ÉCOLE NATIONALE DES CHARTES
UNIVERSITÉ PARIS, SCIENCES & LETTRES

Kelly Christensen
diplômée de doctorat musicologie

D'ALTO à TEI

Modélisation de transcriptions automatiques pour une pré-éditorialisant des textes

Mémoire pour le diplôme de master
« Technologies numériques appliquées à l'histoire »

2022

Résumé

Quand des modèles OCR et HTR extraient les données d'une ressource textuelle numérisée, les informations relatives à la structure physique de l'image risquent de se perdre. Un schéma XML standardisé qui s'appelle ALTO a été créé afin de conserver et structurer ces données non-textuelles et géometriques en les tenant en relation avec le contenu textuel. La plupart des modèles OCR et HTR compte sur ce schéma. Cependant ALTO ne convient pas bien à l'édition numérique ni aux traitements automatique du langage. Les éditeurs et les chercheurs en lettres attendent un schéma XML plus courant dans le monde des humanités numériques : la TEI. Il faut donc un mapping pour transformer un fichier ALTO en fichier TEI sans perdre aucune donnée lors du processus. Cette transformation automatisée permet à conserver les données particulières au schéma ALTO, telles que celles sur la segmentation et sur la structure physique du document numérisé, ainsi qu'à exploiter le contenu textuel de la ressource textuelle. La flexibilité de la TEI et son usage très répandu rendent le schéma idéal pour mieux valoriser les données produites par les modèles OCR et HTR.

Dans le cadre du stage pour obtenir le diplôme de Master 2 « Technologies numériques appliquées à l'histoire », ce mémoire porte sur la modélisation de la transformation de ALTO en TEI. Cette modélisation a été réalisée dans le cadre du projet *Gallic(orpor)a*, financé par la BnF lors d'un stage qui a eu lieu au sein du laboratoire Automatic Language Modelling and Analysis & Computational Humanities entre avril et juillet 2022.

Mots-clés : HTR, OCR, ALTO, TEI, TAL, édition numérique.

Informations bibliographiques : Kelly Christensen, *D'ALTO à TEI, modélisation de transcriptions automatiques pour une pré-éditorialisant des textes*, mémoire de master « Technologies numériques appliquées à l'histoire », dir. Ségolène Albouy, École nationale des chartes, 2022.

Remerciements

Mes remerciements vont tout d'abord à...

Bibliographie

16 Linking, Segmentation, and Alignment - The TEI Guidelines. URL : https://tei-c.org/release/doc/tei-p5-doc/en/html/SA.html#SASOstdf (visité le 25/08/2022).

About – TEI : Text Encoding Initiative. URL : https://tei-c.org/about/ (visité le 25/08/2022).

Alliance, The ARK. *Community*. ARK Alliance. 6 nov. 2020. URL : https://arks.org/community/ (visité le 23/08/2022).

Bawden, Rachel et al. « Automatic Normalisation of Early Modern French ». In : LREC 2022 - 13th Language Resources and Evaluation Conference. 20 juin 2022. DOI : 10.5281/zenodo.5865428. URL : https://hal.inria.fr/hal-03540226 (visité le 11/08/2022).

Bibliothèque nationale de France. *Rapport d'activité 2021 de la Bibliothèque nationale de France*. Paris, France, 1er juill. 2022. URL : https://www.bnf.fr/fr/bnf-rapport-dactivite-2021 (visité le 09/08/2022).

Bobichon, Philippe. « Le Lexicon : Mise En Page et Mise En Texte Des Manuscrits Hébreux, Grecs, Latins, Romans et Arabes ». In : (2009), p. 81. URL : https://cel.archives-ouvertes.fr/cel-00377671 (visité le 25/07/2022).

— *Caviarder*. In : *Codicologia*. Institut de recherche et d'histoire des textes, 2011. URL : http://codicologia.irht.cnrs.fr/theme/liste_theme/413#tr-868 (visité le 25/07/2022).

Camps, Jean-Baptiste et al. « Corpus and Models for Lemmatisation and POS-tagging of Classical French Theatre ». In : *Journal of Data Mining & Digital Humanities* 2021 (Digital humanities in... 14 fév. 2021), p. 6485. ISSN : 2416-5999. DOI : 10.46298/jdmdh.6485. arXiv : 2005.07505 [cs]. URL : http://arxiv.org/abs/2005.07505 (visité le 09/08/2022).

Carlin, Marie et Arnaud Laborderie. « Le BnF DataLab, Un Service Aux Chercheurs En Humanités Numériques ». In : *Humanités numériques* 4 (déc. 2021). URL : https://hal-bnf.archives-ouvertes.fr/hal-03285816 (visité le 11/08/2022).

Caron, Bertrand. *Formats de Données Pour La Préservation à Long Terme : La Politique de La BnF*. Technical Report. Bibliothèque Nationale de France (Paris), oct. 2021. URL : https://hal-bnf.archives-ouvertes.fr/hal-03374030 (visité le 23/08/2022).

Chagué, Alix. *LECTAUREP Contemporary French Model (Administration)*. Zenodo, 12 mai 2022. URL : https://zenodo.org/record/6542744 (visité le 12/08/2022).

Chagué, Alix et Thibault Clérice. « Sharing HTR Datasets with Standardized Metadata : The HTR-United Initiative ». In : Documents Anciens et Reconnaissance Automatique Des Écritures Manuscrites. 23 juin 2022. URL : https://hal.inria.fr/hal-03703989 (visité le 12/08/2022).

Chagué, Alix, Thibault Clérice et Laurent Romary. « HTR-United : Mutualisons La Vérité de Terrain ! » In : *DHNord2021 - Publier, Partager, Réutiliser Les Données de La Recherche : Les Data Papers et Leurs Enjeux*. Lille, France : MESHS, nov. 2021. URL : https://hal.archives-ouvertes.fr/hal-03398740 (visité le 10/08/2022).

Chagué, Alix et al. *HTR-United/Htr-United : V0.1.28*. Zenodo, 10 août 2022. DOI : 10.5281/zenodo.6979746. URL : https://zenodo.org/record/6979746 (visité le 12/08/2022).

Clérice, Thibault. *YALTAi : Segmonto Manuscript and Early Printed Book Dataset*. Zenodo, 10 juill. 2022. DOI : 10.5281/zenodo.6814770. URL : https://zenodo.org/record/6814770 (visité le 12/08/2022).

Clérice, Thibault et Ariane Pinche. *HTRVX, HTR Validation with XSD*. Version 0.0.1. Sept. 2021. DOI : 10.5281/zenodo.5359963. URL : https://github.com/HTR-United/HTRVX (visité le 12/08/2022).

Coquenet, Denis, Clément Chatelain et Thierry Paquet. « Handwritten Text Recognition : From Isolated Text Lines to Whole Documents ». In : *ORASIS 2021*. Saint Ferréol, France : Centre National de la Recherche Scientifique [CNRS], sept. 2021. URL : https://hal.archives-ouvertes.fr/hal-03339648 (visité le 04/08/2022).

DigiPal : Digital Resource and Database of Palaeography, Manuscripts and Diplomatic. London, été 2011. URL : http://www.digipal.eu/.

Gabay, Simon. *E-Ditiones, 17th c. French Sources*. Nov. 2018. URL : https://hal.archives-ouvertes.fr/hal-02388415 (visité le 10/08/2022).

— *FreEM-corpora/FreEMnorm : FreEM Norm Parallel Corpus*. Zenodo, 17 jan. 2022. DOI : 10.5281/zenodo.5865428. URL : https://zenodo.org/record/5865428 (visité le 11/08/2022).

Gabay, Simon, Jean-Baptiste Camps et Ariane Pinche. « SegmOnto ». In : *Création de Modèle(s) HTR Pour Les Documents Médiévaux En Ancien Français et Moyen Français Entre Le Xe-XIVe Siècle*. Paris, France : Ecole nationale des chartes | PSL, nov. 2021. URL : https://hal.archives-ouvertes.fr/hal-03481089 (visité le 25/07/2022).

Gabay, Simon, Thibault Clérice et Christian Reul. *OCR17 : Ground Truth and Models for 17th c. French Prints (and Hopefully More)*. Mai 2020. URL : https://hal.archives-ouvertes.fr/hal-02577236 (visité le 12/08/2022).

GABAY, Simon et al. « Standardizing Linguistic Data : Method and Tools for Annotating (Pre-Orthographic) French ». In : *Proceedings of the 2nd International Digital Tools & Uses Congress (DTUC '20)*. Hammamet, Tunisia, oct. 2020. DOI : 10.1145/3423603.3423996. URL : https://hal.archives-ouvertes.fr/hal-03018381 (visité le 12/08/2022).

GABAY, Simon et al. « Automating Artl@s – Extracting Data from Exhibition Catalogues ». In : *EADH 2021 - Second International Conference of the European Association for Digital Humanities*. Krasnoyarsk, Russia, sept. 2021. URL : https://hal.archives-ouvertes.fr/hal-03331838 (visité le 23/08/2022).

GABAY, Simon et al. « SegmOnto : Common Vocabulary and Practices for Analysing the Layout of Manuscripts (and More) ». In : *1st International Workshop on Computational Paleography (IWCP@ICDAR 2021)*. Lausanne, Switzerland, sept. 2021. URL : https://hal.archives-ouvertes.fr/hal-03336528 (visité le 09/08/2022).

GABAY, Simon et al. « From FreEM to D'AlemBERT ». In : *Proceedings of the 13th Language Resources and Evaluation Conference*. Marseille, France : European Language Resources Association, juin 2022. URL : https://hal.inria.fr/hal-03596653 (visité le 10/08/2022).

GABAY, Simon et al. « Le Projet FREEM : Ressources, Outils et Enjeux Pour l'étude Du Français d'Ancien Régime ». In : *TALN 2022 - Traitement Automatique Des Langues Naturelles*. Sous la dir. d'Yannick ESTÈVE et al. Avignon, France : ATALA, juin 2022, p. 154-165. URL : https://hal.archives-ouvertes.fr/hal-03701524 (visité le 10/08/2022).

GACEK, Adam. *The Arabic Manuscript Tradition : A Glossary of Technical Terms and Bibliography*. Handbook of Oriental Studies 1, The Near and Middle East. Leiden : Brill, 2001. 269 p. ISBN : ISBN 90-04-12061-0.

« Glossaire codicologique français-arabe ». In : *Gazette du livre médiéval* 40.1 (2002), p. 79-80. URL : https://www.persee.fr/doc/galim_0753-5015_2002_num_40_1_1563 (visité le 25/07/2022).

GOODRICH, Gregory. « Kurzweil Reading Machine : A Partial Evaluation of Its Optical Character Recognition Error Rate ». In : *Journal of Visual Impairment and Blindness* (12 jan. 1979).

GOVINDAN, V. K. et A. P. SHIVAPRASAD. « Character Recognition — A Review ». In : *Pattern Recognition* 23.7 (1990), p. 671. ISSN : 0031-3203. URL : https://www.academia.edu/6986960/Character_recognition_A_review (visité le 05/08/2022).

GUIBON, Gaël et al. « Parsing Poorly Standardized Language Dependency on Old French ». In : *Thirteenth International Workshop on Treebanks and Linguistic Theories (TLT13)*. Sous la dir. de V. HENRICH et al. Proceedings of the Thirteenth International Workshop on Treebanks and Linguistic Theories (TLT13). Tübingen, Germany, déc. 2014,

p. 51-61. URL : https://hal.archives-ouvertes.fr/hal-01250959 (visité le 10/08/2022).

JENTSCH, Patrick et Stephan PORADA. « From Text to Data Digitization, Text Analysis and Corpus Linguistics ». In : *Digital Methods in the Humanities : Challenges, Ideas, Perspectives*. Sous la dir. de Silke SCHWANDT. Bielefeld University Press, 2021.

KIESSLING, Benjamin. « Kraken - an Universal Text Recognizer for the Humanities ». In : ADHO 2019 - Utrecht. 2019. URL : https://dh-abstracts.library.cmu.edu/works/9912 (visité le 10/08/2022).

LASSNER, David et al. « Publishing an OCR Ground Truth Data Set for Reuse in an Unclear Copyright Setting. Two Case Studies with Legal and Technical Solutions to Enable a Collective OCR Ground Truth Data Set Effort ». Version 1.0. In : *Fabrikation von Erkenntnis – Experimente in den Digital Humanities. Hg. von Manuel Burghardt* Lisa Dieckmann (2021). Avec la coll. d'Herzog August BIBLIOTHEK, 5). DOI : 10.17175/SB005_006. URL : https://zfdg.de/sb005_006 (visité le 10/08/2022).

MANIACI, Marilena. *Terminologia Des Libro Manoscritto*. Addenda 3. Rome : Istituto centrale per la patologia del libro, 1996.

MARTIN, Louis et al. « CamemBERT : A Tasty French Language Model ». In : *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. ACL 2020. Online : Association for Computational Linguistics, juill. 2020, p. 7203-7219. DOI : 10.18653/v1/2020.acl-main.645. URL : https://aclanthology.org/2020.acl-main.645 (visité le 11/08/2022).

METS : Metadata Encoding and Transmission Standard. BnF - Site institutionnel. URL : https://www.bnf.fr/fr/mets-metadata-encoding-and-transmission-standard (visité le 23/08/2022).

MUEHLBERGER, Guenter et al. « Transforming Scholarship in the Archives through Handwritten Text Recognition : Transkribus as a Case Study ». In : *Journal of Documentation* 75.5 (9 sept. 2019), p. 954-976. ISSN : 0022-0418. DOI : 10.1108/JD-07-2018-0114. URL : https://www.emerald.com/insight/content/doi/10.1108/JD-07-2018-0114/full/html (visité le 04/08/2022).

MUZERELLE, Denis. *Caviarder*. In : *Codicologia*. Institut de recherche et d'histoire des textes, 2011. URL : http://codicologia.irht.cnrs.fr/theme/liste_theme/413#tr-868 (visité le 25/07/2022).

MUZERELLE, Dennis. *Vocabulaire Codicologique : Répertoire Méthodique Des Termes Français Relatifs Aux Manuscrits*. Rubricae 1. Paris : Éd. Cemi, 1985.

Numpy : NumPy Is the Fundamental Package for Array Computing with Python. Version 1.23.1. URL : https://www.numpy.org (visité le 04/08/2022).

OSTOS, Pilar, M. Luisa PARDO et Elena E. RODRÍGUEZ. *Vocabulario de codicologÍa : Versión Española Revisada y Aumentada Del Vocabulaire Codicologique*. Instrumenta Bibliologica. Madrid : Arco/Libros, 1997.

Otsu, Nobuyuki. « A Threshold Selection Method from Gray-Level Histograms ». In : *IEEE Transactions on Systems, Man, and Cybernetics* 9.1 (jan. 1979), p. 62-66.

Pinche, Ariane. « HTR model Cremma Medieval ». In : (21 juin 2022). doi : 10.5281/zenodo.6669508. url : https://zenodo.org/record/6669508 (visité le 10/08/2022).

Pinche, Ariane et Jean-Baptiste Camps. « CremmaLab Project : Transcription Guidelines and HTR Models for French Medieval Manuscripts ». In : *Colloque "Documents Anciens et Reconnaissance Automatique Des Écritures Manuscrites"*. Paris, France, juin 2022. url : https://hal.archives-ouvertes.fr/hal-03716526 (visité le 10/08/2022).

Pinche, Ariane et Thibault Clérice. *HTR-United/Cremma-Medieval : Cortado 2.0.0*. Zenodo, 11 juill. 2022. doi : 10.5281/zenodo.6818057. url : https://zenodo.org/record/6818057 (visité le 12/08/2022).

Regnault, Mathilde, Sophie Prévost et Éric Villemonte de La Clergerie. « Challenges of Language Change and Variation : Towards an Extended Treebank of Medieval French ». In : *TLT 2019 - 18th International Workshop on Treebanks and Linguistic Theories*. Paris, France, août 2019. url : https://hal.inria.fr/hal-02272560 (visité le 10/08/2022).

Scheithauer, Hugo, Alix Chagué et Laurent Romary. « From eScriptorium to TEI Publisher ». In : *Brace Your Digital Scholarly Edition !* Berlin, France, nov. 2021. url : https://hal.inria.fr/hal-03538115 (visité le 23/08/2022).

Schneider, Ben R. « The Production of Machine-Readable Text : Some of the Variables ». In : *Computers and the Humanities* 6.1 (sept. 1971), p. 39-47. issn : 0010-4817, 1572-8412. doi : 10.1007/BF02402324. url : http://link.springer.com/10.1007/BF02402324 (visité le 02/08/2022).

Siddiqi, Imran-Ahmed, Florence Cloppet et Nicole Vincent. « Writing Property Descriptors : A Proposal for Typological Groupings ». In : *Gazette du livre médiéval* 56.1 (2011), p. 42-57. doi : 10.3406/galim.2011.1981. url : https://www.persee.fr/doc/galim_0753-5015_2011_num_56_1_1981 (visité le 02/08/2022).

Stehno, Birgit, Alexander Egger et Gregor Retti. « METAe–Automated Encoding of Digitized Texts ». In : *Literary and Linguistic Computing* 18.1 (1er avr. 2003), p. 77-88. issn : 0268-1145, 1477-4615. doi : 10.1093/llc/18.1.77. url : https://academic.oup.com/dsh/article-lookup/doi/10.1093/llc/18.1.77 (visité le 24/08/2022).

Stein, Achim et Sophie Prévost. *Syntactic Annotation of Medieval Texts : The Syntactic Reference Corpus of Medieval French (SRCMF)*. Narr Verlag, 2013, p. 275. isbn : 978-3-8233-6760-4. url : https://halshs.archives-ouvertes.fr/halshs-01122079 (visité le 10/08/2022).

Stein, Achim et Sophie Prévost. *Syntactic Reference Corpus of Medieval French (SRCMF)*. Stuttgart : ILR University of Stuttgart, 2013. isbn : 899-492-963-833-3. url : http://srcmf.org.

TEI element sourceDoc. url : https://tei-c.org/release/doc/tei-p5-doc/en/html/ref-sourceDoc.html (visité le 23/08/2022).

TEI : Text Encoding Initiative. url : https://tei-c.org/ (visité le 24/08/2022).

Turing, A. M. « Computing Machinery and Intelligence ». In : *Mind* LIX.236 (1er oct. 1950), p. 433-460. issn : 0026-4423. doi : 10.1093/mind/LIX.236.433. url : https://doi.org/10.1093/mind/LIX.236.433 (visité le 04/08/2022).

Zarri, Gian Piero. « Quelques aspects techniques de l'exploitation informatique des documents textuels : saisie des données et problèmes de sortie ». In : *Publications de l'École Française de Rome* 31.1 (1977), p. 399-413. url : https://www.persee.fr/doc/efr_0000-0000_1977_act_31_1_2286 (visité le 01/08/2022).

ref

Glossaire

CREMMALab Consortium pour la reconnaissance d''écriture manuscrite des matériaux anciens. 23

Handwritten Text Recognition La reconnaissance du texte écrit sur une image numérique. 3

HTR-United HTR-United is a catalog and an ecosystem for sharing and finding ground truth for optical character or handwritten text recognition (OCR/HTR). 21–24

IIIF Image API Un service de web qui renvoie une image suite à une requête standardisée HTTP(S). L'URI peut préciser la région, la taille, la rotation, la qualité, les caractéristiques, et le format de l'image demandée.. 39, 40, 43

Inria Institut national de recherche en sciences et technologies du numérique. 17–19, 24

International Image Interoperability Framework Normes internationales de l'exploitation des images numériques et de leurs métadonnées par API. 38

One Document Does it all Un fichier XML TEI qui précise les règles d'un schème TEI personnalisé.. 46

Optical Character Recognition La reconnaissance des polices du texte sur une image numérique. 6

École nationale des chartes Grande école bla bla bla. 19

Acronymes

ALMAnaCH Automatic Language Modelling and Analysis & Computational Humanities. 18, 19, 22

ALTO Analyzed Layout and Text Object. i, 27, 41–45, 47–49, 51, 52, 54, 55, 78

API Application Programming Interface. 38–41, 45

ARK Archival Resource Key. 37–39, 41, 45, 78

BnF Bibliothèque nationale de France. i, 17–19, 38–40, 42, 45

CREMMA Consortium Reconnaissance d'Écriture Manuscrite des Matériaux Anciens. 23, 24

DTS Distributed Text Services. 37, 46

ENC École nationale des chartes. 19

HTML HyperText Markup Language. 43

HTR Handwritten Text Recognition. i, 3–7, 9, 10, 12–15, 17, 19, 20, 22, 23, 26, 27, 37, 40–45, 48, 50, 55

IIIF International Image Interoperability Framework. 37–39, 41, 45, 46, 78

Inria Institut national de recherche en sciences et technologies du numérique. 18

JSON JavaScript Object Notation. 40, 43

METS Metadata Encoding and Transmission Standard. 47, 48

OCR Optical Character Recognition. i, 6, 7, 9–13, 23, 42, 47, 48, 50, 55

ODD One Document Does it all. 46

RDF Resource Description Framework. 37, 46

TAL Traitement automatique des langues. 14, 17–19, 24–26, 28, 40, 45, 46, 54

TEI Text Encoding Initiative. i, 27, 28, 37, 42, 44–46, 52, 54, 55

XML eXtensible Markup Language. 27, 41–45, 47–49, 52

Introduction

Première partie

Présentation du projet

Chapitre 1

Qu'est-ce que l'HTR ?

Qu'est-ce qu'est l'*Handwritten Text Recognition* ou la reconnaissance automatique d'écriture manuscrite ? L'*Handwritten Text Recognition* (HTR) est l'une des meilleurs approches aujourd'hui pour prédire du texte à partir d'une image numérique, y compris les manuscrits et les imprimés. Le développement de cette technologie occupe les chercheurs depuis plus d'un siècle. Pour expliquer comment l'HTR fonctionne, il faut comprendre dans un premier temps ce qu'est une image numérique et comment les différents éléments qui la compose : la mise en page (telle que les zones de textes et les illustrations), les lignes, ainsi que le texte proprement dit.

1.1 Le fonctionnement général de l'HTR

1.1.1 L'image numérique

L'un des objectifs de l'HTR est d'imiter l'œil humain et reconnaître les lignes et les points d'une écriture sur une image numérisée contenant du texte. Une image numérique consiste en un quadrillage qui se compose de petits carrés qui s'appellent des pixels. Issu de l'anglais, le mot pixel veut dire *picture element* et désigne l'élément minimal d'une image numérique. Les pixels sont stockés sous le format *bitmap* ou BMP et chaque pixel peut compter 1 bit, 4 bits, 8 bits, ou 24 bits selon l'encodage de l'image. Quelque soit l'encodage, chaque pixel n'a qu'une seule couleur composée des degrés du rouge, vert, et bleu (système RVB). Vu ensemble, un groupe de pixels peut donner l'impression des courbes d'un objet ou d'un caractère. L'exemple de figure 1.1 montre la diagonale de la lettre « A » écrite en crayon rouge.

Un scanner encode l'image en décomposant une image en unités de pixel et en donnant à chaque pixel un tableau de trois entiers du système RVB (cf. Figure 1.2). Le premier entier qui se trouve dans la donnée tripartite d'un pixel défini l'intensité de la couleur rouge ; le deuxième entier celle de la couleur vert et le troisième celle de la couleur bleu. Avec un peu de distance, l'œil humain, à partir de ce quadrillage, distingue les formes

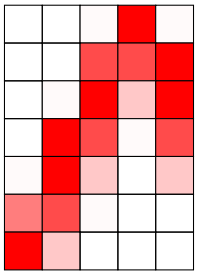


FIGURE 1.1 – Une couleur par pixel

qui constituent une image. L'HTR vise à identifier et reconnaître les points contiguës d'un caractère.

255,255,255	255,255,255	255,250,250	255,0,0	255,250,250
255,255,255	255,255,255	255,75,75	255,75,75	255,0,0
255,255,255	255,250,250	255,0,0	255,200,200	255,0,0
255,255,255	255,0,0	255,75,75	255,250,250	255,75,75
255,250,250	255,0,0	255,200,200	255,255,255	255,200,200
255,125,125	255,75,75	255,250,250	255,255,255	255,255,255
255,0,0	255,200,200	255,255,255	255,255,255	255,255,255

FIGURE 1.2 – Donnée RVB

1.1.2 Les tâches d'un logiciel HTR

Pour aboutir à une prédiction HTR, la tâche consiste en trois étapes : la reconnaissance des différentes zones de la page, la reconnaissance des lignes où se trouve du texte, et la transcription du texte. L'analyse de la mise en page n'est pas obligatoire, mais les

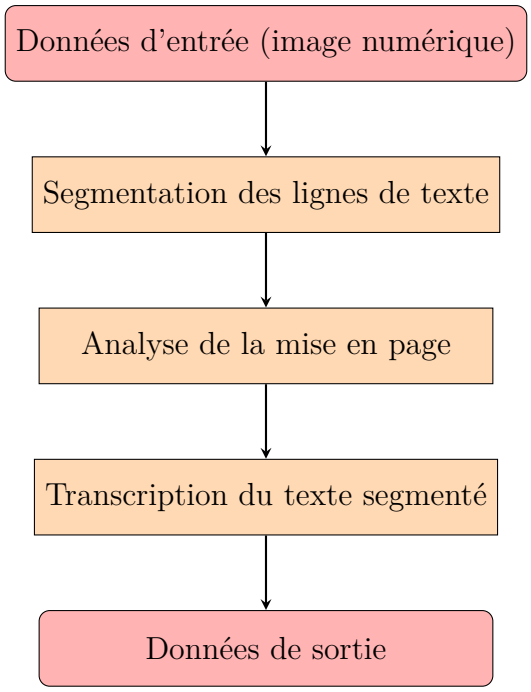


FIGURE 1.3 – Processus d'un logiciel HTR

deux autres tâches sont essentielles à l'HTR. Chacune exige son propre modèle entraîné spécifiquement pour sa tâche. (cf. Figure 1.3)

La segmentation des zones de la page

La première tâche de la reconnaissance du texte est de localiser l'emplacement du texte. Cette étape s'appelle souvent la segmentation et elle est indispensable.[1] Selon son entraînement, un modèle de segmentation recherche les espaces entre les contours d'un caractère ou entre les lignes de texte. Si le modèle est entraîné pour reconnaître du texte écrit en japonais, par exemple, il rechercherait l'ensemble de caractères bordés à gauche et à droit de l'espace et les reconnaîtrait comme étant une ligne de texte. Ainsi que tracer la limite d'une ligne de texte, il faut aussi tracer la limite d'un caractère. L'ensemble de coordonnées qui encadre une telle entité s'appelle un « masque ».[2] Cet objet contiguë permet au modèle de reconnaître du texte.

1. Alix CHAGUÉ, Thibault CLÉRICE et Laurent ROMARY. « HTR-United : Mutualisons La Vérité de Terrain ! » In : *DHNord2021 - Publier, Partager, Réutiliser Les Données de La Recherche : Les Data Papers et Leurs Enjeux*. Lille, France : MESHS, nov. 2021. URL : `https://hal.archives-ouvertes.fr/hal-03398740` (visité le 10/08/2022).

2. Denis COQUENET, Clément CHATELAIN et Thierry PAQUET. « Handwritten Text Recognition : From Isolated Text Lines to Whole Documents ». In : *ORASIS 2021*. Saint Ferréol, France : Centre National de la Recherche Scientifique [CNRS], sept. 2021. URL : `https://hal.archives-ouvertes.fr/hal-03339648` (visité le 04/08/2022).

La transcription

La transcription du texte est l'étape fondamentale de l'HTR. Cette dernière s'effectue à partir des données encadrées dans les masques. Il faut ce qu'on appelle un modèle HTR [3]. La sortie d'un logiciel HTR peut être enrichie et inclure, en plus du texte brut reconnu, les données de la segmentation. En tout cas, la phase de transcription est essentielle dans un protocol de reconnaissance automatique du texte.

1.1.3 Les origines de l'HTR

L'*Handwritten Text Recognition* a évolué à partir des recherches sur l'*Optical Character Recognition*. Les origines de cette dernière méthode remontent au dix-neuvième siècle. Les progrès les plus importants de cette technologie ont eu lieu il y a plus qu'un demi-siècle aux États-Unis. Le développement des logiciels OCR a été réalisé dans le cadre de l'amélioration du traitement en masse des données numérisées. Dans les années soixante, le besoin de gérer et traiter de grandes quantités de données est devenu de plus en plus pressant notamment dans les grandes entreprises, dont les banques. La numérisation des journaux et le traitement en masse des lettres à la poste furent aussi l'un des contextes importants du développement de la reconnaissance automatique du texte sur des images.

Dans les années mille neuf cent soixante, de plus en plus de sociétés souhaitaient adopter les méthodes de traitement des données assisté par ordinateur. Ce traitement a exigé l'encodage des données dans un format directement exploitable par ordinateur. Il y avait alors deux moyens pour la saisie des données : (i) la carte perforée dans laquelle une machine faisait des trous qu'un ordinateur savait lire. En deuxième temps, (ii) la bande magnétique sur laquelle une machine imprimait des bytes, la plus petite unité exploitable par ordinateur.

L'*Optical Character Recognition* comptait sur ce dernier moyen, ainsi que sur une autre technologie, la numérisation des images par scanner. L'atout de l'OCR est qu'en principe la saisie de données se fait automatiquement. À l'époque, les autres moyens d'encoder les données textuelles exigeaient leur saisie manuelle. Dans le cas des cartes perforées, un individu lisait un document et tapait le texte à la perforatrice. Le *Magnetic Tape/Selectric Typewriter* (MT/ST), développé par l'entreprise américaine IBM en 1964, comptait également sur la saisie manuelle des données, mais les encodait sur une bande magnétique au lieu d'une carte perforée.

En 1971, le chercheur Ben R. Schneider a comparé l'OCR et ces deux autres moyens d'encodage du texte. [4] Schneider a déterminé que l'OCR n'avait pas encore surpassé l'ap-

3. l'acronyme HTR peut soit renvoyer uniquement à cette étape, soit plus généralement au processus complet qui comprend la reconnaîssance du texte, mais aussi la segmentation des zones et des lignes

4. Ben R. Schneider. « The Production of Machine-Readable Text : Some of the Variables ». In : *Computers and the Humanities* 6.1 (sept. 1971), p. 39-47. ISSN : 0010-4817, 1572-8412. DOI : 10.1007/

pareil MT/ST d'IBM parce que, malgré sa saisie automatique de données, il exigeait toujours beaucoup de correction à la main, car contrairement aux deux autres méthodes, les données d'entrée du traitement OCR n'étaient pas créées par un humain mais par un scanner. L'appareil MT/ST avait donc une exactitude supérieure à l'OCR à l'époque.

En outre, un logiciel OCR était limité par son jeu de données de polices, puisqu'il prédit du texte en faisant un comparaison entre un caractère qu'il a mémorisé et le motif qu'il a reconnu sur l'image. Même aujourd'hui l'OCR se distingue de son successeur l'HTR par le fait qu'il compte sur une base de données des polices. Contraire à l'OCR dans les années soixante-dix, l'appareil MT/ST pouvait encoder des documents des diverses polices en comptant sur le discernement d'un être humain lors de la saisie des données.

En 1977, Gian Piero Zarri, en résumant l'état de l'art de l'OCR, a remarqué que la reconnaissance du texte sur les manuscrits n'était pas encore faisable.

> Rappelons que la reconnaissance optique des caractères permet la lecture directe du texte par un « scanner » qui se charge d'effectuer le transfert sur bande magnétique ; le texte doit être composé avec des caractères de type « imprimerie » ou « machine à écrire », car la lecture de caractères « manuels » ne semble pas encore actuellement complètement sortie de la phase expérimentale. [5]

Dans les mêmes années, le chercheur américain Ray Kurzweil fait le même constat : *Kurzweil* ou le *Kurzweil Reading Machine* (KRM). L'OCR peut reconnaître plusieurs polices. [6] Cependant, comme dit Zarri, l'OCR reste sous la dépendance de la reconnaissance des polices et donc ne peut pas encore parvenir à la prédiction du texte écrit, ce qui a amené au développement de l'*Handwritten Text Recognition*.

1.1.4 La binarisation

Afin de reconnaître la police d'un caractère, les logiciels OCR du XXe siècle avaient besoin d'un processus préliminaire qui s'appelle la binarisation. Depuis quelques années, cette étape n'est plus nécessaire pour l'HTR, mais il est toujours courant pour les logiciels OCR contemporains. [7] Comme expliquent Patrick Jentsch et Stephan Porada, « The idea is to only extract the pixels which actually belong to the characters and discard any other

BF02402324. URL : http://link.springer.com/10.1007/BF02402324 (visité le 02/08/2022).

5. Gian Piero Zarri. « Quelques aspects techniques de l'exploitation informatique des documents textuels : saisie des données et problèmes de sortie ». In : *Publications de l'École Française de Rome* 31.1 (1977), p. 399-413. URL : https://www.persee.fr/doc/efr_0000-0000_1977_act_31_1_2286 (visité le 01/08/2022).

6. Gregory Goodrich. « Kurzweil Reading Machine : A Partial Evaluation of Its Optical Character Recognition Error Rate ». In : *Journal of Visual Impairment and Blindness* (12 jan. 1979).

7. Patrick Jentsch et Stephan Porada. « From Text to Data Digitization, Text Analysis and Corpus Linguistics ». In : *Digital Methods in the Humanities : Challenges, Ideas, Perspectives*. Sous la dir. de Silke Schwandt. Bielefeld University Press, 2021.

pixel information which, for example, is part of the background [8]. » La binarisation trie les pixels d'une image en deux classes : l'arrière-plan (*background* en anglais) et le premier plan (*foreground* en anglais).

Normalement pour binariser une image, on veut remplacer la valeur composée d'un pixel, c'est-à-dire les trois degrés du rouge, du vert, et du bleu, avec la valeur simple d'un décimal. Ce décimal veut représenter l'échelle des gris dans le pixel. En anglais, ce traitement s'appelle le *grayscale* ou le niveau de gris en français. Aujourd'hui les langages de programmation ont souvent des librairies qui fournissent des méthodes pour rendre une image en niveau de gris automatiquement. Mais dans la figure 1.4, nous donnons un calcul simple qui permet le traitement des niveaux de gris. On commence toujours avec la donnée tripartite du pixel, qu'on veut remplacer par une seule valeur. Représentons chaque partie de la donnée du pixel par les variables p :

$$p_1, p_2, p_3$$

En premier temps, on prend la moyenne des degrés de la couleur, c'est-à-dire la moyenne de p ou $\bar{p}$. La Figure 1.6a montre le résultat de ce calcul superposé à l'image originale.

$$\bar{p} = \sum_{i=1}^{3} \frac{1}{3} p_i = \frac{1}{3}(p_1 + p_2 + p_3)$$

Ensuite, on récupère $\bar{p}$ et la divise par la valeur maximale du p, qui est 255 parce que le degré du rouge, vert, ou bleu d'un pixel ne monte qu'à 255. La Figure 1.6b montre le résultat de ce calcul.

Donnée tripartite du pixel p_1, p_2, p_3	Moyenne du pixel $\sum_{i=1}^{3} p_i$	Valeur niveau de gris du pixel $\frac{\sum_{i=1}^{3} p_i}{\max\{p_i\}}$
255,000,000	$\bar{p} = \frac{255+0+0}{3} = 85$	$\frac{\bar{p}}{255} = 0.33$
255,075,075	$\bar{p} = \frac{255+75+75}{3} = 135$	$\frac{\bar{p}}{255} = 0.53$
255,125,125	$\bar{p} = \frac{255+125+125}{3} = 168.33$	$\frac{\bar{p}}{255} = 0.66$
255,200,200	$\bar{p} = \frac{255+200+200}{3} = 218.33$	$\frac{\bar{p}}{255} = 0.86$
255,250,250	$\bar{p} = \frac{255+220+220}{3} = 251.67$	$\frac{\bar{p}}{255} = 0.99$
255,255,255	$\bar{p} = \frac{255+255+255}{3} = 255$	$\frac{\bar{p}}{255} = 1.00$

FIGURE 1.4 – Évaluation des pixels

Il y a plusieurs techniques de binarisation mais toute a besoin d'un seuil (*threshold*

8. JENTSCH et PORADA, « From Text to Data Digitization, Text Analysis and Corpus Linguistics », p. 107.

en anglais). Le seuil nous permet à trier les pixels en les deux classes : le *background* et le *foreground*. Nos yeux font cette étape facilement, mais un ordinateur a besoin d'un algorithme. L'un des algorithmes les plus courant pour définir le seuil a été élaboré en 1979 par le chercheur Nobuyuki Otsu. [9]

La méthode Otsu de seuillage reste toujours courante dans l'HTR [10] et elle fait partie de la librairie Python `numpy`.. [11] Elle examine la variance entre les deux classes (*background* et *foreground*) pour déterminer un seuil idéal pour le jeu de données. Visualisées dans un histogramme, comme on voit dans la figure 1.5, les données d'un jeu de pixels devraient se lever dans deux sommets et, idéalement, une baisse profonde devrait les diviser. Cette variance veut dire qu'il y a dans l'image une distinction importante entre les contours d'un caractère et l'arrière-plan de l'image. La méthode de seuillage examine la variance entre ces deux classes, le contour et l'arrière-plan, pour générer un seuil adapté aux données.

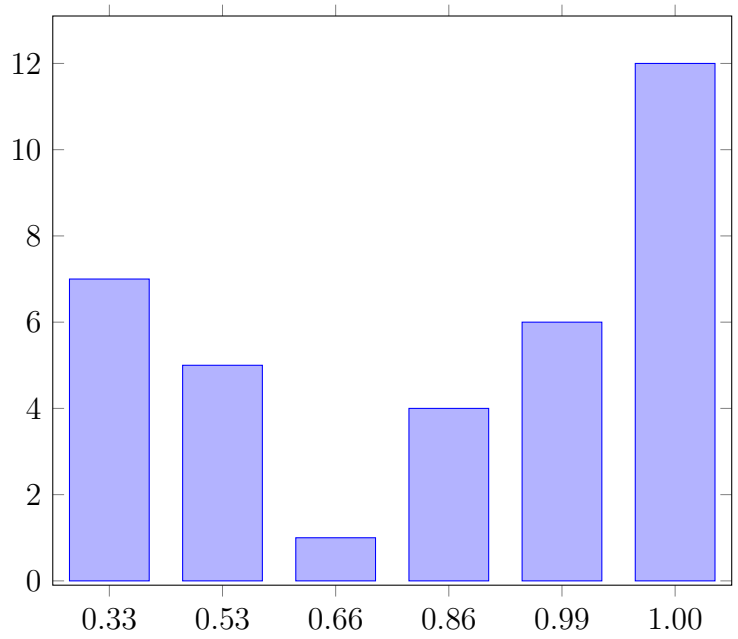


FIGURE 1.5 – Histogramme des valeurs niveau de gris des pixels

Le seuil sert à transformer la valeur numérique de chaque pixel soit en 0, soit en 1. Les pixels dont la valeur tombe au-dessous du seuil prennent la valeur 0 ; les autres, étant déterminés de faire partie du contour du caractère, prennent la valeur 1. La Figure 1.6c montre un exemple du résultat de ce triage. Ainsi, les logiciels OCR et HTR peuvent analyser les valeurs identiques et contiguës dans les données de l'image. Mais le logiciel

9. Nobuyuki OTSU. « A Threshold Selection Method from Gray-Level Histograms ». In : *IEEE Transactions on Systems, Man, and Cybernetics* 9.1 (jan. 1979), p. 62-66.

10. Imran-Ahmed SIDDIQI, Florence CLOPPET et Nicole VINCENT. « Writing Property Descriptors : A Proposal for Typological Groupings ». In : *Gazette du livre médiéval* 56.1 (2011), p. 42-57. DOI : 10.3406/galim.2011.1981. URL : https://www.persee.fr/doc/galim_0753-5015_2011_num_56_1_1981 (visité le 02/08/2022).

11. *Numpy : NumPy Is the Fundamental Package for Array Computing with Python*. Version 1.23.1. URL : https://www.numpy.org (visité le 04/08/2022).

255	255	251.67	85	251.67
255	255	135	135	85
255	251.67	85	218.33	85
255	85	135	251.67	135
251.67	85	218.33	255	218.33
168.33	135	251.67	255	255
85	218.33	255	255	255

(a) La moyenne de chaque pixel

1.00	1.00	0.98	0.33	1.98
1.00	1.00	0.53	0.53	0.33
1.00	0.98	0.33	0.86	0.33
1.00	0.33	0.53	0.98	0.53
0.98	0.33	0.86	1.00	0.86
0.66	0.53	0.98	1.00	1.00
0.33	0.86	1.00	1.00	1.00

(b) L'image en niveau de gris

0	0	0	1	0
0	0	1	1	1
0	0	1	0	1
0	1	1	0	1
0	1	0	0	0
1	1	0	0	0
1	0	0	0	0

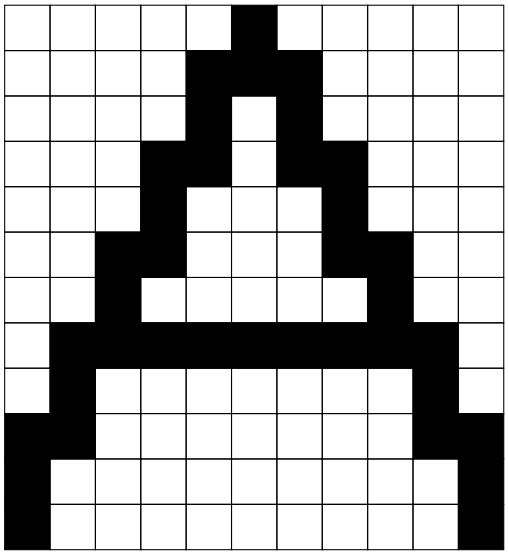
(c) L'image binarisée

FIGURE 1.6 – La binarisation

n'est pas encore prêt à percevoir l'occurence d'un caractère.

1.1.5 L'algèbre linéaire, les matrices, et l'intelligence artificielle

On transforme les pixels d'un caractère en une matrice, telle que celle présentée dans la figure 1.7b. Pourquoi ? À la base, l'ordinateur est une calculatrice. Les logiciels OCR et HTR décomposent une image numérique en représentations mathématiques, les matrices. Ainsi, les matrices permettent aux logiciels de faire des calculs probabilistes et de prédire quel caractère correspond à quelle représentation. Mais pour faire des prédictions, un logiciel a besoin d'une intelligence artificielle.



(a) Le masque de la lettre A,
sur l'image binarisée

$$\begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}$$

(b) La matrice du masque de
la lettre A

FIGURE 1.7 – La matrice d'un caractère

Template matching

Pendant la dernière moitié du XXe siècle, la reconnaissance du texte et l'intelligence artificielle ont connu des progrès importants. En 1950, il y a plus que soixante-dix ans, Alan Turing a publié sa théorisation de l'IA. [12] Pendant les années soixante, au début de l'OCR, l'IA a rendu possible la prédiction du texte représenté par les matrices en correspondant la représentation qu'un logiciel a reconnue à la représentation d'un caractère qu'il avait dans

12. A. M. TURING. « Computing Machinery and Intelligence ». In : *Mind* LIX.236 (1er oct. 1950), p. 433-460. ISSN : 0026-4423. DOI : 10.1093/mind/LIX.236.433. URL : https://doi.org/10.1093/mind/LIX.236.433 (visité le 04/08/2022).

sa base de données. Cette technique primitive est le *template matching* ou le filtrage par modèle. Elle n'est pas le moyen le plus pratique. Les chercheurs V. K. Govindan et A. P. Shivaprasad l'ont apprécié en 1990, après qu'elle a été dépassée par le *feature analysis*.

> [Template matching] directly compares an input character to a standard set of prototypes stored [*modèles stockés*]. The prototype that matches most closely provides recognition. […] This type of technique suffers from sensitivity to noise and is not adaptive to differences in writing style. [13]

Sous la dépendance du *template matching*, un logiciel OCR ne parviendra pas à préduire un caractère si la totalité de sa représentation en matrice n'est pas assez proche du modèle stocké lors de sa programmation.

Feature analysis

Les progrès dans le domaine de l'IA pendant les années soixante-dix et quatre-vingt ont rendu possible le développement d'une nouvelle technique dont profite toujours les logiciels OCR, quoique avec beaucoup d'élaboration depuis. Au lieu de faire correspondre la représentation d'un caractère à une autre, les logiciels OCR décomposent un caractère en ses *features* ou ses aspects. En 1990, Govindan et Shivaprasad ont résumé l'état de cette technique.

> The features may represent global and local properties of the characters. These include strokes, and bays in various directions, end points, intersections of line segments, loops […], and stroke relations, angular properties, sharp protrusions […]. These features have high tolerances to distortions and style variations, and also tolerate a certain degree of translation and rotation. However, the extraction processes are in general very complex and it is difficult to generate masks for these types of features. [14]

Ainsi, le logiciel OCR essaie de faire correspondre l'ensemble de certains aspects d'une représentation à l'ensemble des mêmes aspects aux lesquels le logiciel associe un caractère. Selon Govindan et Shivaprasad, un aspect peut être un trait, le croisement des lignes, une boucle, la relation entre plusieurs traits, la caractéristique des angles, ou une saillie.

Grâce à la technique de *feature analysis*, un logiciel OCR moderne parvient à prendre une décision quant à la représentation d'un caractère bien qu'il n'ait pas trouvé le caractère de la même police, ayant la même représentation, dans sa base de données. *L'Handwritten Text Recognition* utilise aussi du *feature analysis*. Mais à la place de produire les métadonnées sur la police du caractère segmenté et reconnu ainsi que la langue du texte, l'HTR

13. V. K. Govindan et A. P. Shivaprasad. « Character Recognition — A Review ». In : *Pattern Recognition* 23.7 (1990), p. 671. ISSN : 0031-3203. URL : `https://www.academia.edu/6986960/Character_recognition_A_review` (visité le 05/08/2022).

14. Ibid.

analyse les aspects d'un caractère simplement pour prédire le texte. Cependant, cette analyse n'est pas plus simple que celle réalisée par un logiciel OCR.

Le logiciel HTR a besoin d'associer beaucoup d'aspects d'une variété immense à un seul caractère puisqu'une main peut écrire un caractère par plusieurs moyens, selon la position de la lettre dans un mot. En outre, l'écriture d'une main peut varier et une page peut avoir plusieurs mains. Dit simplement, l'OCR analyse les aspects d'un caractère du point de vue d'une langue et d'une police attendue. L'HTR se focalise uniquement sur les aspects topologiques d'un caractère, sans avoir besoin de savoir la langue ni d'avoir appris la police du texte.

Deep learning

Aujourd'hui l'HTR profite du *feature analysis* ainsi que d'un développement plus récent dans l'intelligence artificielle qui s'appelle le *deep learning* ou l'apprentissage profond. Pouvant s'améliorer grâce aux réseaux neurones, un logiciel OCR ou HTR moderne peut prendre les décisions de plus en plus pertinente quand il essaie de prédire du texte à partir de l'analyse des aspects (*feature analysis*). L'une des premieres mises en œuvre des reseaux neurones pour *l'Handwritten Text Recognition* était le projet european *Transkribus.*[15] Un autre projet européen a suivi *Transkribus* et, contrairement au premier, laisse ses données et les architectures de ses modèles ouvertes : *Kraken.*[16] Élaboré dans l'esprit de la science ouverte, le projet *Gallic(orpor)a* a choisi d'utiliser *Kraken* et une interface de transcription, elle aussi ouverte, *eScriptorium.*

1.2 Le modèle HTR

Puisque *Transkribus* et *Kraken* profitent tous les deux de l'apprentissage profond, les processus mis en œuvre par les interfaces graphiques *Transkribus* et *eScriptorium* ressemblent généralement à la même description. Ils commencent avec l'entraînement d'un modèle HTR. Ensuite, ils appliquent le modèle entraîné aux données d'entrées, c'est-à-dire l'image d'un page numérisée. Le taux de réussite se calcule par le pourcentage des caractères que le modèle n'a jamais vus mais qui étaient bien prédits.

15. Guenter Muehlberger et al. « Transforming Scholarship in the Archives through Handwritten Text Recognition : Transkribus as a Case Study ». In : *Journal of Documentation* 75.5 (9 sept. 2019), p. 954-976. ISSN : 0022-0418. DOI : 10.1108/JD-07-2018-0114. URL : https://www.emerald.com/insight/content/doi/10.1108/JD-07-2018-0114/full/html (visité le 04/08/2022).

16. Benjamin Kiessling. « Kraken - an Universal Text Recognizer for the Humanities ». In : ADHO 2019 - Utrecht. 2019. URL : https://dh-abstracts.library.cmu.edu/works/9912 (visité le 10/08/2022).

1.2.1 Les données d'entrée

En premier temps, avant de penser à l'entraînement d'un modèle, il faut bien connaître sa saisie des données. Les données d'entrée d'un modèle vont être les images numériques, composées des pixels. En général, leurs contenus textuels devraient se ressembler afin que le modèle se spécialise dans une écriture particulière. Il est pourtant possible d'entraîner un modèle très généraliste, mais il nécessite un très large corpus d'entraînement.

1.2.2 Les données d'entraînement

Le premier défi de l'entraînement d'un modèle HTR est la création des données d'entraînement. Ces données sont des transcriptions annotées et corrigées à la main à partir des images. La paire formée par une image et sa transcription s'appelle une vérité de terrain ou *ground truth* en anglais, puisque la transcription doit être parfaite ou *vraie*[17]. Afin d'entraîner un modèle HTR, il faut un jeu des vérités de terrain. Alix Chagué décrit les vérités de terrain comme,

> des ensembles de données annotées et corrigées de manière à fournir au modèle des paires composées d'une part d'une image ou d'une portion d'image (entrée) et d'autre part de l'annotation attendue (sortie), qui peut être des coordonnées dans le cas de la segmentation ou un ensemble de caractères pour la transcription. [18]

Lors de l'apprentissage, les vérités de terrain fournissent au modèle en cours son résultat souhaité pour qu'il puisse savoir comment s'améliorer afin de produire les bonnes prédictions ou les prédictions *vraies*. Les données produites reproduire au plus près la prédiction idéale des vérités de terrain. Grâce à l'apprentissage profonde, un modèle peut apprendre comment arriver à la prédiction souhaitée à partir de données d'entraînement.

1.2.3 L'entraînement

Lors de l'entraînement, le modèle HTR (comme un modèle TAL)) s'évalue périodiquement et une dernière fois quand l'entraînement est terminié. La dernière évaluation s'appelle le *score*. Afin d'obtenir un meilleur *score*, on peut optimiser l'entraînement du modèle en appuyant sur plusieurs paramètres.

Par exemple, on peut modifier le nombre de fois que le modèle révise sa manière de faire ses tâches de prédiction. Chaque essaie s'appelle un époch, dérivé de l'anglais *epoch*,

17. David Lassner et al. « Publishing an OCR Ground Truth Data Set for Reuse in an Unclear Copyright Setting. Two Case Studies with Legal and Technical Solutions to Enable a Collective OCR Ground Truth Data Set Effort ». Version 1.0. In : *Fabrikation von Erkenntnis – Experimente in den Digital Humanities. Hg. von Manuel Burghardt* Lisa Dieckmann (2021). Avec la coll. d'Herzog August Bibliothek, 5). doi : 10.17175/SB005_006. url : https://zfdg.de/sb005_006 (visité le 10/08/2022).

18. Chagué, Clérice et Romary, « HTR-United ».

et un plus grand nombre d'époch donne au modèle plus d'essais à s'améliorer. Cependant, plus d'époch pèse plus sur le budget d'un projet puisqu'il consomme plus de puissance de calcul et plus de temps. En outre, on ne veut pas faire passer trop d'époch et risquer le sur-apprentissage d'un modèle, qui s'appelle le *overfitting* en anglais. Cela veut dire que la fonction prédictive du modèle s'est trop bien adaptée à ses données d'entraînement, y compris tout le bruit des données, et elle n'est pas suffisamment généraliste pour réussir sur des données que le modèle n'avais jamais vues. On peut également régler la taille du pas d'apprentissage après chaque époch, c'est à dire son *learning rate*.

On peut aussi modifier la composition du jeu de données traitée lors d'un époch. Ce dernier paramètre s'appelle un *batch size*, et il est aussi un entier, comme l'époch. Disons qu'on veut entraîner notre modèle sur 400 images. Un époch prendrait trop de temps et trop de puissance de calcul s'il essayait de traiter toutes les images de notre jeu de données au même temps. Du coup, on veut le diviser selon notre paramètre *batch size*. Si on déclarait un *batch size* de 100 images, on dirait au modèle qu'il faut itérer sur son jeu de données 4 fois afin de compléter un époch, en rappelant qu'un époch est égal à une passe complète sur le jeu de données d'entraînement, voir la figure 1.8. Ce qu'on appelle l'erreur, soit la différence entre la prédiction du modèle et la vérité de terrain, se calcule à chaque itération d'un *batch* dans un époch. On veut que cette valeur diminue, c'est à dire que la prédiction du modèle se rapproche de plus en plus à la vérité de terrain. Pour optimiser le modèle, on peut choisir entre plusieurs fonctions pour calculer l'erreur, ce qu'on appelle une *loss function*. Dans l'exemple de la Figure 1.8, on pourrait appliquer une *loss function*, telle que le *mean squared error* (MSE), aux toutes les prédiction d'un *batch* afin de connaître l'érreur ou le *loss* du modèle à ce point de l'entraînement.

Dans le cadre du projet *Gallic(orpor)a*, Ariane Pinche a entraîné des modèles HTR sur le corpus gold, que l'équipe a fait à la main et qu'elle et Simon Gabay ont vérifié. Pinche l'a scindé en trois, comme la Figure 1.9 visualise. Une majorité des images (80%) compose le jeu d'entraînement (*training set*). Une petite partie (10%) compose le jeu de validation (*development set*). Et la dernière partie (10%) compose le jeu de test (*testing set*). Les données du *training set* sont ceux qui se divisent en *batch* ; chacune est traitée une fois lors d'un époch. En prenant compte du taux de réussite du modèle sur les données du *training set*, le réseau neural fait des ajustements aux connexions entre ses neurones. Après quelques ajustements, l'entraînement s'arrête pour évaluer le modèle en cours de développement. Les données du *development set* sont utilisées pour donner au réseau une évaluation impartiale de son succès puisqu'elles sont les données qu'il n'a jamais vues. Après chaque moment de validation, le réseau neural oublie les données du *development set*, afin qu'il ne s'apprenne pas par eux, et reprend son entraînement avec les données du *training set*. En fin, les données du *testing set* sont utilisées à la fin de l'entraînement pour donner un *score* final au modèle sur les données qu'il n'a jamais vues.

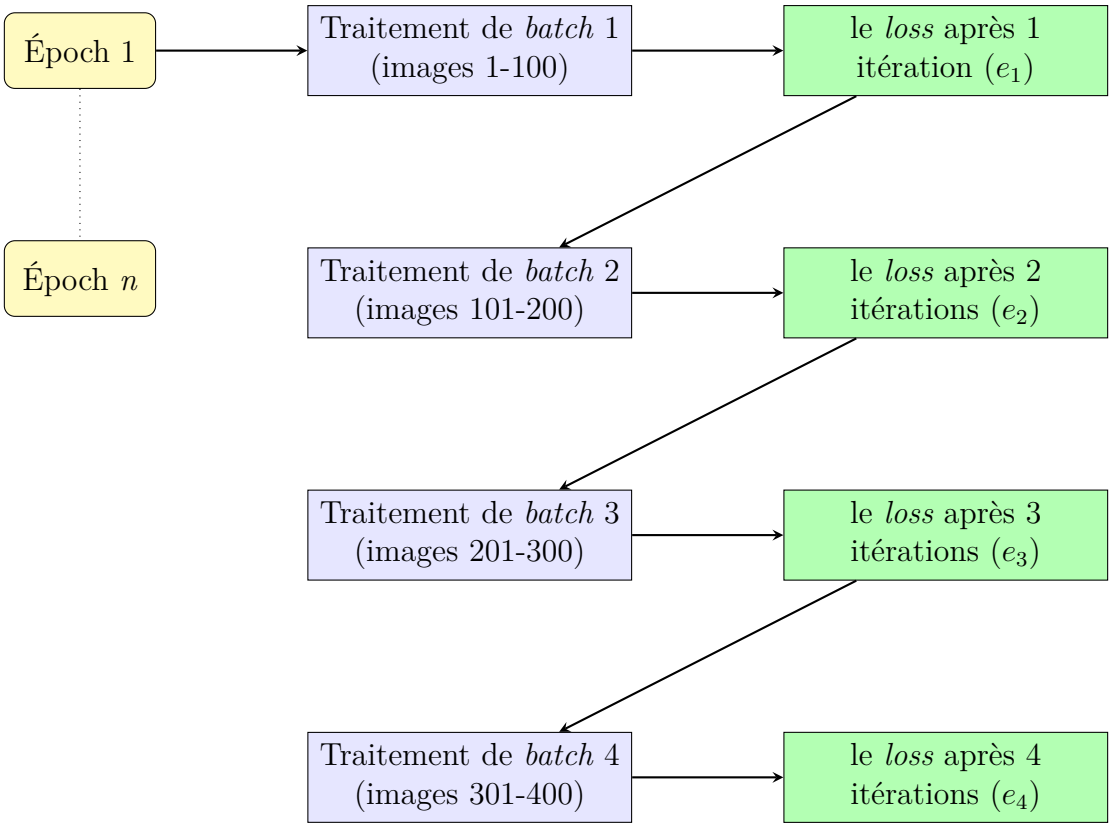


FIGURE 1.8 – La visualisation d'époch 1

FIGURE 1.9 – La division du corpus gold pour l'entraînement d'un modèle

1.2.4 Le résultat de l'entraînement

Le résultat de l'entraînement s'appelle un modèle. L'un des buts du projet *Gallic(orpor)a* était de proposer pour tous les documents du XVe au XIXe siècle deux modèles généralistes. Un se spécialisera dans l'écriture des manuscrits et des incunables. Un autre modèle s'entraînera pour les imprimés.

Chapitre 2

Le rêve du projet *Gallic(orpor)a*

Le projet *Gallic(orpor)a* s'est développé à partir de plusieurs projets précédents et il tire parti de divers domaines de recherche. Ses créateurs, en mettant en valeur leurs propres connaissances, ont visé à assembler un pipeline qui peut traiter tout document dans la base de données Gallica de la Bibliothèque nationale de France (BnF). Les chercheurs spécialisés en *l'Handwritten Text Recognition* (HTR), en le Traitement automatique des langues (TAL), en l'histoire, en la littérature, en la lexicographie et en la stylométrie se sont rassemblés pour réaliser ce pipeline. Le pipeline visait à prédire et analyser du ancien français et du français de l'Ancien Régime, ainsi que les manuscrits et les imprimés, à partir des pages numérisées des documents créés entre 1400 et la révolution française. Cependant, le vrai rêve du projet était de produire un prototype qui servirait d'exemple et pourrait être élaboré dans le but de traiter vraiment tout document source numérisé.

Les ambitions du *Gallic(orpor)a* se sont rendu possibles grâce aux recherches de plusieurs chercheurs et ingénieurs, tel que Laurent Romary, Philippe Gambette, Thibault Clérice, Pedro Suarez Ortiz, Claire Jahan, Caroline Corbières, et Alexandre Bartz. Mais les principaux qui se chargeaient de la surveillance du projet *Gallic(orpor)a* lors de mon stage en 2022 étaient Jean-Baptiste Camps, Simon Gabay, et Ariane Pinche, qui ont développé des modèles HTR pour extraire du texte des document numériques dans la base de données Gallica. Chez Inria, en tant que stagiaire, j'ai aussi travaillé en collaboration avec Benoît Sagot et Rachel Bawden, qui ont développé des outils d'analyse linguistique du texte extrait. Tous ensemble, ces chercheurs de divers spécialités ont contribué leurs connaissances pour produire un processus du traitement polyvalent.

2.1 Le contexte du projet

Bibliothèque nationale de France et le Data Lab

Le Data Lab s'est mis en place au sein de la Bibliothèque nationale de France (BnF) en 2021.[1] Lors de sa première année, le Data Lab a lancé son premier appel aux projets qui mettent en valeur les fonds et les ressources de l'institution phare patrimoniale. Le projet *Gallic(orpor)a* faisait partie des premiers projets acceptés en 2021, à côté des projets *AUREJ* (Accès Unifié aux REssources de la Jouabilité), *GALLICAENV*, *BUZZ-F*, et *AGODA* (Analyse sémantique et Graphes relationnels pour l'Ouverture et l'étude des Débats à l'Assemblée nationale).[2] Ayant sa candidature retenu, *Gallic(orpora)* profitait du financement du Data Lab de la BnF. La plupart du travail sur le projet a eu lieu pendant la première moitié de 2022, suite au mis en place du stage et des vacations par Ariane Pinche, Simon Gabay, et Benoît Sagot.

Inria et l'équipe ALMAnaCH

Inria est l'Institut national de recherche en sciences et technologies du numérique et il compte plusieurs branches dans le monde. La branche parisienne encadre l'équipe ALMAnaCH dont le acronyme veut dire *Automatic Language Modelling and Analysis & Computational Humanities*. Au sein d'ALMAnaCH s'est développé le meilleur modèle TAL pour la langue française, CamemBERT.[3] L'équipe ALMAnaCH encadre les chercheurs, les ingénieurs, les doctorants, et les stagiaires attachés aux projets concernés soit par le traitement automatique des langues, soit par les humanités numériques. L'acronyme du nom fait référence à ces deux pôles de recherche : *Automatic Language Modelling and Analysis* est le traitement automatique des langues, et le *Computational Humanities* est l'humanités numériques. Le projet *Gallic(orpor)a* se situait entre les deux, impliquant l'extraction des données et l'édition des documents historiques ainsi que l'analyse linguistique du texte extrait.

Le directeur de recherches d'ALMAnaCH est Benoît Sagot, qui s'est chargé de l'encadrement du stage du projet *Gallic(orpor)a*. En tant que stagiaire, je faisais partie de l'équipe entre début avril et fin juillet 2022. Pendant le stage, Rachel Bawden a animé un groupe de lecture hebdomadaire et des séminaires mensuelles dont j'ai profité dans

1. Marie Carlin et Arnaud Laborderie. « Le BnF DataLab, Un Service Aux Chercheurs En Humanités Numériques ». In : *Humanités numériques* 4 (déc. 2021). URL : https://hal-bnf.archives-ouvertes.fr/hal-03285816 (visité le 11/08/2022).

2. Bibliothèque nationale de France. *Rapport d'activité 2021 de la Bibliothèque nationale de France*. Paris, France, 1er juill. 2022. URL : https://www.bnf.fr/fr/bnf-rapport-dactivite-2021 (visité le 09/08/2022), p. 123.

3. Louis Martin et al. « CamemBERT : A Tasty French Language Model ». In : *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. ACL 2020. Online : Association for Computational Linguistics, juill. 2020, p. 7203-7219. DOI : 10.18653/v1/2020.acl-main.645. URL : https://aclanthology.org/2020.acl-main.645 (visité le 11/08/2022).

l'intérêt de me tenir au courant sur les nouvelles recherches et les nouveaux enjeux du TAL. Elle a aussi développé un modèle TAL pour le projet *Gallic(orpor)a* et m'a instruit dans sa mise en œuvre. [4] Disposée d'un bureau, j'ai travaillé en présentiel quatre jours par semaine, passant un jour toutes les semaines à l'École nationale des chartes pour travailler à côté de l'une des chefs du projet, Ariane Pinche. Chez Inria, j'ai profité de l'expertise de mes collègues de l'équipe ALMAnaCH, en particulier Alix Chagué et Hugo Scheithauer. L'équipe entière de *Gallic(orpor)a* a aussi profité des serveurs d'Inria, qui prennaient en charge une partie de la puissance de calcul et du stockage de données pour l'interface graphique HTR *eScriptorium*.

École nationale des chartes et l'université de Genève

En tant qu'une école, contrairement à une équipe de recherche comme ALMAnaCH, les rôles de l'École nationale des chartes et l'université de Genève dans le projet *Gallic(orpor)a* concernés l'encadrement des chercheurs qui y ont contribué leurs connaissances. L'École nationale des chartes (ENC) a aussi donné un lieu de travail, dont j'ai profité un jour par semaine. Ariane Pinche, qui était post-doctorante à l'École nationale des chartes, et Simon Gabay, maître-assistant à l'université de Genève, ils ont géré la mise en place du stage et des vacations que la bourse du Data Lab de la BnF a financés pour 2022. L'ENC et l'université de Genève les ont soutenu lors de l'encadrement des vacations et du stage.

Gabay, Pinche, et deux autres chercheurs qui étaient attachés à l'École nationale des chartes pendant le stage, Jean-Baptiste Camps et Thibault Clérice, ont tous contribué au projet *Gallic(orpor)a*. Gabay et Pinche se sont occupés de l'harmonisation des vérités de terrain produites par l'équipe en reliant toute transcription que les vacataires ont faite dans l'interface graphique *eScriptorium*. Pinche et Clérice ont commencé à utilisé les vérités de terrain des documents médiévaux en entraînant des nouveaux modèles de l'HTR et de la segmentation. [5] Par rapport à la segmentation, Jean-Baptiste Camps, Pinche, et Gabay ont développé le syntaxe *SegmOnto* qui servait à harmoniser les vérités de terrain produites pour tout document dans le corpus d'entraînement, y compris les manuscrits et les imprimés. [6] Bien que chaque chercheur ait ses spécialités, ils ont tous collaboré et la

4. Rachel Bawden et al. « Automatic Normalisation of Early Modern French ». In : LREC 2022 - 13th Language Resources and Evaluation Conference. 20 juin 2022. DOI : 10.5281/zenodo.5865428. URL : https://hal.inria.fr/hal-03540226 (visité le 11/08/2022) ; Simon Gabay. *FreEMcorpora/FreEMnorm : FreEM Norm Parallel Corpus*. Zenodo, 17 jan. 2022. DOI : 10.5281/zenodo.5865428. URL : https://zenodo.org/record/5865428 (visité le 11/08/2022).

5. Thibault Clérice. *YALTAi : Segmonto Manuscript and Early Printed Book Dataset*. Zenodo, 10 juill. 2022. DOI : 10.5281/zenodo.6814770. URL : https://zenodo.org/record/6814770 (visité le 12/08/2022).

6. Simon Gabay et al. « SegmOnto : Common Vocabulary and Practices for Analysing the Layout of Manuscripts (and More) ». In : *1st International Workshop on Computational Paleography (IWCP@ICDAR 2021)*. Lausanne, Switzerland, sept. 2021. URL : https://hal.archives-ouvertes.fr/hal-03336528 (visité le 09/08/2022).

division des aspects du projet n'étaient pas aussi fermes qu'ils n'ont pas profité des idées de l'un et de l'autre.

2.2 La portée des données d'entraînement

Le projet *Gallic(orpor)a* visait à développer et mettre en pratique les modèles HTR pouvant traiter tout document français dans la base de données Gallica crée entre le XIVe siècle et la révolution française. Un corpus des documents sources numérisés ainsi qu'une sélection de leurs pages à transcrire étaient choisis comme les données d'entraînement d'un tel modèle HTR. Les membres de ce corpus sont détaillés dans le section A.1 de l'appendice. Le choix se faisait en pensant à la diversité linguistique et à la diversité du genre. Comme montre la Figure 2.1, il y avait les manuscrits, les incunables, et les imprimés. De plus, chaque type de document a porté plusieurs genres littéraires, y compris la poésie, le récit, et le traité. Dans le cas des imprimés, il y avait aussi des pièces de théâtre.

			Ecriture							
Type	Genre	Siècle	Null		antiqua		cursive		gothique	
manuscrit	Poésie	13								2
		15						3		
	Récit	13								7
		14						1		6
		15						6		
		16								1
	Traité	14								1
incunable	Poésie	15								2
	Récit	15								5
		16				1				1
	Traité	15								2
		16				1				2
imprimé	Poésie	16				3				
		17				7				
		18				4				
	Récit	16		1		2		1		2
		17		1		9				
		18				6				
	Théâtre	16				2				
		17				5				
		18				3				
	Traité	16				2				1
		17				3				
		18				12				

FIGURE 2.1 – Diversité linguistique et générique

Pour terminer, un autre souhait quant à la diversité du corpus a porté sur le lieu

de publication ou d'apparition du document, comme montre la Figure 2.2. La plupart des documents choisis de la base de données Gallica étaient sortis de Paris. Une autre partie importante est venue de Lyon. Il y avait aussi un effort d'inclure les manuscrits, les incunables, et les imprimés écrits en français qui sont venus des villes hors de la France, tel que Londres, Amsterdam, et Rome.

FIGURE 2.2 – Diversité géographique

Après avoir établi le corpus, les pages ciblées dans chaque document étaient transcrites par des vacataires en utilisant l'interface graphique *eScriptorium*, qui permet à la fois la transcription du texte et la segmentation de la page. La segmentation de la page se faisant en mettant les étiquettes précises sur les masques des lignes et les bloques de texte ; ces étiquettes suivaient le vocabulaire *SegmOnto*. Ensuite, les vérités de terrain produites avec l'interface graphique *eScriptorium* étaient transférées vers les dépôts Github du bon siècle. L'outil `HTRVX` du projet HTR-United a analysé toute transcription transféré. L'outil s'est alerté aux erreurs potentielles de la segmentation et il a facilité le nettoyage des données.

2.3 Les prédécesseurs du projet

Comme expliqué avant, *Gallic(orpor)a* a rassemblé les recherches de plusieurs projets précédents. Il a profité des glossaires codicologiques, des progrès dans la prédiction et la segmentation des documents, des progrès dans l'analyse linguistique, et des catalogues

des données. Le glossaire *SegmOnto* se servait à harmoniser les vérités de terrain pour les manuscrits et les imprimés transcrits. Les modèles HTR étaient à la fois un support à la production des vérités de terrain, en faisant en premier temps une transcription préliminaire, et le but du projet, étant de produire les modèles entraînés sur le vocabulaire *SegmOnto*. Le catalogue HTR-United, spécifiquement son outil `HTRVX`, a surveillé l'harmonisation des transcriptions produites comme des vérités de terrain. [7] En outre, le catalogue HTR-United a publié gratuitement les vérités de terrain du projet *Gallic(orpor)a* pour que d'autres projets et d'autres modèles HTR puissent en profiter. [8] Pour terminer, l'analyse linguistique était aussi un objectif du projet *Gallic(orpor)a*, et la mise en œuvre de cet aspect comptait sur les modèles de la langue française construits par les chercheurs de l'équipe ALMAnaCH.

Le glossaire codicologique

Le projet *Gallic(orpro)a* a harmonisé ses données selon le vocabulaire *SegmOnto*, qui est expliqué en détail dans le chapitre 3. Ayant géré les données produites selon cette codicologie, j'ai aussi contribué à l'élaboration et le perfectionnement du vocabulaire. La décision d'utiliser le syntaxe descriptif des lignes et des zones de *SegmOnto* a été prise bien en avance de la mise en œuvre du projet *Gallic(orpor)a*. La généralité du vocabulaire était déterminée d'être conforme à la diversité ciblée des documents traités dans le cadre du projet. Puisque *Gallic(orpor)a* visait à livrer un prototype d'un pipeline qui pourrait traiter tout document source numérisé, la généralisation des étiquettes appliquées aux lignes et aux zones était impérative, et le vocabulaire de *SegmOnto* était jugé la meilleure solution.

La segmentation et la prédiction du texte

Les progrès de la reconnaissance du texte sont expliqués en détail dans le chapitre 1. Un logiciel HTR commence par la segmentation de la page, et dès qu'il sait où se trouvent les caractères, les mots, et les lignes du texte il le prédit à partir de l'écriture. Ces deux tâches se font selon les compétences qu'il a appris lors de son entraînement. Dans le but de produire les vérités de terrain pouvant entraîner les modèles HTR pour les manuscrits, les incunables, et les imprimés dans la base de données Gallica, le projet *Gallic(orpor)a* a profité de l'expertise de Simon Gabay et d'Ariane Pinche, qui s'occupaient de la relecture des vérités de terrain et la gestion des corpus d'or.

7. Thibault Clérice et Ariane Pinche. *HTRVX, HTR Validation with XSD*. Version 0.0.1. Sept. 2021. DOI : 10.5281/zenodo.5359963. URL : https://github.com/HTR-United/HTRVX (visité le 12/08/2022).

8. Alix Chagué et Thibault Clérice. « Sharing HTR Datasets with Standardized Metadata : The HTR-United Initiative ». In : Documents Anciens et Reconnaissance Automatique Des Écritures Manuscrites. 23 juin 2022. URL : https://hal.inria.fr/hal-03703989 (visité le 12/08/2022).

Les progrès dans la prédiction du texte sur les imprimés de l'Ancien Régime ainsi que son analyse ont aidé le projet *Gallic(orpor)a*. Par rapport aux progrès dans l'OCR des imprimés françaises de l'Ancien Régime, Gabay a entraîné les modèles sur les vérités de terrain des imprimés du XVIe au XVIIIe siècle.[9] En collaboration avec d'autres chercheurs, il a travaillé sur le jeu de données OCR17+ qui a fourni des vérités de terrain des imprimés du XVIIe siècle.[10] Pour tester le pipeline, dans l'attente des modèles nouvellement entraînés sur les données produites dans le cadre de *Gallic(orpor)a*, j'ai utilisé un modèle de segmentation et un modèle d'HTR que Gabay a développé dans le cadre de son projet *E-ditiones*.[11]

Pour la reconnaissance du texte sur les documents médiévaux, le projet CREMMA-Lab est clef. Géré dans le cadre des études postdoctorales d'Ariane Pinche, le Consortium Reconnaissance d'Écriture Manuscrite des Matériaux Anciens, ou CREMMA, est un dépôt des images et leurs transcriptions corrigées à la main, c'est-à-dire des vérités de terrain. Afin d'entraîner un modèle HTR, il faut un jeu des vérités de terrain.[12] Le projet CREMMALab fournit un jeu des vérités de terrain de 13 manuscrits médiévaux qui se composent de 21 656 lignes de texte transcrites.[13] Sur les données du projet, Pinche a entraîné un modèle HTR qui est désormais disponible sur Zenodo.[14] Le projet *Gallic(orpor)a* en a profité pour aider à la création des vérités de terrain pour les manuscrits médiévaux de son propre jeu de données.

L'harmonisation et la partage des données

Le projet HTR-United mis en commun les vérités de terrain générées par tout projet *open source*.[15] Sa base de données, gratuitement mise en ligne par GitHub, contient les images et leurs transcriptions faites par plusieurs projets de recherche, et elle porte sur les documents de plusieurs périodes historiques et écritures. Un modèle HTR peut être entraîné sur ces jeux de données. Par exemple, Alix Chagué a entraîné un modèle HTR

9. Simon Gabay et al. « Standardizing Linguistic Data : Method and Tools for Annotating (Pre-Orthographic) French ». In : *Proceedings of the 2nd International Digital Tools & Uses Congress (DTUC '20)*. Hammamet, Tunisia, oct. 2020. DOI : 10.1145/3423603.3423996. URL : https://hal.archives-ouvertes.fr/hal-03018381 (visité le 12/08/2022).

10. Simon Gabay, Thibault Clérice et Christian Reul. *OCR17 : Ground Truth and Models for 17th c. French Prints (and Hopefully More)*. Mai 2020. URL : https://hal.archives-ouvertes.fr/hal-02577236 (visité le 12/08/2022).

11. Simon Gabay. *E-Ditiones, 17th c. French Sources*. Nov. 2018. URL : https://hal.archives-ouvertes.fr/hal-02388415 (visité le 10/08/2022).

12. Chagué, Clérice et Romary, « HTR-United ».

13. Ariane Pinche et Jean-Baptiste Camps. « CremmaLab Project : Transcription Guidelines and HTR Models for French Medieval Manuscripts ». In : *Colloque "Documents Anciens et Reconnaissance Automatique Des Écritures Manuscrites"*. Paris, France, juin 2022. URL : https://hal.archives-ouvertes.fr/hal-03716526 (visité le 10/08/2022).

14. Ariane Pinche. « HTR model Cremma Medieval ». In : (21 juin 2022). DOI : 10.5281/zenodo.6669508. URL : https://zenodo.org/record/6669508 (visité le 10/08/2022).

15. Alix Chagué et al. *HTR-United/Htr-United : V0.1.28*. Zenodo, 10 août 2022. DOI : 10.5281/zenodo.6979746. URL : https://zenodo.org/record/6979746 (visité le 12/08/2022).

sur les vérités de terrain du *LECTAUREP Project*, soutenu par Inria et les Archives Nationales, qui sont mis en commun sur la base de données HTR-United. [16]

Dans l'esprit de la science ouverte, le projet *Gallic(orpor)a* a transféré toute vérité de terrain de ses dépôts GitHub vers le catalogue HTR-United. À la fin du stage, en juillet 2022, Ariane Pinche et Thibault Clérice ont entraîné un modèle pour les manuscrits médiévaux en utilisant les transcriptions que l'équipe du projet *Gallic(orpor)a* ont produites. Ces vérités de terrain sont désormais mises en commun sur HTR-United et Pinche et Clérice ont lié le premier modèle publié du projet *Gallic(orpor)a* avec le catalogue HTR-United et le dépot du projet CREMMA (Consortium Reconnaissance d'Écriture Manuscrite des Matériaux Anciens). [17] La partage des données du projet est l'un de ses objectifs.

Ainsi qu'à contribuer au catalogue, le projet *Gallic(orpor)a* a aussi profité des outils de HTR-United. Le dernier met en commun des outils qui ont pour but d'harmoniser les données ajoutés à son catalogue. Ces outils peuvent être intégrés dans un *workflow* de GitHub, ce que l'équipe de *Gallic(orpor)a* a fait. L'un de ces outils est `HTRVX`, qui se prononce comme le personnage Asterix, et il a rendu possible à l'équipe du projet nettoyer les transcriptions sorties de *eScriptorium*. [18] En exemple, `HTRVX` relit les transcriptions et les cherche pour les erreurs communes. L'existence d'un tel outil et sa disponibilité gratuite grâce au projet HTR-United a beaucoup aidé le projet *Gallic(orpor)a*.

L'analyse linguistique

Après l'extraction et le nettoyage des données des documents source de Gallica, le projet *Gallic(orpor)a* a envisagé à analyser le texte. Dans cet objectif, il a profité des progrès dans l'analyse linguistique des anciens états de la langue française. L'analyse linguistique du français de l'Ancien Régime, tel que ce qui se voit dans les écrits de Molière, est un domaine de recherche actuellement en plein développement. Depuis une dizaine d'années, les chercheurs dans la linguistique computationnelle ont élaboré des outils pour analyser le français autre que le français contemporaine, dont les recherches sont déjà animées par l'application commerciale et les jeux de données plus nombreuses.

L'histoire de l'analyse linguistique et du Traitement automatique des langues (TAL) est hors de ce mémoire. Néanmoins, les projets qui ont précédés *Gallic(orpor)a* et sur lesquels il a compté méritent de discussion. Achim Stein a bordé le sujet de l'analyse linguistique du français du Moyen Âge dans son article de 2013. [19] Stein a montré que, pour les

16. Alix Chagué. *LECTAUREP Contemporary French Model (Administration)*. Zenodo, 12 mai 2022. URL : `https://zenodo.org/record/6542744` (visité le 12/08/2022).

17. Ariane Pinche et Thibault Clérice. *HTR-United/Cremma-Medieval : Cortado 2.0.0*. Zenodo, 11 juill. 2022. DOI : `10.5281/zenodo.6818057`. URL : `https://zenodo.org/record/6818057` (visité le 12/08/2022).

18. Clérice et Pinche, *HTRVX, HTR Validation with XSD*.

19. Achim Stein et Sophie Prévost. *Syntactic Annotation of Medieval Texts : The Syntactic Reference Corpus of Medieval French (SRCMF)*. Narr Verlag, 2013, p. 275. ISBN : 978-3-8233-6760-4. URL : `https://halshs.archives-ouvertes.fr/halshs-01122079` (visité le 10/08/2022).

chercheurs qui ont débuté d'appliquer les progrès dans l'analyse linguistique à l'étude du français des anciens états, il faut faire attention aux propriétés syntaxiques et morphologiques propres à la langue. Du coup, une architecture qui pourrait parvenir aux résultats souhaités pour l'anglais du Moyen Âge n'aura pas forcement le même taux de réussite avec le français du Moyen Âge à cause de différences syntactiques et morphologiques dans la langue.

Achim Stein et Sophie Prévost ont créé un *treebank* pour l'ancien français, le *Syntactic Reference Corpus of Medieval French* (SRCMF), qui avance toujours l'analyse linguistique des anciens états du français. [20] En 2014, Prévost est des autres chercheurs, Gael Guibon, Isabelle Tellier, Matthieu Constant, et Kim Gerdes, ont ajouté aux conclusions de Stein que la variation lexique de l'ancien français pose aussi un défi à l'analyse linguistique. [21] En 2019, Mathilde Regnault, Prévost, et Éric Villemonte de La Clergerie ont utilisé le SRCMF avec le MCVF (Modéliser le changement : les voies du français) pour encore élaborer notre connaissance de l'évolution du français. [22] Telles études linguistiques ont tourné la terre pour que l'analyse du texte extrait des manuscrits médiévaux et des imprimés historiques puissent voir le jour aujourd'hui.

Gallic(orpor)a a profité des progrès dans l'analyse linguistique du français historique utilisé dans les imprimés et les manuscrits de ses corpus. Jean-Baptiste Camps, Simon Gabay, Paul Fièvre, Thibault Clérice, et Florian Cafiero ont créé *Deucalion* qui est un modèle TAL pour le français de l'Ancien Régime, entraîné sur les livrets des drames du XVIIe siècle. [23] Encore plus récent est le projet *FREEM*qui veut dire *French Early Modern* [24] Dans une présentation du projet à la conférence du Traitement Automatique des Langues Naturelles à Avignon en juin 2022, les auteurs ont décrit l'un des modèles qu'ils ont développé à partir de l'étude linguistique de l'ancien français.

> L'étude de la langue ne nécessitant pas uniquement un seul outil, mais toute une gamme de solutions, nous avons adopté une approche holistique du pro-

20. Achim Stein et Sophie Prévost. *Syntactic Reference Corpus of Medieval French (SRCMF)*. Stuttgart : ILR University of Stuttgart, 2013. ISBN : 899-492-963-833-3. URL : http://srcmf.org.

21. Gaël Guibon et al. « Parsing Poorly Standardized Language Dependency on Old French ». In : *Thirteenth International Workshop on Treebanks and Linguistic Theories (TLT13)*. Sous la dir. de V. Henrich et al. Proceedings of the Thirteenth International Workshop on Treebanks and Linguistic Theories (TLT13). Tübingen, Germany, déc. 2014, p. 51-61. URL : https://hal.archives-ouvertes.fr/hal-01250959 (visité le 10/08/2022).

22. Mathilde Regnault, Sophie Prévost et Éric Villemonte de La Clergerie. « Challenges of Language Change and Variation : Towards an Extended Treebank of Medieval French ». In : *TLT 2019 - 18th International Workshop on Treebanks and Linguistic Theories*. Paris, France, août 2019. URL : https://hal.inria.fr/hal-02272560 (visité le 10/08/2022).

23. Jean-Baptiste Camps et al. « Corpus and Models for Lemmatisation and POS-tagging of Classical French Theatre ». In : *Journal of Data Mining & Digital Humanities* 2021 (Digital humanities in... 14 fév. 2021), p. 6485. ISSN : 2416-5999. DOI : 10.46298/jdmdh.6485. arXiv : 2005.07505 [cs]. URL : http://arxiv.org/abs/2005.07505 (visité le 09/08/2022).

24. Simon Gabay et al. « Le Projet FREEM : Ressources, Outils et Enjeux Pour l'étude Du Français d'Ancien Régime ». In : *TALN 2022 - Traitement Automatique Des Langues Naturelles*. Sous la dir. d'Yannick Estève et al. Avignon, France : ATALA, juin 2022, p. 154-165. URL : https://hal.archives-ouvertes.fr/hal-03701524 (visité le 10/08/2022).

> blème, en misant sur la création d'un modèle de langue dédié au français d'Ancien Régime, D'AlemBERT [25], qui devrait venir en soutien des différentes tâches de TAL envisagées. [26]

Le développement des nouveaux modèles TAL pour les anciens états du français a rendu possible la mise en œuvre d'un tel étape d'analyse au pipeline de *Gallic(orpro)a*.

2.4 Le pipeline

Ainsi que la création des vérités de terrain, qui est expliqué dans la section précédente de ce chapitre (Section 2.2), le projet *Gallic(orpor)a* avait pour but de créer un pipeline pouvant traiter tout document source de la base de données Gallica. En mettant en œuvre les modèles entraînés, il visait à générer une ressource lexicographique qui porte sur le document source. En commençant par les pages numérisés du document source, la ressource numérique présentera quatre types d'information :

1. les métadonnées à propos du document source
2. les données topologiques et linguistiques prédites par *l'Handwritten Text Recognition* (HTR)
3. le texte pré-éditorialisé, extrait du document
4. le texte analysé par les outils Traitement automatique des langues (TAL)

Chaque type d'information veut servir une utilité différente que la lectrice éventuelle ou le lecteur éventuel de la ressource pourrait désirer. Les métadonnées s'informent sur trois types de document concerné par le pipeline : (1) la ressource lexicographique qu'il créé, (2) le fac-similé numérique du document source, (3) le document source physique qui se conserve quelque part, sinon qui a été conservé avant sa disparition. Les données produites par les modèles HTR présentent la segmentation de la mise en page et la prédiction du texte. Ensuite, le texte pré-éditorialisé est présenté, sans sa segmentation, d'une manière plus convenable à l'analyse linguistique. Et en fin, la ressource présente son analyse du texte prédit grâce à l'application de certains modèles TAL.

Une visualisation du pipeline se voit dans le Figure 2.3. Elle montre en bleu les saisies des données, en vert les fichiers préliminaires que le pipeline construit, en orange sa première sortie, et en violet les divers moyens d'exploitation de sa sortie. Puisque le pipeline va générer des métadonnées et les données topologiques et linguistiques, il a besoin d'au moins deux saisies des données. L'un porte sur les métadonnées des trois types de document concernés (la ressource lexicographique elle-même, le fac-similé numérique

25. Simon Gabay et al. « From FreEM to D'AlemBERT ». In : *Proceedings of the 13th Language Resources and Evaluation Conference*. Marseille, France : European Language Resources Association, juin 2022. URL : `https://hal.inria.fr/hal-03596653` (visité le 10/08/2022).

26. Gabay et al., « Le Projet FREEM ».

du document source, le document source physique). L'autre saisie des données est l'image numérique elle-même. La première saisie peut se composer de plusieurs sources de données en ligne, afin de fournir à la ressource lexicographique autant de détail que possible. La deuxième saisie se compose du fac-similé numérique dans le format simple des images numériques, tel que JPEG, TIFF, ou PNG.

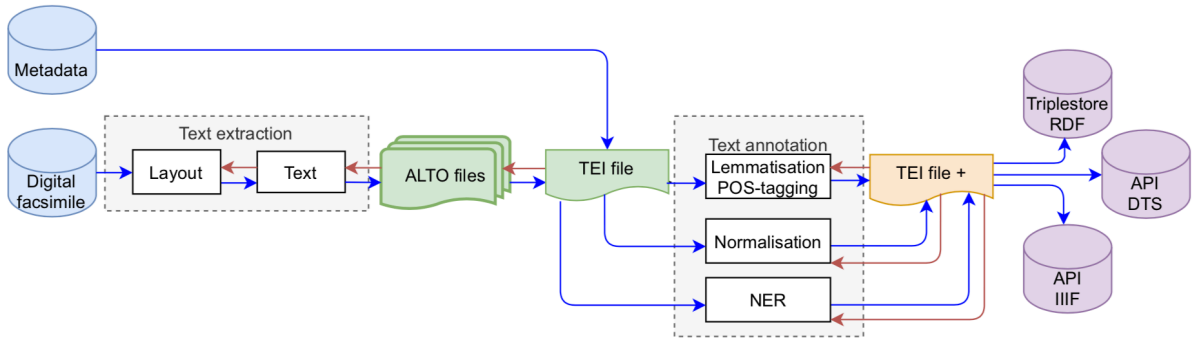


FIGURE 2.3 – Pipeline

En vert, la Figure 2.3 montre les premiers fichiers préliminaires créés par le pipeline, les fichiers ALTO. Ce format d'un fichier XML est un schéma particulièrement convenable à l'encodage de l'analyse de la mise en page que fait un modèle de segmentation et la prédiction du texte que fait un modèle HTR. Son acronyme veut dire en anglais *l'Analyzed Layout and Text Object*, ou la mise en page analysée et l'objet de texte. Comme se voit par la nature bipartite du nom *Analyzed Layout and Text Object*, les fichiers ALTO aligne la mise en page et le texte prédit. Le pipeline créé ce genre de fichier préliminaire en appliquant les modèles HTR aux données d'entrée visuelles.

Après la création des premiers fichiers préliminaires, le pipeline créé la première version du fichier TEI, qu'il va plus tard enrichir avec l'analyse linguistique du texte prédit. Ce premier fichier TEI occasionne la création des métadonnées. Comme montre la flèche bleue dans la Figure 2.3, les métadonnées sont récupérées depuis les sources en ligne et ensuite nettoyées et intégrées au fichier TEI. Le fichier TEI préliminaire récupère aussi les données des fichiers ALTO. Il est le combinaison de ces deux genres de données que rend le fichier TEI bien fait pour présenter toutes les données créés par le pipeline.

À la suite de la récupération, du nettoyage, et enfin de la transformation des données des deux sources pour les conformer au schéma TEI, le pipeline continue à traiter le texte qu'il a récupéré des fichiers ALTO. Le pipeline trait deux fois le texte prédit. En premier temps, il parse les données récupérées des fichiers ALTO et extrait toute ligne de texte imbriquée dans une zone qui fait partie du texte principal. Cela veut dire qu'il n'extrait pas de numéro de page, d'en-tête, etc. Les lignes de texte ainsi sélectionnées sont présentées comme le texte pré-éditorialisé du document source. Elles représentent une espèce de transcription du texte, en ignorant la mise en page et les autres types d'écriture sur la page qui sont communiqués autre part dans le fichier TEI.

Ce texte pré-éditorialisé peut servir à l'analyse d'une utilisatrice ou d'un utilisateur qui veut appuyer sur le texte tel qu'il était dans le document source. Ce texte sert aussi à la génération d'un texte analysé par les outils TAL. Le résultat de ce dernier traitement est visualisé dans la Figure 2.3 comme la sortie en orange, le fichier TEI enrichi, *TEI file +*. Pour terminer, les données de la sortie peuvent être exploitées dans plusieurs formats, que la Figure 2.3 visualise en violet.

Chapitre 3

Au commencement, il y avait les *guidelines SegmOnto*

3.1 La problématique

Un manuscrit se définit par ses moyens de création. Étant rédigé à la main, souvent avant la croissance de l'imprimerie, un manuscrit est un objet singulier dans le monde. Il est vrai que d'autres ressources peuvent présenter le même texte, les mêmes images, ou bien la même musique que présent un manuscrit. Cependant, un manuscrit n'a pas d'autre exemplaire. Ses contenus sont réalisés par et se répandent dans sa constitution matérielle particulière. Un manuscrit est unique avec son écriture, parfois de plusieurs mains, ses fautes d'écriture, ses parties abîmées, décorées, ou révisées, sa provenance et son histoire en tant qu'objet rare qui se transfère entre des individus, des familles, et des organisations. De plus, chaque propriétaire peut encore modifier la ressource en ajoutant ou retirant des pages, en corrigeant ou blâmant du texte ou des images, ainsi qu'en changeant la reliure et les informations portant sur l'édition.

Pour étudier un manuscrit, il faut donc développer un vocabulaire qui sait décrire les divers aspects de l'objet composé. Après tout, le pouvoir de bien définir les termes d'une étude est à la base de l'analyse. Sans un vocabulaire bien élaboré et cohérent, les arguments d'une chercheuse ou d'un chercheur ne seront pas compréhensibles. L'harmonisation d'un vocabulaire est encore plus importante eu égard à la communication des découverts et à la collaboration entre plusieurs personnes, surtout si elles ont des spécialités différentes. Ces deux activités, la communication et la collaboration, sont fondamentales à la recherche et exigent donc l'élaboration d'un vocabulaire cohérent pour décrire des manuscrits.

Voici les défis de nommage dans l'étude des manuscrits et voici l'un des obstacles que le projet *Gallic(orpor)a* a essayé de franchir. Imaginons, par exemple, qu'on veut décrire le texte principal qui se trouve sur la page d'un document. On peut y appliquer l'étiquette descriptive *Main Zone* ou bien *Principal Text*. En lisant des articles scientifiques où cha-

cun utilise une étiquette différente, un humain arriverait à reconnaître que les étiquettes différentes parlent de la même partie de la page. Mais pour un outil informatique, si la même région d'une page ne porte pas la même étiquette, il n'arriverait pas à les associer sans être instruit à chercher plusieurs versions du même concept. Dans ce cas, l'analyse serait trop compliquée à effectuer à l'échelle. C'est ainsi que la description des manuscrits est importante au projet *Gallic(orpor)a*. Le projet *Gallic(orpor)a* cherchait donc à profiter d'un vocabulaire bien élaboré et cohérent, qui pourrait décrire soit les manuscrits, soit les imprimés historiques. Il y avait plusieurs projets qui avaient cherché à répondre à cette problématique, mais celui que le projet *Gallic(orpor)a* a choisi est le vocabulaire du projet *SegmOnto*.

3.2 Les solutions proposées

Plusieurs projets avaient proposé des solutions quant à la description normalisée des manuscrits. Hors de la France, les vocabulaires ont été élaborés notamment en anglais et pour les manuscrits médiévaux. Un exemple important est la base de données *DigiPal* (Digital Resource and Database of Palaeography, Manuscripts and Diplomatic), qui n'est plus mis à jour mais qui a été développé au sein du département des humanités numériques à King's College London. [1] L'un de ses auteurs, Peter Stokes, travaille actuellement en France et continue dans la même veine en contribuant au projet *SegmOnto* qui était développé dans un environnement francophone, même si son vocabulaire est rédigé en anglais. [2] Mais le projet *SegmOnto* n'est pas le premier projet français qui a essayé d'élaborer un lexique pour les documents historiques. Ni est le projet *DigiPal* le premier en Europe. Avant leur création, la codicologie en France et en Europe suivait le modèle de Denis Muzerelle et son *Vocabulaire codicologique*.

3.2.1 Le Vocabulaire international de la codicologie

En 1985, Denis Muzerelle a conçu un vocabulaire codicologique qui avait pour but de fournir des médiévistes avec des termes uniformes pouvant décrire les aspects d'un manuscrit. [3] Depuis l'apparition de son vocabulaire en français, des autres chercheurs sont venus pour adapter les termes de Muzerelle en d'autres langues. Marilena Maniaci

1. *DigiPal : Digital Resource and Database of Palaeography, Manuscripts and Diplomatic.* London, été 2011. URL : `http://www.digipal.eu/`.

2. Simon Gabay, Jean-Baptiste Camps et Ariane Pinche. « SegmOnto ». In : *Création de Modèle(s) HTR Pour Les Documents Médiévaux En Ancien Français et Moyen Français Entre Le Xe-XIVe Siècle*. Paris, France : Ecole nationale des chartes | PSL, nov. 2021. URL : `https://hal.archives-ouvertes.fr/hal-03481089` (visité le 25/07/2022).

3. Dennis Muzerelle. *Vocabulaire Codicologique : Répertoire Méthodique Des Termes Français Relatifs Aux Manuscrits*. Rubricae 1. Paris : Éd. Cemi, 1985.

a publié une version du *Vocabulaire codicologique* pour l'italien en 1996. [4] Pilar Ostos, Luisa Pardo, et Elena Rodríguez en ont créé un pour l'espagnol l'année suivante. [5] Parfois appelé le *Vocabulaire international de la codicologie*, l'édition multilingue du *Vocabulaire codicologique* que Muzerelle a commencé en 1985 était maintenue jusqu'à l'édition d'une version 1.1. en 2002-2003. (besoin de citation)

3.2.2 La Codicologia

Aujourd'hui, la paléographie et l'étude des manuscrits peuvent profiter de l'application web *Codicologia* qui réunit le *Vocabulaire codicologique* ainsi que deux autres bases de données similaires : le projet multilingue *Lexicon* et le *Glossaire codicologique arabe*. Ses trois bases de données spécialisent dans divers écritures. Le *Vocabulaire codicologique* a été développé pour les manuscrits de l'écriture latin. Piloté par Philippe Bobichon, le projet *Lexicon* présente un vocabulaire en français pour décrire les manuscrits écrits en latin, roman, grec, hébreu, et arabe. [6] Un vocabulaire spécialisé plus profondément pour l'arabe a été élaboré dans le *Glossaire codicologique arabe* d'Anne-Marie Eddé et Marc Geoffroy. [7] Ce dernier a été conçu au sein de l'Institut de recherche et d'histoire des textes après les modèles de Muzerelle et le vocabulaire codicologique en arabe d'Adam Gacek. [8]

L'application web *Codicologia* rassemblent ces projets et présente un vocabulaire bien étendu. Par exemple, *Codicologia* fournit 15 termes pour décrire une faute d'écriture dans un manuscrit. Certains de ses termes possèdent eux-même plusieurs définitions que les divers bases de données fournissent. Le terme *caviarder*, par exemple, a une définition courte dans le vocabulaire français de Muzerelle.

> Supprimer un mot, un passage..., en le recouvrant largement d'encre, de façon à ce qu'il ne puisse être lu. [9]

Selon le *Lexicon* de Bibichon, par contre, le *caviarder* se définit d'une manière plus détaillé et vise à expliquer l'étymologie du mot afin de préciser son usage dans le cadre des manuscrits des divers écritures.

---

4. Marilena Maniaci. *Terminologia Des Libro Manoscritto*. Addenda 3. Rome : Istituto centrale per la patologia del libro, 1996.

5. Pilar Ostos, M. Luisa Pardo et Elena E. Rodríguez. *Vocabulario de codicologÍa : Versión Española Revisada y Aumentada Del Vocabulaire Codicologique*. Instrumenta Bibliologica. Madrid : Arco/Libros, 1997.

6. Philippe Bobichon. « Le Lexicon : Mise En Page et Mise En Texte Des Manuscrits Hébreux, Grecs, Latins, Romans et Arabes ». In : (2009), p. 81. URL : `https://cel.archives-ouvertes.fr/cel-00377671` (visité le 25/07/2022).

7. « Glossaire codicologique français-arabe ». In : *Gazette du livre médiéval* 40.1 (2002), p. 79-80. URL : `https://www.persee.fr/doc/galim_0753-5015_2002_num_40_1_1563` (visité le 25/07/2022).

8. Adam Gacek. *The Arabic Manuscript Tradition : A Glossary of Technical Terms and Bibliography*. Handbook of Oriental Studies 1, The Near and Middle East. Leiden : Brill, 2001. 269 p. ISBN : ISBN 90-04-12061-0.

9. Denis Muzerelle. *Caviarder*. In : *Codicologia*. Institut de recherche et d'histoire des textes, 2011. URL : `http://codicologia.irht.cnrs.fr/theme/liste_theme/413#tr-868` (visité le 25/07/2022).

> Le mot [*caviarder*] apparaît en 1907 (noircir à l'encre) : il désigne alors un procédé appliqué par la censure russe, sous Nicolas Ier. Dans certains manuscrits grecs, le détail rempli d'encre est surmonté d'un point et d'un trait court destinés à le neutraliser. Ce procédé est très souvent utilisé parmi d'autres, pour la censure des manuscrits hébreux effectué sous l'autorité de l'inquisition, en Italie, à la fin du xvie siècle et au début du xviie. [10]

Étant élaboré à partir d'un corpus très diversifié, le *Lexicon* de Bibichon a moins de termes qu'a le *Vocabulaire codicologique* mais ses termes sont plus généralisés. Le vocabulaire de Muzerelle, par contre, fait plus de distinctions entre les aspects d'un manuscrit et donc a plus de termes distincts par rapport aux deux autres vocabulaires de l'application *Codicologia*.

En réunissant les trois bases de données, sans privilégier aucun, *Codicologia* présente un vocabulaire codicologique vraiment vaste. Cependant, l'application *Codicologia*, comme toutes ses bases de données, vise à répondre au manque de cohérence dans la manière par laquelle la communauté scientifique décrit les manuscrits. Le grandeur de son vocabulaire pose un problème à cet objectif. Ayant plus de deux milles termes en français—certains d'entre eux ont eux-même plusieurs définitions—la solution proposée par *Codicologia* livre un vocabulaire bien harmonisé et documenté mais trop étendu pour être appliqué à l'échelle dans une approche informatique.

Sans un corpus d'entraînement gigantesque, qui couterait une somme énorme, l'apprentissage automatique ne peut pas faire de distinction au niveau des termes conçus par Muzerelle et les autres auteurs des bases de données de *Codicologia*. Aujourd'hui, un modèle ne peut pas s'entraîner sur des milles des étiquettes possibles et arriver à distinguer entre, par exemple, 15 types de faute d'écriture. Un humaine peut le faire, et pour cette raison les bases de données de *Codicologia* sont utiles. Mais leurs vocabulaires ne conviennent pas bien à une approche informatique.

3.3 Les *guidelines* de *SegmOnto*

Le projet *SegmOnto* propose un vocabulaire plus petit qui peut pourtant décrire une grande diversité de documents historiques, y compris les manuscrits et les imprimés. Cet objectif est encore plus compliqué à achever qu'un vocabulaire spécialisé aux manuscrits. Décrire les documents d'une diachronie longue, et sans préférence d'une écriture en particulier, exige un équilibre délicat entre la généralité et la particularité. Pour y arriver, les *guidelines* du projet *SegmOnto* limite le nombre de termes dans son vocabulaire, sans en priver aucun d'une identité distincte.

10. Philippe Bobichon. *Caviarder*. In : *Codicologia*. Institut de recherche et d'histoire des textes, 2011. URL : `http://codicologia.irht.cnrs.fr/theme/liste_theme/413#tr-868` (visité le 25/07/2022).

Les *guidelines* se divisent en deux catégories : les « zones » et les « lines ». La première parle des régions sur la page, y compris les régions de texte et les régions sans texte, tel qu'une image. Pour la plupart de temps, la catégorie de la ligne veut décrire les différents types de lignes de texte. Mais une ligne du vocabulaire *SegmOnto* peut aussi tracer la ligne d'une partition musicale ou une ligne réelle sur la page qui n'oriente pas des autres systèmes d'écriture, telle qu'une ligne qui divise la page en deux. Chacune de ces deux catégories se compose d'une liste des étiquettes, et chacune d'elles cherche à parvenir à l'équilibre entre la généralité et la particularité. Une étiquette devrait pouvoir être appliquée à soit un manuscrit, soit un imprimé, de peu importe quelle langue et quelle écriture.

3.3.1 Les zones

— **CustomZone** : une zone qui ne convient pas à aucune d'autres catégories de zone.
— **DamageZone** : une zone qui contient des marques de dégâts sur le document source, tel qu'un trou.
— **DecorationZone** : une zone qui contient un élément graphique, y compris de la peinture et les petits dessins dans la marge de la page.
— **DigitizationArtefactZone** : une zone qui contient un item qui n'appartient pas au document source mais est lié au processus de la numérisation, tel qu'une règle pour montrer la mesure du document.
— **DropCapitalZone** : une zone qui contient une initiale ; l'initiale peut prendre l'espace de plusieurs lignes de texte ou porter une décoration importante, tel que de l'historicisation, l'ornementation, ou des dessins.
— **MainZone** : une zone qui contient le texte principal du document source.
— **MarginTextZone** : une zone qui contient le texte dans la marge du document source.
— **MusicZone** : une zone qui contient une partition musicale.
— **NumberingZone** : une zone qui contient des numéros de page, y compris les numéros rédigés en chiffres romans.
— **QuireMarksZone** : une zone qui contient des notes en bas page destinées à la fabrication du document source pour garder les pages dans le bon ordre.
— **RunningTitleZone** : une zone qui contient une version du titre du document ou d'une section du document qui se trouve en tête de la page.
— **SealZone** : une zone qui contient un sceau sur le document source.
— **StampZone** : une zone qui contient l'empreint d'un tampon sur le document source.
— **TableZone** : une zone qui contient une table.

— **TitlePageZone** : une zone souvent sur l'une des premières pages du document source qui contient toutes les informations concernant le titre et l'édition du document.

3.3.2 Les lignes

— **CustomLine** : une ligne qui ne convient pas à aucune d'autres catégories de ligne.
— **DefaultLine** : une ligne qui contient du texte attendu dans la zone.
— **DropCapitalLine** : une ligne qui contient l'initiale.
— **HeadingLine** : une ligne qui contient le texte d'un titre, tel que celui d'une section ou d'un chapitre.
— **InterlinearLine** : une ligne qui traverse la page pour marquer une limite.
— **MusicLine** : une ligne de la portée d'une partition.

Deuxième partie

Exposition de la préparation et du travail d'analyse

Chapitre 4

Un pipeline visant à tout rassembler

Le pipeline du projet *Gallic(orpor)a* se déroule dans cinq étapes. En premier temps, il récupère les fac-similés numériques des documents sources sur Gallica. En deuxième temps, il applique des modèles HTR aux fac-similés téléchargés afin de produire une prédiction du texte et une transcription de la mise en page. Ensuite, il crée un fichier TEI préliminaire qui réunit les données produites par les modèles HTR et les métadonnées récupérées de plusieurs sources en ligne. Le quatrième étape enrichit le fichier TEI avec une analyse linguistique du texte de la transcription. En fin, le pipeline export les données du fichier TEI en divers formats, y compris les données conformant aux schèmes RDF, DTS, et IIIF.

4.1 La récupération des fac-similés numériques

L'accès aux pages numérisées d'un document source est une condition essentielle à toute étape du pipeline de *Gallic(orpor)a*. Afin de transcrire le document et produire la ressource lexicographique du format TEI, il faut d'abord dire au logiciel comment il peut accéder au fac-similé numérique. De plus, les images doivent être d'une qualité aussi bonne, sans être pourries par trop de bruit, que le logiciel HTR peut bien reconnaître le texte et les segments dedans. Et en fin, bien qu'il ne soit pas essentiel à l'étape de la transcription de l'image, l'image devrait s'associer aux métadonnées qui porte sur le document pour que le pipeline puisse enrichir la ressource lexicographique avec les informations portant sur le document transcrit.

4.1.1 Archival Resource Key (ARK)

Deux solutions existent pour répondre aux questions d'accès et de métadonnées. En premier temps, il existe dans le monde archivistique une espèce de clef numérique qui se connaît par l'acronyme ARK, qui veut dire en anglais *Archival Resource Key*. Cette clef est un identifiant unique qui indique une ressource, soit numérique soit physique, dont

la conservation une institution, telle que la Bibliothèque nationale de France, se charge. Le schème ARK est actuellement maintenu par la communauté ARK Alliance.[1] L'identifiant unique facilite la récupération d'une ressource en particulière, sans la confondre avec d'autres exemplaires du même document. La précision qu'accorde l'ARK améliore l'exactitude d'un processus de traitement automatique.

La mise en place d'un système de fichiers

Le pipeline se sert de l'identifiant ARK afin de gérer le lien entre les images numériques et le document source. Une partie du système de fichiers du pipeline de *Gallic(orpor)a* est visualisée dans la Figure 4.1, qui montre l'exemple de deux documents sources dans lequel chacun a trois pages numérisées en format JPEG. Le chemin d'accès à chaque image se compose donc de l'identifiant ARK. Ainsi son association au document source est conservée et accessible au logiciel.

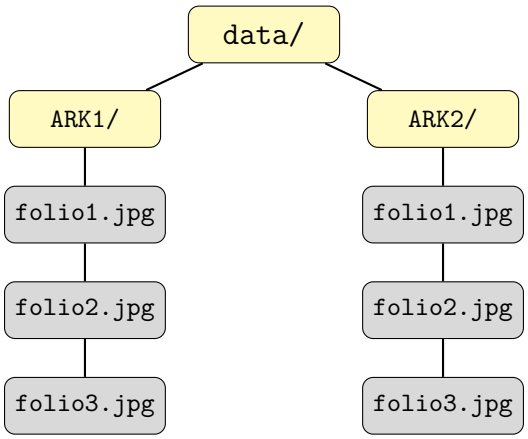


FIGURE 4.1 – Système de fichiers

4.1.2 International Image Interoperability Framework (IIIF)

En plus d'associer les images à leur document source, l'identifiant ARK sert aussi à récupérer les images depuis l'internet. Ce genre de requête se fait par un outil généralisé qui s'appelle une API (*Application Programming Interface*). Une telle interface facilite la communication entre deux ordinateurs pour qu'ils puissent échanger d'information. Ainsi, un ordinateur peut demander aux serveurs de la Bibliothèque nationale de France les images d'un document source qu'ils conservent.

Comme des autres institutions participantes, la BnF met en pratique une API spécialisée à l'échange des données visuelles qui vient d'une initiative internationale qui s'appelle *l'International Image Interoperability Framework*. Le IIIF s'engage à standardiser

1. The ARK ALLIANCE. *Community*. ARK Alliance. 6 nov. 2020. URL : `https://arks.org/community/` (visité le 23/08/2022).

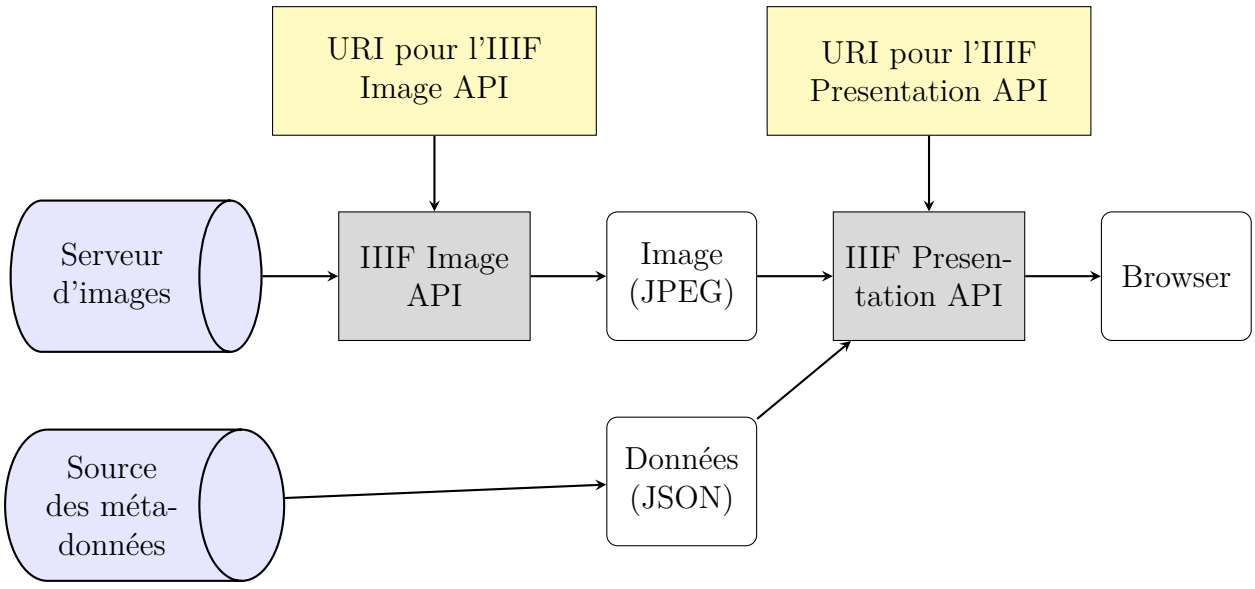
la gestion des ressources visuelles mises en ligne avec le but d'améliorer l'exploitation et la partage des ressources et de leurs métadonnées. Suite à une requête standardisée HTTP(S), l'IIIF Image API renvoie l'image. L'URI se compose de quatre éléments généralisés, suivis par cinq éléments qui peut préciser (1) la région, (2) la taille, (3) la rotation, (4) la qualité, et (5) le format de l'image demandée. Puisque la BnF s'engage à l'initiative de l'IIIF, toute image sur sa base de données Gallica peut être requêtée par l'URL que l'IIIF Image API attend.

scheme	`https://`
server	`gallica.bnf.fr`
prefix	`/iiif`
identifier	`/ark:12148/bpt6k1057726c`
folio	`/f1`
region	`/full`
size	`/full`
rotation	`/0`
quality	`/native`
format	`.jpg`

(a) URI pour l'IIIF Image API

scheme	`https://`
server	`gallica.bnf.fr`
prefix	`/iiif`
identifier	`/ark:12148/bpt6k1057726c`
manifest	`/manifest`
format	`.json`

(b) URI pour l'IIIF Presentation API



(c) IIIF Image API et IIIF Presentation API

FIGURE 4.2 – IIIF APIs

La Figure 4.2 montre comment construire les URI qui se servent de la requête à une API IIIF. Prenons le document source de l'identifiant ARK bpt6k1057726c. Pour récupérer ses pages numérisées depuis la base de données Gallica, on envoie une requête à l'IIIF Image API ; l'exemple pour récupérer la totalité de la première page du document en format JPEG, sans rotation ni d'autre modification, se voit dans la Figure 4.2a. Comme

la Figure 4.2c visualise, une telle requête HTTPS envoyée à l'IIIF Image API renverra un objet numérique de l'image. Afin d'accès aux métadonnées de l'image, on enverrait une requête d'une autre structure à la Presentaiton API, montrée dans la Figure 4.2b, qui renverrait des données en format JSON.

4.1.3 La mise en pratique

Dans le cadre du projet ARTL@S en 2020, Caroline Corbières a développé un script pour importer à partir de l'IIIF Image API les pages d'un fac-similé numérique stocké sur les serveurs de la BnF.[2] Le projet *Gallic(orpor)a* a profité du travail de Corbières et l'a utilisé pour récupérer des fac-similés numériques ciblés par l'utilisatrice ou l'utilisateur du pipeline. En ajoutant une option au script (`-l`) je l'ai modifié afin de permettre qu'une utilisatrice ou utilisateur puisse limiter les nombres de pages récupérées. La limite évite le téléchargement de trop d'images ainsi qu'économise l'énergie dépensée. Ainsi, l'utilisatrice ou l'utilisateur peut tester la mise en œuvre des modèles HTR ou TAL sur un panel restreint des données.

4.2 L'application des modèles HTR

Le pipeline du projet *Gallic(orpor)a* visait à transcrire les ressources textuelles avec un modèle HTR qui convient bien au type du document, en rappelant qu'un modèle se spécialise dans une écriture particulière grâce à son entraînement. Cela exige donc que le pipeline a d'accès aux modèles entraînés et qu'il prend la décision automatiquement sur quel modèle convient auquel document. L'utilisateur ou l'utilisatrice doit donner des modèles au pipeline, pour qu'il puisse les appliquer aux données visuelles. Par contre, l'utilisateur ou l'utilisatrice ne doit pas forcément prendre la décision. Le pipeline saura quel modèle il doit appliquer en interrogeant les métadonnées du document et en recherchant sa langue et son siècle de création.

4.2.1 L'entraînement des modèles

Dans le cadre du projet *Gallic(orpor)a*, l'équipe a entraîné des nouveaux modèles généralisés, l'un pour les manuscrits et les incunables d'une écriture latine, l'autre pour les imprimés créées avant la révolution française et aussi d'une écriture latine. Ces modèles ont besoin des vérités de terrain produites lors du projet *Gallic(orpor)a*. Bien que le pipeline ne refasse pas la création de ces vérités de terrain, ne de l'entraînement des modèles, il

2. Simon GABAY et al. « Automating Artl@s – Extracting Data from Exhibition Catalogues ». In : *EADH 2021 - Second International Conference of the European Association for Digital Humanities*. Krasnoyarsk, Russia, sept. 2021. URL : `https://hal.archives-ouvertes.fr/hal-03331838` (visité le 23/08/2022).

compte de cet aspect du projet puisqu'il a besoin des modèles. Néanmoins, le pipeline peut s'adapter aux divers modèles et mettre en œuvre le modèle souhaités d'une 'utilisatrice ou d'un l'utilisateur. En s'adaptant aux modèles publiés ainsi qu'aux modèles toujours en cours, qu'une utilisatrice ou un utilisateur lui donne directement, le pipeline facilite les expériences scientifiques quant à l'HTR appliqué aux plusieurs corpus.

La création des vérités de terrain

Lors de la première moitié de 2022, l'équipe du projet *Gallic(orpor)a* se composait des vacataires qui se chargeaient de la création des vérités de terrain pour les nouveaux modèles HTR. Les chefs du projet Simon Gabay et Ariane Pinche ont sélectionné un corpus de documents à transcrire. Les membres de l'équipe se spécialisaient dans certains types de document sur la liste, tels que les manuscrits et les incunables médiévaux. Un vacataire prendrait l'un des documents du corpus et saisir son identifiant ARK dans l'interface de transcription *eScriptorium*, dont je parle dans le chapitre 1. L'interface imite ce que le pipeline du *Gallic(orpor)a* fait, ainsi que le script de Corbières, en téléchargeant les images IIIF du document.

L'interface *eScriptorium* facilite la segmentation et la transcription des pages. Les vacataires l'ont utilisaient pour segmenter les zones de la page et y mettre les étiquettes conformant au vocabulaire *SegmOnto*. Portant les étiquettes de *SegmOnto*, les transcriptions aident à entraîner les modèles avec un vocabulaire cohérent pour tout type de document, y compris les manuscrits et les imprimés. L'interface *eScriptorium* export les vérités de terrain dans un fichier XML ALTO avec des images télécharges en format JPEG ou PNG.

4.2.2 La sélection des modèles

Dans l'intérêt de maximiser l'automatisation du pipeline, la mise en œuvre d'un modèle s'effectue automatiquement, selon une analyse des métadonnées du document à traiter. Grâce au fait que les images sont diffusées par une API IIIF, le pipeline a d'accès à leurs métadonnées. Comme montre la Figure 4.2c, une requête au format de l'URI pour l'IIIF Presentation API renvoie des données qui portent sur la date de création ou d'apparition du document ainsi que la langue principale de son texte. Ces deux critères informent le pipeline du type de modèle à privilégier pour le document. Néanmoins, en l'absence du modèle parfait donné lors de l'installation de l'application *Gallic(orpor)a*, les paramètres du pipeline met en œuvre un modèle de segmentation et un modèle HTR désigné les modèles par défaut.

4.2.3 La sortie des modèles segmentation et HTR

Les modèles HTR produisent des fichiers ALTO, qui est un schème XML. La structure de données souhaitées dans le fichier final du pipeline est également de l'XML, qui se définie par l'imbrication des données. Mais au final, le pipeline produira un fichier TEI. La sortie des modèles HTR se conforme au schème ALTO et donc a besoin d'être transformée en TEI.

4.2.4 Le schème ALTO

La Bibliothèque nationale de France définit le schéma ALTO comme, « un schéma XML standardisé, qui permet de stocker les informations relatives à la structure physique et au texte extrait par OCR ». [3] Les logiciels HTR exportent aussi leurs prédictions dans le format XML ALTO. Il y a les schémas XML qui conviennent bien à l'encodage du texte, tel que la TEI. Mais le liaison entre le texte et la mise en page, transcrite par un modèle de segmentation, est mieux établi par le schéma ALTO qui spécialise dans l'encodage de la structure physique d'une ressource textuelle.

Le schéma ALTO a été développé dans le cadre du projet METS (*Metadata Encoding and Transmission Schema*). Le dernier date de 2001. [4] Comme résume la Bibliothèque nationale de France,

> METS renseigne alors sur la structure logique de la page (nature sémantique des blocs de texte, par exemple titre, partie d'article, légende d'illustration, etc.), tandis qu'ALTO localise des contenants (blocs, lignes, etc.) sur la matrice de l'image de la page. [5]

Le schéma ALTO vise donc à renseigner sur la mise en page ainsi que sur le texte d'une page. Certains de ses éléments, tel que l'élément ALTO `<polygon>`, par exemple, précise les coordonnées d'une région ou d'une zone sur la page qu'avait prédite un modèle de segmentation. Cependant, certains attributs, tel que l'attribut `@CONTENT`, s'attribuent à certains parties de l'image du texte prédit. Par exemple, un schème ALTO encoderait une ligne de texte disant une phrase incomplète, *oyseaux lesq̃lz ie esperoye pren*, dans l'architecture XML qui ressemble à ce qui se voit dans la Figure 4.3. Tout cela veut dire que le schéma ALTO réussit à réunir les données structurelles de la mise en page et les données textuelles de la transcription prédite.

Le schéma XML ALTO peut se réaliser dans plusieurs formats, et pas uniquement celui qui se voit dans la Figure 4.3. Dans l'exemple de la Figure 4.3, les données textuelles

3. Bertrand Caron. *Formats de Données Pour La Préservation à Long Terme : La Politique de La BnF*. Technical Report. Bibliothèque Nationale de France (Paris), oct. 2021. URL : `https://hal-bnf.archives-ouvertes.fr/hal-03374030` (visité le 23/08/2022), p. 75.

4. *METS : Metadata Encoding and Transmission Standard*. BnF - Site institutionnel. URL : `https://www.bnf.fr/fr/mets-metadata-encoding-and-transmission-standard` (visité le 23/08/2022).

5. Caron, *Formats de Données Pour La Préservation à Long Terme*, p. 75.

```
<TextLine ID="eSc_line_1ed06324"
    TAGREFS="LT825"
    BASELINE="1193 982 2263 969"
    HPOS="1184"
    VPOS="877"
    WIDTH="1079"
    HEIGHT="127">
    <Shape>
        <Polygon POINTS="1193 982 1184 903 1303 877 1307 877 2255 890 2263
   969 2263 995 1193 1004"/>
    </Shape>
    <String
        CONTENT="oyseaux˜lesqlz ie esperoye pren"
        HPOS="1184"
        VPOS="877"
        WIDTH="1079"
        HEIGHT="127">
    </String>
</TextLine>
```

FIGURE 4.3 – L'encodage d'une ligne de texte en ALTO

sorties de la fonction prédictive du modèle HTR se trouvent dans l'attribut `@CONTENT` dans ce format. L'exemple montre un élément qui indique la région sur l'image source qu'occupe la ligne de texte, l'élément `<String>`. Son attribut `@CONTENT` porte sur le contenu textuel prédit sur cette ligne de texte.

Sinon, la sortie d'un logiciel HTR peut prendre les autres formats ALTO qui existent. L'un des défis pour le pipeline a été de le faire adapter aux divers formats d'ALTO qu'un logiciel HTR pourrait produire. Par exemple, le contenu textuel d'une ligne de texte sera encodé dans l'attribut `@CONTENT` de l'élément ALTO `<String>`, comme se voit dans la Figure 4.3, à la sortie de l'interface *eScriptorium*. Cependant, à la sortie de l'interface depuis la ligne de commande (CLI, ou *Command Line Interface*) de *Kraken*, le contenu textuel d'une ligne de texte est divisé entre les mots et les caractères reconnus dans la ligne. Par conséquent, le pipeline ne peut pas chercher le contenu de la ligne de texte dans l'attribut `@CONTENT` de l'élément `<String>`. Il doit reconstruire le contenu de la ligne de texte à partir de plusieurs éléments `<String>` qui représentent les mots dans une ligne de texte, au lieu de la ligne de texte elle-même.

4.3 Réunir la transcription et les métadonnées

Jusqu'ici, le pipeline a récupéré les images numériques en format JPEG ou PNG depuis l'IIIF Image API et les a traité avec les modèles HTR en produisant des fichiers XML ALTO. Ensuite, le pipeline recherchera les métadonnées en plusieurs formats, y compris JSON, XML, et HTML, depuis l'internet. Avec une telle diversité de structures de données, un format robuste est exigé pour tout rassembler et le mettre en ordre. Le format

choisi pour parvenir à ce défi est l'XML TEI. Ce format se réalise dans un fichier XML, qui veut dire qu'il imbrique les données dans une façon hiérarchisée. Mais contrairement au schème XML ALTO, le schème TEI se spécialise dans l'édition numérique du texte. La communauté du TEI décrit l'objectif du projet ainsi :

> The Text Encoding Initiative (TEI) is a consortium which collectively develops and maintains a standard for the representation of texts in digital form. Its chief deliverable is a set of Guidelines which specify encoding methods for machine-readable texts, chiefly in the humanities, social sciences and linguistics. [6]

En plus de la représentation du texte, le TEI dispose aussi des éléments XML qui conviennent bien à la représentation des transcriptions, y compris leurs données topologiques portant sur la mise en page. Le TEI est donc un format idéal pour fusionner les transcriptions sorties des modèles HTR, les métadonnées récupérées des sources en ligne, et le texte extrait des transcriptions. En plus, le TEI est un format très utilisé et beaucoup d'outils numériques s'y adaptent déjà.

4.3.1 L'extrait des données des fichiers ALTO

Les données sorties du traitement HTR sont du schème XML ALTO. L'un des objectifs du pipeline est de ne pas perdre aucune donnée produite par les modèles HTR, c'est-à-dire une donnée encodée dans le schéma ALTO. Il doit donc transformer la sortie des modèles en le format souhaité, le TEI. Il faut modéliser la transformation d'ALTO à TEI et la justifier puisqu'il y a plusieurs traductions possibles entre l'ALTO et le TEI.

Nous de l'équipe de *Gallic(orpor)a* avons conclu que l'élément XML TEI `<sourceDoc>` convient bien à l'encodage de toute donnée des modèles HTR portant sur le texte prédit et de la mise en page. Des autres chercheurs, tel que Hugo Scheithauer, Alix Chagué, et Laurent Romary, ont arrivés à la même conclusion. [7] Les guidelines de la Text Encoding Initiative définie l'élément `<sourceDoc>` ainsi :

> `<sourceDoc>` contains a transcription or other representation of a single source document potentially forming part of a dossier génétique or collection of sources. [8]

Les données topologiques et linguistiques sont balisés dans les éléments XML descendant du `<sourceDoc>`. Ces choix sont décrits et justifiés dans le chapitre 5.

6. *TEI : Text Encoding Initiative*. URL : `https://tei-c.org/` (visité le 24/08/2022).

7. Hugo Scheithauer, Alix Chagué et Laurent Romary. « From eScriptorium to TEI Publisher ». In : *Brace Your Digital Scholarly Edition !* Berlin, France, nov. 2021. URL : `https://hal.inria.fr/hal-03538115` (visité le 23/08/2022).

8. *TEI element sourceDoc*. URL : `https://tei-c.org/release/doc/tei-p5-doc/en/html/ref-sourceDoc.html` (visité le 23/08/2022).

4.3.2 Le récupération des métadonnées

En plus de transformer les données des fichiers ALTO, le pipeline récupère les métadonnées sur le fac-similé numérique et le document source physique, ainsi que créer les métadonnées sur la ressource lexicographique elle-même. Toutes ses métadonnées sont encodées dans l'élément XML TEI `<teiHeader>`. Cet élément est essentiel au schème TEI, mais le pipeline réussit à construire ses descendants et les remplir avec les données si le fac-similé physique s'est trouvé dans la base de données Gallica. Sinon, le pipeline laisse vide la plupart des éléments obligatoires du `<teiHeader>`.

Avec l'identifiant ARK, qu'il prend du système de fichiers, le pipeline récupère les métadonnées du document source depuis les sources en ligne. Même pour les fac-similés hébergés par divers institutions, le pipeline récupère les métadonnées diffusées directement par l'API IIIF. Les métadonnées récupérés de l'API IIIF, géré par l'institution hôte du document numérique, sont liées—si possible—avec des autres sources de données. Dans l'exemple des documents numériques de la base de données Gallica, le pipeline récupère les métadonnées quant au document source depuis l'API IIIF que la BnF met à disposition. Si cette requête a réussi, le pipeline va récupérer les données du catalogue général de la BnF en passant une requête à l'API *SRU* de la BnF qui veut dire, en anglais, le *Search/Retrieve via URL*. Pour terminer, le pipeline recherche encore des métadonnées dans le catalogue du Système Universitaire de Documentation (SUDOC) qui portent sur les institutions patrimoniales en France qui hébergent les documents sources.

4.3.3 La construction du fichier préliminaire TEI

Les données produites par les modèles HTR ainsi que les métadonnées récupérées depuis les sources en ligne sont réunies dans le fichier TEI. Les premiers dans l'élément `<sourceDoc>` et les dernières dans l'élément `<teiHeader>`. Ensuite, le pipeline commence à traiter les données intégrées au document. D'abord, il extrait les lignes de texte qui font partie des régions de la page considérées comme les principaux. Cette sélection du texte est comme une transcription puisqu'il ignore les en-têtes, les numéros de pages, etc. La transcription est ainsi encodée dans l'élément XML TEI `<body>`.

4.4 L'analyse linguistique et le fichier final

L'avant dernier étape du pipeline est d'analyser le texte. D'abord, les mots sont lemmatisés et reconnu selon leur nature, tel que nom ou verbe. La reconnaissance de la nature d'un mot se connaît par le terme anglais *part of speech* ou POS. La lemmatisation, un autre traitement lexical, divise un segment de texte en les parties et les indexer. Tout lexème pourrait ensuite être normalisé avec un modèle TAL qui transforme le mot prédit par le modèle HTR en sa version normalisée. Par exemple, un modèle TAL peut

transformer le mot prédit *nostre* en le mot normalisé *nôtre.*[9] En fin, un modèle TAL NER (*Named-Entity Recognition*) peut encore traiter le texte pour reconnaître les entités nommées, tel qu'une personne ou un lieu. Le pipeline met en œuvre plusieurs modèles TAL afin d'analyser ses divers aspects linguistiques du texte.

4.4.1 L'ODD (One Document Does it all)

Un fichier TEI se soumettent aux règles qui assurent l'uniformité. Afin de mettre en pratique les traitements à l'échelle pour tout document produit par le pipeline, il faut que tout élément du TEI soit utilisé de la même manière. Pour parvenir à une telle régularisation, le TEI dispose d'un document qui applique des règles personnalisées au produit souhaité du pipeline. Ce document se connaît par son acronyme ODD, qui veut dire *One Document Does it all* ou un fichier qui tout fait. Dans le cadre du stage, j'ai aidé à la rédaction d'un ODD qui explique en détail tout aspect du fichier TEI sorti du pipeline.

4.5 L'exploitation des données

En fin, les données encodées dans le fichier enrichi et final TEI peuvent être exploitées en tant qu'un fichier TEI ainsi que dans divers formats secondaires. Le logiciel TEI Publisher, par exemple, peut aisément publier un fichier TEI et permettre les utilisateurs de naviguer les données dans un visionneur de l'édition en ligne. Le fichier TEI peut aussi être exploité par les fichiers de transformation XSL. Il y a plusieurs formats secondaires qui pourraient servir à l'exploitation des données encodées dans le fichier TEI.

L'équipe a envisagé trois formats secondaires par lesquels les données produites par le pipeline peuvent être exploitées. L'un de ses formats est l'IIIF, le même qui a permis la construction du fichier TEI. Dans le fichier TEI, chaque attribut `@source` d'un élément `<zone>` descendant de l'élément `<sourceDoc>` contient un URI qui permet de visionner la partie de l'image concernée. Par exemple, si l'attribut `@source` descend d'un élément `<zone>` qui porte sur une ligne de texte, sa valeur donnerait dans un browser ou dans un visionneur uniquement la partie de l'image source qui contient cette ligne de texte. Ainsi les données du fichier final TEI peuvent être exploitées afin de visionner les bloques de textes, lignes de textes, les mots, et les caractères transcrits. Les deux autres formats secondaires profitent plutôt des métadonnées du fichier TEI ; ils sont le DTS (Distributed Text Services) et le RDF (Resource Description Framework).

9. Bawden et al., « Automatic Normalisation of Early Modern French ».

Chapitre 5

L'analyse des structures des données XML

5.1 ALTO : *Analyzed Layout and Text Object*

5.1.1 Qu'est-ce qu'est l'ALTO ?

Le format XML ALTO s'est évolué à partir du projet européen METAe en 2003. Le projet qui a conditionné la création d'ALTO s'occupait du développement des logiciels dont des institutions patrimoniales pourraient servir à la création et à l'exploitation des fac-similés numériques de leur fonds. Le but du projet était d'extraire à partir des pages numérisées les informations portant sur la mise en page et les autres données structurelles. D'ici 2003, les logiciels OCR étaient déjà bien mis en pratique. L'enjeu à l'époque était d'élaborer un schème de données qui soumettrait le texte extrait à la logique structurelle de la page et du document. Tandis qu'un logiciel OCR reconnaît le texte du titre d'un chapitre et le texte de son sous-titre, un nouveau format devrait pouvoir distinguer les deux lignes de texte que le logiciel a reconnues et ensuite les hiérarchiser, en disant que le sous-titre est subordonné au titre du chapitre.

Le projet METAe a donc développé un format METS (*Metadata Encoding and Transmission Standard*) qui avait pour but d'augmenter les données textuelles extraites par un logiciel OCR avec la logique de la mise en page et du document. Bien que les logiciels OCR aient souvent exporté leurs prédictions dans un format du texte brut, le format METS visait à hiérarchiser le mélange des diverses données dans un format XML. Par exemple, à travers d'un arbre XML, le texte d'un sous-titre descendrait de la région dans laquelle les caractères de cette ligne de texte s'encadrent sur la page numérisée. En outre, au moins l'un d'eux, soit la donnée sur le texte du sous-titre, soit les données sur l'emplacement, porterait quelque chose pour dire que dans la logique du document il est un sous-titre.

Le schème METS a réussi à combiner les métadonnées de la ressource numérique

ainsi que de l'objet text qui a été transcrit avec ses données structurelles et textuelles grâce au format hiérarchisé de l'XML. Mais le format primordial METS n'avait pas répondu à la question de comment bien structurer les dernières données, celles qui sont produites par un logiciel OCR ou HTR. Les créateurs du format ALTO ont décrit son prédécesseur comme un « emballage » (*wrapper*) pour la structure de données ALTO. [1] Tandis que le format METS organise les métadonnées et la logique du document, telles que l'occurence et l'ordre des pages, le format ALTO s'insère sous l'arbre de chaque page afin de décrire la transcription produite pour l'image numérique.

Normalement, un fichier XML ALTO décrit une page (ou une image) d'un document. Mais, comme se voit dans l'exemple donné dans l'exposition du schème quand il était nouveau en 2003, modélisé dans la Figure 5.1, l'élément `<Layout>` peut en fait contenir plusieurs éléments `<Page>`. [2] Néanmoins, la plupart de logiciels HTR qui utilisent le format ALTO crée un fichier per page numérisée.

```
<Layout>
  <Page ID="XXX" PHYSICAL_IMG_NR="000" HEIGHT="000" WIDTH="000" STYLEREFS="
    XXX">
    <PrintSpace ID="XXX" HPOS="000" VPOS="000" HEIGHT="000" WIDTH="000">
      <TextBlock ID="XXX" HPOS="000" VPOS="000" HEIGHT="000" WIDTH="000">
        <TextLine ID="XXX" HPOS="000" VPOS="000" HEIGHT="000" WIDTH="000">
          <String ID="XXX" HPOS="000" VPOS="000" HEIGHT="000" WIDTH="000"
    CONTENT="XXX"/>
          <Sp ID="XXX" HPOS="000" VPOS="000" HEIGHT="000" WIDTH="000"/>
          <String ID="XXX" HPOS="000" VPOS="000" HEIGHT="000" WIDTH="000"
    CONTENT="XXX"/>
        </TextLine>
        <TextLine ID="XXX" HPOS="000" VPOS="000" HEIGHT="000" WIDTH="000">
          <String ID="XXX" HPOS="000" VPOS="000" HEIGHT="000" WIDTH="000"
    CONTENT="XXX"/>
          <Sp ID="XXX" HPOS="000" VPOS="000" HEIGHT="000" WIDTH="000"/>
          <String ID="XXX" HPOS="000" VPOS="000" HEIGHT="000" WIDTH="000"
    CONTENT="XXX"/>
        </TextLine>
      </TextBlock>
    </PrintSpace>
  </Page>
  <Page ID="XXX" PHYSICAL_IMG_NR="000" HEIGHT="000" WIDTH="000" STYLEREFS="
    XXX">
    <PrintSpace ID="XXX" HPOS="000" VPOS="000" HEIGHT="000" WIDTH="000">
<!-- ... -->
</Layout>
```

FIGURE 5.1 – La structure ALTO version 1, circa 2003

Le format ALTO est dans sa quatrième version, mais la structure actuelle ressemble

1. Birgit STEHNO, Alexander EGGER et Gregor RETTI. « METAe–Automated Encoding of Digitized Texts ». In : *Literary and Linguistic Computing* 18.1 (1er avr. 2003), p. 77-88. ISSN : 0268-1145, 1477-4615. DOI : 10.1093/llc/18.1.77. URL : https://academic.oup.com/dsh/article-lookup/doi/10.1093/llc/18.1.77 (visité le 24/08/2022), p. 81.
2. Ibid.

bien au modèle qu'ont présenté les auteurs Birgit Stephno, Alexader Egger, et Gregor Retti en 2003. Dans sa première version, montrée dans la Figure 5.1, les éléments les plus petits étaient les segments de texte (`<String>`) et les espaces entre mots (`<Sp>`), balisés dans une ligne de texte (`<TextLine>`) qui appartient à un bloque de texte (`<TextBlock>`). Tous ces éléments XML porte un identifiant unique (`@ID`) et quatre coordonnées portant sur le rectangle dans lequel s'encadre le contenu de l'élément. Le contenu textuel est représenté dans l'attribut `@CONTENT` de l'élément `<String>`.

5.1.2 La structure actuelle des fichiers XML ALTO

Aujourd'hui, l'élément le plus petit d'une structure de données ALTO est un glyphe (`<Glyph>`), au lieu d'un segment de caractères (`<String>`). Par conséquent, dans le nouveau format, le contenu textuel est représenté deux fois, une fois comme l'attribut `@CONTENT` de l'élément classique `<String>` et une deuxième comme le même attribut de l'élément `<Glyph>`. Un comparaison entre les deux arborescences est visualisé dans la Figure 5.3. Comme montre la sous-figure 5.3b, la nouvelle architecture se permet d'aller en plus de détail. Certains logiciels, tel que l'interface *eScriptorium*, produisent toujours les fichiers ALTO avec une variation de l'ancienne structure où l'élément `<String>` n'est pas répétable et représente le contenu textuel de la ligne.

En général, toute donnée portant sur la mise en place de la page se dispose de quatre coordonnées qui ensemble tracent un rectangle. Les valeurs des attributs `@HPOS` et `@VPOS` font les coordonnées x,y du point le plus haut à gauche du rectangle, comme se voit dans la Figure 5.2. La valeur de l'attribut `@HEIGHT` compte la différence entre la coordonnée y du point le plus haut et la coordonnée y du point le plus bas. La valeur de l'attribut `@WIDTH` calcule aussi la différence entre le côté gauche du carré et son côté droit. Ces quatre attributs sont attribués aux éléments `<PrintSpace>`, `<TextBlock>`, `<TextLine>`, `<String>`, `<Sp>`, et `<Glyph>`.

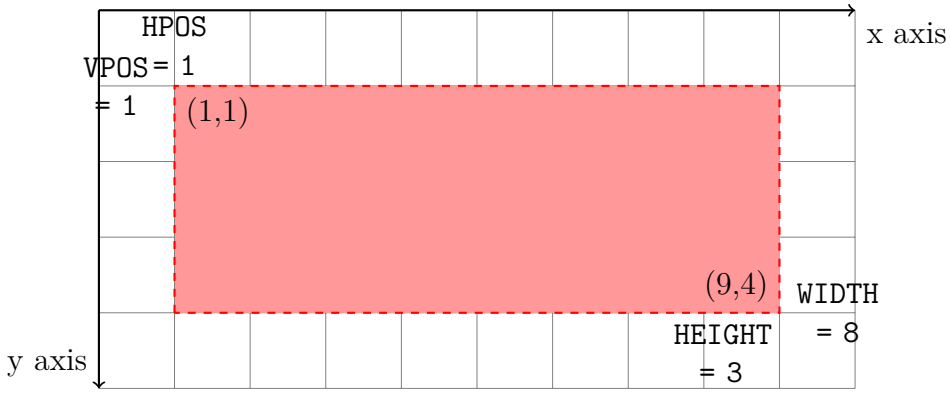
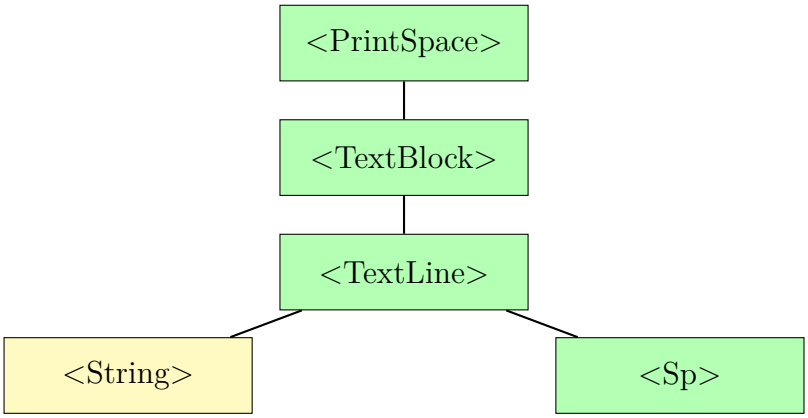


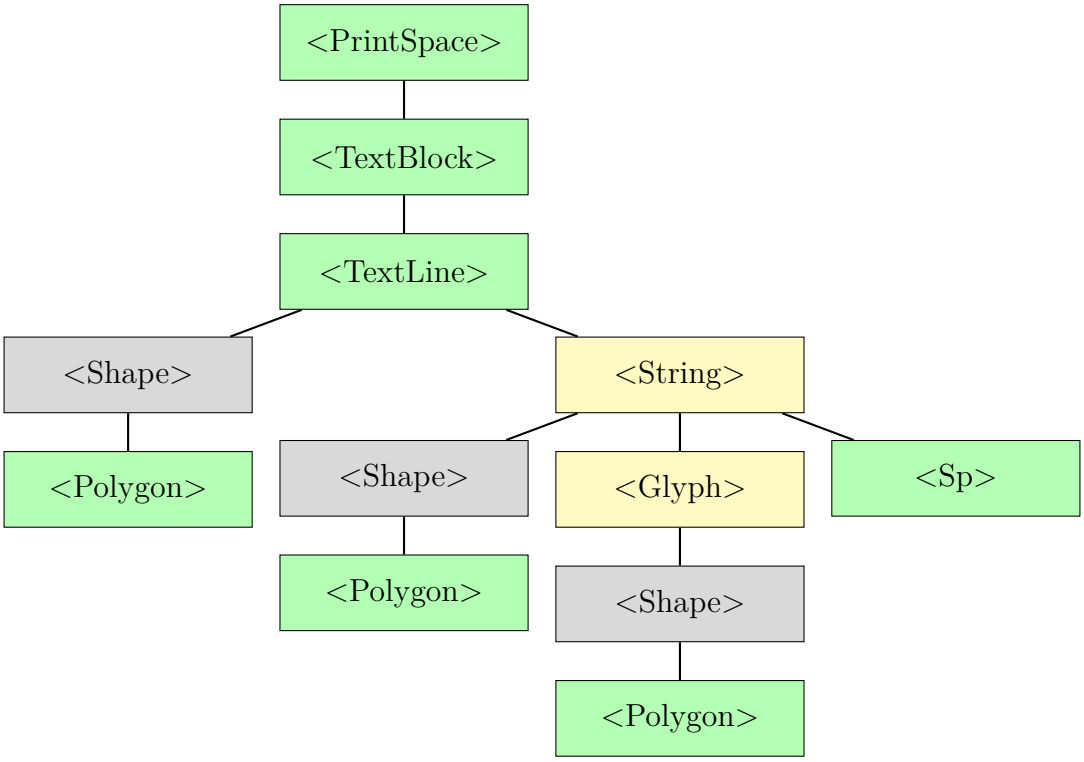
FIGURE 5.2 – Les coordonnées d'un masque en rectangle

L'arborescence actuelle du format ALTO diffère de l'original car, aujourd'hui, elle peut préciser les coordonnées d'un polygone en plus d'un rectangle. Cela est un déve-

loppement dans la technologie des logiciels OCR et HTR. La visualisation dans la sous-figure 5.3b montre le nouvel élément `<Polygon>` qui descend directement d'un élément (`<Shape>`) qui lui-même ne porte pas d'attribut ni d'intérêt dans l'arborescence que de baliser les informations du polygone. Cet élément est indiqué en gris dans la sous-figure 5.3b. La Figure 5.3 indique tout élément qui contient du texte en jaune dans les deux arborescences. Le contenu textuel est toujours balisé dans l'élément `<String>`, qui porte sur le segment ou sur le mot d'une ligne de texte (`<TextLine>`). Mais en allant jusqu'au détail du glyphe dans l'arborescence actuelle, l'élément `<Glyph>` représente tout caractère composant un mot (`<String>`).



(a) Ancienne structure



(b) Nouvelle structure

FIGURE 5.3 – Modélisation des formats ALTO

Certains attributs actuels dans l'arborescence, tel que l'attribut `@BASELINE` de l'élément `<TextLine>` et l'attribut `@POINTS` de tout élément `<Polygon>`, prennent comme valeur une chaîne d'entiers. Montrée dans la Figure 5.4, cette chaîne veut représenter des pairs de coordonnées x,y. Chaque point articule une extrémité soit d'une ligne qui trace le baseline de la ligne de texte (`@BASELINE`) soit une ligne qui encadre la région reconnue par un modèle de segmentation (`@POINTS`). Pour les deux attributs `<@POINTS>` et `<@BASELINE>`, le format ALTO encode chaque entier dans une chaîne dont les composants sont séparés par espace, où la valeur de l'axe des x précède la valeur de l'axe des y. Un polygone (`@POINTS`) peut avoir plusieurs points tout le long de son périmètre. Par contre, le baseline d'une ligne de texte (`@BASELINE`) compte toujours quatre entiers puisqu'il n'a qu'un début et une fin, donc deux pairs x,y.

```
<Layout>
  <Page ID="XXX" PHYSICAL_IMG_NR="000" WIDTH="000" HEIGHT="000">
    <PrintSpace HPOS="000" VPOS="000" HEIGHT="000" WIDTH="000">
      <TextBlock ID="XXX">
        <TextLine ID="XXX" HPOS="000" VPOS="000" HEIGHT="000" WIDTH="000"
    BASELINE="1 2 3 4">
          <Shape>
            <Polygon POINTS="1 2 3 4 5 6 7 8"/>
          </Shape>
          <String ID="XXX" CONTENT="AB" HPOS="000" VPOS="000" WIDTH="000"
    HEIGHT="000" WC="1.0">
            <Shape>
              <Polygon POINTS="1 2 3 4 5 6 7 8 9 10"/>
            </Shape>
            <Glyph ID="XXX" CONTENT="A" HPOS="000" VPOS="000" WIDTH="000"
    HEIGHT="000" GC="1.0">
              <Shape>
                <Polygon POINTS="1 2 3 4 5 6 7 8"/>
              </Shape>
            </Glyph>
            <Glyph ID="XXX" CONTENT="B" HPOS="000" VPOS="000" WIDTH="000"
    HEIGHT="000" GC="1.0">
              <Shape>
                <Polygon POINTS="1 2 3 4 5 6 7 8"/>
              </Shape>
            </Glyph>
          </String>
        </TextLine>
<!-- ... -->
      </TextBlock>
    </PrintSpace>
  </Page>
</Layout>
```

FIGURE 5.4 – La structure ALTO version 4, circa 2022

5.2 TEI : *Text Encoding Initiative*

La raison pour laquelle la TEI a été choisie pour fusionner toute donnée du pipeline *Gallic(orpor)a* est parce qu'elle est un format XML souple, qui peut s'adapter facilement à plusieurs types de document et d'édition numérique. Cela veut dire qu'une exposition détaillée du schème TEI n'est pas possible. La manière pour encoder une ligne de texte n'est pas aussi fixée que cela du schème ALTO, par exemple.

Tandis qu'une ligne de texte dans un fichier ALTO est balisée dans l'élément `<TextLine>`, une ligne de texte dans un document TEI peut être encodée dans plusieurs façons. Elle peut suivre l'élément vide `<lb/>` ou elle peut être à côté d'autres lignes de texte, toutes balisées ensemble dans un élément tel que `<p>` ou `<div>`. De plus, parce que la TEI permet de classer les composants d'une ligne de texte selon la logique du document ou de la langue, certains mots ou phrases peuvent être balisés dans d'autres éléments, tel que l'élément `<date>` pour encoder une année. Dans le même ordre des idées, une ligne de texte peut aussi être balisée dans les éléments qui expliquent sa fonction dans le document. Par exemple, une ligne de texte peut être un item dans une liste (`<item>`) ou la salutation à la fin d'une lettre (`<salute>`).

Prenant l'exemple d'une lettre, la Figure 5.5 montre l'encodage de sa salutation dans les deux schèmes, ALTO et TEI. On voit que l'ALTO excelle à préciser l'emplacement des mots (et des caractères) sur la page d'un document. Mais après la reconnaissance de la lettre, dont la certitude du modèle se représente par l'attribut `@WC` (*word certainty*), le schème ALTO ne donne pas d'autre information. L'encodage dans le format TEI, par contre, enrichit la ligne de texte avec beaucoup d'information. Grâce à l'élément TEI `<salute>` on sait que la ligne de texte est la salutation d'une lettre ou quelque autre forme de communication. De plus, l'encodage appuie sur la date en-tête pour attribuer au mot *demain* une date précise qui est encodée dans l'attribut `@when`. En fin, le schème TEI dispose d'un système pour réunir les occurrences du même concept dans un texte, tel qu'une personne. L'encodage dans la Figure 5.5c utilise l'attribut `@ref` pour dire que l'occurence du mot *chérie* fait référence à une personne à laquelle a été donnée, dans les métadonnées du document TEI, l'identifiant `"Kelly"`.[3]

5.2.1 Qu'est-ce qu'est la TEI ?

Comme montrent les exemples de la Figure 5.5, le schème TEI se spécialise à la représentation d'un texte et à son édition numérique. Il facilite l'enrichissement du texte avec les métadonnées, telles que les références aux autres endroits dans le document ainsi que la classification de la nature d'un mot ou d'une phrase. Les normes de la TEI sont souples à exprès, afin de permettre les encodages personnalisés qui se focalisent sur les

3. Dans le TEI, les identifiants n'ont pas de mot-dièse, mais quand ils sont référencés dans le document la référence en porte un.

12 août 2022

Coucou ! J'ai fait une réservation
pour ton anniversaire.

À demain ma chérie,

(a) L'exemple d'une lettre

```
<TextLine ID="line1" HPOS="0" VPOS="0" HEIGHT="40" WIDTH="200" BASELINE="0
    40 200 40">
  <Shape>
    <Polygon POINTS="...."/>
  </Shape>
  <String ID="seg1" CONTENT="À" HPOS="..." VPOS="..." WIDTH="..." HEIGHT="
    ..." WC="1.0">
  <!-- ... -->
  <String ID="seg2" CONTENT="demain" HPOS="..." VPOS="..." WIDTH="..."
    HEIGHT="..." WC="0.888">
  <!-- ... -->
  <String ID="seg3" CONTENT="ma" HPOS="..." VPOS="..." WIDTH="..." HEIGHT="
    ..." WC="0.9">
  <!-- ... -->
  <String ID="seg3" CONTENT="chérie," HPOS="..." VPOS="..." WIDTH="..."
    HEIGHT="..." WC="0.91">
  <!-- ... -->
</TextLine>
```

(b) La dernière ligne de la lettre encodée dans le schème ALTO

```
<salute>À <date when="2022-08-13">demain</date> ma <name ref="#Kelly" type=
    "person">chérie</name>,</salute>
```

(c) La dernière ligne de la lettre encodée dans le schème TEI

FIGURE 5.5 – Le comparaison de l'encodage d'une ligne de texte en ALTO et TEI

aspects différents d'un texte. Le même texte peut donc être encodé en TEI dans plusieurs manières, selon les besoins et les objectifs des personnes qui se chargent de l'encodage.

Les normes de la TEI sont maintenues par une communauté internationale et leur usage est très répandu dans le monde. L'association est soutenue par le financement des institutions patrimoniales qui comptent sur ses *guidelines* et contribuent des cas d'utilisation. Sur son site web, l'association explique qu'elle continue à modifier ses normes selon les besoins des utilisateurs.

> The scope of the TEI is constantly expanding and the Guidelines are in steady ongoing development to keep pace with the emerging needs of the TEI community. [4]

La croissance de la TEI rend le schème très approprié à l'édition et à l'échange puisque beaucoup d'institutions ont développé des outils numériques qui l'utilisent.

La souplesse de la TEI est à la fois un avantage et un défi à surmonter. Puisque le schème permet de plusieurs encodages du même document, il est donc possible de réaliser plusieurs transformations d'un encodage en ALTO vers un encodage en TEI. Mais pour mettre en œuvre une transformation automatique à l'échelle, il faut une seule modélisation qui s'adapte à tout type de document dont la transcription est encodée en ALTO. En outre, l'enrichissement du texte possible dans la TEI est compliqué à réaliser par ordinateur. Tandis qu'un humain pourrait voir la date en-tête sur la lettre dans la Figure 5.5a et puis savoir que la date référencée dans la salutation est le jour suivant, le 13 août, un logiciel ne pourrait pas faire le liaison entre les deux données si facilement. Donc, bien qu'il puisse savoir, grâce au TAL, que le mot *demain* veut parler d'une date, il ne saurait pas de quelle date parle la lettre ; par contre, une lectrice ou un lecteur humain la saurait avec facilité. Voici quelques défis d'une transformation d'ALTO à TEI.

5.2.2 Les éléments de base de la TEI

La TEI peut s'adapter à plusieurs types de documents mais elle exige toujours certains éléments de la racine qui donnent au schème son arborescence générale. Depuis la racine `<TEI>` d'un document TEI, il faut au moins ces deux descendants : le `<teiHeader>` et le `<body>`. Comme le schème ALTO, le schème TEI a besoin des métadonnées à propos du document encodé et de l'encodage lui-même. Le document TEI imbrique les métadonnées dans l'élément `<teiHeader>`. L'élément `<body>` porte sur les données qui constituent la transcription ou la représentation du document ou des documents ; le dernier sera le cas où le document TEI réalise une édition critique qui ressemblent plusieurs exemplaires d'une œuvre, par exemple. Pour résumer, la TEI a besoin d'au moins les métadonnées, encodées dans le `<teiHeader>`, et les données qui représentent le texte, encodées dans le `<body>`.

4. *About – TEI : Text Encoding Initiative*. URL : `https://tei-c.org/about/` (visité le 25/08/2022).

Après ces deux éléments obligatoires, le schème TEI autorise d'autres éléments facultatifs de descendre directement de la racine `<TEI>`. L'un d'eux est l'élément `<sourceDoc>` dont nous parlons dans la section 4.3.1. La TEI définit le `<sourceDoc>` comme un élément qui peut contenir *une transcription ou une représentation d'un seul document source, qui se réserve le pouvoir à faire partie d'un dossier génétique ou d'une collection d'autres sources.* [5] (traduction par l'autrice) Comme se justifie dans la section 4.3.1, le projet *Gallic(orpor)a* a choisi d'encoder toute donnée du fichier ALTO dans l'élément TEI `<sourceDoc>`. Le schème TEI destine l'élément `<sourceDoc>` à la transcription d'un document source. Un fichier ALTO contient une telle transcription, produite par un logiciel OCR ou HTR. Le `<sourceDoc>` convient bien aux données d'un fichier ALTO car les éléments qui descendent du `<sourceDoc>` portent sur la mise en page ainsi que sur la transcription des images de texte.

Un autre élément facultatif qui descend de la racine `<TEI>` est l'élément `<standOff>`. Le projet *Gallic(orpor)a* s'appuyait aussi sur cet élément parce qu'il est destiné à la représentation des annotations au texte. L'avant dernière étape du pipeline du projet est l'analyse linguistique du texte prédit par le logiciel HTR. Le résultat de cette analyse est une version du texte annotée qui pourrait se différer sensiblement da la transcription. Selon les *guidelines* de la TEI :

> The `standOff` element is intended to hold content that does not fit well in the `text` (e.g. because it is not transcribed from the source), nor in the `teiHeader` (e.g. because it is not metadata about the source or transcription). Examples include [...] annotations indicating the morphosyntactic features of a text, and annotations commenting on or associating parts of a text with additional information. [6]

Comme s'est révélé par les *guidelines*, l'élément `<standOff>` convient bien au texte annoté et normalisé. Ainsi, la transcription du texte tel qu'il s'apparaît dans le document source, avec toute saute de ligne et faute d'orthographe, se trouvera dans les éléments `<sourceDoc>` et `<text>`, qui descend du `<body>`. Par contre, la version du texte qui n'existe pas dans le document source mais qui sert bien à l'analyse du document se trouvera dans l'élément facultatif `<standOff>`. Pour résumer, les quatre éléments pertinents qui descendent de la racine `<TEI>` sont visualisés dans la Figure 5.6.

5. *TEI element sourceDoc.*

6. *16 Linking, Segmentation, and Alignment - The TEI Guidelines.* URL : `https://tei-c.org/release/doc/tei-p5-doc/en/html/SA.html#SASOstdf` (visité le 25/08/2022).

```xml
<TEI>
  <teiHeader><!-- métadonnées --></teiHeader>
  <sourceDoc><!-- transcription --></sourceDoc>
  <body>
    <text><!-- texte pré-éditorialisé --></text>
  </body>
  <standOff><!-- texte annoté --></standOff>
</TEI>
```

FIGURE 5.6 – Les éléments de base du schème TEI

Chapitre 6

À la recherche des métadonnées

6.1 Uniquement l'essentiel

6.1.1 Documents de divers types et de plusieurs époques

Expliquer le défi de modéliser un `<teiHeader>` qui est à la fois assez généralisé pour convenir aux divers documents et assez précisé pour servir aux utilisateurs et à la recherche.

6.1.2 Exemples des métadonnées souhaitées

Donner un exemple des métadonnées d'un imprimé (cf. fig. 6.1) :

```
<sourceDesc>
  <biblStruct>
    <monogr>
      <author xml:id="author">
        <persName>
          <surname>Balzac</surname> <!-- auteur -->
          <forename>Honoré</forename>
        </persName>
      </author>
      <title>The Wild Ass's Skin</title> <!-- titre -->
      <editor role="translator">Ellen Marriage</editor>
      <editor role="preface">George Saintsbury</editor>
      <pubPlace key="FR">Paris</pubPlace>
      <imprint>
        <pubPlace>London</pubPlace> <!-- lieu de publication -->
        <publisher>Dent</publisher> <!-- éditeur -->
        <date when="1906">1906</date> <!-- date de publication -->
      </imprint>
    </mongr>
  </biblStruct>
```

FIGURE 6.1 – Exemple des métadonnées d'un imprimé encodées en TEI (emprunté de `teibyexample.org` – à changer)

Et donner un exemple d'un incunable (cf. fig. 6.2) :

```
<sourceDesc>
  <msDesc>
    <msContents>
      <biblStruct>
        <monogr>
          <author xml:id="author">
            <persName>
              <surname>Tory</surname> <!-- auteur -->
              <forename>Geoffroy</forename>
            <persName>
          </author>
          <title>Champ fleury</title> <!-- titre -->
          <imprint>
            <pubPlace>Paris</pubPlace> <!-- lieu de publication / lieu d'
    apparition -->
            <publisher>
              <persName>
                <surname>Gourmont</surname> <!-- éditeur -->
                <forename>Gilles de</forename>
              </persName>
            </publisher>
          </imprint>
        </mongr>
      </biblStruct>
```

FIGURE 6.2 – Exemple des métadonnées d'un incunable encodées en TEI (emprunté du cours TEI de J-B Camps 2015 – à changer)

Expliquer comment l'objectif du projet *Gallic(orpor)a* de traiter des documents d'une diachronie longue pose un défi à la récupération et encodage généralisée des métadonnées.

6.2 Où se trouvent les métadonnées des sources de Gallica

Présenter les deux sources de métadonnées ciblées par l'application `alto2tei`.

6.2.1 L'IIIF Image API

L'API du manifest IIIF contient des données rudimentaires sur le document. Elles sont envoyées dans un format JSON. Donner un exemple (cf. fig. 6.3) :

6.2.2 L'API SRU de la BnF

L'API du catalogue général de la BnF contient des données bien précises sur le document. Elles sont envoyées dans un format XML-Unimarc. Donner un exemple (cf. fig. 6.4).

```
{"Metadata":
    {
    "Label": "Title",
    "Value": "The Wild Ass's Skin", # titre

    "Label": "Creator",
    "Value": "Honoré Balzac" # auteur
    }
}
```

FIGURE 6.3 – Exemple des métadonnées envoyées par l'API IIIF

```
  <mxc:datafield tag="200" ind1="1" ind2=" ">
    <mxc:subfield code="a">The Wild Ass's Skin</mxc:subfield> <!-- titre --
    >
    <mxc:subfield code="b">Texte imprimé</mxc:subfield>
  </mxc:datafield>
  <mxc:datafield tag="210" ind1=" " ind2=" ">
    <mxc:subfield code="a">Londres</mxc:subfield> <!-- lieu de publication
    -->
    <mxc:subfield code="c">Dent</mxc:subfield> <!-- éditeur -->
    <mxc:subfield code="d">1906</mxc:subfield> <!-- date de publication -->
  </mxc:datafield>
[...]
  <mxc:datafield tag="700" ind1=" " ind2="|">
    <mxc:subfield code="a">Balzac</mxc:subfield> <!-- auteur -->
    <mxc:subfield code="b">Honoré</mxc:subfield>
  </mxc:datafield>
```

FIGURE 6.4 – Exemple des métadonnées envoyées par l'API SRU de la BnF

6.3 Une solution

Présenter les métadonnées du document qu'on a déterminé d'être essentielle / assez généralisées parmi les divers types de document.

Troisième partie

Mise en opérationnelle du projet

Chapitre 7

La génération du `<teiHeader>`

7.1 La récupération des données

Parler de la récupération des données depuis les deux APIs discutés (cf. fig. **??**) ainsi que le fichier YAML de configuration.

7.1.1 Du manifest IIIF à un dictionnaire Python

Montrer le mapping des données du manifest au dictionnaire Python.

7.1.2 De l'Unimarc à un dictionnaire Python

Montrer le mapping des données du catalogue au dictionnaire Python.

7.2 L'analyse des données

Parler de la stratégie d'atténuation des risques en sélectionnant les données fiables. Les métadonnées du catalogue général de la BnF sont utilisées uniquement si le même exemplaire physique du document numérisé sur Gallica a bien été trouvé. Sinon, on risque de mettre les données d'un autre exemplaire de l'oeuvre que celui qui a été transcrit et donc introduire des fausses données, tel que le cote ou même l'éditeur et la date de publication de l'exemplaire.

7.3 Le modèle du `<teiHeader>`

Montrer le mapping des données au `<teiHeader>`.

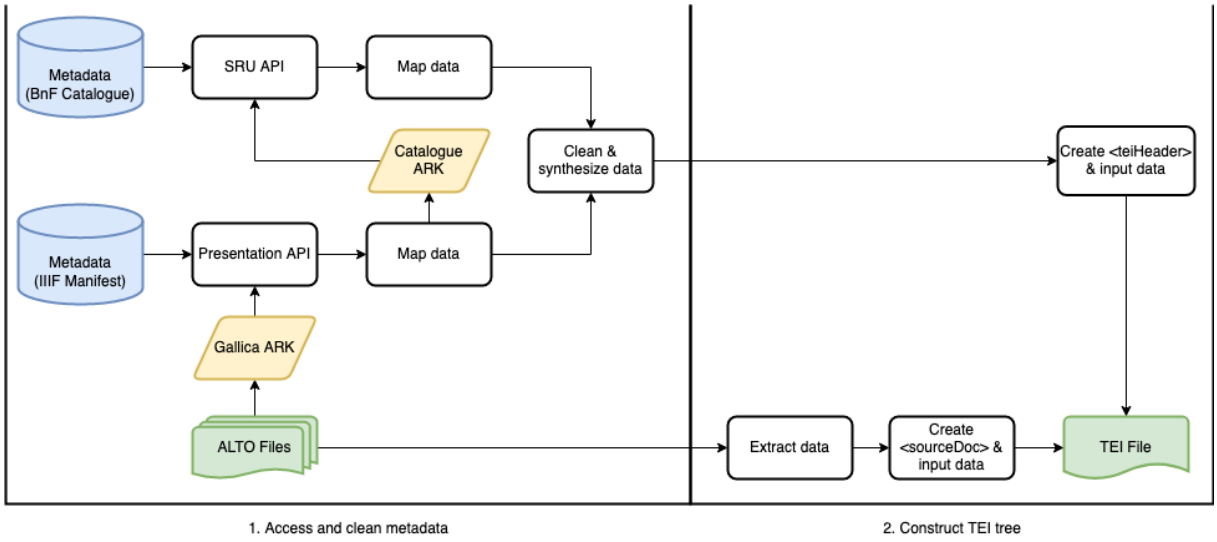


FIGURE 7.1 – Workflow

Chapitre 8

La modélisation de la <sourceDoc>

8.1 Le modèle du <sourceDoc>

Montrer le mapping des données prédites par les modèles HTR et segmentation vers les éléments TEI de la <sourceDoc>.

8.1.1 Niveau de la ligne de texte

Des exemples / tables ...

8.1.2 Niveau d'un mot ou d'une espace

Des exemples / tables ...

8.1.3 Niveau du glyphe

Des exemples / tables ...

8.2 Documenter ce modèle dans l'ODD

Présenter le travail d'avoir écrit l'ODD *SegmOnto-Gallicorpora*.

Chapitre 9

Les données textuelles produites

9.1 La génération du `<body>` grâce au lexique *SegmOnto*

Expliquer comment j'ai généré le `<body>` en prenant les lignes de texte de la `<sourceDoc>` selon leurs étiquettes.

9.2 L'analyse linguistique

Parler d'un travail futur qui pourra prendre le fichier XML-TEI que j'ai créé et extraire les lignes de texte du `<body>` et les passer aux modèles TAL pour faire des analyses linguistiques. Montrer un exemple de ces données et comment cela marcherait avec un travail préliminaire que j'aurai fait d'ici la fin de stage et/ou le travail d'un ancien stagiaire que j'ai appliquée aux données Gallic(orpor)a.

Conclusion

Annexe A

Données

A.1 Portée des données d'entraînement

Type	Genre	Forme	Écriture	Siècle	Langue	Titre
manuscrit	poésie	vers	gothique	13	fro	Français 20050 - chansonnier de Saint-Germain-des-Près
manuscrit	récit	prose	gothique	13	fro	Français 23117, légendier
manuscrit	récit	prose	gothique	13	fro	Français 6447, légendier
manuscrit	récit	prose	gothique	13	fro	NAF 23686, légendier
manuscrit	récit	prose	gothique	13	fro	Français 13496, légendier
manuscrit	récit	vers	gothique	13	fro	Français 860 - Roland, Gaydon, Ami et Amile, Jourdain de Blaye, Auberi le Bourguignon
manuscrit	récit	vers	gothique	13	fro	Français 12615 - chansonnier de Noailles
manuscrit	récit	vers	gothique	13	fro	Français 1443 - Garin le Lohérain (C) et Girbert de Metz
						Français 12603 - Fierabras, mais aussi Chevalier as deux espees

Table des figures

Liste des tableaux

Table des matières