessus

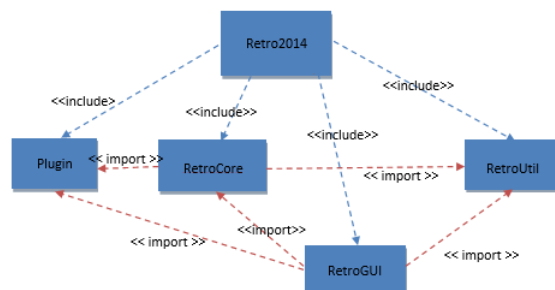


FIGURE 4.4 – Architecture du Retro2014

## 4.4 L'architecture générale du système

Concernant l'architecture du code source, Retro2014 est composé de 4 sous-projets : « Plugin », « RetroCore », « RetroGUI » et « RetroUtil ». On détaille des descriptions et fonctions de sous-projets, après il y a un diagramme pour présenter les relations entre eux.

### 4.4.1 Plugin

Ce sous-projet qui est un paquet contient des modules d'extension d'application pour compléter le logiciel hôte pour lui apporter de nouvelles fonctionnalités. Le type de sortie est une librairie Plugin. Il est séparé en 3 parties (répertoires) :

- DatabaseObjects : Les objets fondamentaux pour stocker des données du projet.
  - Database : comme une base de données (dataset) pour sauvegarder tous les données à traiter.
  - Document : contient tous les patterns (EoC) d'un document.
  - Cluster : après le traitement de Clustering, tous les informations peuvent sauvegarder dans un objet de la classe Cluster. Un Cluster est composé d'une liste de Patterns et ses paramètres de priorités.
  - Pattern : représentant un élément générique qui possède une liste de signatures.
  - Signature : est un pointeur de caractéristiques représentant un Pattern.
- Interfaces : Les interfaces et des classes abstraites de Clustering permettant de développer un module de Clustering, de descripteur ou de la lecture du document.
  - IClusteringPlugin : comme une base de données (dataset) pour sauvegarder tous les données à traiter.
  - IConfig : s'intègre au plugin configuration avec la méthode de sauvegarder.
  - IDescriptorPlugin : une interface pour ajouter le descripteur.
  - IDocumentReaderPlugin : une interface pour charger la base de données.
- PluginTools : il est un outil de traitement image qui contient les fonctionnalités de débruitage de l'image et normalisation l'image.

### 4.4.2 RetroCore

Ce sous-projet contient des méthodes cœur du logiciel, par exemple les méthodes de transcription, les méthodes de traitement et la gestion du projet. Le type de sortie est une librairie RetroCore. Il est séparé en 3 parties :

- Model : Les éléments fondamentaux de la base de données du projet Retro : des paramètres de configuration, l'énumération d'exceptions, l'information sur logiciel.
- OcrTypo : il est composé des fonctionnalités liées à l'algorithme TemplateMatching, à l'export et à l'emplacement des données. Il sert à réaliser la transcription automatique.
  - FontModel : définir l'objet Font. Et on considère un Font comme un modèle d'apprentissage.
  - IOCR : l'interface de OCR moteur (Optical Character Recognition) est la reconnaissance optique de caractères. Celui-ci permet de récupérer le texte dans l'image d'un texte imprimé et de le sauvegarder dans un fichier pouvant être exploité dans un traitement de texte pour enrichissement.
  - TemplateMatchingOCREngine : héritage « IOCR ». Il est un OCR moteur utilisant TemplateMatching pour la transcription automatique.
- ViewModel : il définit les ViewModel éléments du Retro2014 en l'architectural pattern – MVVM.

### 4.4.3 RetroGUI

Le sous-projet principal de l'application qui fait appel aux autres sous-projets. IL est composé de IHM d'application. Le type de sortie est l'Application Windows.

#### 4.4.4 RetroUtil

Ce sous-projet incorpore les outils liés à l’affichage de l’écran dynamique de démarrage et à la conversion d’une image.

#### 4.4.5 Diagramme du package

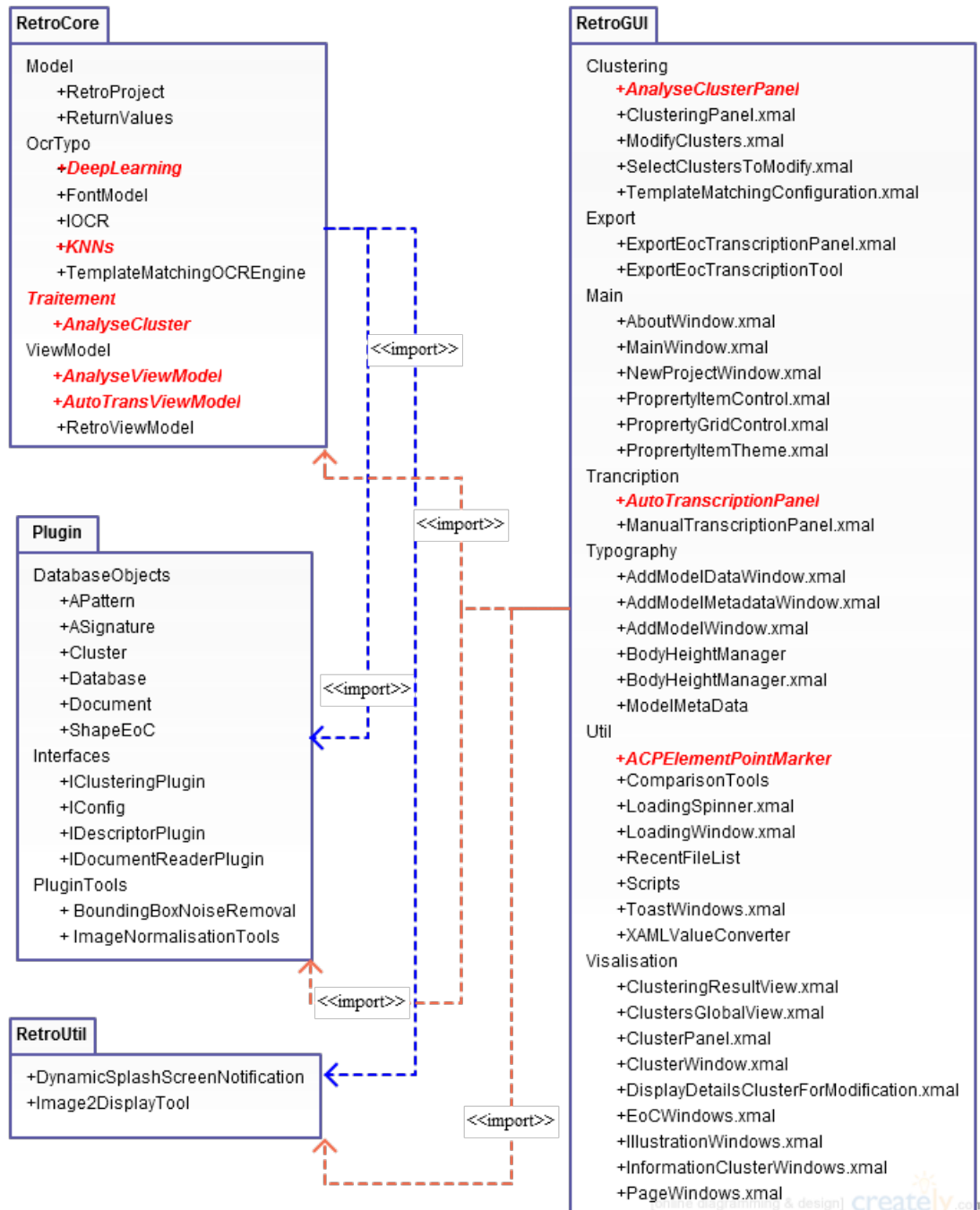


FIGURE 4.5 – Packages Diagramme

L'architecture finale est comme le Packages Diagramme fig.4.5. Un sous-projet est un package et les classes ou les IHMs sont regroupés dans différent répertoires. Les fichiers avec "+" avant sont des fichier .cs(classe), sauf les fichiers avec extension .xmal. Les classes nouvelles ou les IHMs ajoutés(fichiers .xmal) sont rouge.

## 4.5 Contraintes de développement et d'exploitation

Toutes les fonctionnalités de Retro ancien ne doivent pas être modifiées et doivent intégrer au développement du projet. Le projet développé doit être Open-Source constitue une autre contrainte car ceci implique un code parfaitement commenté (en anglais), l'utilisation de l'anglais pour les noms de variables et autres, et un code qui soit le plus modulaire possible.

### 4.5.1 Cycle de Développement & Design Pattern

#### Cycle de Développement

On applique la méthode Agile pour le développement du Retro2014. Le cycle de développement du logiciel prendront en compte toutes les étapes de la conception d'un logiciel. La conception de logiciel met en œuvre un ensemble d'activités qui à partir d'une demande d'informatisation d'un processus.[?]

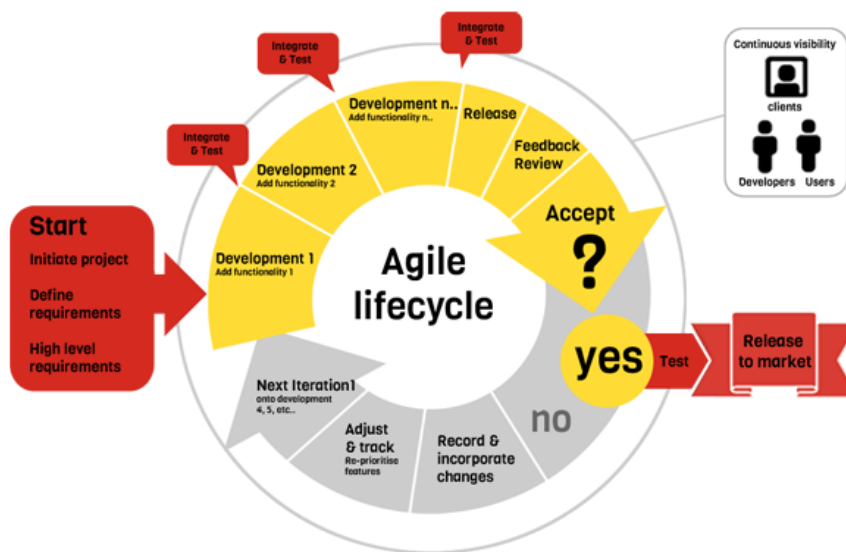


FIGURE 4.6 – Agile cycle

### 4.5.2 Patron de conception(Design Pattern)

**IHM** Retro2014 est développé de la manière WPF(Windows Presentation Foundation) et la mode "MVVM". Donc on continue d'utiliser cette manière.

**Patron de conception** est une solution qu'on adapte la méthode "MVVM" sous C# qui s'est construit au fur et à mesure que les développeurs créaient des applications utilisant le XAML. Cette méthode permet de séparer la vue de la logique et de l'accès aux données en accentuant les principes de binding et d'événement.

1. **Modèle** : correspond aux données. Il s'agit en général de plusieurs classes qui permettent d'accéder aux données. Peu importe la façon dont on remplit ces données (base de données, service web, etc), c'est ce modèle qui est manipulé pour accéder aux données.
2. **View** : correspond à tout ce qui sera affiché. En pratique, il s'agit du fichier .xaml.
3. **View-Modèle** : ou modèle de vue est le lien entre le modèle et la vue. Il s'agit d'une classe qui fournit une abstraction de la vue. Il s'appuie sur la puissance du binding pour mettre à disposition de la vue les données du modèle.

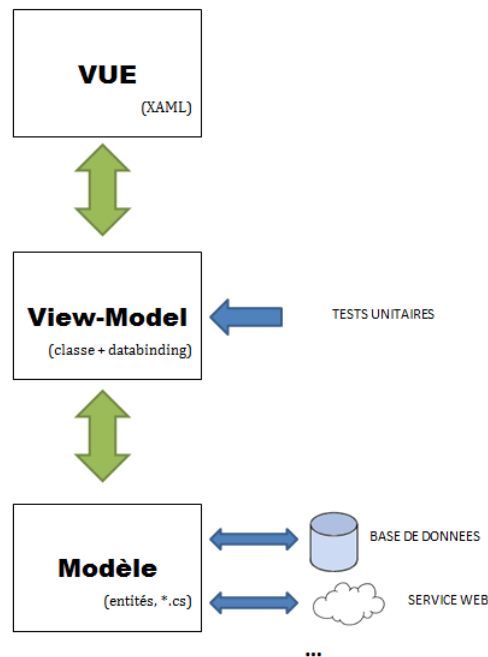


FIGURE 4.7 – MVVM Modèle



# Fouille de données visuelles

**Fouille de données visuelles** découvre d'informations intéressantes dans un paquet de données, en anglais Data Mining. Il est fortement lié à l'apprentissage automatique(machine learning). Pour bien comprendre le mécanisme du logiciel Retro et savoir comment designer le logiciel, on fait des études dans des aspects suivant.

## 5.1 Rappel : Analyse en Composantes Principales (ACP)

Méthode factorielle de dimension pour l'exploration statistique de données quantitatives complexes. Représentations graphiques des individus, des variables et simultanée ; qualité de représentation. L'analyse en Composantes Principales (ACP) est un grand classique d'analyse des données pour l'étude exploratoire ou la compression d'un grand tableau  $n \times p$

### 5.1.1 Objectifs

On dispose d'un tableau de données  $X(n \times p)$ . Ce tableau définit deux nuages de points :

**Nuage de points-variables** : coordonnées des vecteurs variables tracées dans le repère dont les axes représentent les individus( espace de dimension  $n$ ).

**Nuage de points-individus** : coordonnées des vecteurs individus tracées dans le repère dont les axes représentent les variables (espace de dimension  $p$ ).

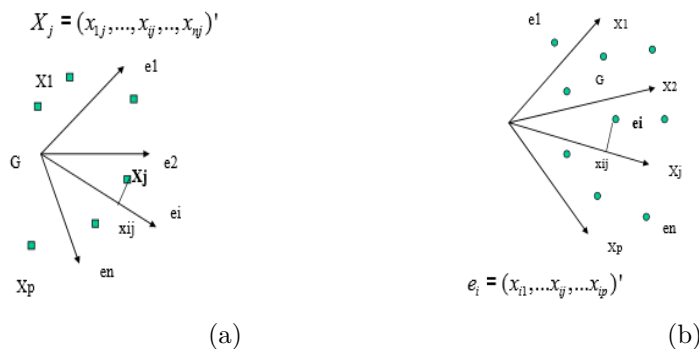


FIGURE 5.1 – (a)le nuage des points variables représenté dans l'espace de dim  $n$  défini par les individus;(b)le nuage des points individus représenté dans l'espace de dim  $p$  défini par les variables

A cause des difficulté à mettre en évidence les relations globales existant entre les variables dès que  $p > 3$ , on va retirer les relations vraiment caractéristiques(proximités entre variables et individus), ceci en limitant la perte d'information. On va déterminer un sous-espace de dimension  $q < p$ ( $q$  nouveaux axes) (ou  $q < n$ ), sur lequel projeter les nuages de points relatifs qui soit visuelle, soit le moins déformant possible. Il faut que l'axe sur lequel on projette permette la dispersion maximale. Le sous-espace est appelé espace factoriel du nuage.

**L'espace factoriel** On peut définir  $q$  nouvelles variables/individus comme axes du repère du nuage de points-individus/points-variables : les composantes principales.

- Le meilleur axe (premier axe factoriel) sera celui sur lequel le nuage de points projeté est de dispersion, tel que le nuage projeté est d'inertie maximale.
- Le second axe sera celui qui, après le premier est tel que le nuage projeté est d'inertie maximale, tout en étant orthogonal au premier.

### 5.1.2 Les étapes d'une ACP

#### 1) Choix du tableau $X$

**Travail toujours sur le tableau centré** : on montre que tout axe factoriel passe par le centre de gravité - le nouveau repère est centré en  $G$ .

Si  $X$  n'est pas réduit, l'importance que prendront les variables dans le calcul des composantes principales est fonction de leur ordre de grandeur ; une variable d'écart-type faible. Des variables de fort écart-type construiront les premières composantes principales : les calculs ne sont pas faux, et conduisent à la même interprétation mais la lecture des résultats risque d'être brouillée. Donc on commence souvent par centrer et réduire (multiplier chaque membre de  $X$  par  $1/n$ ).

#### 2) Analyse directe

**Construction de l'espace factoriel du nuage de points-individus** Détermination de l'origine est qu'on choisit les axes principaux d'inertie du ACP. Les étapes de choisir les axes sont comme suivant :

- rechercher un axe  $\Delta_{u1}$  maximisant l'inertie expliquée  $I_{\Delta_{u1}}$ . On note  $E_1 = \Delta_{u1}$
- rechercher un axe  $\Delta_{u2}$  orthogonal à  $E_1$ , maximisant l'inertie expliquée  $I_{\Delta_{u2}}$ . On note  $E_2 = E_1 \oplus \Delta_{u1}$ .
- ...
- rechercher un axe  $\Delta_{uk}$  orthogonal à  $E_{k-1}$ , maximisant l'inertie expliquée  $I_{\Delta_{uk}}$ . On note  $E_k = E_{k-1} \oplus \Delta_{uk}$ .

Pour calculer des vecteurs de l'axe, il faut créer la matrice des variances-covariances et récupérer des vecteurs et valeurs propres. La matrice est définie par  $V = {}^tX * X$  ( $X$  est la matrice individus/variable modifiée)

#### 3) Analyse duale

**Projection des individus sur les vecteurs propres** On multiplie la matrice individus/variables modifiée et les vecteurs propres. Puis on obtient une matrice contenant les nouvelles coordonnées de chaque individu projeté sur les vecteurs propres.

#### 4) Interprétation de ces analyses

Choix du nombre d'axes  $q$  à retenir, construction des nuages de points projetés sur ces axes, interprétation des axes principaux et étude des proximités entre points.

#### 5) Synthèse des résultats

Construction éventuelle du tableau  $C$  réduit (tableau des composantes principales) et visualisation des nuages de points associés.

## 5.2 Apprentissage automatique

**Définition informelle** L'apprentissage automatique observe d'un phénomène et construit d'un modèle de ce phénomène. En suite on peut prévoir et analyse du phénomène automatiquement grâce au modèle. Il y a deux grand paradigmes d'apprentissage :

- **Cas non-supervisé**(parfois dénommé "clustering") : il divise un groupe hétérogène de données, en sous-groupes de manière que les données considérées comme les plus similaires soient associées au sein d'un groupe homogène et qu'au contraire les données considérées comme différentes se retrouvent dans d'autres groupes distincts. Il n'y a pas de sortie à priori. Il parfois applique dans des domaines en dessus :
  - **Positionnement** : représenter des objets dans le plan (un point par objet). Applications : visualisation globale d'un jeu de données, analyse visuelle (groupes, corrélation, etc.)
  - **Classification(Clustering)** : trouver dans un ensemble d'objets des groupes homogènes(classes) et bien distincts les uns des autres. Application : typologie de clients, regroupement de gènes, regroupement de pages web.
  - **Recherche de schémas fréquents** : trouver des groupes d'objets fréquemment ensembles, trouver des séquences fréquentes d'actions.Applications : recommandations, offres marketing, etc.
- **Cas supervisé** : on cherche à produire automatiquement des règles à partir d'une base de données d'apprentissage contenant des exemples qui sont déjà pré-classés en représentant un groupe d'individus homogènes et en permettant de classer les nouveaux arrivants. Le processus en deux étapes : la construction d'un modèle sur les données dont la classe est connue(la base d'apprentissage), la utilisation pour classification des nouveaux arrivant.
  - **Discrimination** : il y a  $q$  classes d'objets. Puis placer une nouvelle observation  $x$  dans une des  $q$  classes. Applications : diagnostic médical, reconnaissance de caractères, etc.
  - **Ranking/scoring** :donner des objets intéressants (grands qu sens de l'ordre)- c'est à dire, si un objet est plus intéressant qu'an autre, on donne un score d'intérêt à un objet. Application : recherche d'information( page rank de Google), suggestions (amazon).
  - **Régression** : associer un objet de l'ensemble complexes à une nouvelle observation. Application : associer un arbre de syntaxe à texte, traduction automatique.

### 5.2.1 Systèmes OCR

La reconnaissance optique de caractères part de l'image numérique réalisée par un scanner optique d'une page ou un appareil photo numérique et produit en sortie un fichier texte en divers formats (texte simple, formats de traitements de texte, XML. . . ) Les étapes de traitement peuvent être schématisées ainsi :

1. **Pré-analyse de l'image** : le but est d'améliorer éventuellement la qualité de l'image. Ceci peut inclure le redressement d'images inclinées, des corrections de contraste, image binaire, la détection de contours.
2. **Segmentation en lignes et en caractères** (ou Analyse de page) : vise à isoler dans l'image les lignes de texte et les caractères à l'intérieur des lignes.
3. **Reconnaissance** : après normalisation, une instance à reconnaître est comparée à une bibliothèque de formes connues, et on retient la forme la plus « proche » (ou les  $N$  formes les plus proches), selon une distance ou une vraisemblance. Les techniques de reconnaissance se classent en quelques grands types :
  - (a) Classification par Caractéristiques(Features) : une forme à reconnaître est représentée par un vecteur de valeurs numériques calculées à partir de cette forme. Le rôle du classificateur est de

déterminer à quelle classe de caractères la forme à reconnaître appartient le plus vraisemblablement.

- (b) Méthodes métriques : consistent à comparer directement la forme à reconnaître, au moyen d'algorithmes de distance, avec un ensemble de modèles appris.
  - (c) Méthodes statistiques : dans le domaine de la reconnaissance d'écriture manuscrite, il est fréquemment fait appel aux méthodes probabilistes/statistiques comme les chaînes de Markov.
4. **Post-traitement** utilisant des méthodes linguistiques et contextuelles pour réduire le nombre d'erreurs de reconnaissance : systèmes à base de règles, ou méthodes statistiques basées sur des dictionnaires de mots, de syllabes.
  5. Génération du format de sortie, avec la mise en page pour les meilleurs systèmes.

### 5.3 K-Nearest Neighborhoods(KNNs)

La méthode de K plus proches voisins est une méthode basée sur l'apprentissage par analogie et l'apprentissage supervisé.

**Principe** Elle est simple et directe. Elle ne nécessite pas d'apprentissage mais simplement le stockage des données d'apprentissage. Un exemple de classe inconnue est comparée à toutes les données stockées. On choisit pour le nouveau exemple la classe majoritaire parmi ses K plus proches voisins(elle peut donc être lourde pour des grandes bases de données) au sens d'une distance choisie.

### 5.4 L'algorithme du KNNs

Afin de trouver les K plus proches d'une donnée à classer, on peut choisir la distance euclidienne. Soient deux données représentées par deux vecteurs  $x_i$  et  $x_j$ , la distance entre ces deux données est donnée par

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^d (x_{ik} - x_{jk})^2} \quad (5.1)$$

---

**Algorithm 1:** Algorithme du plus proches voisin (1-ppv)

---

```

input : Données d'apprentissage :  $X^{train} = (x_1^{train}, \dots, x_n^{train})$ ; Classes des données
         d'apprentissage :  $Z^{train} = (z_1^{train}, \dots, z_n^{train})$ ; Données de test :  $X^{test} = (x_1^{test}, \dots, x_m^{test})$ 
output: Classes des données de test :  $Z^{test} = (z_1^{test}, \dots, z_m^{test})$ 

1 special treatment of the first line;
2 for  $i \leftarrow 1$  to  $m$  do
3   special treatment of the first element of line i;
4   for  $j \leftarrow 1$  to  $n$  do
5     Calculer la distance euclidienne entre  $x_i^{test}$  et  $x_j^{train}$  en utilisant l'équation 5.1;
6      $d_j \leftarrow d(x_i^{test}, x_j^{train})$  ;
7   end
8   Calculer la classe  $z_i^{test}$  du ième exemple qui vaut la classe de son ppv;
9   trouver l'indice du ppv de  $x_i^{test}$   $ind\_ppv_i \leftarrow \operatorname{argmin}_{j=1}^n d_j$  ;
10  trouver la classe du ppv de  $x_i^{test}$  (qui est  $x_{ind\_ppv_i}^{train}$ );
11   $z_i^{test} = z_{ind\_ppv_i}^{train}$  ;
12 end

```

---

Ensuite, étendez votre code pour le cas des K-pvv pour une valeur de  $K \geq 1$ . L'algorithme KNNs est donné par le pseudo-code 2 ci-après.

---

**Algorithm 2:** Algorithme du plus proches voisin (K-ppv)
 

---

```

input : Données d'apprentissage :  $X^{train} = (x_1^{train}, \dots, x_n^{train})$ ; Classes des données
         d'apprentissage :  $Z^{train} = (z_1^{train}, \dots, z_n^{train})$ ; Données de test :  $X^{test} = (x_1^{test}, \dots, x_m^{test})$ ;
         Nombre des ppv  $K$ 
output: Classes des données de test :  $Z^{test} = (z_1^{test}, \dots, z_m^{test})$ 

1 for  $i \leftarrow 1$  to  $m$  do
2   for  $j \leftarrow 1$  to  $n$  do
3     Calculer la distance euclidienne  $d_{ij}$  entre  $x_i^{test}$  et  $x_j^{train}$  en utilisant l'équation 5.1;
4      $d_j \leftarrow d_{ij}$ ;
5   end
6   Calculer la classe  $z_i^{test}$  du ième exemple qui vaut la classe de son ppv;
7   \*trouver les K-ppv de  $x_i^{test}$                                      *\
8   Trier les distances  $d_j$  selon un ordre croissant pour  $j = 1, \dots, n$ ;
9   Récupérer un même temps les indices  $IndVoisins$  avant le tri des  $d_j$ ;
10  Récupérer les classes des  $K$  premiers ppv à partir des indices  $IndVoisins$  et en trouver la
    classe majoritaire :  $C_k \leftarrow 0 (k = 1, \dots, K)$ ;
11  for  $k \leftarrow 1$  to  $K$  do
12     $ind\_voisin_k \leftarrow IndVoisins_k$ ;
13     $h \leftarrow z_{ind\_voisin_k}^{train}$ ;
14     $C_h = z_h + 1$ ;
15  end
16  \*trouver la classe du ppv de  $x_i^{test}$ :
    (la classe majoritaire de celles de ses K-ppv)                       *\
17   $z_i^{test} = arg \sum_{k=1}^K C_k$ ;
18 end
  
```

---

## 5.5 Caractères de l'algorithme KNNs

- Les attributs ont le même poids : certains exemples peuvent être moins classant que d'autres. On peut centrer et réduire pour éviter les biais.
- Apprentissage paresseux : KNNs n'a pas besoins de préparation avant le classement. Mais il a besoin de technique d'indexer pour large base de données. Tous les calculs sont fait lors du classement.
- Calcul du score d'une classe : on peut changer les résultats, variantes possibles.

## 5.6 Neuron Network - Deep learning

Apprentissage en profondeur(Deep learning) est une branche de l'apprentissage automatique(machine learning) basées sur l'apprentissage de modèles de données.

### 5.6.1 Apprentissage en profondeur(Deep learning)

**La profondeur** est la longueur de plus long chemin d'entrée à sortie. La profondeur de Classique Neuron Network égale le nombre de couche (e.g. le nombre de couche cachée plus 1 lequel représente la couche

sortie). La profondeur de 2 est suffisante dans de nombreux cas pour approcher n'importe quelle fonction avec une précision arbitraire. Par exemple, pour les portes logiques, les neurones formels (à seuil), les neurones avec une fonction d'activation sigmoïdale etc. Mais cette possibilité a un prix : le nombre de nœuds requis dans le graphe (c'est-à-dire le nombre d'opérations, mais aussi le nombre de paramètres à entraîner) peut devenir très large.

Le sens de profondeur des processus cognitifs est que les humains organisent leurs idées et leurs concepts de façon hiérarchique. Les humains apprennent d'abord des concepts simples, et les combinent ensuite pour représenter des concepts plus abstraits. En suite les humains construisent des solutions en combinant de nombreux niveaux d'abstraction et de traitement.

### 5.6.2 Modèle probabiliste pour les architectures profondes

On s'intéresse particulièrement au modèle de la machine de Boltzmann, donc certaines variantes sont utilisées dans des architectures profondes comme les Deep Belief Networks et les Deep Boltzmann Machines.

#### Deep Neural Network(DNN)

Deep neural network est un artificiel neurone network(ANN) avec multiple caches couches entre la couche d'entrée et la couche sortie. Familier à ANNs, DNNs peut être modèle complexe non-linéaire relation. Un DNN peut être discriminant entraîné par l'algorithme de propagation arrière. Le poids mise à jour via stochastique gradient descendant utilisant l'équation suivant :

$$\Delta w_{ij}(t+1) = \Delta w_{ij}(t) + \eta \frac{\partial C}{\partial w_{ij}} \quad (5.2)$$

$\eta$  : le taux d'apprentissage

$C$  : le coût fonction, qui correspond au type d'apprentissage (non – supervisé, supervisé, renforcement) et la fonction d'activation.

#### Deep Belief Networks(DBNs)

Deep Belief Network, DBNs est un probabiliste, un modèle générative contenant plusieurs couches caches. Il est aussi considéré comme composé de modules d'apprentissage simple.

Le résultat de pré-entraînement DNN peut être utilisé comme les poids initiaux par DBNs. En suite DBNs peut utiliser propagation arrière ou d'autre algorithme discriminants pour réglage fin les poids. Cette méthode est vraiment utile quand la base d'apprentissage est limitée, puisque les poids mal initialisés peuvent avoir un impact significatif sur la performance du modèle final. Ces poids sont préformés dans une région de l'espace de poids qui est plus proche des pondérations optimales (comparant à l'initialisation aléatoire). Cela permet à la fois une meilleure capacité de modélisation et de convergence plus rapide de la phase de réglage fin.

DBNs sont des modèles graphiques qui apprennent à extraire une représentation hiérarchique profonde des données d'apprentissage. Ils modélisent la distribution conjointe entre observée vecteur  $x$  et la  $l$  couche cachée  $h^k$  :

$$P(x, h^{(1)}, \dots, h^{(l)}) = \left( \prod_{k=0}^{l-2} P(h^k | h^{k+1}) \right) P(h^{l-1} | h^l) \quad (5.3)$$

Où  $x = h^0$ . La variables (vecteur) aléatoire associée à la couche  $k$  est  $h^k$ . Les deux dernières couches ont une distribution jointe qui est donnée par une RBM (la dernière de la pile). Les RBMs du dessous

servent seulement à définir les probabilités conditionnelles  $P(h^k|h^{k+1})$  pour les unités visible  $h^k$  sur les unités cachées  $h^{k+1}$  de la RBM au niveau k, et  $P(h^{l-1}|h^l)$  est la distribution visible caché conjointe dans le RBM de haut niveau.

### Restricted Boltzmann machines(RBM)

Un DBN peut être entraîné en la manière de non-supervisé, couche par couche où les couches sont construit par RBM. Un RBM est un non-orienté, générative basée sur l'énergie avec une couche d'entrée et seule couche cachée. Connexions existent seulement entre les unités visibles de la couche d'entrée et les unités cachées de la couche cachée. Il n'y a pas de connexions entra la couche entre des nœuds. La couche visible sert à importer des données d'apprentissage. La couche cachée sert à extraire des caractéristiques [4]. C'est à dire, donnant des valeurs d'unité visible, les valeurs d'unités cachée n'ont pas de corrélation. Inversement il est aussi.

$$P(h|v) = \prod_{j=1}^N P(h_j|v) \quad (5.4)$$

$$P(v|h) = \prod_{i=1}^M P(v_i|h) \quad (5.5)$$

Le possibilité d'activation d'unité cachée :

$$P(h_j = 1|v) = \sigma(h_j) = \frac{1}{1 + e^{-h_j}} \quad (5.6)$$

Ceci est illustré dans la figure ci-dessous.

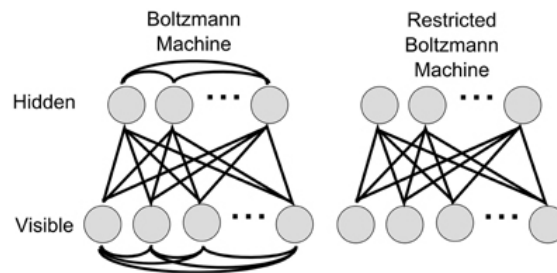


FIGURE 5.2 – RBM structure

Échantillon d'un DBN :

- l'échantillon un  $h^{l-1}$  de la RBM du dessus (numéro  $l$ , par exemple en faisant du Gibbs
- pour k de  $l-1$  à 1 : l'échantillon les unités visibles  $h^k$  étant données les unités cachées  $h^{k+1}$  dans la RBM k.
- retourner le dernier échantillon produit  $h^k$ , qui est le résultat de la génération par le DBN.

La méthode d'entraînement du RBMs est proposée par Geoffrey Hinton pour implémenter avec le modèle « Produit de l'expert » qui est connu comme Divergence Contrastives (CD). CD fournit une approximation de la méthode du maximum de vraisemblance qui devrait idéalement être appliquée pour l'apprentissage les poids de RBM. Dans l'entraînement d'un seul RBM, mise à jour des poids sont effectuées avec la montée gradient par l'équation suivante :

$$\Delta w_{ij}(t+1) = \Delta w_{ij}(t) + \eta \frac{\partial \log(p(v))}{\partial w_{ij}} \quad (5.7)$$

$P(v) = \frac{1}{Z} \sum h e^{-E(v,h)}$  : La probabilité de vecteur visible.  $Z$  est la fonction partition pour normalisation.  $E(v, h)$  est l'énergie fonction assigne l'état de network.

Une fois qu'un RBM est formé, un autre RBM peut être empilé au sommet de celui-ci pour créer un modèle multi-couche. Chaque fois qu'un autre RBM est empilé, la couche visible d'entrée est initialisée à un vecteur de formation et les valeurs pour les unités dans les couches de RBM déjà formé sont affectées en utilisant les poids et les préjugés actuels. La couche finale des couches déjà formés est utilisée comme entrée à le nouveau RBM. Le nouveau RBM est ensuite formé à la procédure ci-dessus, et alors ce processus peut être répété jusqu'à ce qu'un critère d'arrêt requis soit atteint.

### L'algorithme proposé avec RBM

Le principal du greedy layer-wise non-supervisé entraînement peut appliquer au DBNs avec RBM autant que les construction blocs pour chaque couche. On fait principalement des études sur l'algorithme que M. Yann LeCun a proposé en 2006 et fait des références au site du M. Yann LeCun [5] et un article [6] du M. Hanlin Goh. Puis on va essayer de réaliser cet algo dans le travail suivant.

L'entrée matrice  $X$  regardant comme une matrice de signatures. Le processus comme suivant :

1. Entraîner la première couche comme un RBM, entrée  $X=h(0)$ , comme la couche visible.
2. Utilisant la première couche, on obtient la représentation (la matrice du poids  $W$ ) laquelle sera utilisé comme la base d'entrées pour la deuxième couche. La représentation peut être choisie pour l'activation moyenne  $p(h^{(1)} = 1|h^{(0)})$  ou les échantillons de  $p(h^{(1)}|h^{(0)})$ .
3. Entraîner la deuxième couche comme un RBM, utilisant des données transformées (échantillons ou activation moyenne) comme la base d'entraînement ( pour la couche visible du RBM).
4. Répéter l'étape 2 et 3 jusqu'à le nombre de couche désiré. Chaque fois il se propage soit des échantillons sois des valeurs moyennes.

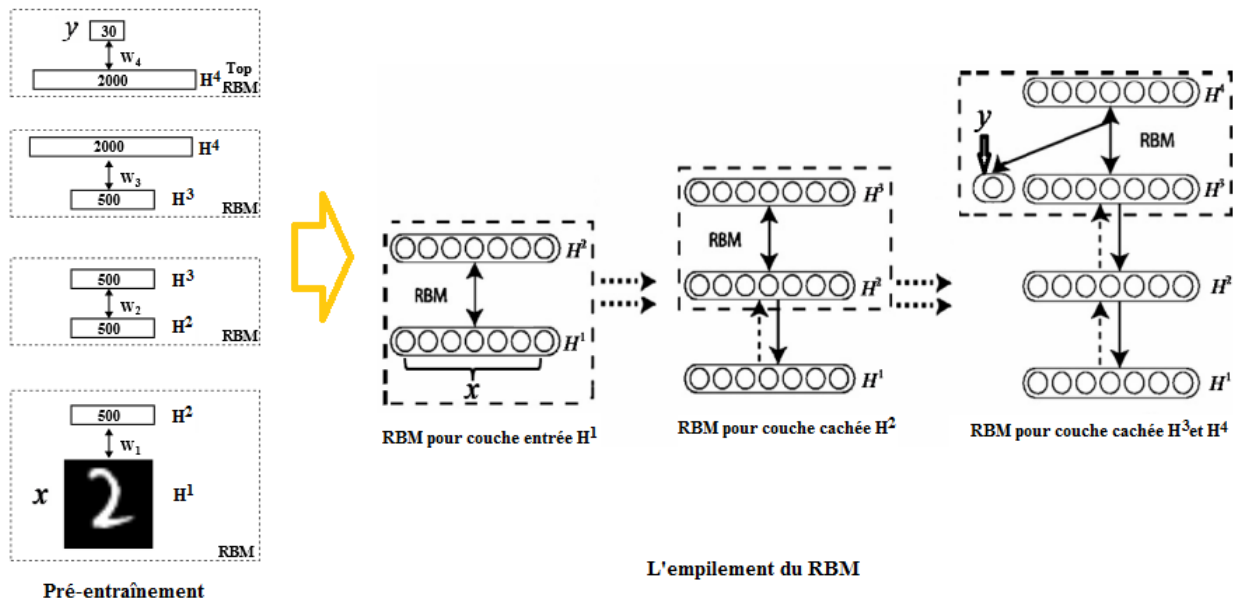


FIGURE 5.3 – Pré-entraînement par RBM

5. Affiner tous les paramètres d'architecture profonde adhérent (fig.5.4 au proxy pour le DBN log (possibilités) ou au critère d'entraînement supervisé (après l'ajout de machines d'apprentissage supplémentaire pour convertir la représentation appris en prédictions supervisés, par exemple un classificateur linéaire).



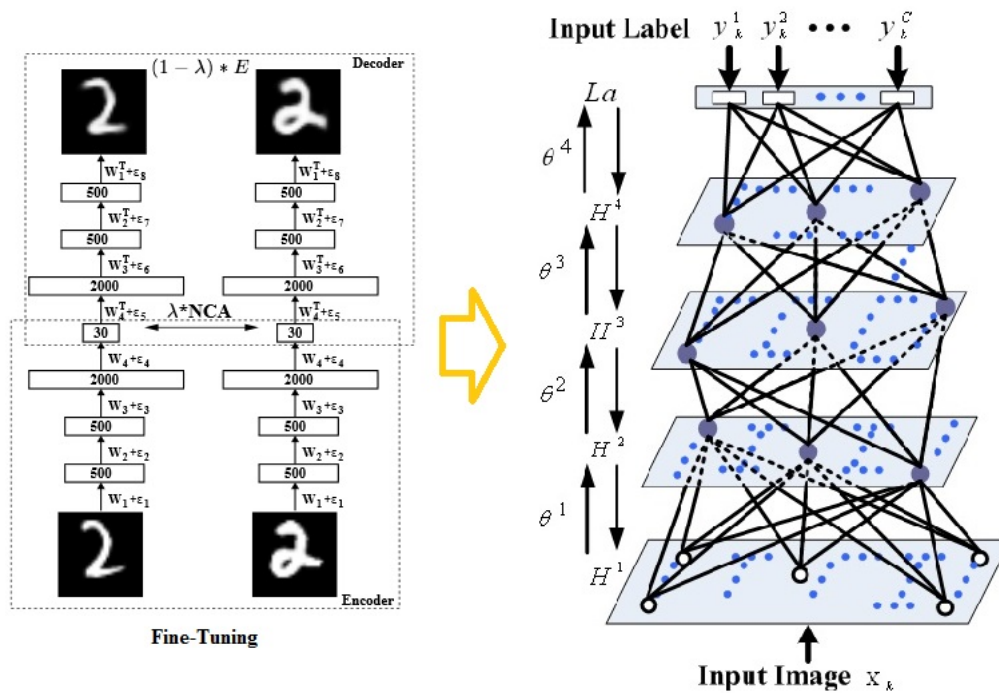


FIGURE 5.4 – Fine-Tuning par arrière-propagation

## 5.7 Forces et Faiblesses de DBNs

La méthode du Neuron Network permet d'approcher tout sorte de fonction. Mais elle a des faiblesses ce qu'il faudra faire des attentions dans l'application :

- Coûteux en apprentissage : le temps d'exécution l'apprentissage est trop long probable à cause du calcul complexes, la possibilité d'élaguer le réseau en connexions ou peu applicable sur de larges BD.
- Effet boîte noire : les comportements difficiles à expliquer.

# Design et Proposition

Avant de commencer à développer, il faudra d'étudier le logiciels existants, les théories correspondantes des fonctionnalités à développer et faire des propositions sur des fonctions à développer. Cette partie va donc vous permettre de comprendre quels ont été choix faits pour la suite du travail.

## 6.1 Proposition de fonctionnalités

Cette partie va donc vous permettre de comprendre quels libraires et méthodes ont été choix faits pour la suite du travail.

### 6.1.1 Optimisation visuelle de clusters

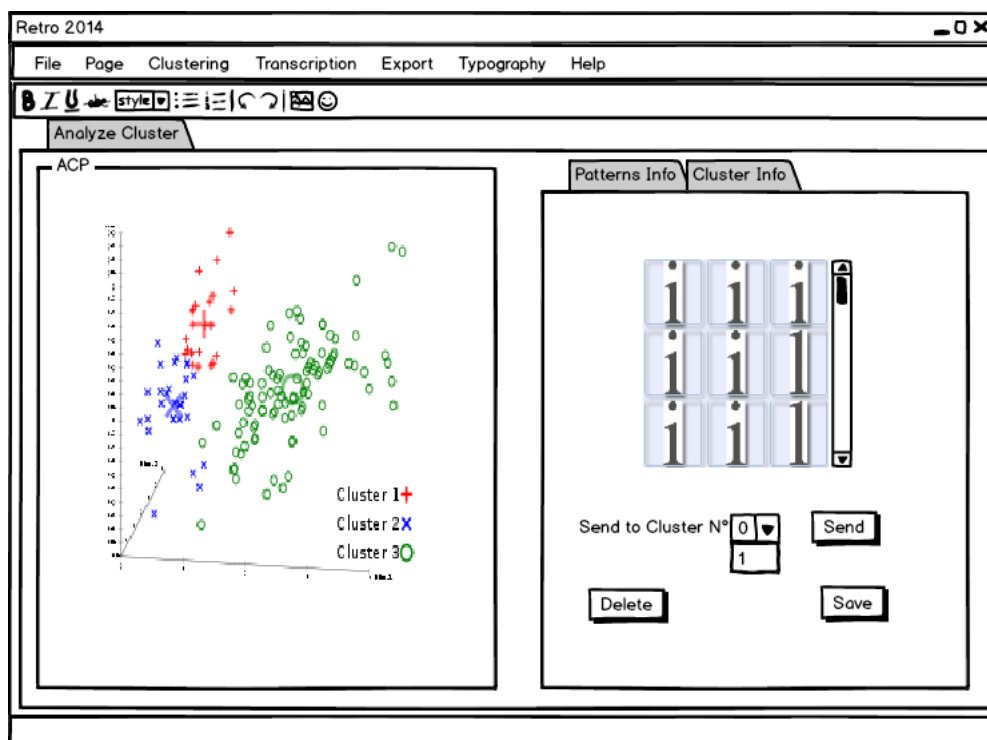


FIGURE 6.1 – Maquette d'analyse Cluster

#### A) Projection par ACP

La réalisation du ACP utilise la librairie "Accord.NET"- classe "PrincipalComponentAnalysis", référencé à un exemple du ACP *Principal Component Analysis (PCA)* sur le site Accord.Net[7]. Après on peut voir le résultat de projection sur un plan. Au niveau du visuelle, On implémente une librairie "WPF DynamicDataDisplay v0.3"[2] pour présenter visuelle de données. Ce librairie peut réaliser différent interaction visuelle de données.

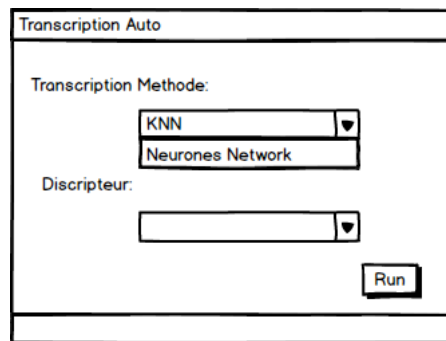
## B) Manipulation interactive des objets projetés

**1. Détection le point sur le plan ACP** La figure du ACP peut détecter la position du souris et afficher la position et id du pattern(point) avec une étiquette quand le souris flotte au pattern(point). En cliquant le pattern(point), on peut trouver directement le pattern dans la liste de patterns pour des manipulations à la suivant.

**2. Modification du Pattern** On peut effectuer deux manipulations sur pattern : supprime et envoi à autre Cluster. Après sélectionnant un pattern dans la liste du patterns, on peut le supprimer ou l'envoyer afin d'améliorer la qualité du Cluster.

### 6.1.2 Reconnaissance de clusters

L'objet du projet est d'explorer de l'ensemble des éléments du cluster pour mieux le reconnaître.



The image shows a software interface titled "Transcription Auto". It contains two dropdown menus. The first is labeled "Transcription Methode:" and has "KNN" selected, with "Neurones Network" visible below it. The second is labeled "Discripteur:" and is currently empty. A "Run" button is located at the bottom right of the interface.

FIGURE 6.2 – Maquette d'Auto Transcription

## Environnement existant

En notre conditions, il y a peu de modèles mais de multiples exemples de l'objet à reconnaître. Celui-ci est contraire aux conditions normales du Neuron Network qui a pleins de modèles à apprendre mais peu d'exemples à reconnaître. Après regardant des résultats expérimentation [4] et sachant des propriétés de quelques algorithmes, on propose de choisir deux méthodes du machine learning : KNNs et DBNs, pour comparer les effets d'algo dans notre situation.

KNNs est une méthode pour une petite quantité de base d'apprentissage, qui garde la base d'apprentissage et pas d'entraînement. Le complexe du temps a rapport au nombre de base d'apprentissage. En contrairement, DBNs a besoins d'une grand nombre d'exemple d'apprentissage et le temps d'apprentissage est sensible.

## A) L'implémentation du KNNs

Pour l'algorithme du KNNs, il va sauvegarder la base d'apprentissage, après calculer la distance entre un exemple de test et la base d'apprentissage.

- **Données entrées** : un répertoire d'apprentissage qui contient les exemples images et des fichiers xml d'exemple. On peut extraire les caractéristiques/signatures d'image avec un descripteur comme des données d'entré et la classe à laquelle le modèle appartient.
- **Données sorties** : les classes de la base de test(changer le fichier .xml du cluster).
- **Libraires** : *Accord.MachineLearning.KNearestNeighbors*

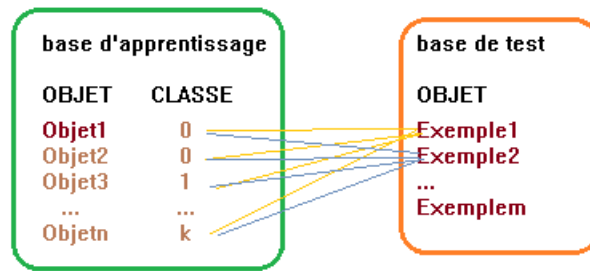


FIGURE 6.3 – KNNs

## B) L'implémentation du DBNs

**Proposition d'algorithme du DBNs** Je voudrais proposer de implémenter un algorithme qui combine le Deep Belief Network avec Restricted Boltzmann machines. celui est expliqué dans la section 5.6.2. Le processus d'auto transcription en DBNs est désigné dans l'image fig.6.4

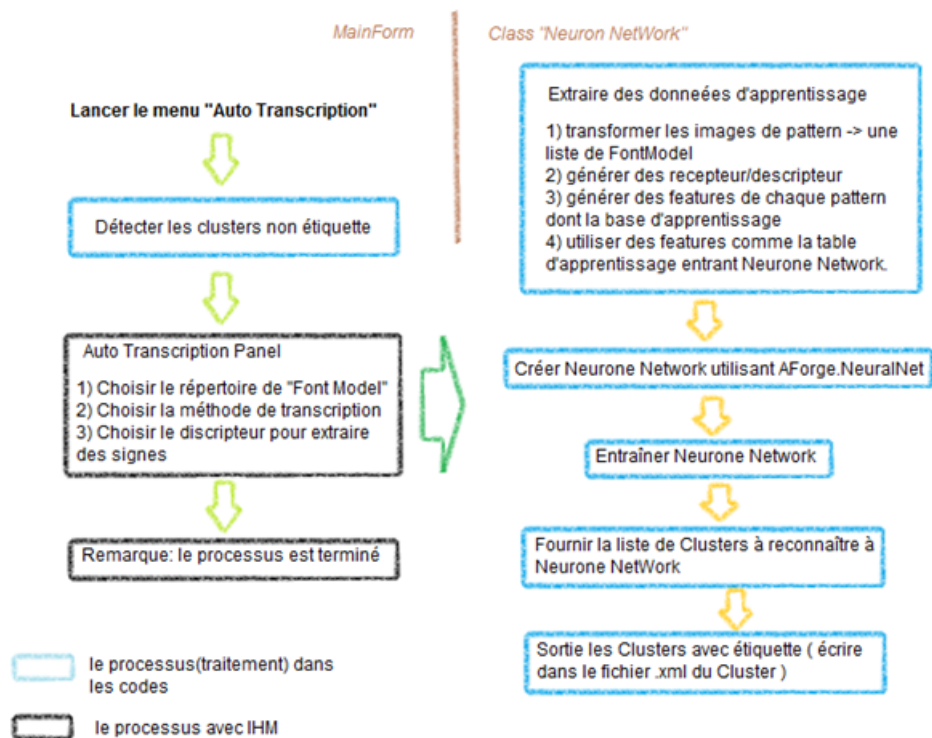


FIGURE 6.4 – Design le processus de DBN

La structure d'algorithme est présentée dans la figure fig. 6.5

**L'architecture du DBN** On désigne des composants du network est comme suivant :

- **DBN structure** : une couche entrée(nombre de neurone : la dimension de caractéristique), une couche sortie(nombre de neurone : nombre de classe de la base d'apprentissage), trois couches cachées(500,500,2000).
- **RBM structure** : une couche d'unité visible, une couche d'unité cachée. Le nombre de neurone correspond la couche arrière et la couche précédente.
- **Libraires** : *Accord.Neuro.Networks.DeepBeliefNetwork* , *Accord.Neuro.Networks.RestrictedBoltzmannMachine*

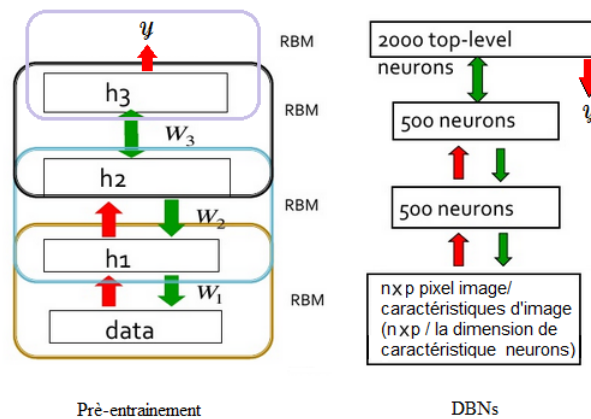


FIGURE 6.5 – DBNs design

- **Constructeur** : `DeepBeliefNetwork(int inputsCount, params RestrictedBoltzmannMachine[] layers, RestrictedBoltzmannMachine(IStochasticFunction function, int inputsCount, int hiddenNeurons).`
- **Matrice d'entrée** : signatures du « ZernikeDescriptor »

On peut faire des références aux deux logiciels : Deep Belief Network and Boltzmann Machines [7] et Neural Network OCR [8].

## 6.2 Conditions de fonctionnement

L'application RETRO s'applique facilement, mais en terme de temps de traitement, sur l'analyse de clusters et sur l'auto transcription, l'exécution de l'application risque d'être très coûteuse. Cela pourrait prendre des heures voir quelques jours.

On peut considérer que la nouvelle version de RETRO s'assure en termes de temps de traitement puisque les méthodes de clustering et les descripteurs utilisés dans PFE-Intégration et interfaçage de logiciels de clustering et de transcription ont été repris tels quels via l'utilisation des DLLS.

## 6.3 Cahier de spécification

Afin de réfléchir précisément au travail à effectuer pour le bon développement de l'application, un cahier de spécification (CDS) a été créé. Celui ci définit précisément l'architecture du projet, ses besoins et ses fonctionnalités. Pour plus de précision quant à l'étude et la réflexion sur le PFE, vous pouvez donc vous reporter au CDS.

# Travail réalisé- Développement

Plusieurs interfaces principales et le menu principal ont été mises à jour dans cette version de Retro. Mais il y a encore plusieurs fonctions existant attendue à développer ou modifier, donc pour dérouler ce projet on que corrige des fonctions correspondant avec des fonctionnalités à développer(voir la section 7.1). L'objectives principaux sont le développement d'améliorer la qualité du cluster, l'IHM "Analyse Cluster"(voir section 7.2) et le développement de reconnaître automatique.

## 7.1 Mise à jour de RETRO

Après la prise en main de Retro, j'ai jugé possible le fait d'y apporter la modification d'affichage des images d'un cluster et du cluster globalement dans le panel "View/Edit". J'ajoute un propriété "RepresentativePath" dans la classe "Cluster" étant lié avec les fichier .xaml pour l'affichage d'image de représentative du cluster. La deuxième modification que je fais est la charge les descripteurs existants du cluster dès lors que il ouvert le Retro projet.

## 7.2 Ajout du module "Analyse Cluster"

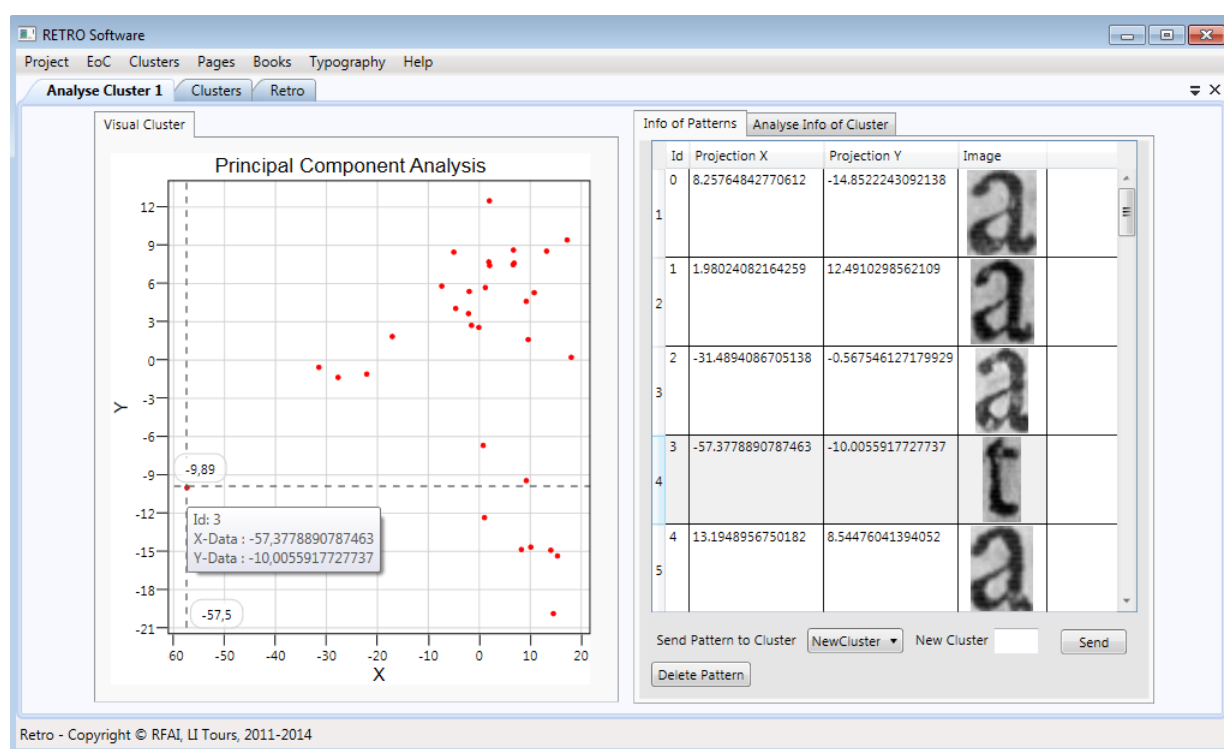


FIGURE 7.1 – Panel d'Analyse Cluster

### 7.2.1 IHM du "Analyse Cluster"

Après choisit un cluster dans le panel "View/Edit", on lance le menu "Cluster->Analyse Cluster"(fig.7.2) entrant dans l'IHM suivant fig.7.1

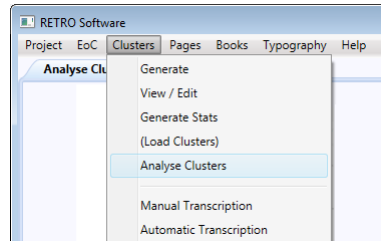


FIGURE 7.2 – Main menu

### 7.2.2 Ajout de fonctionnalités

#### Visuelle Cluster-ACP

Dans le "TabControl"-"Visual Cluster" à gauche, on extrait les signatures "Zernike" du pattern comme entrée du ACP. La projection de deux premiers composants affiche par un "ChartPlotter" dynamique qui peut agrandir et réduire le schéma ACP. Les points projetés dans le schéma présentant un pattern quand le souris flotte sur un point, une étiquette affiche l'id et la position du pattern. En cliquant le point à gauche, la liste du pattern à droite va afficher le pattern correspondant. On montre un exemple sur la figure fig.7.1

L'interactif plan 3D peut être agrandi ou réduit par le souris. En cliquant à droite sur le plan, un menu flottant va afficher des fonctions supplémentaires comme la figure fig.7.3.

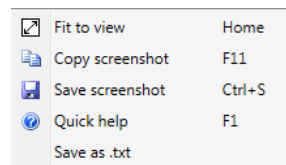


FIGURE 7.3 – Menu flottant

#### Affichage la liste de patterns

Dans le "TabControl"-"Info of patterns" à droite, on affiche tous les patterns dans le cluster : l'id, la position dans le plan ACP, l'image du pattern. On ne peut pas modifier les contenus dans la liste. Tant à la limite d'analyse, on a testé un cluster avec 1008 patterns. Toutes les fonctions marchent bien.

#### Supprimer le pattern sélectionné

Après sélectionnant un pattern dans la liste à droite, on peut supprimer le pattern sélectionné et en le même temps le fichier .xml du cluster est modifié.

#### Envoi le pattern sélectionné

Après sélectionnant un pattern dans la liste à droite, on peut l'envoyer vers soit d'autre cluster existe, soit un nouveau cluster. Le comboBox à côté contient l'id et l'étiquette du cluster. Le textbox, qui est rempli

de l'étiquette du nouveau pattern, va être affiché quand le combobox choisit "NewCluster". Tandis que l'opération est validé, le fichier .xml du cluster est modifié.

### Affichage des données d'analyse cluster

Dans le tableltem "Analyse Info of Cluster", on peut voir des données du cluster : id, label, nombre de patterns, le descripteur de signature du pattern. (voire fig.7.7)

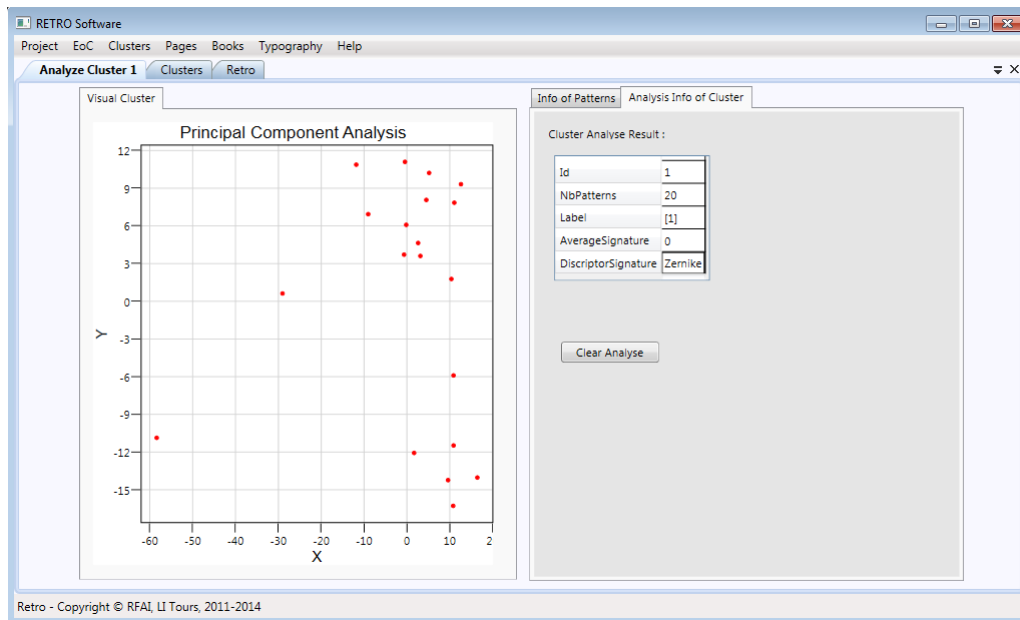


FIGURE 7.4 – Analyse Information du Cluster

### Nettoyage le panel "Analyse Cluster"

Dans le tableltem "Analyse Info of Cluster", le button "Clean Analyse" nettoie toutes les données sur le panel "Analyse Cluster". Le résultat est comme la figure fig.7.5

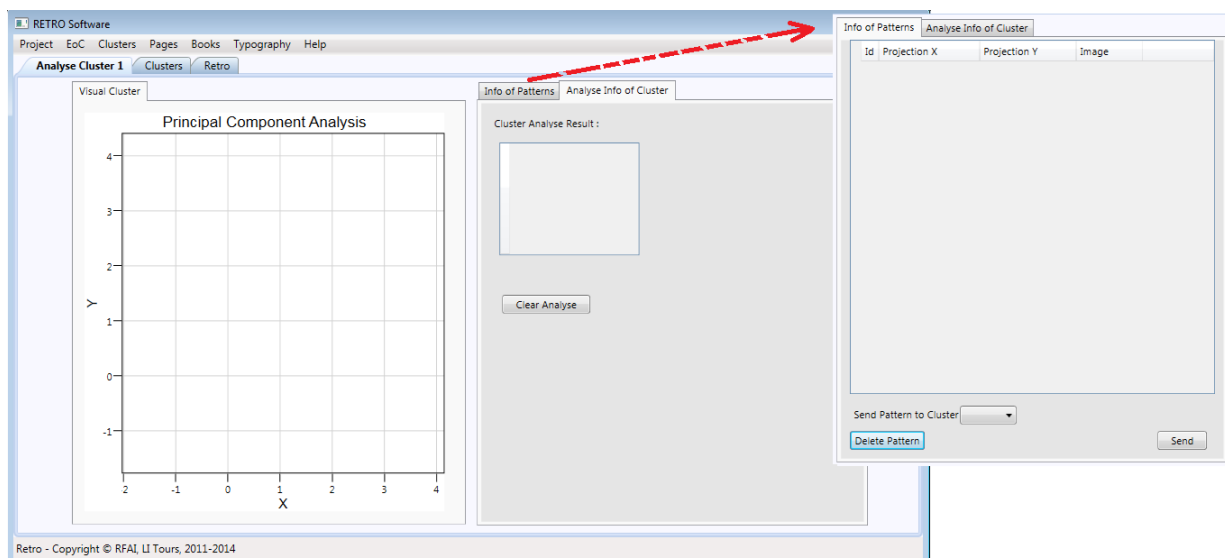


FIGURE 7.5 – Nettoyage le panel de l'Analyse Cluster



### 7.2.3 Libraires et classes correspondants

Les libraires et les classes associés à ce module sont :

— **Sous-projet RetroGUI**

- Répertoire "clustering" : le fichier .xml *AnalyseClusterPanel* est l'IHM d'analyse cluster qui est contrôlé par le viewmodel "AnalyseViewModel".
- Répertoire "util" : la classe *ACPElementPointMarker* est la classe qui présente les points dans le plan ACP. On ajoute des propriétés pour des fonctions supplémentaires comme l'étiquette flottant, action du souris etc.
- Libraire : *DynamicDataDisplay.PointMarkers*, *DynamicDataDisplay.DataSources*, *DynamicDataDisplay.Charts.Navigation*.

— **Sous-projet RetroCore**

- Répertoire "Treatment" : la classe *AnalyseCluster* est la classe qui contient des méthodes de traitement des données d'un cluster , par exemple ACP, traitement dsignature etc.
- Répertoire "ViewModel" : ce répertoire est composé des "ViewModel" qui est une lien entre l'IHM et des classes qui traite des données. On ajoute une nouvelle classe "AnalyseViewModel" pour l'IHM "Analyse Cluster".
- Libraire : *Accord.Statistics.Analysis*, *Accord.Math*, *DynamicDataDisplay.DataSources*, *DynamicDataDisplay.DataSources*, *DynamicDataDisplay.PointMarkers*.

## 7.3 Ajout du module "Auto Transcription"

### 7.3.1 IHM du "Auto Transcription"

En cliquant le menu "Cluster->Auto Transcription", on lance le panel d'auto transcription fig.7.6.

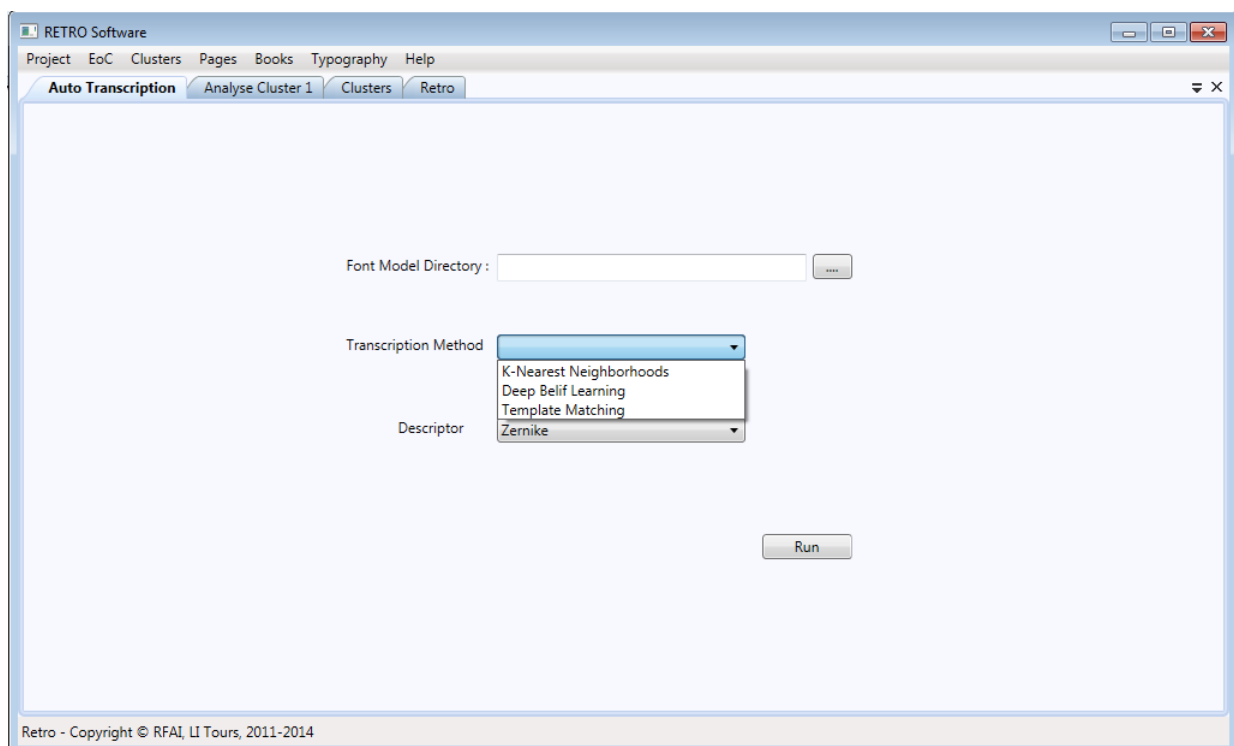


FIGURE 7.6 – Panel d'Auto Transcription

### 7.3.2 Ajout de fonctionnalités

On ajoute deux méthodes d'auto transcription "K-Nearest Neighborhoods" et "Deep Belief Network". Dans le panel "Auto Transcription", on peut choisir une méthode de transcription via le comboBox, une description via le comboBox aussi et mettre le chemin du répertoire du modèle de l'apprentissage existant dans le textBox (voir fig.7.6).

### 7.3.3 Appliquer la méthode de transcription

#### Jeu de modèle

Pour la base d'apprentissage, on applique le répertoire "R80\_Gering\_cicero\_v2" qui contient des modèles "FontModel" qui est composé d'un image originel, un image binaire et un fichier .xml de propriétés du caractère.

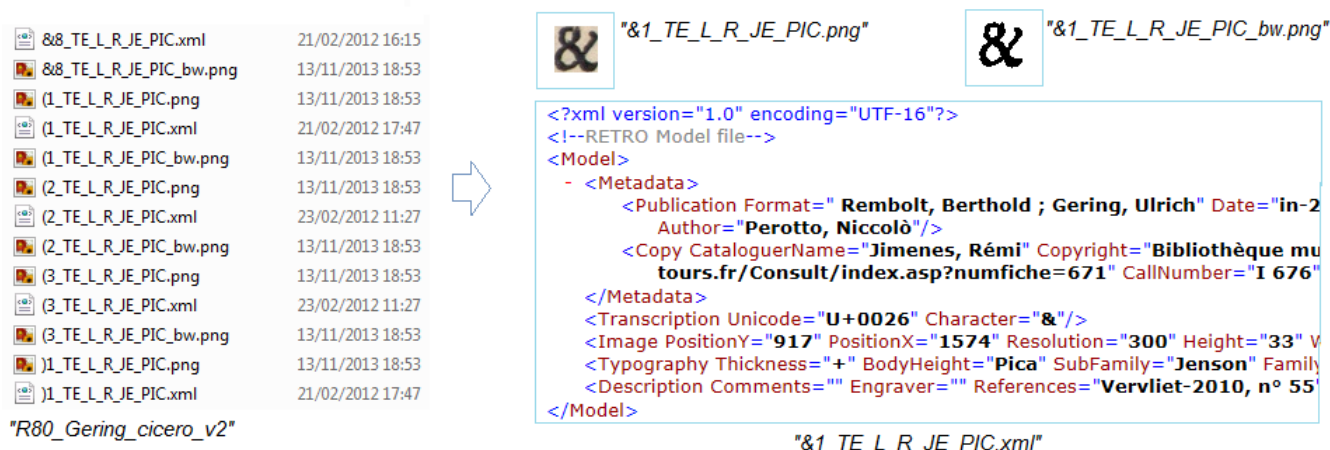


FIGURE 7.7 – Jeu de modèle pour l'apprentissage

#### Test

Après on fait une série de tests avec différente base d'apprentissage et différente base de test pour comparer le performance du "KNNs" et "DBNs". Le résultat du test est présenté dans la tableau suivant tab.7.1 :

Méthode	Base de l'app.	Base du test	Temps de l'app.	Temps du test	Temps total	Reco. %
KNNs	70	5	13047,75ms	22006,26ms	35054,01ms	0.6
	180	20	37418,14ms	35203,01ms	72621,15ms	0.3
	473	70	61945,54ms	26353,51ms	88299,05ms	0.11
DBNs	70	5	19min37s	12910,73ms	19min50s	0.0
	180	20	3h2min	41216,3575ms	3h3min	0.0

TABLE 7.1 – KNNs et DBNs Test

**Note** : pour KNNs le temps de l'apprentissage est le temps de préparation comme extraire des données entrées pour l'algo, sauvegarder des modèles etc. Le nombre de base d'apprentissage est le nombre du caractère à apprendre. Le nombre de base du test est le nombre du cluster à tester.

On peut voir sur le tableau tab.7.1 que KNNs peut reconnaître bien les caractères quand la base d'apprentissage est petite. Avec l'augmentation de la base d'apprentissage, la mesure de distance perd sa supériorité donc le taux de reconnaissance abaisse. L'expérimentation du DBNs est échec car le taux de reconnaissance est zéro. En conséquence de la qualité et la quantité de base d'apprentissage et des faiblesses de l'algorithme, l'effet de transcription automatique n'est pas idéal.

### 7.3.4 Libraires et classes correspondant

Les libraires et les classes associés à ce module sont :

— **Sous-projet RetroGUI**

- Répertoire "transcription" : le fichier .xml *AutoTranscriptionPanel* est l'IHM d'auto transcription qui est contrôlé par le viewmodel "AutoTransViewModel".

— **Sous-projet RetroCore**

- Répertoire "OcrTypo" : on ajoute la classe *DeepLearning* et la classe *KNNs* étant des méthodes du "Machine Learning" implémentées pour le reconnaît des clusters.
- Répertoire "ViewModel" : On ajoute une nouvelle classe "AutoTransViewModel" pour l'IHM "Auto Transcription".
- Libraire : Accord.Neuro, Accord.Statistics.Analysis, Accord.Math, AForge.Neuro, AForge.Imaging.Filters.

Note : un document doxygen du code est fourni pour bien comprendre ou améliorer les classes de ces algorithmes.

## 7.4 Rendus

Pour déployer le logiciel dans d'autre machine, on devra obtenir une version exécutable. Les fichiers nécessaires sont groupés dans le répertoire comme suivant fig.7.10 :

- Toutes les librairie utilisées dans le projet
- le logiciel exécutable
- Les dossiers *XML\_Files* et *Plugins* .
  - *XML\_Files* : il contient des .xml de méthodes du descripteur et du Clustering(voir fig. 7.9).
  - *Plugins* : ce dossier est composé des fichiers .dll pour l'ajout des méthodes du Clustering, du descripteur et du document reader(voir fig. 7.8).

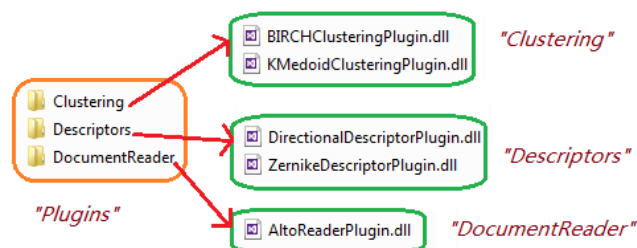


FIGURE 7.8 – Le répertoire du *Plugins*

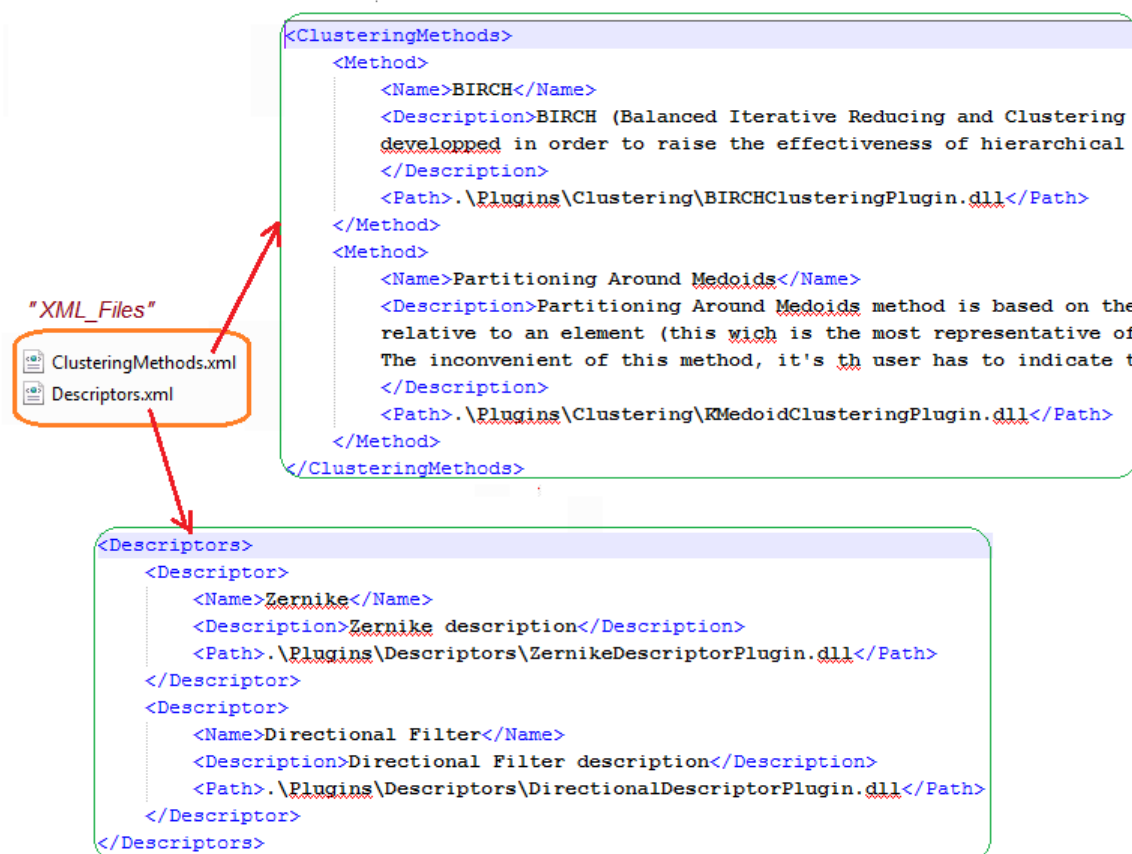


FIGURE 7.9 – Le répertoire du *XML\_Files*

Plugins	03/05/2015 17:04
XML_Files	03/05/2015 17:04
Accord.dll	07/12/2014 19:16
Accord.MachineLearning.dll	07/12/2014 19:18
Accord.Math.dll	07/12/2014 21:02
Accord.Neuro.dll	06/12/2014 15:48
Accord.Statistics.dll	07/12/2014 19:18
AForge.dll	16/07/2013 00:04
AForge.Imaging.dll	16/07/2013 00:04
AForge.Math.dll	16/07/2013 00:04
AForge.Neuro.dll	16/07/2013 00:04
AvalonDock.dll	11/05/2014 18:33
AvalonDock.Themes.dll	11/05/2014 18:33
Clustering.dll	11/05/2014 18:33
DynamicDataDisplay.dll	27/04/2009 23:42
Plugin.dll	01/05/2015 15:50
Retro2014.exe	03/05/2015 12:02
RetroCore.dll	03/05/2015 12:02
RetroLib.dll	11/05/2014 18:33
RetroUtil.dll	03/05/2015 12:02
WPFToolkit.dll	02/03/2010 11:09

FIGURE 7.10 – Le répertoire exécutable

# Gestion de projet

## 8.1 Suivi du projet

Afin d'établir un suivi, un projet a été créé sous Redmine Polytech'Tours en utilisant le gestionnaire de version SVN. Ainsi tous les documents, sources de test et des codes du PFE sont disponibles à tout public et l'application est sous licence GNU.

Le projet sous GitHub peut être téléchargé à l'adresse[9] : <https://github.com/CHENJing88/PFE.git>  
Un démo d'utilisation est sur YouTube [10] : <https://www.youtube.com/watch?v=G-1IC3VK5y0>

## 8.2 Planning

Le figure suivant fig.8.1 présente le diagramme de Gantt prévisionnel, établi au commencement du projet.

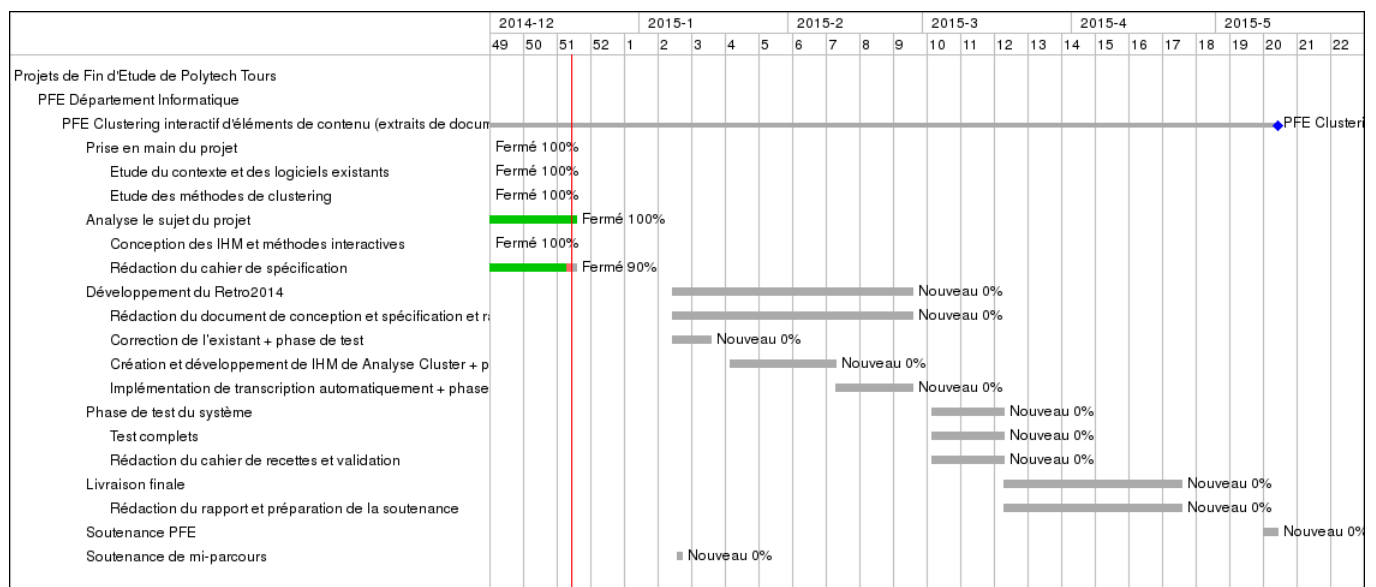


FIGURE 8.1 – Diagramme de Gantt prévisionnel

La figure 8.2 présente le diagramme de Gantt réel tel qu'il a été suivi tout au long de la réalisation du projet.

**Le déroulement** On gère le projet en utilisant la méthode Agile, donc je discute régulièrement avec M. Ramel pour faire des changements de tâche et rendre des comptes rendus pour des études. Par rapport à la gestion du projet, j'ai utilisé le Redmine Polytech'Tours au début après je change au Git qui est une plateforme open source. Il est plus pratique pour partager et mettre à jour le projet au bureau.

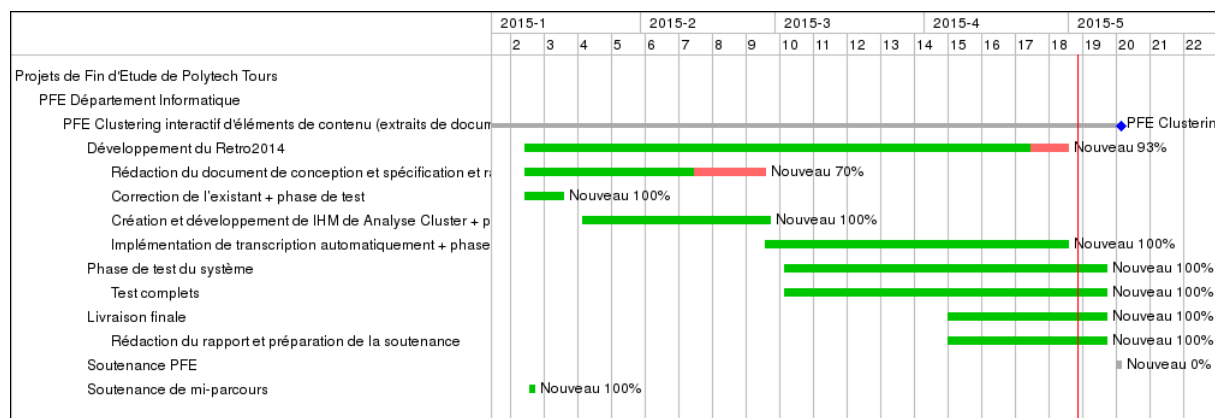


FIGURE 8.2 – Gant réel

**Le changement** Comparant avec le diagramme prévisionnel, le planning prévu a été respecté jusqu'au janvier 2015. A partir de ce moment là, quelques changements ont été effectués. D'abord on a changé quelques fonctionnalités pendant le développement en concernant l'utilisation de l'utilisateur et ajouter certaines fonctionnalités nécessaires. Après on annule la rédaction du rapport du cahier de recette et de validation, car c'est pas nécessaire.

**Le retard** Au niveau du temps de retard, le développement du transcription automatique est en retard un mois à cause de l'algorithme qu'on veut implémenter est trop difficile à comprendre et mis en place. On perd le temps sur l'étude de l'algorithme DBNs et l'implémentations. Pour gagner le temps j'avais commencé de rédiger le rapport du projet en le même temps.

En conséquence de peu d'expériences de gestion un grand projet du 4/6 mois, j'ai pas prévu des difficultés et la complexité du développement d'un grand logiciel. De ce côté là, ce projet m'a permis de faire plus attentions de suivre le diagramme prévu et de faire le changement à temps en face de difficulté.

# Conclusion

---

Ce PFE a été l'occasion d'un projet sur une longue période. Pendant le développement, on fait un peu de rajustement considérant des manipulations pratiques, donc il a un peu de différences avec le cahier de spécification qui est pour bien maîtriser le projet et de commencer le développement.

La difficulté d'abord rencontrée du PFE était de trouver une façon d'affichage la visuelle projection du Cluster et d'ajouter des fonctionnalités intersection. En appliquant la librairie 3D, c'est pas facile de trouver une classe existante pour réaliser un effet qu'on souhaite. Donc j'essaie plusieurs façons enfin de réaliser des fonctionnalités prévues.

Les autres difficultés étaient de comprendre et implémenter l'algorithme DBNs et d'améliorer le complexe du temps d'application. A cause la méthode d'apprentissage en fondeur proposé en récente années et pleins d'articles font d'études sur cela, je veux essayer d'appliquer cette méthode à notre cas du clustering bien que l'effet n'arrive pas comme prévu.

Le développement d'application est dans la période débutant, donc il mieux de regrouper ou déplacer les classes du sous-projet pour établir une architecture plus sain afin de réduire des changement coûteux en période médiale ou finale, par exemple on peut intégrer des méthodes d'outils dans un répertoire ou un sous-projet. Quant à la transcription automatique, l'algorithme du DBNs a besoin d'améliorer encore.

A la fin, j'ai plus d'expériences de développer un logiciel un peu complexe et rends compte un processus complètement du développement. Ce projet aussi m'a permis d'apprendre l'importance de la gestion du projet et l'organisation soi-même.

# Bibliographie

---

- [1] Framework accord.net. <http://accord-framework.net/>.
- [2] 3d-dynamic data display. <http://dynamicdatadisplay.codeplex.com/>.
- [3] Paradiit. <https://sites.google.com/site/paradiitproject>.
- [4] Bilinear deep belief network. [http://www4.comp.polyu.edu.hk/~csshzhong/Bilinear\\_Deep\\_Belief\\_Network.html](http://www4.comp.polyu.edu.hk/~csshzhong/Bilinear_Deep_Belief_Network.html).
- [5] The mnist database of handwritten digits. <http://yann.lecun.com/exdb/mnist/>.
- [6] Universite Pierre et Marie Curie Paris VI Hanlin Goh. Learning deep visual representations. *HAL*, 2013.
- [7] Accord samples gallery. <http://accord-framework.net/samples.html>.
- [8] Neural network ocr. <http://www.codeproject.com/Articles/11285/Neural-Network-OCR>.
- [9] Github source du projet. <https://github.com/CHENJing88/PFE.git>.
- [10] Youtube démonstration de l'utilisation de retro2014. <https://github.com/CHENJing88/PFE.git>.





# Clustering interactif d'éléments de contenu(extraits de documents)

---

Rapport de Projet de fin d'études 2014

**Résumé :** Le premier objectif de ce projet est mis en place la visualisation et l'optimisation des clusters. Le deuxième objectif est le développement d'une méthode de reconnaissance automatiquement de clusters. On effectue deux méthodes pendant le projet, une est l'apprentissage en profondeur(DBN) avec Restrictive Boltzman machines(RBM), l'autre est K-Plus proche voisins(KPPV)

**Mots clefs :** Projet de fin d'etude(PFE), Retro, Analyse en composantes principales(ACP), Apprentissage en profondeur,KPlus proche voisins(KPPV)

**Abstract:** The first objective of this final year project is to effectuate the visual data (for a cluster) and improvement the quality of the data(a cluster). We develop two methods of transcription automate: Deep Belief Networks(DBNs) with Restrictive Boltzman machines(RBM) and KNearest Neighborhoods(KNNs)

**Keywords:** Final-year project,Retro, Principle Component Analysis(PCA), Machine Learning, Deep Belief Networks(DBNs), KNearest Neighborhoods(KNNs)

---

## Encadrant

Jean-Yves Ramel  
[jean-yves.ramel@univ-tours.fr](mailto:jean-yves.ramel@univ-tours.fr)

Université François-Rabelais, Tours

## Étudiant

Jing CHEN  
[jing.chen-2@etu.univ-tours.fr](mailto:jing.chen-2@etu.univ-tours.fr)

DI5 2014 - 2015