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

HTR

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.

Deux pistes méthodologiques se dessinent :

1. Rassembler dans un premier temps des lettres qui sont de la même main, pour voir quels sont les résultats du modèle entraîné par H. Souvay lors d'un précédent stage [1] ;
2. Reprendre un modèle déjà entraîné à travailler sur plusieurs mains ; c'est l'option qui été privilégiée par le projet Lectaurep [2]).

1.2 Choisir un corpus d'entraînement

Les 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. La distribution des mains y est variable selon les tomes :

1. Le premier volume [3] présente une grande variété de mains s'enchaînant fréquemment les unes aux autres ;

1. 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.

2. Alix Chagué, *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. 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).

2. Le deuxième volume [4] présente en revanche une meilleure cohérence paléographique : la même main peut se suivre sur un bon nombre de pages consécutives, facilitant l'entraînement d'un modèle sur une écriture particulière. Nous avons repris ce volume, en partie utilisé par H. Souvay pour ses tests, afin de constituer un premier sous-corpus paléographiquement cohérent ;
3. Le troisième volume [5], où les mains du deuxième volume se retrouvent largement et a pu être joint au précédent.

1.2.1 Main 1

Nous avons établi une liste de 30 images (soit 30 doubles pages) au sein du 2e et du 3e volume attestant une écriture homogène que nous dénommons *Main 1*. Nous avons pour cela sélectionné les lettres afin de ne travailler que sur un seul type d'écriture, sachant que les changements de main interviennent souvent en milieu de page. Quelques corrections de la main de CdS apparaissent ponctuellement.

1.2.2 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 Segmentation et annotation des zones d'écriture

Nous avons procédé à une première expérience de transcription sur le sous-corpus *Main 1* avec le logiciel e-Scriptorium installé localement.

1.3.1 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.

4. Id., *Correspondance générale, seconde copie, 2e volume, 1815-1821*, URL : `https://constance-de-salm.de/archiv/#/document/11216` (visité le 11/04/2022).
5. Id., *Correspondance générale, seconde copie, 3e volume, 1822-1828*, URL : `https://constance-de-salm.de/archiv/#/document/11217` (visité le 12/04/2022).
6. Le dépouillement se trouve dans le fichier ./htr/mains/brouillonsCDS.md

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].

Cetaines 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 :*

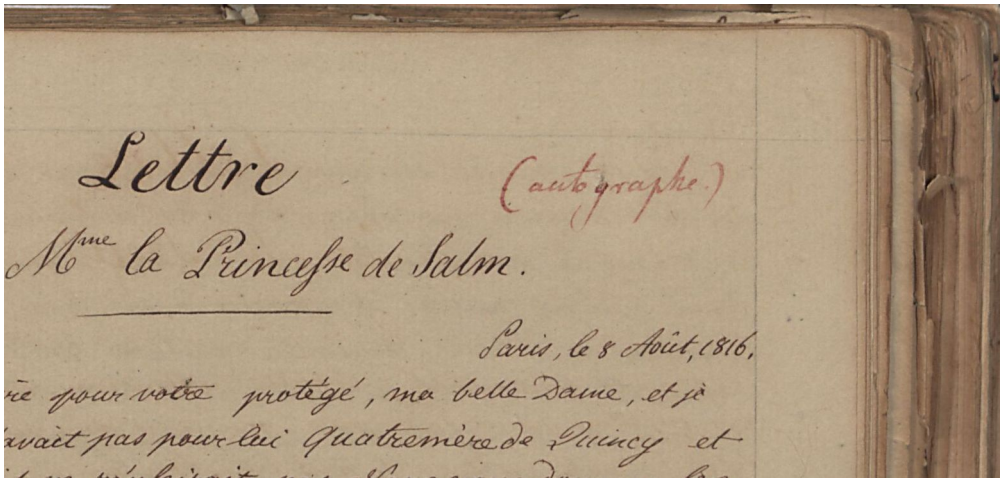

Figure 1.1 – Rubrique "autographe".

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).

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 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.

1.3.2 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.*

1.3.3 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 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.

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

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).

1.4 Mise en oeuvre de la reconnaissance d'écriture

Dans le cadre de son stage, H. Souvay a initié l'entraînement d'un modèle HTR à partir d'un petit volume de vérités de terrain [13]. La méthodologie employée était la suivante :

> Nous avons décidé de tenter la transcription automatique sur un sous-ensemble du corpus composé de copies de lettres compilées dans des recueils. [...] Les mains sont relativement constantes dans le temps dans ce sous-ensemble contrairement au reste du corpus. Même proches, ces mains demeurent différentes. Nous avons donc opté pour l'entraînement d'un modèle multi-mains, c'est à dire un modèle non-spécialisé capable de transcrire plusieurs mains représentées dans le corpus d'entraînement [14]

Il s'agit dans un premier temps d'augmenter le volume des vérités de terrain pour améliorer les performances du modèle entraînés par H. Souvay.

L'objectif visé est de dépasser un taux de précision de reconnaissance de 90% pour chaque main.

1.5 Automatiser la correction des prédictions

Nous avons suivi la démarche explicitée dans la documentation du projet DAHN [15] et proposé quelques développements aux scripts issus de ce projet.

1.5.1 Analyser les mots

Nous avons appliqué le script d'analyse de mots spellcheck-texts.py [16] à nos prédictions HTR [17].

Les corrections sont plus nombreuses sur des prédictions *Handwritten Text Recognition* (HTR) que sur des prédictions OCR, surtout avec un modèle encore peu entraîné. La

13. H. Souvay, *La Correspondance de Constance de Salm (1767-1845) : Rapport de Stage…*
14. *Ibid.*, p. 6-7.
15. F. Chiffoleau, *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).
16. 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).
17. 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 en procédant à des permutations de lettres : il est en mesure de proposer des formes considérées comme justes dans une limite de deux fautes par mot [18]

correction est donc un travail conséquent. Il est en outre à mener avec prudence. Le risque de la correction automatique est de remplacer involontairement des prédictions justes.

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 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 collecte-mots, qui fouille les vérités de terrain déjà constituées et permet de valider rapidement 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.

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 [19] écrit par F. Chiffoleau a dû en outre être perfectionné afin de procéder à une tokénisation des mots, pour corriger avec exactitude les formes erronées présentes dans le texte.

1.5.2 Résoudre les difficultés de lecture : le cas des noms propres

Dans la mesure où les patronymes représentent une difficulté majeure dans la transcription du texte, il est nécessaire de se reporter aux notices du publiées sur le site constance-de-salm.de. Afin de relier facilement les reproductions numériques de la correspondance et les notices publiées sur le site, on a écrit un script intitulé images.py [20] qui gère les images par dossiers en générant un tableau de données pour toutes les images du dossier désigné ; on a également mis en place un notebook pour une utilisation simplifiée de ce script.

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 dans deux cas de figure :

— Lorsqu'une prédiction 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

19. 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).

20. S. Biay, *Images.Py*, 6 avr. 2022, URL : `https://github.com/sbiay/CdS-edition/blob/fb90ad201a4c8c6fde4fe4e97d7068ebca98f6a3/donnees/images.py` (visité le 19/04/2022).

en *celle*), tantôt de la reconnaissance d'un *e* à la place d'un *a* (on corrigera en *cela*) ;

— Lorsque les erreurs d'une prédiction donnent naissance à une orthographe attestée par un autre mot : *Dans* ***vu*** *siècle où tous les talens.*

Dans un premier temps nous avons procédé selon une méthode d'automatisation qui neutralisait les corrections ambiguë (*cele* était intégré à la liste globale des corrections avec une absence de lemme afin d'être exclu de la correction automatique) et l'on excluait du processus de correction toutes les orthographes attestées, comme *vu*.

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* ;

— Enfin, les prédictions proposant un mot attesté à la place d'un autre ne sont pas rares non plus, car elles peuvent concerner des mots très fréquents : *le* confondu avec *la*, *nous* confondu avec *vous*.

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 très fréquents (*un, une, le, la*) a rendu nécessaire l'application des corrections au niveau de chaque ligne d'écriture, car les appliquer à une page entière aurait fatalement entraîné des corrections erronnées. La probabilité que les mots *le* et *la* soient tous les deux présents dans une ligne d'écriture avec l'un juste et l'autre faux est en effet très faible : on peut sans grand dommage changer tous les *le* d'une ligne en *la*. En revanche, modifier tous les *le* en *la* à l'échelle d'une page produirait à l'évidence un résultat catastrophique.

Conserver en mémoire les différentes résolutions possibles d'une forme pour les proposer à la correction d'une prochaine page risque en outre d'ajouter au travail du correcteur celui de sélectionner la bonne lecture d'un mot… déjà juste !

On a par conséquent 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 afin de faciliter le travail de sélection.

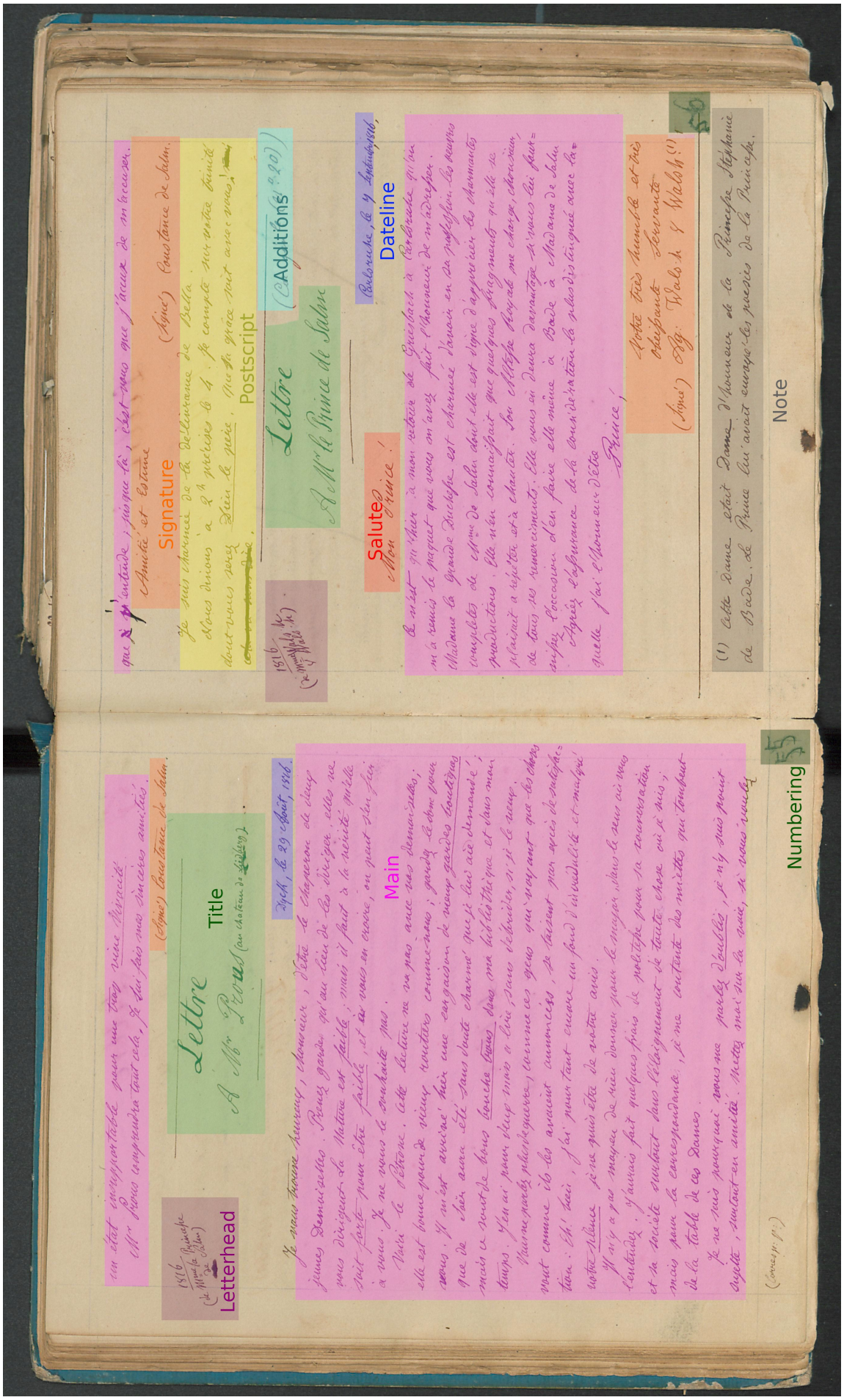

FIGURE 1.2 – Exemple de typage des zones de texte sur une double page.

Annexes

Annexe A

Normes de transcription

A.1 Accentuation

L'usage des accents a été normalisé selon les règles modernes.

A.2 Majuscules et minuscules

La casse a été respectée sans appliquer les règles modernes : *je lis les Journaux Allemands.*

A.3 Coupure des mots

La coupure des mots a été respectée : *d'avantage, Ç'a été.*

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.

A.6 Ponctuation

Les signes de ponctuation ont été transcrits fidèlement.

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*.

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), *Images.Py*, 6 avr. 2022, URL : https://github.com/sbiay/CdS-edition/blob/fb90ad201a4c8c6fde4fe4e97d7068ebca98f6a3/donnees/images.py (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).