INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Kévin DESPOULAINS
Gaël GENDRON
Corentin GUILLOUX
Valentin FOUCHER
Enzo CRANCE
Charlotte RICHARD
Laure DU MESNILDOT
Timothée NEITTHOFFER

Élèves ingénieurs de l'INSA Rennes

Année universitaire 2018/2019

Rapport de spécification fonctionnelle

Projet 4INFO

Logiciel de génération de données d'apprentissage pour la reconnaissance d'écriture manuscrite

Encadrants

Bertrand COÜASNON (*IRISA*)

Erwan FOUCHÉ & Julien BOUVET (*Sopra Steria*)

Projet en collaboration avec

Jean-Yves LE CLERC (*Archives départementales*)

Sophie TARDIVEL (*Doptim*)

Table des matières

Chapitre 1

Introduction

Ce projet nous a été proposé par l'équipe IntuiDoc de l'IRISA, en étroite collaboration avec la startup Doptim et avec le soutien de Jean-Yves LE CLERC, conservateur du patrimoine aux archives départementales d'Ille-et-Vilaine. Tout au long de l'année, nous serons encadrés par Bertrand COÜASNON, enseignant-chercheur membre d'IntuiDoc, Erwan FOUCHÉ, chef de projet chez Sopra Steria, Julien BOUVET, ingénieur chez Sopra Steria également. Nous serons aussi accompagnés par Sophie TARDIVEL, responsable et *data scientist* chez Doptim.

Ce projet a pour but de fournir un programme permettant de concevoir des bases d'apprentissage automatiquement pour l'entraînement de divers systèmes de reconnaissance d'écriture manuscrite ainsi que leur exploitation. Ces reconnaisseurs seront, par exemple, capables de retranscrire de manière informatique des documents manuscrits (registres paroissiaux, registres d'état civil, documents d'entreprise, ...) pour les rendre plus exploitables. Ce projet permettra donc de gagner du temps sur la compréhension de documents anciens en rendant l'entraînement de systèmes complexes plus simple.

Dans ce rapport, nous définirons premièrement les spécifications générales du projet, ainsi que celles de ses éléments. Nous détaillerons donc les spécifications concernant la préparation des données en entrée, le stockage dans une base de données, les liens avec un système de reconnaissance d'écriture manuscrite, et enfin l'interface utilisateur. Nous expliquerons ensuite l'architecture logicielle du projet, la façon dont les éléments précédemment cités seront construits et liés ensemble. Enfin, nous conclurons en abordant la suite du projet.

Chapitre 2

Spécifications fonctionnelles

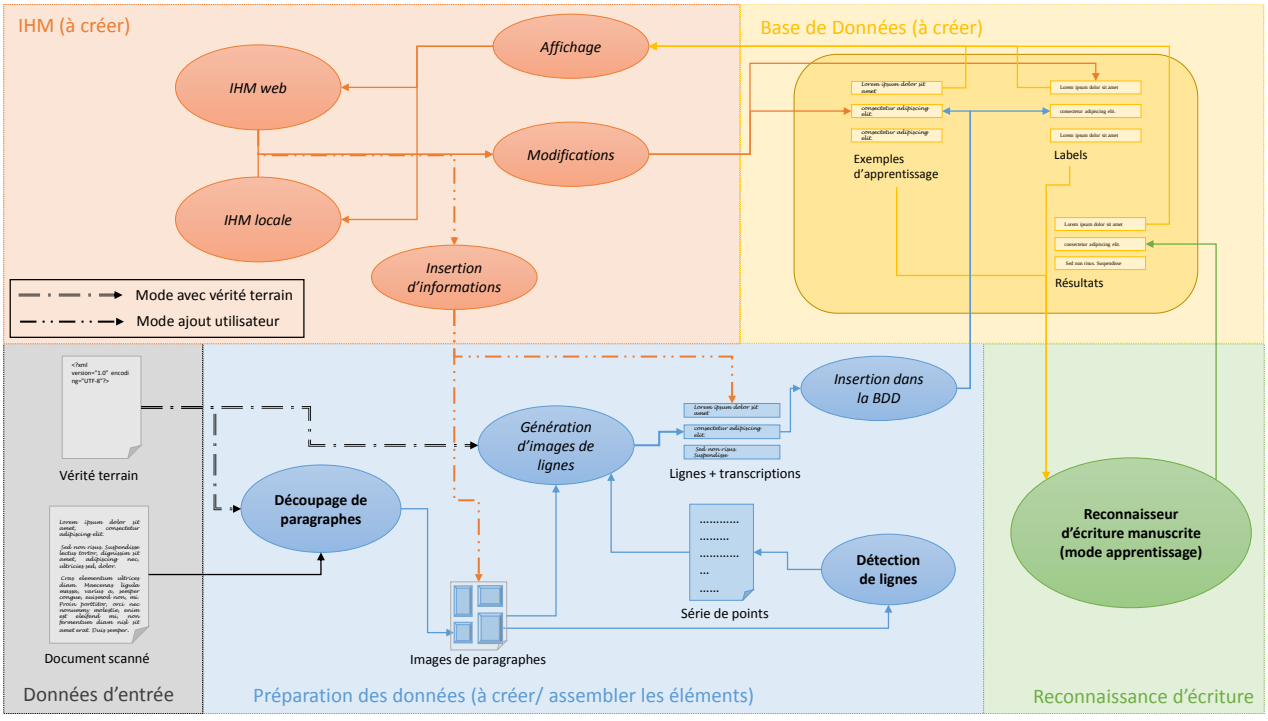
2.1 Spécifications générales

2.1.1 Aspect général du projet

Le projet se découpe en quatre grandes parties qui sont :
- la préparation des données ;
- le stockage des données dans une BDD ;
- l'interface avec le reconnaisseur ;
- l'IHM.

Ces parties sont résumées sur le schéma 1, déjà abordé dans le rapport de pré-étude.

Figure 1 : Décomposition du projet (mode apprentissage)



Le projet se présente sous la forme d'un programme, principalement écrit en Scala. Par la suite tous les éléments décrits seront considérés comme écrits en Scala et pour toute exception le langage utilisé sera précisé.

La partie découpe de documents prend en entrée des documents scannés contenant de l'écriture manuscrit, en extrait les informations contenues sous la forme de couples imagette-transcription et les stocke dans la BDD.

Le Framework de Base de Donnée utilisé est codé en Java. Les langages Scala et Java étant interopérables car tournant tous deux sur le JVM, notre code Scala est donc compatible avec ce Framework. La base de données est écrite en Java. Scala, étant exécuté sur la JVM, est compatible avec elle.

Ces informations stockées sont mises à disposition d'un classifieur pour qu'il puisse apprendre dessus. Ce classifieur ne fait pas partie du cadre de l'application, mais vient se greffer à celle-ci. Le programme doit donc faire en sorte que les données de la BDD puissent être accessibles par le classifieur. Le classifieur n'étant pas intégré au programme, il peut être changé à tout moment. L'application doit néanmoins pouvoir le faire fonctionner en 3 modes : apprentissage, évaluation et production. Les données stockées dans la base sont ainsi divisées en sous-ensembles, chacun correspondant à un mode de fonctionnement.

Le tout est géré par l'utilisateur grâce à une interface Web, en JavaScript pour une plus grande accessibilité et une meilleure portabilité. Ce choix a également pour but de faciliter l'évolution de l'application vers une solution multi-utilisateurs distants, où plusieurs personnes s'y connecteraient via internet. Chaque partie sera expliquée plus en détails dans la suite du rapport.

2.1.2 Côté utilisateur

Le logiciel à créer fonctionnera donc suivant plusieurs modes de fonctionnement :
- apprentissage ;
- évaluation ;
- production.

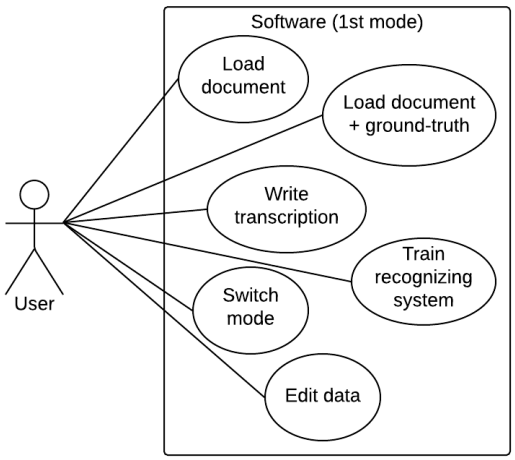
Les trois modes ne sont pas à implémenter avec la même priorité, seul le mode apprentissage est nécessaire pour obtenir un résultat fonctionnel. Les deux modes suivants sont à implémenter suivant le temps restant.

Le mode apprentissage a pour objectif de faire apprendre un système de reconnaissance d'écriture. Le logiciel génère une base d'apprentissage pour un système de reconnaissance d'écriture manuscrite et fournit les données produites en entrée du reconnaisseur. Celui-ci peut alors les utiliser pour apprendre à retranscrire correctement du texte.

Du point de vue de l'utilisateur du logiciel, par l'intermédiaire de l'IHM il peut :

1. charger un document dans la base de données, accompagné d'une vérité-terrain ;
2. charger un document dans la base sans vérité-terrain ;
3. annoter lesdits document pour leur associer une nouvelle transcription ;
4. lancer l'apprentissage du système de reconnaissance ;
5. modifier les données contenues dans la base :
   (a) supprimer un document ;
   (b) déplacer un document vers un autre sous-ensemble.
6. changer de mode de fonctionnement.

Figure 2 : Diagramme de cas d'utilisation (mode apprentissage)

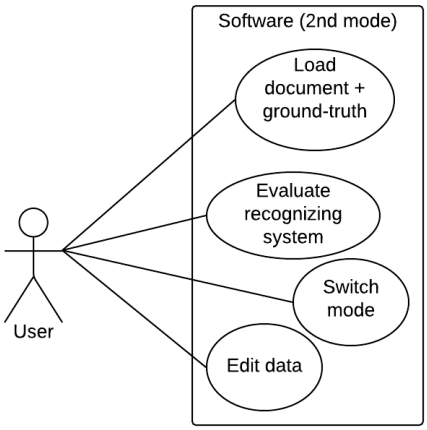


Le mode évaluation a pour objectif d'évaluer un système de reconnaissance, ce mode est en général utilisé après avoir effectué un apprentissage avec le reconnaisseur. Il permet de vérifier l'efficacité de l'apprentissage effectué.

Dans ce mode, toujours par l'intermédiaire de l'IHM, l'utilisateur peut :

1. charger un document, accompagné d'une vérité-terrain (la vérité-terrain est obligatoire dans ce mode de fonctionnement, pour que l'évaluation soit correcte) ;
2. lancer l'évaluation du système de reconnaissance ;
3. modifier les données contenues dans la base :
   (a) supprimer un document ;
   (b) déplacer un document vers un autre sous-ensemble.
4. changer de mode de fonctionnement.

Figure 3 : Diagramme de cas d'utilisation (mode évaluation)

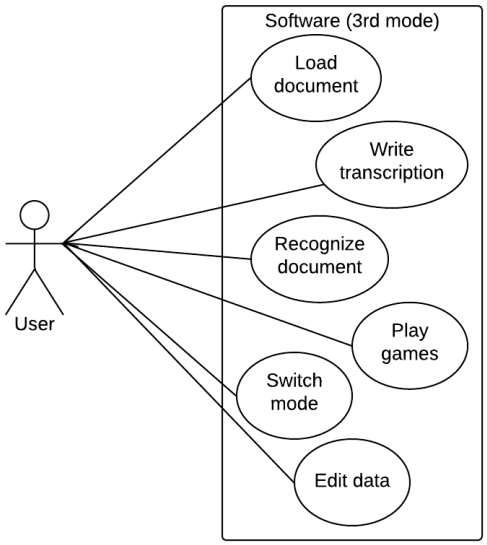


Le mode production a pour objectif d'utiliser un système de reconnaissance afin d'obtenir une transcription pour des documents n'en ayant pas. Ce mode est en général utilisé sur un reconnaisseur entraîné, dont l'apprentissage a été évalué, et ayant donc obtenu de bons résultats. Il s'agit également d'un mode de démonstration, permettant d'utiliser le reconnaisseur de manière ludique. Il pourra être utilisé lors de la démonstration de fin de projet.

Dans ce mode, via l'intermédiaire de l'IHM, l'utilisateur peut :

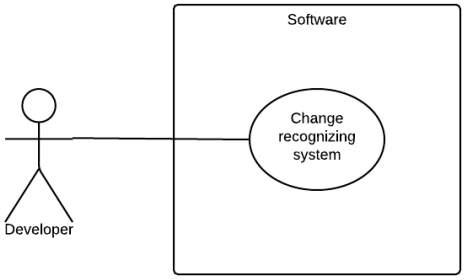
1. charger un document sans vérité-terrain ;
2. écrire une transcription manuellement, pour compléter ou corriger la transcription du reconnaisseur ;
3. jouer à des mini-jeux en utilisant le reconnaisseur (le contenu des mini-jeux n'a pas encore été établi, cette option étant annexe au projet) ;
4. modifier les données contenues dans la base :
   (a) supprimer un document ;
   (b) déplacer un document vers un autre sous-ensemble.
5. changer de mode de fonctionnement.

Figure 4 : Diagramme de cas d'utilisation (mode production)



Le logiciel ne possède pas de système de reconnaissance d'écriture intégré. Cette partie, comme précisé précédemment, y est simplement greffée. Il n'est pas prévu de pouvoir changer de reconnaisseur via l'interface graphique, cette opération étant rare et pouvant même ne jamais avoir lieu au cours de la vie du logiciel. Néanmoins, il doit être possible pour un développeur de système de reconnaissance d'écriture manuscrite utilisant notre logiciel de changer le reconnaisseur. Cette possibilité sera donc incluse dans le projet.

Figure 5 : Diagramme de cas d'utilisation (pour développeur)



2.2 Préparation des données

Cette partie du projet a pour but de réaliser un premier traitement sur les données d'entrée. Ces données sont aux formats GEDI et PiFF, présentés dans le dernier rapport. Nous devons fournir un logiciel qui soit capable de traiter ces formats (**PR_FO_1** et **PR_FO_2**). Nous avons validé la proposition que nous avions faite, qui était de choisir PiFF comme format de traitement unique. Nous fournirons donc un convertisseur de GEDI vers PiFF, et nous garantirons par modélisation logicielle la possibilité pour les futurs utilisateurs d'écrire d'autres convertisseurs des formats de leur choix vers PiFF s'ils le souhaitent (**GEN_EVOL**).

En exploitant le fichier PiFF, ce module du projet génèrera les imagettes qui formeront les exemples. Les coordonnées utilisées pour la découpe d'image seront calculées grâce au polygone identifiant le paragraphe présent dans le fichier d'entrée, ainsi que les détecteur de lignes. Cette découpe pourra être réalisée soit en lignes, soit en paragraphes, ce qui correspond à deux reconnaisseurs différents à utiliser par la suite. Par ailleurs, les fonctions de manipulation d'images utiliseront la bibliothèque graphique OpenCV, que nous avions déjà sélectionnée auparavant. Les deux options seront disponibles sur le logiciel.

Nous devrons récupérer la vérité terrain si elle existe dans le fichier d'entrée, afin de pouvoir l'afficher sur l'IHM et permettre à l'utilisateur de la corriger ainsi qu'au reconnaisseur de s'entraîner avec, en lui fournissant des exemples que notre logiciel aura créés en associant les imagettes à leur transcription.

2.3 Base de données

2.4 Interface avec le reconnaisseur

2.5 Interface Homme-Machine

Chapitre 3

Architecture logicielle

3.1 Description des modules

3.2 Interactions

Chapitre 4

Organisation du travail

4.1 Organisation du travail

Figure N : LOT 1 - X

Règles	Description
XX_1	Règle XX_1

Formats PR_FO_1 : Permettre de traiter le format PiFF PR_FO_2 : Permettre de traiter le format GEDI Traitement PR_TR_1 : Fonction de détection de lignes intégrée au logiciel PR_TR_2 : Permettre un découpage des images en lignes PR_TR_3 : Localiser les paragraphes PR_TR_4 : Permettre un découpage des images en paragraphes Représentation des données PR_RE_1 : Associer les images à leur transcription PR_RE_2 : Associer la vérité terrain à une transcription PR_RE_3 : Permettre de générer une vérité terrain si besoin, grâce à un reconnaisseur

GEN_ERGO : Ergonomie du logiciel GEN_EVOL : Logiciel évolutif

Chapitre 5

Conclusion

INSA Rennes
20 Avenue des Buttes de Coësmes
CS 70839
35708 Rennes Cedex 7

Tél. +33 (0) 2 23 23 82 00
Fax +33 (0) 2 23 23 83 96

www.insa-rennes.fr

INSA

UNIVERSITE BRETAGNE LOIRE

Cti Commission des Titres d'Ingénieur

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR, DE LA RECHERCHE ET DE L'INNOVATION