

Contenu de la documentation

Présentation	3
Contexte	3
Objectifs	3
1 Reconnaissance automatique des écritures manuscrites	5
1.1 Problématique	5
1.1.1 Des sources écrites par plusieurs mains	5
1.1.2 Objectif : éditer	6
1.2 Choisir des collections d'évaluation	6
1.3 Préparer le traitement d'un dossier	7
1.4 Choisir une application : Transkribus ou eScriptorium ?	7
1.5 Annoter les régions et les lignes d'écriture	8
1.5.1 Régions	9
1.5.2 Typer les lignes d'écriture	10
1.5.3 Phénomènes graphiques particuliers	10
1.6 Entraîner des modèles de segmentation des pages	11
1.7 Tester et entraîner des modèles de reconnaissance d'écriture	11
1.8 Injecter les transcriptions manuelles dans les prédictions	11
1.9 Automatiser la correction des prédictions	12
1.9.1 Champ d'application et limites	12
1.9.2 Analyser les mots	13
1.9.3 Gérer les résolutions ambiguës	14
Annexes	15
A Normes de transcription	19
A.1 Accentuation	19
A.2 Majuscules et minuscules	19
A.3 Séparation des mots	19
A.4 Orthographe	19
A.5 Abréviations	20

A.6	Ponctuation	20
A.7	Passages biffés, palimpsestes	20
A.8	Passages illisibles	20
	Bibliographie	21

Présentation

Contexte

Constance de Salm (1767-1845), femme de lettres française, a entretenu une vaste correspondance à partir de son mariage avec de nombreux intellectuels en Allemagne, en France, en Russie.

Le projet de publier numériquement sa correspondance est né de l'intérêt pour les relations entre noblesses française et allemande au sein du Deutsches Historisches Institut Paris (DHIP). Il en a résulté la production d'un site *Wordpress* adossé au système de base de données Die Virtuelle Forschungsumgebung für die Geistes- und Sozialwissenschaften (FuD). Les notices de plus de 11000 lettres, publiées sur le site constance-de-salm.de, associent la reproduction numérique des documents manuscrits (lettres, copies, brouillons, recueils) avec leurs métadonnées descriptives, ainsi qu'une transcription de la première ligne de chaque lettre.

Objectifs

L'objectif du stage consiste à mettre en place un flux de production automatisé pour l'édition des lettres au format XML-TEI. On s'appuiera pour cela sur les instruments et la documentation produits dans le cadre du projet Digital Edition of historical manuscripts (DAHN), fondé sur l'édition de la correspondance de Paul d'Estournelles de Constant (1852-1924)¹.

Il s'agit en particulier d'identifier les points de difficultés que posent le traitement de ce vaste corpus tant du point de vue de la transcription automatisée des documents que du point de vue de leur encodage au format TEI.

Il serait notamment souhaitable, au terme du stage de disposer d'un flux de production pour l'édition d'un volume de recueil de lettres.

1. Floriane Chiffoleau, *DAHN Project*, GitHub, URL : <https://github.com/FloChiff/DAHNProject> (visité le 05/04/2022).

Chapitre 1

Reconnaissance automatique des écritures manuscrites

1.1 Problématique

1.1.1 Des sources écrites par plusieurs mains

Quatre à cinq mains différentes ont été repérées jusqu'à présent dans la correspondance de Constance de Salm (CdS) (mais aucune enquête paléographique complète n'a été menée). Cette variété des écritures est un problème majeur pour l'automatisation des transcriptions.

Les choix effectués dans le cadre du projet Lecture Automatique de Répertoires (Lectaurep) ont permis de guider notre démarche. L'alternative méthodologique a été décrite ainsi par A. Chagué :

Quand on se lance dans une campagne de transcription reposant sur la reconnaissance d'écritures manuscrites, on passe généralement par une série de questions qui sont les mêmes d'un projet à l'autre. Parmi ces questions, il y a celle des modèles de transcription et de leur rapport à la variation des écritures. Doit-on entraîner un modèle pour chaque type d'écriture présent dans un corpus de documents ? Au contraire, peut-on se contenter d'entraîner un seul modèle tout terrain (qu'on appellera mixte ou générique) ?¹

Les résultats probants obtenus par le projet Lectaurep en suivant l'option d'entraînement d'un modèle mixte² nous ont convaincu d'emprunter cette voix.

Deux séries de tests méritaient dès lors d'être effectuées :

1. Alix Chagué, *Création de modèles de transcription pour le projet LECTAUREP #1*, Lectaurep : l'intelligence artificielle appliquée aux archives notariales, URL : <https://lectaurep.hypotheses.org/475> (visité le 05/04/2022).

2. Id., *Création de modèles de transcription pour le projet LECTAUREP #2*, Lectaurep : l'intelligence artificielle appliquée aux archives notariales, URL : <https://lectaurep.hypotheses.org/488> (visité le 05/04/2022).

1. Reprendre les tests sur le modèle entraîné de zéro par H. Souvay lors d'un précédent stage consacré à la correspondance de CdS³ ;
2. Reprendre un modèle générique entraîné pour le projet Lectaurep .

1.1.2 Objectif : éditer

Il faut prendre en considération dans les choix méthodologiques du processus d'*Handwritten Text Recognition* (HTR) les objectifs à atteindre. Il s'agit d'éditer le texte en restant proche de l'usage scribal, sans donc le corriger :

1. La transcription ne résout pas les abréviations ;

1.2 Choisir des collections d'évaluation

Afin de donner les meilleures chances aux tests à effectuer avec le modèle entraîné par H. Souvay, nous sommes repartis des mêmes vérités de terrain, issues de la seconde copie de la correspondance générale. Ces recueils de lettres constituent la part du corpus la plus normée sur le plan de l'écriture et de la mise en page, leur qualité de conservation assurant en outre de bonnes conditions à la reconnaissance d'écriture. Nous avons particulièrement exploité les trois premiers volumes de cet ensemble qui en compte six⁴.

La variété des écritures se partage de manière contrastée entre des mains dominantes et des mains rares. Généralement, deux mains dominantes se partagent un recueil ; leur distribution peut être discontinue. Quant aux mains rares, elles n'occupent que quelques feuillets par recueil ; nous ne les avons pas retenu pour les tests.

Nous avons également analysé les écritures du recueil de la correspondance adressée par J.P.E. Martini à CdS afin d'élargir la variété de notre corpus de tests. Nous y avons distingué deux mains⁵.

On a privilégié pour les corpus de test et d'entraîner des modèles des reproductions favorables à une bonne reconnaissance de l'écriture, évitant en particulier les problèmes de transparence qui font ressortir au recto l'encre du verso.

Concernant l'écriture personnelle de CdS, le site ne publie aucune lettre originale de sa main, mais 52 brouillons (*Entwurf*). Entrainer un modèle de reconnaissance sur cette

3. Hippolyte Souvay, *La Correspondance de Constance de Salm (1767-1845) : Rapport de Stage*, rapport de stage de seconde année de master Humanités numériques et computationnelles, École nationale des chartes-Institut historique allemand à Paris, 2021.

4. Constance de Salm, *Correspondance générale, seconde copie, 1^{er} volume, 1785-1814*, URL : <https://constance-de-salm.de/archiv/#/document/11215> (visité le 11/04/2022) ; Id., *Correspondance générale, seconde copie, 2^e volume, 1815-1821*, URL : <https://constance-de-salm.de/archiv/#/document/11216> (visité le 11/04/2022) ; Id., *Correspondance générale, seconde copie, 3^e volume, 1822-1828*, URL : <https://constance-de-salm.de/archiv/#/document/11217> (visité le 12/04/2022).

5. Une présentation des mains peut être parcourue sur le dépôt du projet

écriture suppose un travail délicat de transcription pour une écriture particulièrement cursive (compter environ deux semaines pour disposer d'une bonne vingtaine de pages).

1.3 Préparer le traitement d'un dossier

L'archive photographique de la correspondance de Constance de Salm comporte des documents non inventoriés. Afin de n'engager dans notre chaîne de traitement que des documents effectivement inventoriés, nous avons consacré un *notebook* à la préparation du traitement d'un dossier⁶.

Après l'étape préliminaire de l'import local et de la conversion des images au format Jpeg (afin de ne pas travailler avec le format Tiff, trop lourd), il est nécessaire d'établir la liste des images associées à une notice de l'inventaire. Nous avons pour cela écrit un script python⁷ qui analyse les noms des fichiers convertis et importés localement, croise ces noms avec les données de l'inventaire et écrit en sortie un fichier Json qui liste, pour chaque notice l'inventaire contenant l'une des images du dossier, l'URL de cette notice sur le site <https://constance-de-salm.de> et la liste complète des images attachées à cette notice. Le fichier donne par ailleurs la liste des images qui ne sont liées à aucune notice de l'inventaire.

Une fois le dossier analysé et le fichier produit, les commandes que nous avons écrites dans le *notebook* permettent de n'importer dans le dossier de travail que les images correspondant à une notice de l'inventaire.

1.4 Choisir une application : Transkribus ou eScriotorium ?

Au moment de notre stage, les deux principales applications permettant de procéder à la transcription automatique des écritures manuscrites sont eScriotorium et Transkribus.

Differentes considérations peuvent conduire à opter pour l'une ou l'autre de ces applications⁸. Le facteur nous apparaissant comme le plus déterminant a trait aux compétences d'ingénierie des personnes chargées de mener la campagne de transcription.

6. Sébastien Biay, *Préparer Le Traitement d'un Dossier*, 20 mai 2022, URL : https://github.com/sbiay/CdS-edition/blob/6c4e4d4cff3101a154b9fa7e4a248e7ac87ff7ee/htr/Preparer_le_traitement_dune_source.ipynb (visité le 23/05/2022).

7. Id., *donneesImages.Py*, 6 avr. 2022, URL : <https://github.com/sbiay/CdS-edition/blob/main/htr/py/donneesImages.py> (visité le 19/04/2022).

8. Nous avons assisté le 9 mai 2022 à l'atelier organisé au sein du Data-Lab de la Bibliothèque nationale de France (BnF) et dont le programme est détaillé dans le billet d'Olivier Jacquot, *Transkribus / eScriotorium : transcrire, annoter et éditer numériquement des documents d'archives, ateliers à la BnF*, Carnet de la recherche à la Bibliothèque nationale de France, URL : <https://bnf.hypotheses.org/12575> (visité le 10/05/2022).

L'écosystème applicatif Transkribus est celui qui propose le plus grand choix de services, tant pour les utilisateurs ayant des compétences d'ingénierie élevées (logiciel Expert Client) que pour les néophytes (Transkribus Lite). Conjuguées à la facilité de prise en main de Transkribus Lite, les fonctionnalités de gestion des versions de transcription offertes par Transkribus Expert Client rendent cet écosystème le mieux à même d'héberger des campagnes de transcription de grande ampleur, faisant appel à de multiples transcripteurs, voire à de la production participative (ou *crowdsourcing*).

L'application eScriptorium, à un stade de développement moins avancé, avec une interface dotée de moins de fonctionnalités que Transkribus (gestion des versions de transcription, annotation du texte), mobilise davantage de compétences d'ingénierie. En revanche, la gratuité totale de son utilisation et qui plus est la culture *open-source* portée par la communauté qui développe et utilise eScriptorium (ouverture des données de modèles, de vérités de terrain, développement d'outils auxiliaires à la transcription, à la gestion de fichiers, propositions de standards d'annotation) rendent cette application tout à fait adéquate aux projets impliquant un petit nombre de transcripteurs ayant une bonne culture d'ingénierie au préalable, notamment au sein d'institutions désireuses de promouvoir la science ouverte.

Nous avons opté pour eScriptorium dans le cadre de ce stage en raison du dynamisme de la communauté eScriptorium au sein de l'École nationale des chartes (ENC) (à travers le projet Consortium Reconnaissance d'Écriture Manuscrite des Matériaux Anciens (Cremma)).

Nous avons testé son utilisation à partir d'une installation locale, faisant appel aux seules ressources d'un ordinateur portable, à savoir sans serveur ni carte graphique externe. Cette méthode nous a permis de procéder à des entraînements de modèle à partir de petits volumes de vérités de terrain. Si des entraînements plus massifs s'avéraient nécessaires, il serait alors impératif de se tourner vers une infrastructure dotée de plus grandes capacités de calcul, ce que, par exemple, un partenariat entre le DHIP et le projet Cremma rendrait possible.

1.5 Annoter les régions et les lignes d'écriture

L'annotation des régions et des lignes d'écritures répond à deux fonctions distinctes :

- Permettre l'entraînement d'un modèle de segmentation ;
- Transformer leur contenu afin de l'affecter à des éléments déterminés de l'arborescence XML-TEI qu'il faudra construire⁹.

Cette réflexion sur les besoins de la transformation vers le format TEI a été nourrie par les *Guidelines* de l'édition de correspondance du projet DAHN¹⁰. Par ailleurs, F. Chif-

9. C'est aussi un enjeu central du projet **Galli(corpor)a**

10. F. Chiffolleau, *Correspondence : Guidelines*, DAHN Project, 10 janv. 2022, URL : [https :](https://)

foleau a formulé une ontologie pour les régions et lignes des écrits de correspondance en langue française pour le XXe siècle¹¹ dans le cadre du projet SegmOnto : A Controlled Vocabulary to Describe the Layout of Pages (SegmOnto)¹². Afin de rendre notre propre typologique générique et de pouvoir exploiter l'outil de validation d'annotation HTRUC¹³, nous avons repris les types SegmOnto en exploitant les types `CustomZone` et `CustomLine` lorsqu'il était nécessaire de les personnaliser.

1.5.1 Régions

L'entraînement d'un modèle de segmentation à reconnaître et annoter automatiquement des types de régions d'écritures est un travail complexe. La mise en page des lettres répond à des principes clairs pour l'oeil humain ; il présente en revanche d'importantes variations métriques dans l'espace de la page. Le meilleur exemple de ces variations se trouve dans les recueils : les lettres ont été transcrrites les unes à la suite des autres ; un début de lettre peut dès lors se trouver à n'importe quelle hauteur de la page.

Ajouter un commentaire sur l'espacement des lignes de l'en-tête

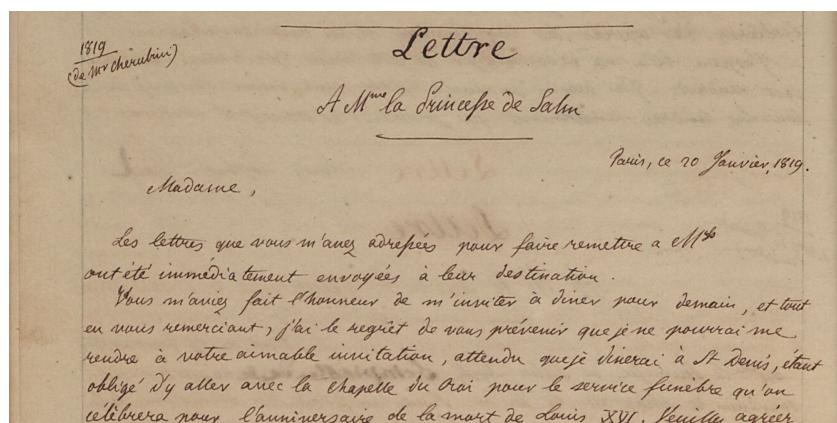


FIGURE 1.1 – Début d'une lettre présentant une disposition aérée des éléments (SALM (Constance de), *Correspondance générale, seconde copie, 2^e volume, 1815-1821*, URL : <https://constance-de-salm.de/archiv/#/document/11216> (visité le 11/04/2022)).

La reconnaissance de la fin d'une lettre (dont la signature alignée à droite de la page est le premier mais non le seul élément) est encore plus délicate, car elle ne se manifeste jamais par un élément visuellement massif comme un titre.

[//github.com/FloChiff/DAHNProject/blob/8df8dfc6053a7dd57a6c5510d1e56bb336ce1d04/Correspondence/Guidelines/Documentation-Correspondance.pdf](https://github.com/FloChiff/DAHNProject/blob/8df8dfc6053a7dd57a6c5510d1e56bb336ce1d04/Correspondence/Guidelines/Documentation-Correspondance.pdf) (visité le 07/04/2022).

11. Id., [Correspondance En Langue Française, XXe s.] SegmOnto, 10 déc. 2021, URL : https://github.com/SegmOnto/examples/tree/main/sources/lettre_fr_XXe (visité le 07/04/2022).

12. Simon Gabay, Jean-Baptiste Camps, Ariane Pinche et Claire Jahan, « SegmOnto : Common Vocabulary and Practices for Analysing the Layout of Manuscripts (and More) », dans 1st International Workshop on Computational Paleography, Lausanne, 2021, URL : <https://hal.archives-ouvertes.fr/hal-03336528> (visité le 20/04/2022).

13. Thibault Clérice, *HTRUC, HTR-United Catalog Tooling (Pronounced EuchTruc)*, version 0.0.1, nov. 2021, URL : <https://github.com/HTR-United/HTRUC> (visité le 20/05/2022).

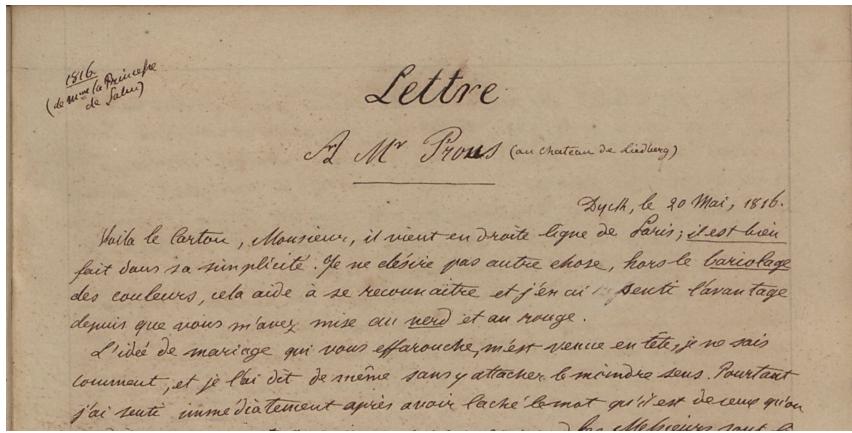


FIGURE 1.2 – Début d'une lettre présentant une disposition resserrée des éléments (SALM (Constance de), *Correspondance générale, seconde copie, 2^e volume, 1815-1821*, URL : <https://constance-de-salm.de/archiv/#/document/11216> (visité le 11/04/2022)).

En raison de cette complexité, nous avons opté pour une typologie de régions resserrée. Tandis que l'ontologie de F. Chiffolleau proposait un type de région pour chaque élément significatif de l'écrit de correspondance¹⁴, nous n'avons retenu que trois types de régions principaux pour annoter la structure des lettres :

- **MainZone** : pour le corps du texte ;
- **CustomZone :opener** : pour l'en-tête des lettres ;
- **CustomZone :closer** : pour la clôture des lettres.

la définition des régions est formelle, visuelle

La figure 1.3 ci-dessous propose une mise en oeuvre de ce typage des régions.

1.5.2 Typer les lignes d'écriture

Les types de lignes dont on propose l'utilisation sont :

1.5.3 Phénomènes graphiques particuliers

CdS a corrigé certains mots de sa main :

- En rayant une lettre, un mot ou plusieurs mots, ou bien en réécrivant par dessus le texte. Dans de nombreux cas cela consiste en une simple lettre barrée ; le typage de la ligne demanderait alors beaucoup d'effort pour un résultat minime ;
- En réécrivant dans l'interligne : il est alors pertinent d'utiliser le type de ligne e-Scriptorium **Correction**.

Un ensemble de solutions d'encodage des corrections a été proposé dans le cadre du projet DAHN¹⁵. J'envisage plutôt **ne pas encoder ces éléments dans la phase**

14. F. Chiffolleau, [Correspondance En Langue Française, XXe s.]...

15. Id., *Few Tips for Reading the Text Files*, DAHN Project, URL : <https://github.com/FloChiff/DAHNProject/tree/master/Project%20development/Texts> (visité le 11/04/2022).

d’HTR, et de ne les aborder que la phase d’édition. Il sera de toute façon nécessaire, lors de la reprise manuelle de l’édition TEI, de suivre la reproduction du manuscrit à éditer. En outre, introduire des caractères tels que £, €, etc. dans la transcription générerait du bruit dans l’entraînement du modèle HTR et imposerait une phase de nettoyage pour les réutilisations éventuelles des vérités de terrain.

En somme, il s’agirait de **transcrire tout ce qui est lisible** (y compris les lettres biffées, lorsque c’est possible), en privilégiant le dernier état du texte dans le cas où la correction a été superposée à la première couche d’écriture.

1.6 Entrainer des modèles de segmentation des pages

Cette section est à écrire.

1.7 Tester et entraîner des modèles de reconnaissance d’écriture

Cette section est à écrire.

1.8 Injecter les transcriptions manuelles dans les prédictions

Le test et l’entraînement des modèles de reconnaissance d’écriture impose la production de transcriptions manuelles du texte. Il nous est apparu essentiel que cette tâche un peu fastidieuse soit pleinement valorisée dans le processus d’édition et que ces transcriptions théoriquement parfaites servent non seulement à l’entraînement des modèles mais soient aussi exploitées pour la production de l’édition finale du texte.

La méthode la plus simple pour joindre les fichiers XML-Alto contenant les transcriptions manuelles aux fichiers contenant la prédiction automatique du texte des autres pages d’un même dossier est de regrouper ces fichiers ensemble. Or, nous avons voulu tenir compte de la possibilité que les transcriptions manuelles ne recouvrent pas toutes les lignes d’écriture d’une page. Certaines mains n’étant attestées que de manière sporadique, en compagnie d’autres écritures, la méthodologie d’entraînement impose de ne transcrire que l’écriture propre au test ou à l’entraînement, laissant les autres écritures de côté. Il résulte de cette nécessité que les fichiers XML-Alto contenant les transcriptions manuelles peuvent être lacunaires, et qu’ils ne peuvent donc pas se substituer aux fichiers contenant la prédiction complète des lignes d’écriture d’une page.

Il était donc nécessaire de concevoir une méthode de remplacement, dans les fichiers

contenant la prédiction automatique du texte, des seules lignes pour lesquelles nous avions produit des transcriptions manuelles. Cibler de manière précise des lignes d'écriture dans un fichier XML-Alto est rendu possible par l'identifiant unique de chaque élément contenant une ligne de texte (`TextLine`). Nous avons donc écrit un script python¹⁶ capable d'analyser toutes les lignes d'écriture des fichiers de nos vérités de terrain et de comparer leur identifiant avec ceux des lignes des fichiers des prédictions automatiques portant les mêmes noms. En cas de correspondance entre les identifiants, la transcription manuelle vient remplacer la prédiction du texte.

1.9 Automatiser la correction des prédictions

Une fois que l'on dispose d'un modèle de reconnaissance d'écriture suffisamment bien entraîné pour donner des prédictions satisfaisantes pour toutes les mains principales d'une source, on peut réaliser des prédictions sur l'ensemble de la source.

Les corrections à appliquer à ces prédictions HTR restent nombreuses, ce qui appelle à trouver des solutions d'automatisation. Cette tâche requiert néanmoins de la prudence. Le risque de son automatisation est notamment de remplacer involontairement des prédictions justes ou de remplacer des prédictions fausses par d'autres prédictions fausses. Le contrôle des propositions automatiques de correction est donc nécessaire, bien qu'un trop grand nombre de données à contrôler puisse nuire gravement à la rentabilité du processus.

L'automatisation de la correction des prédictions a pour objectif d'accélérer le passage de la prédiction au format XML-TEI. Le résultat de cette correction est imparfait ; par conséquent cette correction n'intervient pas dans le processus d'entraînement d'un modèle HTR qui dépend de transcriptions les plus justes possibles. Une fois les modèles HTR correctement entraînés, la correction automatique permet de résoudre rapidement un certains nombres d'erreurs en amont la transformation au format TEI, où une correction manuelle approfondie du texte est nécessaire pour son établissement définitif.

Nous avons suivi la démarche explicitée dans la documentation du projet DAHN¹⁷ et proposé quelques développements aux scripts issus de ce projet.

1.9.1 Champ d'application et limites

La correction automatisée se concentre sur l'orthographe des mots. Elle n'aborde pas la ponctuation et s'appuie sur des dictionnaires où l'accentuation des mots est normalisée selon l'usage moderne (alors que l'édition finale doit respecter l'usage scribal), et ce afin

16. S. Biay, *injectTranscript.Py*, 20 mai 2022, URL : <https://github.com/sbiay/CdS-edition/blob/main/htr/py/injectTranscript.py> (visité le 03/06/2022).

17. F. Chiffolleau, *How to Do a Post-OCR Correction for TEXT Files*, DAHN Project, 8 avr. 2022, URL : <https://github.com/FloChiff/DAHNProject/blob/8df8dfc6053a7dd57a6c5510d1e56bb336ce1d04/Project%20development/Documentation/Post-OCR%20correction%20for%20TEXT%20files.md> (visité le 11/04/2022).

de ne pas multiplier les corrections pour un même lemme. Enfin, elle ne traite pas le problème des mots mal prédits dont l'orthographe est attestée ailleurs dans les vérités de terrain ; par exemple, dans la prédition *Dans vu siècle où tous les talens...*, la prédition erronée *vu* pour *un* ne sera pas corrigée, car le mot *vu* est attesté ailleurs. Nous avions tenter l'automatisation de ce type de correction, mais considérant qu'il impose de passer en revue tous les mots dont l'orthographe est déjà attestée ailleurs dans nos vérités de terrain, cette opération faisait perdre plus de temps qu'elle n'en faisait gagner.

1.9.2 Analyser les mots

Nous avons appliqué le script d'analyse de mots `spellcheck-texts.py`¹⁸ à nos prédictions HTR¹⁹.

Afin de faciliter la correction des dictionnaires générés par le script pour chaque page (chaque proposition de correction doit en effet être contrôlée), on a développé ce script initialement écrit par F. Chiffoleau pour afficher le contexte du mot et en conserver la mémoire, ce qui limite le besoin d'allers-retours entre le dictionnaire à corriger et l'image ou la prédition d'origine.

Dans le but d'optimiser la performance de l'analyse des mots on a développé une fonction appelée `collecteMots`, qui fouille les vérités de terrain déjà constituées et permet de valider automatiquement les mots déjà rencontrés dans le traitement de la correspondance de CdS, évitant de proposer systématiquement les corrections à partir de l'orthographe française modernisée pour les mots à l'orthographe ancienne déjà rencontrée et validée ; cette méthode permet en outre d'enregistrer les formes graphiques des noms propres, qui dès lors ne sont plus non plus considérées comme erronées.

Les corrections s'avérant nombreuses, le script `textCorrection.py`²⁰ écrit par F. Chiffoleau a dû être perfectionné afin de procéder à une tokénisation des mots, pour corriger avec exactitude (de manière spécifique pour chaque ligne) les formes erronées présentes dans le texte. Nous avons pour cela utilisé le module Spacy²¹.

En outre, il s'est avéré nécessaire de modifier la méthode d'application des corrections aux fichiers XML-Alto des prédictions en optant pour l'écriture d'un authentique

18. S. Biay et F. Chiffoleau, *spellcheckTexts.Py*, 6 avr. 2022, URL : <https://github.com/sbiay/CdS-edition/blob/main/htr/py/spellcheckTexts.py> (visité le 19/04/2022).

19. Ce script est fondé sur l'utilisation du module publié par Tyler Barrus, *Pyspellchecker : Pure Python Spell Checker Based on Work by Peter Norvig*, version 0.6.3, URL : <https://github.com/barrust/pyspellchecker> (visité le 19/04/2022). Celui-ci procède à une recherche de correspondances entre les formes du texte et un dictionnaire de référence par des permutations de lettres : il est en mesure de proposer des formes considérées comme justes dans une limite de deux fautes par mot. Par exemple, il reconnaît que la meilleure proposition pour le mot *deusx* est *deux*, mais n'est pas capable d'associer la forme *pubiées* aux mots de la famille de *publier*

20. S. Biay et F. Chiffoleau, *textCorrection.Py*, 6 avr. 2022, URL : <https://github.com/sbiay/CdS-edition/blob/fb90ad201a4c8c6fde4fe4e97d7068ebca98f6a3/htr/py/textCorrection.py> (visité le 19/04/2022).

21. *spaCy : Industrial-strength Natural Language Processing in Python*, URL : <https://spacy.io/> (visité le 27/04/2022).

arbre XML et non d'une imitation d'arbre au format `txt`, comme c'était le cas dans l'état du fichier que nous avons repris. En effet, l'injection des transcriptions manuelles en lieu et place des prédictions dans les seuls fichiers appartenant au corpus d'entraînement de la reconnaissance d'écriture entraîne une modification irrémédiable de l'indentation de ceux-ci. L'indentation de ces fichiers étant devenue différente des autres fichiers des prédictions, il n'était plus possible de s'appuyer sur l'identité des indentations pour repérer les lignes de textes à remplacer. Il devenait donc obligatoire de s'appuyer sur la hiérarchie de l'arbre XML pour appliquer ces corrections.

1.9.3 Gérer les résolutions ambiguës

Appliquer des scripts de correction automatique, on l'a signalé plus haut, comporte le risque d'appliquer partout des corrections ne se justifiant que dans certains cas et ainsi de générer des fautes. Le problème de l'ambiguïté des corrections se pose lorsqu'une prédition peut se prêter selon le contexte à plusieurs résolutions différentes : par exemple *cele*, qui peut résulter tantôt de l'oubli d'un *l* (on corrigera en *celle*), tantôt de la reconnaissance d'un *e* à la place d'un *a* (on corrigera en *cela*).

Dans un premier temps nous avons procédé selon une méthode d'automatisation qui neutralisait les corrections ambiguës : *cele* était intégré à la liste globale des corrections avec une absence de lemme afin d'être exclu de la correction automatique.

Cette méthode présentait plusieurs inconvénients :

- Une fois que l'on avait procédé à des corrections pour les mots d'une page, le script qui les intégrait au fichier rassemblant toutes les corrections contrôlait qu'une forme ne puisse pas être associée à plusieurs corrections. Lorsqu'une ambiguïté était repérée, il fallait intervenir sur les deux fichiers pour neutraliser la correction. Devenu fréquent, ce processus diminuait le bénéfice de temps attendu de la correction automatique ;
- D'autre part, il s'est avéré que les corrections ambiguës sont nombreuses, car il suffit d'une faute sur un petit mot pour le rendre ambigu avec un autre mot : *ue* peut être corrigé en *rue* ou en *une*; *veu*s peut être corrigé en *veux* ou en *vous*; *ceste* peut être corrigé en *cesse* ou en *cette*.

Plutôt que de neutraliser la correction de ces mots, il s'est donc avéré nécessaire de prendre en charge ces ambiguïtés.

Il fallait pour cela résoudre une nouvelle difficulté : opérer des corrections automatiques sur de petits mots fréquents a rendu nécessaire l'application des corrections au niveau de chaque ligne d'écriture, car les appliquer à une page entière aurait sans doute entraîné des corrections erronnées.

Afin de faciliter la sélection de la bonne correction parmi une liste de propositions, on a par écrit une nouvelle fonction (`ordreOccurrences`) dont le rôle est de classer les mots

attestés dans les vérités de terrain par ordre décroissant de nombre d'occurrences. Ainsi, le mot le plus fréquent est toujours proposé comme premier choix au correcteur.

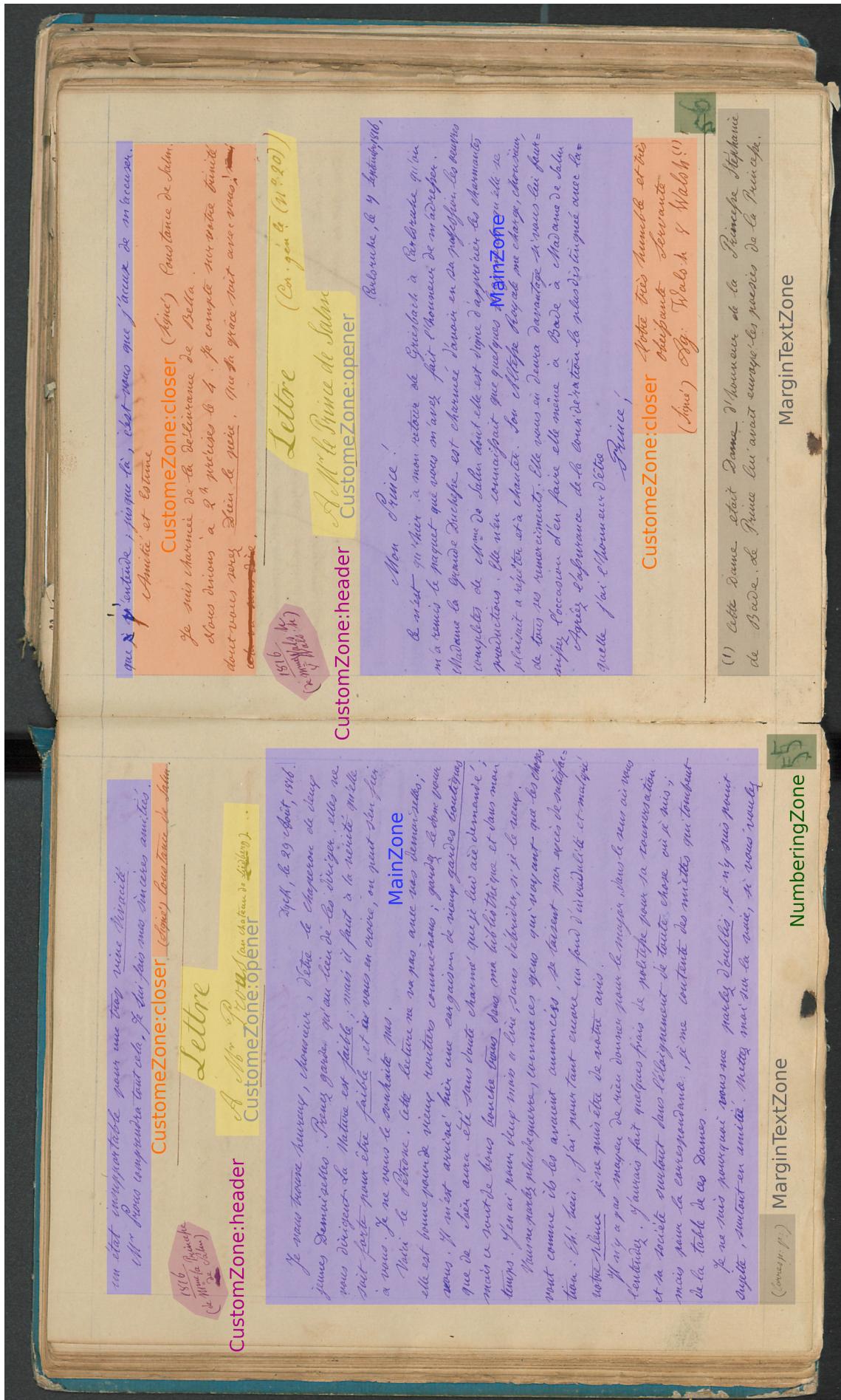


FIGURE 1.3 – Exemple de typage des zones de texte sur une double page.

Annexes

Annexe A

Normes de transcription

A.1 Accentuation

L’usage scribal a été respecté sans normalisation : en cas d’oubli de l’accent sur la préposition *à* on a transcrit *a*.

A.2 Majuscules et minuscules

La casse a été respectée sans appliquer les règles modernes : *je lis les Journaux Allemands*. Les accents ont été appliqués sur les majuscules.

A.3 Séparation des mots

La séparation des mots respecte l’usage graphique du scribe, mais sans imiter l’espacement réel des mots. Ainsi, les élisions, agglutinations ou encore les lexicalisations (consacrées ou fautives) ont été respectées : *d’avantage*, *C’a été*, *tédeum*. Lorsqu’il n’y a aucun doute sur le fait que deux mots sont distincts, même s’ils sont très proches dans l’espace de la page, ils ont été séparés d’une espace.

Nous n’avons pas restitué de trait d’union lorsque l’usage moderne l’imposerait : *portez vous bien*.

Dans le cas particulier de l’écriture personnelle de Constance de Salm, les mots sont très souvent écrits dans un même mouvement de la plume. Dans ce cas seulement, ils ont été transcrits sans espace séparatrice.

A.4 Orthographe

L’orthographe des mots a été respectée : *enfans*, *momens*, *sentimens*, *cahos*.

Lorsque l’orthographe était erronée et changait la prononciation du mot, on a transcrit le mot sans le corriger : *Mr. Pron*s pour *Mr. Prou*s.

A.5 Abréviations

Les abréviations ont été transcrisées sans être résolues : *9bre* pour novembre, *Mr.* pour Monsieur.

L’abréviation *ll* pour livres (unité monétaire) a été transcrise par le caractère .

A.6 Ponctuation

Les signes de ponctuation ont été transcrits fidèlement, y compris les points marquant une pause de la plume sans articulation syntaxique : *je ne sais pas . si vous en serez bien aise*. Les tirets ont été transcrits par le caractère -.

A.7 Passages biffés, palimpsestes

Pour la transcription des phénomènes complexes tels que les passages biffés ou les palimpsestes, on a appliqué les conventions préconisées par la convention de Leyde¹, retenues dans le cadre du Cremma².

On a transcrit tout ce qui était lisible, y compris les lettres biffées, lorsque c’était possible, privilégiant le dernier état du texte et en plaçant le passage corrigé entre crochets : [abc].

On a remplacé chaque lettre biffée illisible par un point et placé l’ensemble des lettres concernées entre crochets : [...] (*pour deux lettres illisibles*).

A.8 Passages illisibles

Pour les problèmes de déchiffrement du texte, la convention de Leyde n’a pas d’autre préconisation que la mention en apparat³

1. « Leiden Conventions », dans *Wikipedia*, 2021, URL : https://en.wikipedia.org/w/index.php?title=Leiden_Conventions&oldid=1004624327 (visité le 05/05/2022).

2. A. Pinche, Séminaire "Création de modèle(s) HTR pour les documents médiévaux en ancien français et moyen français entre le Xe-XIV^e siècle" : compte-rendu de la séance n° 2, CREMMALAB, 2021, URL : <https://cremmalab.hypotheses.org/seminaire-creation-de-modeles-htr/compte-rendu-de-la-seance-n-2> (visité le 05/05/2022).

3. *No sigla were suggested for corruptions (i.e. letters that are legible or restorable, but not understood). Instead, it was proposed that these should be dealt with in an apparatus* (« Leiden Conventions »...).

Bibliographie

Scripts

- BARRUS (Tyler), *Pyspellchecker : Pure Python Spell Checker Based on Work by Peter Norvig*, version 0.6.3, URL : <https://github.com/barrust/pyspellchecker> (visité le 19/04/2022).
- BIAY (Sébastien), *donneesImages.Py*, 6 avr. 2022, URL : <https://github.com/sbiay/CdS-edition/blob/main/htr/py/donneesImages.py> (visité le 19/04/2022).
- *injectTranscript.Py*, 20 mai 2022, URL : <https://github.com/sbiay/CdS-edition/blob/main/htr/py/injectTranscript.py> (visité le 03/06/2022).
- BIAY (Sébastien) et CHIFFOLEAU (Floriane), *spellcheckTexts.Py*, 6 avr. 2022, URL : <https://github.com/sbiay/CdS-edition/blob/main/htr/py/spellcheckTexts.py> (visité le 19/04/2022).
- *textCorrection.Py*, 6 avr. 2022, URL : <https://github.com/sbiay/CdS-edition/blob/fb90ad201a4c8c6fde4fe4e97d7068ebca98f6a3/htr/py/textCorrection.py> (visité le 19/04/2022).