Contenu de la documentation

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 Sozialwissenschften (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

Quatre à cinq mains différentes ont été repérées jusqu'à présent dans la correspondance de 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 mixte [2] nous ont convaincu d'emprunter cette voix.

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

1. Reprendre les tests sur le modèle entraîné de zéro par H. Souvay lors d'un précédent stage consacré à la correspondance de CdS [3] ;

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).

3. Hippolyte Souvay, *La Correspondance de Constance de Salm (1767-1845) : Rapport de Stage*,

2. Reprendre un modèle générique entraîné pour le projet Lectaurep .

1.2 Choisir un corpus d'entraînement

Afin de donner les meilleurs 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 six [4].

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 mains [5].

1.2.1 Écriture de CdS

Le site ne publie aucune lettre originale de la main de CdS, mais 52 brouillons (*Entwurf*) [6].

Entraîner un modèle de reconnaissance sur cette écriture supposerait un travail délicat de transcription pour une écriture particulièrement cursive (compter environ deux semaines pour disposer d'une bonne vingtaine de pages), mais l'investissement peut en valoir la peine.

1.3 Tester et entraîner des modèles de reconnaissance d'écriture

Cette section est à écrire.

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, 1er volume, 1785-1814*, URL : `https://constance-de-salm.de/archiv/#/document/11215` (visité le 11/04/2022) ; Id., *Correspondance générale, seconde copie, 2e volume, 1815-1821*, URL : `https://constance-de-salm.de/archiv/#/document/11216` (visité le 11/04/2022) ; Id., *Correspondance générale, seconde copie, 3e 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

6. Le dépouillement se trouve dans le fichier ./htr/mains/brouillonsCDS.md

1.4 Entraîner des modèles de segmentation des pages

1.4.1 Typer les régions d'écriture

Le typage est utile en ce qu'il permet de traiter de manière différentielle des régions et des lignes afin de les affecter à des éléments distincts de l'arborescence XML-TEI qu'il faudra construire.

Il faut donc réfléchir aux besoins de cette transformation vers le format TEI. Les *Guidelines* de l'édition de correspondance du projet DAHN permettent de guider cette réflexion [7]. Par ailleurs, F. Chiffoleau a formulé une ontologie pour les régions et lignes des écrits de correspondance en langue française pour le XXe siècle [8] dans le cadre du projet SegmOnto : A Controlled Vocabulary to Describe the Layout of Pages (SegmOnto) [9].

La suite de cette partie est dépassée.

Certaines régions peuvent être directement appliquées :

— **Main**
— **Title**
— **Signature**: salutation and signature of the sender;
— **Letterhead**
— **Numbering**
— **Salute**
— **Dateline**: place and date of writing for the letter.

Il pourrait être pertinent de modifier l'usage de :

— **Additions**: *cette catégorie est utilisée ailleurs dans SegmOnto, pour les documents administratifs* [10] *; elle intervient dans le traitement du document postérieurement à sa rédaction. Cette pertinence reste cependant à confirmer. Cette catégorie pourrait également s'appliquer aux rubriques :*

Il pourrait être pertinent de reprendre ou de créer d'autres concepts :

— **Note**: *pour les notes infrapaginales (utilisé dans SegmOnto pour les imprimés* [11]

— **Postscritp**: *cela repmplacerait le rôle à l'origine assigné à* Additions. *J'opterais*

7. F. Chiffoleau, *Correspondence : Guidelines*, DAHN Project, 10 janv. 2022, URL : `https://github.com/FloChiff/DAHNProject/blob/8df8dfc6053a7dd57a6c5510d1e56bb336ce1d04/Correspondence/Guidelines/Documentation-Correspondance.pdf` (visité le 07/04/2022).

8. 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).

9. 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).

10. A. Chagué, *[Documents Administratifs, XIXe s.]* SegmOnto, 10 déc. 2021, URL : `https://github.com/SegmOnto/examples/tree/main/sources/administratif_XIXe` (visité le 07/04/2022).

11. *[Imprimés]*, SegmOnto, 10 déc. 2021, URL : `https://github.com/SegmOnto/examples/tree/main/sources/prints/BnF_cb120117553` (visité le 07/04/2022).

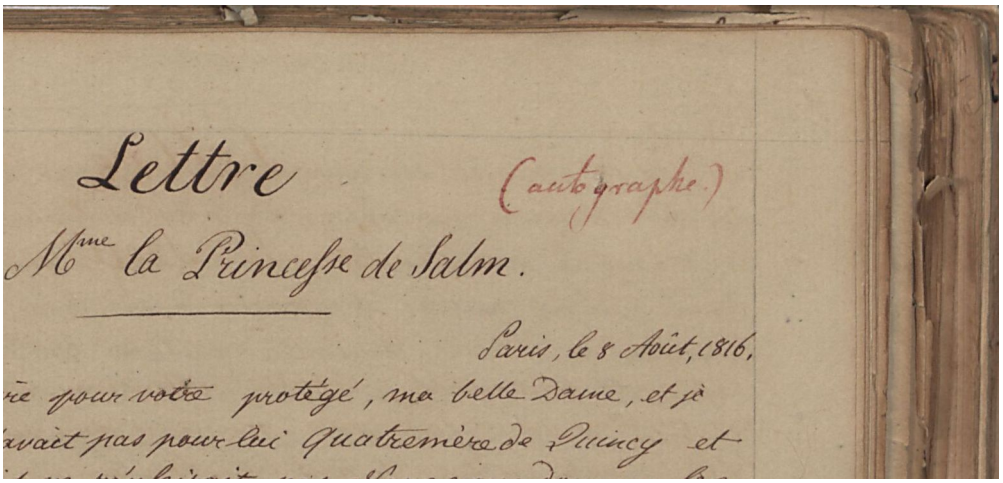

Figure 1.1 – Rubrique "autographe".

bien pour le jaune car il ne va pas me servir par ailleurs, et qu'on ne risque guère d'avoir un tampon proche du post-scriptum.

La figure 1.2 ci-dessous propose une mise en oeuvre de ce typage des régions.

1.4.2 Typer les lignes d'écriture

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

— **Main**

— **Verse** : *les passages en vers sont relativement nombreux*

— **Correction** : *catégorie existant par défaut dans e-Scriptorium, elle s'appliquerait uniquement pour les corrections appliquées dans l'interligne.*

1.4.3 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 DAHN [12]. J'envisage plutôt **ne pas encoder ces éléments dans la phase d'HTR**, et de ne les aborder que la phase d'édition. Il sera de toute façon necessaire, 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 génèrerait du

12. F. Chiffoleau, *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).

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.

1.5 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 *Handwritten Text Recognition* (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 DAHN [13] et proposé quelques développements aux scripts issus de ce projet.

1.5.1 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 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

13. Id., *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).

terrain ; par exemple, dans la prédiction *Dans **vu** siècle où tous les talens…*, la prédiciton erronée *vu* pour *un* ne sera pas corrigée car le mot *vu* est attesté ailleurs [14].

1.5.2 Analyser les mots

Nous avons appliqué le script d'analyse de mots spellcheck-texts.py [15] à nos prédictions HTR [16].

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 pour afficher le contexte du mot et en conserver la mémoire, ce qui limite un peu les allers-retours entre le dictionnaire à corriger et l'image ou la prédiction 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 ainsi une recherche plus coûteuse dans un dictionnaire généraliste de la langue française, évitant également le contrôle de ces mots par l'éditeur.

Les corrections précédemment validées sont, elles aussi, mobilisées lors de l'analyse des prédictions, ce qui permet de réexploiter facilement des corrections.

Les corrections s'avérant nombreuses, le script textCorrection.py [18] écrit par F. Chiffoleau a dû être perfectionné afin de procéder à une tokénisation des mots, pour corriger avec exactitude les formes erronées présentes dans le texte. Nous avons pour cela utilisé le module Spacy [19].

1.5.3 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 prédiction peut se prêter selon le contexte à plusieurs résolutions différentes : par exemple

14. Nous avons 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 fait perdre plus de temps qu'elle n'en fait gagner.

15. Sébastien Biay et F. Chiffoleau, *spellcheckTexts.Py*, 6 avr. 2022, URL : `https://github.com/sbiay/CdS-edition/blob/fb90ad201a4c8c6fde4fe4e97d7068ebca98f6a3/htr/py/spellcheckTexts.py` (visité le 19/04/2022).

16. 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 [17]

18. 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).

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

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 : *uue* peut être corrigé en *rue* ou en *une* ; *veus* 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.

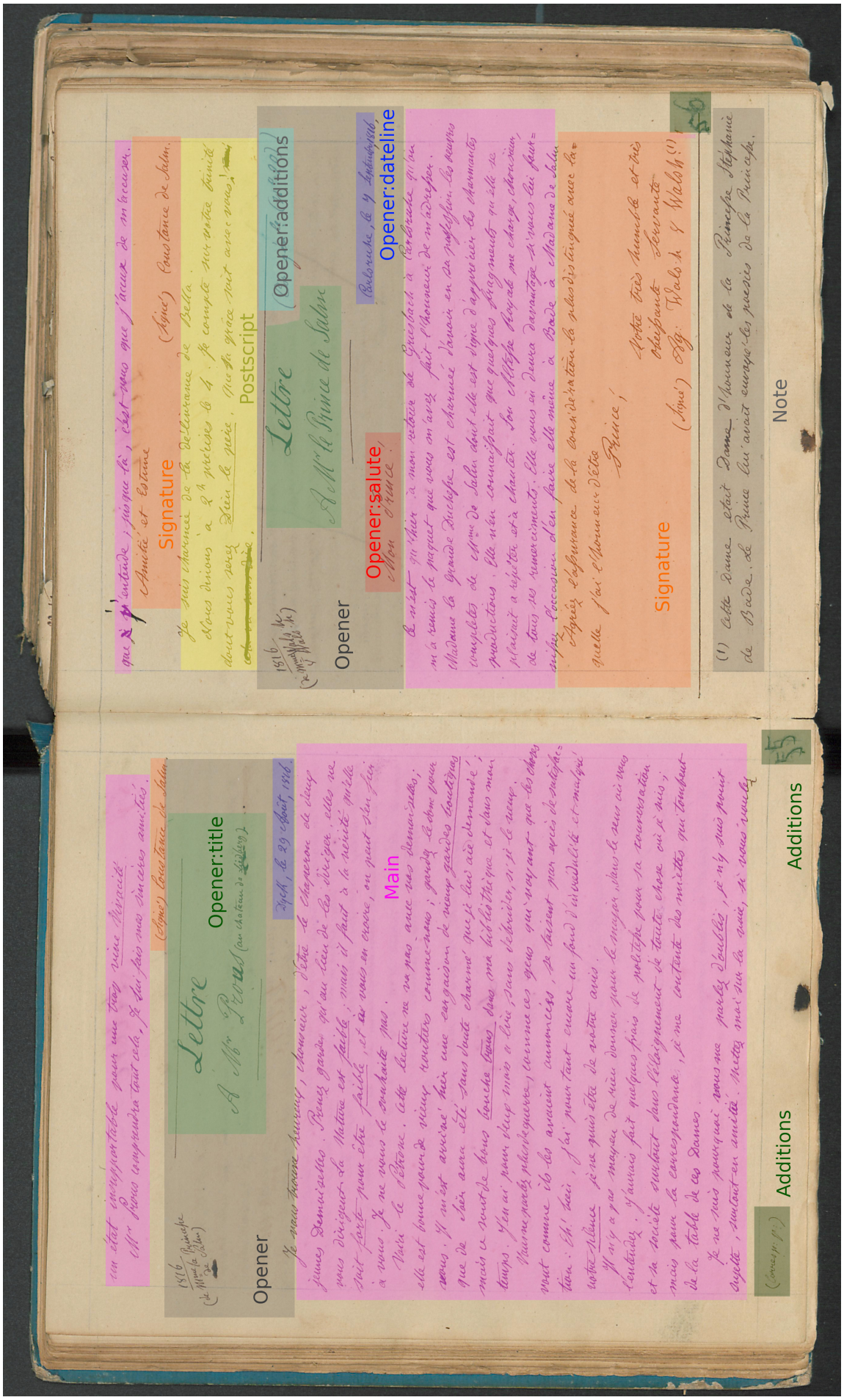

FIGURE 1.2 – Exemple de typage des zones de texte sur une double page.

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 transcrira *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 a été respectée : *d'avantage, Ç'a été, tédeum*.

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

A.4 Orthographe

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

A.5 Abréviations

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

L'abréviation *ll* pour livres (unité monétaire) a été transcrite 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 Corrections

On transcrit tout ce qui est lisible, y compris les lettres biffées, lorsque c'est possible. On privilégie le dernier état du texte dans le cas où la correction a été superposée à la première couche d'écriture.

Lorsque l'orthographe est erronée, on transcrit le mot sans le corriger : *Mr Prons* pour *M. Prous*.

A.8 Lettres illisibles

On remplace chaque lettre illisible par le caractère #.

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) et Chiffoleau (Floriane), *spellcheckTexts.Py*, 6 avr. 2022, URL : `https://github.com/sbiay/CdS-edition/blob/fb90ad201a4c8c6fde4fe4e97d7068ebca98f6a3/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).