UNIVERSITÉ DE NANTES

Composition d'un Livre d'Heures en français, et application des outils de reconnaissance de textes de l'état de l'art

par Baptiste Auffray, Samy Gascoin-Fontaine, Félix Pinel
Encadrantes : Béatrice Daille, Christine Jacquin

24 mai 2018

Résumé

Ce rapport a pour objectif de détailler et expliquer le travail réalisé durant ce travail encadré de recherche. Ce travail porte sur la composition d'un Livre d'Heures en français. Les Livres d'Heures sont des écrits liturgiques du Moyen-Age très populaires à leur époque, ce qui leur a permis de nous parvenir en grand nombre aujourd'hui. Ces livres sont une source importante d'informations sur la vie du Moyen-Age, leur étude est donc primordiale, mais complexe. C'est dans ce but que le projet HORAE a été créé afin de faciliter l'étude des Livres d'Heures en utilisant les technologies de traitement automatique des langues. En passant par l'étude des types de textes constituant un Livre d'Heures, l'analyse de la constitution et des proportions des textes d'un livre ainsi que la récolte des textes utilisés dans les livres, l'objectif est de créer un programme de génération d'un corpus de texte en français. Il doit être similaire à un Livre d'Heures et doit permettre d'étudier les performances des algorithmes de l'état de l'art en traitement du langage afin d'en récupérer la structure d'un véritable Livre d'Heures. Le travail que nous avons réalisé est accessible sur ce lien Github : https://github.com/Tee-veeNovae/GenerationCorpus_HORAE_TER.

Table des matières

1 Introduction **4**

2 Qu'est ce qu'un Livre d'Heures ? **4**
2.1 Les Livres d'Heures au Moyen-Age 4
2.2 La structure d'un Livre d'Heures 4

3 Trouver les textes sources **5**
3.1 Sources de recherches 5
3.2 Documentation des types de textes 5
3.3 Récupération des textes 6

4 Analyse et modélisation d'un Livre d'Heures **8**

5 Génération de corpus d'un Livre d'Heures **9**
5.1 Génération semi-aléatoire 9
5.2 Génération par le XML TOC_Arsenal 9
5.3 Génération par le Livre d'Heures traduit 10
5.4 Méthode sélectionnée et travail réalisé 10
5.4.1 Classe parserXML, exploration du XML et génération du corpus . . 10
5.4.2 Programme de récupération des textes pour la génération 11

6 Application d'algorithmes d'état de l'art sur le corpus **13**

7 Conclusion **13**

A Annexes **14**
A.1 Abréviations des types de textes 14
A.2 Proportions Heures de la Vierge 15
A.3 Proportions Heures de la croix 15
A.4 Proportions Heures du saint-esprit 16
A.5 Proportions Heures des morts 16
A.6 Proportions Suffrages 17

Remerciements

Nous tenons dans un premier temps à exprimer notre gratitude à Béatrice Daille et Christine Jacquin, pour avoir pris le temps de nous écouter, répondre à nos questions et nous orienter dans notre travail.

1 Introduction

Ce projet de Travail Encadré de Recherche (TER) s'est effectué en lien avec le projet de recherche Hours - Recognition, Analysis, Editions (HORAE)[IRH], un projet de l'Agence Nationale de la Recherche (ANR).[dR] HORAE est multidisciplinaire, il regroupe le travail de l'Institut de Recherche et d'Histoire des Textes (IRHT), de Teklia et du Laboratoire des Sciences du Numérique de Nantes (LS2N). Il a pour but d'étudier les pratiques religieuses du milieu du Moyen-Age à travers des Livres d'Heures. Le LS2N s'investit dans ce projet, particulièrement au niveau du traitement automatique des langues.

Le but de ce TER était de générer un corpus de textes en français, s'apparentant à la composition textuelle d'authentiques Livres d'Heures. Ceci a nécessité de reconnaître et analyser l'architecture de différents Livres d'Heures afin d'en extraire une architecture générale ainsi que les types de textes utilisés. Une fois un corpus créé, la deuxième étape consistait à trouver et étudier des algorithmes de traitement du langage et les appliquer sur le corpus afin de récupérer automatiquement la structure d'un Livre d'Heures. Il s'agissait alors de comparer les performances de ces algorithmes sur le corpus en lui faisant subir des dégradations artificielles. Ce rapport fait l'inventaire du travail réalisé durant ce TER afin de comprendre ce qui a été fait et ainsi faciliter sa reprise. Tout le matériel et le travail réalisé sont accessibles sur ce lien github : https://github.com/Tee-veeNovae/GenerationCorpus_HORAE_TER.

2 Qu'est ce qu'un Livre d'Heures ?

2.1 Les Livres d'Heures au Moyen-Age

Les Livres d'Heures sont des ouvrages médiévaux liturgiques destinés aux catholiques pratiquants afin qu'ils puissent faire leurs prières quotidiennes au cours des heures canoniales de la journée, soit environ huit fois par jour. Il s'agissait d'un rassemblement de textes appropriés aux heures liturgiques. On voit apparaître ce genre d'ouvrage en France et aux Pays-Bas dans le courant du XIIIe siècle, succédant au psautier et il devient populaire aux alentours du XVe siècle.

Cette popularité a fait qu'aujourd'hui, les Livres d'Heures sont les livres du Moyen-Age qui sont le plus parvenus jusqu'à nous. Comme ils sont enluminés, ils constituent une source importante d'informations sur la vie et sur la chrétienté du Moyen-Age. Des exemples de Livres d'Heures français connus de nos jours incluent le Livre d'Heures d'Étienne Chevalier et les Très Riches Heures du duc de Berry. [Wik][Paw]

2.2 La structure d'un Livre d'Heures

Il existe d'importantes différences régionales et temporelles entre les Livres d'Heures, mais certaines sections sont presque toujours présentes, d'autres plus régulières. Les deux ouvrages donnés en exemple, qui sont pourtant de la même période et tous les deux français, comportent des différences importantes. Les sections que l'on retrouvera dans tout Livre d'Heures comportent les heures de la Vierge, la section la plus importante de l'ouvrage. Cette section est composée de psaumes, de cantiques et d'hymnes, liés au moins partiellement à la Vierge. Ces textes sont eux-mêmes séparés par des versets et des répons. On retrouve dans les Livres d'Heures également les psaumes pénitentiels, section dont les textes avaient pour but de demander le pardon pour les péchés. Vers la fin de l'ouvrage vient l'office des morts, contenant des prières cléricales écrites à l'occasion de funérailles, et servant au laïc à commémorer la mort de cette personne.

Au cours du XVème siècle, deux parties sont devenues un standard dans la composition des Livre d'Heures : le calendrier et le suffrage des saints. Le calendrier se trouve au début de l'ouvrage et permet au fidèle d'avoir connaissance des fêtes religieuses et des saints du jour. Le suffrage des saints se trouve typiquement en fin d'ouvrage, il est constitué de prières et de suppliques à l'intention de saints.

Dans le texte en XML qui nous a servi d'outil de compréhension des Livres d'Heures, les sections sont très semblables à celles des ouvrages que l'on a pris en exemple précédemment [ref du xml arsenal]. Le principal ajout est l'introduction de deux nouvelles heures : à côté des longues heures de la Vierge se trouvent également les heures du Saint-Esprit et les heures de la Croix. Chacune de ces heures, et l'office des morts (appelé parfois 'les heures des morts') sont divisées selon les heures de prières cléricales. Celles-ci se répartissent tout le long de la journée. Elles commencent à l'aube avec les Laudes, puis Prime, Tierce, Sexte, None, Vêpres, les complies à l'heure du coucher, et enfin les matines au cours de la nuit.

3 Trouver les textes sources

3.1 Sources de recherches

Afin de nous aider dans nos recherches sur les livres, plusieurs documents nous ont été fournis. Les premiers documents sont des fichiers expliquant les objectifs et les résultats du projet BOH. Ils regroupent aussi les différentes heures d'un Livre d'Heures. Le deuxième document est une arborescence XML, elle représente un Livre d'Heures en Latin structuré pour correspondre à celle du livre en question. Ce travail a été réalisé manuellement par Dominique Stutzmann, travaillant pour l'IRHT.

Les premiers documents nous ont permis de lister les types de textes les plus communs qui sont présents dans un Livre d'Heures. Aussi, nous avons pu construire des proportions de présence des types de textes dans un Livre d'Heures. Ces informations seront utiles par la suite pour la génération du corpus de textes. Le document XML nous a permis, quant à lui, de confirmer et approfondir les types de textes utilisés dans les Livres d'Heures, ainsi que développer une technique de génération de corpus.

Nos recherches sur des transcriptions de Livre d'Heures sur internet nous ont menés à un Livre d'Heures entièrement retranscrit en français. Cette retranscription est très détaillée, on y observe la page source de chaque traduction ainsi que des descriptions des différentes illustrations et enluminures.[dB]

3.2 Documentation des types de textes

Afin de savoir où rechercher des textes intéressants pour la création de notre corpus, nous avons créé une documentation sur les différents types de textes qui peuvent apparaître dans un Livre d'Heures. Après avoir étudié les documents fournis précédemment, nous avons extrait 11 types de textes significatifs dans un Livre d'Heures, que nous allons détailler ci-près [Leb] :

- **Antienne *(Antiphona)* :** Il s'agit d'un texte court, entre une et deux lignes. Placé autour d'un texte central qui peut être un psaume, un groupe de psaumes ou un cantique. Le début de l'antienne est placé avant le texte central puis l'antienne est remise en entier après le texte central.[cl]

- **Absolution et Bénédiction :** Texte lu avant les leçons de matines.

- **Cantique *(Canticum)* :** Ce sont des extraits de textes tiré de la bible, on les utilise dans les Livres d'Heures de la même manière que des psaumes.

- **Capitule *(Capitulum)* :** Chaque heure, sauf matines, est doté d'un capitule. Il s'agit d'un extrait d'un chapitre de la Bible qui sert de lecture courte.

- **Invitatoire *(Invitatorium)* :** Il s'agit d'un verset d'un psaume de la bible qui introduit les matines, celui-ci invitant à la prière. L'un des invitatoires les plus courants est le psaume 94 mais les psaumes 66, 99 et 23 peuvent aussi être choisis.[es]

— **Leçon ou Lecture *(Lectio)* :** Ce sont des extraits de l'Écriture qui se disent à matines.

— **Oraison *(Oratio)* :** La structure d'une oraison comporte habituellement une invocation à Dieu, un ou plusieurs considérant, une demande proprement dite, un but ou une finalité exprimée et une conclusion. On la retrouve à chaque heure sauf les matines.

— **Psaumes *(Psalmus)* :** Ils proviennent de la Bible et sont regroupés dans le chapitre des Psaumes au nombre de 150. Il s'agit de textes poétiques composés de versets.

— **Répons *(Responsorium)* :** Le répons est un court chant de méditation après une lecture, pour l'essentiel, il est emprunté à l'Écriture, surtout les versets des Psaumes.

— **Verset ou Versicule *(Versus)* :** Petite phrase, habituellement empruntée aux versets de Psaumes, appelant une réponse (souvent un répons).

— ***Obsecro te* :** Il s'agit d'une supplication à la Vierge, afin de recevoir son assistance au moment de la mort. Cette prière est dite en général au début d'un Livre d'Heures.

Nous avons pu observer par ces recherches que la majeure partie des textes constituant les Livres d'Heures sont extraits de la Bible, il est donc essentiel de récupérer ces textes de la Bible pour la création du corpus. Pour les textes ne provenant pas de la Bible (comme pour l'*Obsecro te*), nous avons recherché des sources qui peuvent correspondre au type de texte. Des abréviations pour chaque type de textes ont été mis en place et disponible en A.1

3.3 Récupération des textes

Le but de notre sujet étant de générer un corpus de texte similaire à celui d'un Livre d'Heures, nous avons dû trouver des sources de textes liturgiques de formes ressemblantes à celles de nos différents types de textes présents dans un Livre d'Heures dans lesquels nous pouvions puiser afin de recréer un corpus en français de la façon la plus convaincante possible.

Les Livres d'Heures étant composés d'extraits de la Bible ou de courts textes semblables à des versets, nous avons essayé de trouver un exemplaire de la Bible dans un format facilement exploitable et légal.

La version dans un format le plus utilisable que nous avons pu trouver se trouve sur le site bible-en-ligne.net, qui présente la Bible sous forme de pages HTML statiques avec du texte brut pour chaque chapitre de chaque livre, à l'inverse par exemple de versions au format .pdf ou d'applications web de lecture et navigation de la Bible avec du texte formaté et annoté, qui présentent du bruit lors de l'extraction de texte (espaces et retours à la ligne en trop, annotations, problèmes d'encodage...).

Nous avons donc extrait le texte de chaque page de bible-en-ligne.net grâce à l'outil Boilerpipe, qui permet de détecter le contenu "principal" d'une page HTML et/ou d'en extraire le texte. Nous avons ainsi pu récupérer le contenu de chaque chapitre de la bible dans des fichiers texte différents nommés par numéro de chapitre dans des dossiers nommés par livre, mais le texte obtenu présentait toujours quelques problèmes de présentation, comme la numérotation classique de chaque verset en début de ligne (provenant de la présentation sur le site) ou parfois (pas toujours !) des permaliens vers les versets ou les autres chapitres du livre. Nous avons donc dû passer par une autre étape de nettoyage afin d'obtenir un format de données parfaitement utilisable de notre côté.

Nous avons développé un script Python qui traite les différents problèmes constatés et ne conserve dans les fichiers finaux qu'un verset d'un chapitre d'un livre donné

par ligne. Ainsi, pour récupérer un verset particulier, par exemple Exode 4 :20, il n'y a qu'à lire la ligne 20 du fichier /exode/4.txt (à noter que les dossiers sont nommés suivant la nomenclature dans les URLs de bible-en-ligne.net, en une chaîne de caractères minuscules, et en comprenant les erreurs du site, comme "agee" et "esaie").

Ce script de Python de traitement, textextracting.py, parcourt récursivement les fichiers du dossier textes_sources/bible-en-ligne/ et les ouvre un par un en lecture. Il ouvre ensuite en écriture le fichier homonyme dans le dossier textes_sources/bible_txt et y écrit chaque caractère pertinent du fichier ouvert en lecture. Nous avons dû gérer les cas où le texte récupéré de BoilerPipe avait un en-tête pour les chapitres, les permaliens présents, la numérotaion précédant chaque verset, les espaces et les retours à la ligne en trop... Le code obtenu est assez complexe et difficile à comprendre, car tout doit être traité au fil du flux de caractères, sans retours possibles, à coup de boucles et de variables temporaires, et au fur et à mesure des problèmes découverts. On note que la fonction de base d'ouverture de fichier en écriture de Python ne crée pas de fichiers lui-même et renvoie donc une erreur si le fichier n'existe pas. Pour remédier à ce problème, nous avons copié le contenu du dossier textes_sources/bible-en-ligne/ dans textes_sources/bible_txt au préalable, les fichiers étant tous de toute façon réécrits de zéro dans le script.

Nous avons créé un fichier indexant les longueurs en caractères de chaque verset de la Bible, nommé "index_longueurs.txt", dans le cadre d'une méthode de requête de texte décrite plus bas. Celui-ci est généré par le script Python createindexlongueurs.py, qui crée le fichier d'index vide, puis ouvre en lecture chaque fichier de textes_sources/bible_txt récursivement. Pour chaque ligne de chaque fichier, on écrit dans l'index la longueur de la ligne, le numéro de la ligne (donc du verset), et l'adresse relative du fichier en cours de lecture, sur des lignes différentes de l'index. Ainsi, nous obtenons des informations sur les longueurs de chaque verset par groupes de 3 lignes, ce qui est facile à parcourir.

Nous avons également remarqué que certains livres de la Bible manquaient sur bible-en-ligne.net (Tobie, Judith, les Machabées, Sagesse, Siracide, Baruch), nous les avons donc ajoutés en les tirant du site missionweb.free.fr et en les formatant et les nettoyant à la main. En rétrospective, il aurait peut-être été préférable de travailler avec cette source, mais nous ne l'avons trouvée qu'à la fin.

Notre objectif au départ était de trouver des alternatives en français à chaque type de texte qui pouvait apparaître dans un Livre d'Heures, en se basant sur leurs définitions, mais il nous a été difficile de trouver des informations ou des versions en français directs pour certains d'entre eux.

Certains textes semblent aujourd'hui ne plus avoir la même signification qu'autrefois, ou du moins de ne plus se présenter sous le même format que celui observé dans les Livres d'Heures, difficile alors de trouver des équivalents dans un format similaire. Par exemple, les Bénédictions trouvables sur Internet se présentent sous la forme de paragraphes se terminant par Amen, ou de plusieurs phrases à la suite de la taille de versets, toutes terminées par Amen. Or, celles que nous avons pu trouver dans nos livres ne semblent pas ressembler à ces versions plus modernes.

D'autres textes dans les livres que nous avons étudiés ne semblaient pas appartenir à un type de texte en particulier, ou semblaient faire partie de plus grands ensembles de textes. En effet, certains des types de textes que nous avons pu observer comme les Invitatoires ou les Capitules étaient composés de plus petits morceaux de textes comme des versets bibliques. Dans ces cas-ci, nous avons opté pour les remplacer par des versets bibliques tirés au hasard mais de longueurs égales ou similaires.

Les Livres d'Heures contiennent également des prières spécifiques comme "Obsecro Te", "Gloria Patri" et "Pater Noster" pour lesquelles nous avons pu trouver une traduction précise. Nous avons également pu trouver sur le site cfc-liturgie.fr[**?**] une page d'antiennes qui semblaient correspondre au format attendu. Ce même site propose également d'autres types de textes que nous cherchions tels que des hymnes et des cantiques, mais ceux-ci n'avaient encore une fois pas la même forme que ceux dans les Livres d'Heures, nous ne pouvions donc pas les utiliser en remplacement des originaux.

4 Analyse et modélisation d'un Livre d'Heures

Après avoir identifié et récupéré les textes utiles à la génération du corpus, une analyse structurelle d'un livre est nécessaire afin de dégager la disposition et les proportions des différents textes selon la section et la sous-section dans laquelle on se situe.

Cette opération a été réalisée manuellement en se basant sur le Livre d'Heures fourni dans le rapport du projet BOH. Nous avons utilisé cette version pour réaliser cette étape car son exploration à la main est plus aisée que la version XML ou qu'une version numérisée. Il s'agissait aussi de la seule version à notre disposition au moment où nous avons réalisé ce travail.

Dans cette analyse, nous recherchons les types de textes identifiés précédemment et nous les comptons afin d'obtenir des proportions pour chaque section et sous-section. Un type "Autres" a été ajouté puisque certains textes ne font pas partie des types identifiés, comme des prières par exemple. l'*Obsecro te* a été omis volontairement, puisqu'il s'agit bien d'un type de texte mais il représente une section d'un Livre d'Heures en lui-même, il ne réapparaîtra donc pas dans d'autres sections d'un Livre d'Heures.

Les sections concernées par cette analyse sont les heures de la Vierge, les heures de la croix, les heures du saint-esprit, les heures des morts et les suffrages. Chaque section est divisée en sous-sections qui peuvent être les suivantes : Les *Matins*, les *Lauds*, les *Prime*, les *Tierce*, les *Sext*, les *None*, les *Vespers*, les *Complines*. Ces sous-sections n'apparaissent pas tous dans chaque section, il y a aussi certaines sous-sections plus spécifiques à des sections comme les *Nocturnum* qui apparaissent seulement dans les heures des morts.

Voici les proportions pour chaque section, représentées dans des tableaux en annexe :

— **Heures de la Vierge :** On remarque que cette heure est la plus importante d'un Livre d'Heures puisqu'elle contient environ 226 textes. La mâtine est la sous-section la plus importante car elle possède 58 textes et c'est aussi la sous-section des heures de la Vierge qui possède le plus de types de textes différents. Les proportions complètes sont accessibles en annexe A.2.

— **Heures de la croix :** Il s'agit d'une heure possédant une constitution moyenne de textes avec 85 textes. Les différentes sous-sections sont toutes très similaires, il semble même qu'entre chaque sous-section, la structure des textes reste la même. Les proportions complètes sont accessibles en annexe A.3.

— **Heures du saint-esprit :** Tout comme les heures de la croix, cette heure a une constitution moyenne de textes avec 68 textes. Les première et dernière sous-sections de cette heure (Mâtine et Compline) semblent plus importantes car elles possèdent plus de textes que les autres. Mais on observe une forme de répétition dans la structure des textes de chaque sous-section. Les proportions complètes sont accessibles en annexe A.4.

— **Heures des morts :** Il s'agit d'une heure possédant une certaine importance puisqu'elle est composée d'environ 156 textes. Comme précédemment, deux sous-sections (Vêpres et Laude) prennent une place plus importante avec un nombre de textes plus élevé. A contrario, la mâtine possède un nombre de textes très faible, seulement 5. Les proportions complètes sont accessibles en annexe A.5.

— **Suffrages :** D'après les documents du projet BOH, les suffrages n'ont que la mâtine en sous-sections. Cette section possède environ une centaine de textes,

qui se répartissent entre antienne, oraison, répons et verset. Les proportions complètes sont accessibles en annexe A.6.

5 Génération de corpus d'un Livre d'Heures

Afin de générer un corpus de Livre d'Heures, plusieurs méthodes ont été envisagées, une seule a été retenue, mais nous pensons qu'il est important de détailler chacune des méthodes. Ceci permettra de reprendre et utiliser plus facilement n'importe quelle de ces méthodes à la suite de ce projet de recherche si cela est nécessaire. Chacune de ces méthodes présente des avantages et des désavantages que nous allons détailler ci-dessous.

5.1 Génération semi-aléatoire

Notre première idée de génération de corpus est une génération semi-aléatoire. Cette idée nous est venue à la suite du travail effectué sur les types de textes constituant un Livre d'Heures ainsi que celui effectué sur les proportions des types de textes. Ces informations nous permettraient de construire un programme de création de corpus qui respecteraient les données récoltées.

Pour chaque heure, on choisit aléatoirement un texte à ajouter qui correspond aux proportions déterminées précédemment. Cette méthode semble être une relativement bonne idée mais nous trouvons que cette construction est trop aléatoire. En effet, en utilisant un choix de textes aléatoire, nous pouvons nous retrouver avec une succession de textes illogiques par rapport à celle d'un Livre d'Heures (par exemple, deux ou trois antiennes à la suite).

Pour pallier cela, nous avons pensé à mettre en place des règles de structure pour chaque type de textes. Par exemple, nous pouvons observer qu'un psaume est encadré par une antienne, ainsi si on choisit aléatoirement d'ajouter un psaume, on ajoutera en plus une antienne pour correspondre à la structure d'un Livre d'Heures. Si nous mettons en place suffisamment de règles de structure, cela nous donnera une construction semi-aléatoire qui respectera beaucoup plus la structure globale d'un véritable Livre d'Heures.

Cette méthode nous apporte plusieurs avantages : d'abord, sachant que le programme est semi-aléatoire, il nous permettrait de construire des corpus différents à chaque exécution. Il serait aussi possible de modifier simplement les règles de structuration pour mieux correspondre à différents Livre d'Heuress. En revanche, une fois le corpus créé, il sera difficile de déterminer s'il correspond bien à la structure d'un véritable Livre d'Heures. De même, une fois les algorithmes de structuration du corpus en XML exécutés, il sera difficile de déterminer si le XML créé est juste puisque nous ne possédons de XML de base défini comme correct, pour comparer avec celui généré.

5.2 Génération par le XML TOC_Arsenal

Comme nous avons pu le dire précédemment, un fichier XML, nommé TOC_Arsenal, nous a été fourni par le LRHT et correspond à la structure et au contenu d'un Livre d'Heures. Ce fichier est très bien renseigné, et nous pouvons retrouver pour de nombreux textes leur correspondance en attribut. En explorant le fichier XML et en récupérant ces correspondances, il est possible de remplacer les textes en latin par les textes en français qui correspondent. Il faut pour cela bien récolter tous les textes utiles pour un Livre d'Heures, ce qui a été fait précédemment, et les trier méthodiquement pour pouvoir les retrouver simplement avec les correspondances du fichier XML. De plus, lors de l'exploration et la modification du fichier XML, il est possible de générer le corpus qui lui correspond.

Cette méthode présente plusieurs avantages : tout d'abord, lors de la création du corpus, nous sommes assurés que la structuration des textes est correcte puisqu'elle

se base sur TOC_Arsenal basé lui-même sur un véritable Livre d'Heures. Ensuite, nous ne créons pas simplement un corpus mais la structuration XML qui lui correspond, ainsi, après l'exécution des algorithmes de structuration du corpus en XML, il sera possible de comparer facilement le résultat avec le XML généré lors de la création du corpus. Par contre, l'un des désavantages est que l'on ne peut générer qu'un seul et même corpus.

5.3 Génération par le Livre d'Heures traduit

En effectuant des recherches sur les sources de textes utiles pour la génération du corpus, nous avons découvert un site internet qui propose la traduction française d'un livre d'heure du Moyen-Age.[dB] Ceci nous a ouvert une nouvelle possibilité de génération de corpus. Il s'agit, comme pour la Bible, de récupérer toute cette traduction en utilisant Boilerpipe.

Cette méthode peut être très intéressante car elle ne nécessite pas de sources de textes pour générer le corpus. En revanche, la récupération du Livre d'Heures traduit demanderait beaucoup de temps car il est réparti sur de nombreuses pages internet (une page web par page du livre). De plus, on ne peut pas s'assurer de la qualité de la traduction et que des informations ne figurant pas dans le livre ne sont pas ajoutées par le traducteur.

5.4 Méthode sélectionnée et travail réalisé

Parmi les trois méthodes de génération de corpus que nous avons exposées précédemment, nous avons décidé de n'en réaliser qu'une seule, au vu du temps que nous avons pour réaliser ce TER. Nous avons choisi de réaliser la méthode de génération par le fichier XML TOC_Arsenal.

5.4.1 Classe parserXML, exploration du XML et génération du corpus

Dans le but d'explorer le fichier, nous avons décidé d'utiliser Java avec DOM. Il s'agit de l'acronyme de Document Object Model, c'est une spécification du W3C pour proposer une API qui permet de modéliser, de parcourir et de manipuler un document XML.

Afin d'utiliser l'API DOM, nous avons créé une classe parserXML qui va posséder la capacité d'ouvrir un fichier XML, de l'explorer, de créer une copie du fichier XML en français et de créer un corpus. Ces fonctionnalités sont réparties sur différentes méthodes que nous allons expliquer ci-dessous avec les variables utiles à la classe.
Voici le descriptif des variables de la classe parserXML, elles sont toutes privées :

- **factoryRead, builderRead et documentRead :** Respectivement de type *DocumentBuilderFactory, DocumentBuilder* et *Document*, ces variables servent à l'ouverture et à la lecture du fichier XML source.

- **factoryWrite, builderWrite et documentWrite :** Respectivement de type *DocumentBuilderFactory, DocumentBuilder* et *Document*, ces variables servent à l'ouverture et à l'écriture d'une nouvelle arborescence XML de sortie.

- **transformerFactory, transformer, source et sortie :** Respectivement de type *TransformerFactory, Transformer, DOMSource* et *StreamResult*, ces variables servent à enregistrer l'arborescence XML de sortie dans un fichier XML.

- **pattern et matcher :** Respectivement de type *Pattern* et *Matcher*, ces variables permettent de détecter un pattern d'expression régulière afin de les supprimer dans le corpus final.

— **fw :** De type *FileWriter*, cette variable permet d'enregistrer dans un fichier le corpus créé.

— **txtRecup :** De type *RecupTexte_itf*, cet objet permet de récupérer les textes utiles à la création du corpus. Cette classe sera décrite ci-dessous.

Voici le descriptif des méthodes de la classe parserXML :

— **public parserXML(*String fichierXML, String fichierXMLSortie*) :** Il s'agit du constructeur de la classe parserXML, elle prend en paramètre le nom du fichierXML à explorer et celui du fichier XML à créer. Il va initialiser les différentes variables citées précédemment.

— **public *void* lancerExploration(*String fichierTXTSortie*) :** Il s'agit de la méthode qui va lancer une méthode récursive d'exploration, elle prend en paramètre le nom du fichier du corpus. Elle va récupérer le nœud racine du XML en lecture et lancer la méthode récursive explorationXML que nous allons détailler ci-dessous.

— **private *void* explorationXML(*Node baliseRead, Node baliseWrite*) :** Cette méthode est une méthode récursive qui va explorer le fichier XML en lecture et créer le XML en écriture. Il explore petit à petit les nœuds fils du nœud père. Lorsque la méthode arrive sur un nœud de type texte, il lancera de nouvelles méthodes qui permettent de déterminer le texte en français le plus adapté pour remplacer le texte en latin. La récursion se stoppe lorsqu'il n'y a plus de nœuds fils pour le nœud en cours d'exploration. Il générera aussi durant l'exploration, le corpus. Comme vous pouvez le voir, cette méthode est privée afin qu'elle ne puisse être lancée seulement par la méthode lancerExploration.

— **public *String* getTextCorresp(*String attributesValue*) :** Cette méthode va prendre en paramètre la valeur des attributs corresp du fichier XML, elle va analyser cet attribut pour en retirer de l'information et déterminer le texte à récupérer. Elle retourne le texte qui correspond à l'attribut analysé par une autre méthode dont nous parlerons ci-dessous.

— **private *String* analyseCorresp(*String typeText, String chapitre, String verset*) :** Cette méthode privée ne peut être appelée que par getTextCorresp. Elle va prendre en paramètre le type de texte, le chapitre et le verset du texte recherché et utiliser l'objet de type *RecupTexte_itf* qui sera décrit ci-dessous.

— **public *String* getHiTrad(*String text*) :** Cette méthode retourne une traduction pour les titres des textes d'un Livre d'Heures en paramètre.

Une fois le corpus généré, certains caractères d'espacement servant à l'indentation du XML sont encore présent dans le corpus. Pour les supprimer, un script python nommé unindent.py doit être exécuté sur le corpus avant d'y appliquer n'importe quel algorithme. Vous pouvez retrouver la classe parserXML et le script python sur le github qui est associé à ce projet.

5.4.2 Programme de récupération des textes pour la génération

Une fois tous nos textes en français organisés dans des fichiers dédiés, nous avons créé un système de requête de textes d'un type donné en Java. Ce programme est le seul qui accède directement aux fichiers de textes sources et permet de fournir une interface pour qu'il n'y ait plus qu'à appeler des méthodes pour récupérer du texte dans un type ou avec une longueur demandée du côté de la génération de corpus.

Ce programme est divisé en 3 fichiers que nous allons détailler ici : Texte_type.java, la structure de donnée dans laquelle on récupère et transfère le contenu d'un texte

et des méta-informations sur celui-ci, RecupTexte_Itf.java, l'interface Java contenant les différentes méthodes pouvant être appelées pour récupérer du texte, et RecupTexte_Impl.java, qui implémente les méthodes.
Le programme de récupération de texte (abbrégé en RecupTexte) fournit plusieurs méthodes pour récupérer du texte dans un format particuler à la demande. Le fichier d'interface ne servant qu'à exposer les méthodes disponibles, nous allons nous concentrer sur l'implémentation de celles-ci.

La méthode **getVersetBible(String livre, int chapitre, int verset)** permet de récupérer un Texte_type contenant le texte du verset au numéro donné du chapitre donné du livre donné (écrit traditionnellement sous la forme "livre chapitre :verset"), avec comme valeur pour l'attribut type "verset", et les attributs livreBible, chapitreBible, et versetBible contiennent les informations correspondantes. Si les arguments donnés ne correspondent à aucun verset, un Texte_type d'erreur est retourné, avec la raison de l'erreur dans l'attribut contenu.

Lorsque cette méthode est appelée, elle regarde la ligne à l'argument "verset" dans le fichier n.txt, n étant l'argument "chapitre", dans le dossier nommé comme l'argument "livre" dans notre système de fichiers. Si la ligne est bien trouvée dans le bon fichier, un Texte_type contenant toutes les informations est retourné.

La méthode **getAntiennes()** permet de récupérer une antienne parmi toutes celles que nous avons pu trouver. Pour cela, elle entre chaque ligne du fichier textes_sources/autres_txt/antiennes.txt dans une liste ArrayList, choisit un indice de la liste au hasard et retourne le contenu de l'élément de la liste à l'indice tiré dans un Texte_type dont l'attribut type est "antienne".

Les méthodes **getObsecroTe()**, **getPaterNostre()**, **getGloriaPatri()** permettent de récupérer le texte des prières correspondantes, respectivement Obsecro Te, Gloria Patri et Pater Nostre. Elles lisent simplement les fichiers correspondant dans le dossier textes_sources/autres_txt et retournent leur contenu dans un Texte_type dont l'attribut type correspond au nom du fichier lu (sans l'extension .txt).

La méthode **getVersetParLongueur(int longueur, int écartAcceptable)** permet de récupérer au hasard un verset de la longueur en caractères donnée, et dans les cas où il n'en existe pas, retourne un verset parmi ceux de longueur comprise par l'intervalle formé par la valeur de l'argument "écartAcceptable", c'est-à-dire les longueurs avoisinnant la longueur donnée par plus ou moins écartAcceptable. Par exemple, getVersetParLongueur(100,5) pourra renvoyer un verset de 95, 96, 97, 98, 99, 101, 102, 103, 104 ou 105 caractères, s'il n'existe pas de verset de 100 caractères dans la Bible.

Bien entendu, parcourir tout les versets dans nos fichiers pour connaître leur longueur à chaque appel de cette fonction serait extrêmement gourmand en temps et en mémoire. C'est donc pour cela que nous avons créé l'index des longueurs détaillé ce-dessus, afin de ne plus avoir à parcourir qu'un seul fichier pour savoir quel verset chercher. Et pour ne pas avoir à parcourir plusieurs fois ce fichier d'index, nous avons créé une fonction privée chargeIndexLongueurs() qui parcourt le fichier et entre les informations dans des listes nommées listeLongueurs, listeVersets et listeAdressesFichiers en attributs de la classe du programme pour un accès rapide et facile aux données. Puisque les données sont entrées dans ces listes dans l'ordre, elles se remplissent en parallèle et les données de chaque liste pour un indice donné correspondent bien au même verset. Cette fonction n'est appelée qu'une première fois si les listes d'index sont vides, et seulement quand on a besoin de cet index.

Une fois l'index chargé, on parcourt la liste des longueurs listeLongueurs et on garde les indices des cellules où la valeur est égale à la longueur donnée en argument, ou proche de "écartAcceptable" caractères, puis on stocke ces indices dans des listes nommées valeursExactes et assezProche. Si la liste valeurExactes contenant les indices des versets de bonne longueur n'est pas vide, on tire un de ces indices au hasard, on regarde à quel verset correspond cet indice dans les listes listeVersets et listeAdressesFichiers, et on utilise la méthode getVersetBible décrite ci-dessus pour le retourner. Si la liste valeurExactes est vide (il n'y a donc pas de verset de la longueur demandée), on regarde si la liste assezProche n'est pas vide, et s'il y a bien au moins un élément, on trouve le verset correspondant de la même manière que décrit

pour les versets de bonne longueur. Si même la liste assezProche est vide (il n'y a pas de versets de la longueur demandée même malgré la laxité laissée), on retourne un Texte_type d'erreur contenant un message d'erreur, avec les valeurs données en argument.

Une surcharge de cette méthode avec un seul argument pour la longueur a été exposée, elle cherche dans ce cas seulement les versets de longueur exactes en appelant la méthode de départ avec un "écartAcceptable" de 0 ;

6 Application d'algorithmes d'état de l'art sur le corpus

Une fois la composition du Livre d'Heures effectuée, l'étape suivante prévue dans le TER était d'appliquer des algorithmes de recherche de textes de l'état de l'art afin de mesurer leurs performances sur un corpus reconstitué, en français. L'étape de composition de Livre d'Heures nous a pris plus de temps qu'initialement prévu, nous n'avons donc pas avancé là dessus.

Nous nous sommes intéressés au format d'annotation CoNLL-U sur les conseils de nos encadrantes. L'utilisation de cet outil aurait été notre prochain objectif dans l'optique d'appliquer ces algorithmes.

7 Conclusion

Nous n'avons pas pu effectuer tout le travail proposé dans ce TER, car nous avons dû au préalable nous former dans des technologies de traitement des langues comme DOM et l'extraction de page HTML avec Boilerpipe. L'étude de la structure des Livres d'Heures et la composition d'un ouvrage de ce type en nous basant sur des règles découvertes avec des Livres d'Heures 'exemples', nous a permis de toucher du doigt ce qu'était un travail de recherche.

Nous n'avons également utilisé qu'une seule des trois méthodes de composition que nous avons évoquées par manque de temps, mais appliquer les autres, particulièrement celle utilisant des règles, permettrait de comparer les résultats obtenus en appliquant des algorithmes de l'état de l'art de reconnaissance de texte.

En tant que groupe d'étudiants en ALMA, ce TER nous a permis de jeter un oeil sur le domaine du TAL en étudiant et traitant des textes, et de constater le travail impressionnant effectué dans les Humanités Numériques, les possibilités d'études et de recherche sur les documents historiques tels que les Livres d'Heures.

A Annexes

A.1 Abréviations des types de textes

Abréviation	Mots
A (-)	Antiphon
Abs (-)	Absolution
Ben (-)	Benediction
Cant (+)	Canticle
Cap (-)	Capitulum
Inv (-)	Invitatory
Les (+)	Lesson
Or (~)	Oration
Ps (+)	Psalm
R (-)	Response
Recom	Recommendation
V(-)	Versicle

Figure 1 – Abréviations des types de textes

A.2 Proportions Heures de la Vierge

Heures de la vierge	Matins	Lauds	Prime	Terce	Sext	None	Vespers	Compline
A (-)	18	13	3	3	3	3	13	3
Abs (-)	1							
Ben (-)	1							
Cant (+)	1	2					1	1
Cap (-)		1	1	1	1	1	1	1
Inv (-)	1							
Les (+)	3							
Or (~)		4	4	4	4	4	4	4
Ps (+)	10	7	3	1	2	2	5	3
R (-)	8	3	2	1	5	4	4	5
Recom								
V(-)	5	2	3	4	4	4	4	5
Autres (~) prière	10	3	2	2	2	2	2	2
total	58	35	18	16	21	20	34	24

Figure 2 – Proportions pour les heures de la Vierge

A.3 Proportions Heures de la croix

Heures de la croix	Matins	Prime	Terce	Sext	None	Vespers	Compline
A (-)	1	1	1	1	1	1	1
Abs (-)							
Ben (-)							
Cant (+)							
Cap (-)							
Inv (-)							
Les (+)							
Or (~)	1	1	1	1	1	1	1
Ps (+)							
R (-)	3	3	3	3	3	3	3
Recom							1
V(-)	3	3	3	3	3	3	3
Autres (~) prière	4	4	4	4	4	4	4
total	12	12	12	12	12	12	13

Figure 3 – Proportions pour les heures de la croix

A.4 Proportions Heures du saint-esprit

Heures saint-esprit	*Matins*	*Prime*	*Tierce*	*Sext*	*None*	*Vespers*	*Compline*
A (-)	1	1	1	1	1	1	1
Abs (-)							
Ben (-)							
Cant (+)							
Cap (-)							
Inv (-)							
Les (+)							
Or (~)	1	1	1	1	1	1	1
Ps (+)							
R (-)	3	2	2	2	2	2	3
Recom							1
V(-)	3	2	2	2	2	2	3
Autres (~) prière	3	3	3	3	3	3	3
total	11	9	9	9	9	9	12

Figure 4 – Proportions pour les heures du saint-esprit

A.5 Proportions Heures des morts

Heures des morts	*Vespers*	*Matins*	*Nuit une*	*Nuit deux*	*Nuit trois*	*Lauds*
A (-)	12		6	6	6	12
Abs (-)						
Ben (-)						
Cant (+)	1					2
Cap (-)						
Inv (-)		1				
Les (+)						
Or (~)	1					1
Ps (+)	6	1	3	3	3	8
R (-)	8		5	5	6	8
Recom						
V(-)	7		6	6	9	7
Autres (~) prière	1	3	4	4	4	1
total	36	5	24	24	28	39

Figure 5 – Proportions pour les heures des morts

A.6 Proportions Suffrages

Suffrages	*Matins*
A (-)	23
Abs (-)	
Ben (-)	
Cant (+)	
Cap (-)	
Inv (-)	
Les (+)	
Or (~)	28
Ps (+)	
R (-)	23
Recom	1
V(-)	23
Autres (~) prière	
total	98

Figure 6 – Proportions pour les suffrages

Références

[cl] cfc liturgie. Antiennes pour les 150 psaumes pendant le temps pascal. http://www.cfc-liturgie.fr/index.php?option=com_content&task=view&id=987&Itemid=206.

[dB] Abbaye du Barroux. Un livre d'heures imprimé et sa traduction en français. http://livreheurestraduit.pagesperso-orange.fr/Plandusite.htm.

[dR] Agence Nationale de Recherche. Projet horae. http://www.agence-nationale-recherche.fr/Projet-ANR-17-CE38-0008.

[es] Liturgie et sacrements. Invitatoire. https://liturgie.catholique.fr/lexique/invitatoire/.

[IRH] LS2N IRHT, Teklia. Horae project. https://horae.digital/.

[Leb] J. Lebigue. L'office des heures. https://irht.hypotheses.org/2204.

[Paw] G. Pawlowski. Les livres d'heures. http://www.cosmovisions.com/textLivresHeures.htm.

[Wik] Wikipédia. Livre d'heures. http://fr.wikipedia.org/wiki/Livre_d'heures.