Publié dans Ressi, no 21, décembre 2021,à citer pour faire référence à ce papier

L'océrisation d'imprimés anciens : les sciences de l'information au service des Humanités

Florence Burgy, Haute Ecole de Gestion, Genève

Résumé

Cet article présente un projet de recherche, en collaboration avec le Bodmer Lab, qui a consisté à océriser des imprimés latins de la Renaissance, afin d'en obtenir une transcription et de la rendre explorable par la recherche plein texte.

Quatre logiciels d'océrisation gratuits et open source ont été testés, avec comme métriques la précision, le rappel, et la F-mesure (F1) au niveau des caractères et au niveau des mots. Tesseract et OCR4all étaient les plus performants, mais ce dernier présentait un problème technique qui rendait son utilisation complexe. Tesseract, qui présentait alors une F1 de 78.62% (caractères) et 31.78% (mots) a donc été retenu pour la suite du projet.

Différentes méthodes ont été testées pour améliorer les résultats obtenus. Toutes les méthodes n'étaient pas nécessairement efficaces, mais grâce à certaines, une F1 de 80.06% au niveau des caractères et de 34.58% au niveau des mots a pu être obtenue.

Abstract

This article presents a research project, in collaboration with Bodmer Lab, which consisted of using an OCR software on Latin prints from the Renaissance, in order to obtain a transcription and make it explorable through full-text research.

Four free and open source OCR software were tested, with accuracy, recall, and F-measurement (F1) at character and word level as metrics. Tesseract and OCR4all were the best performers, but the latter presented a technical problem that made its use complex. Tesseract, which at the time had an F1 of 78.62% (characters) and 31.78% (words) was therefore chosen for the rest of the project.

Different methods were tested to improve the results obtained. Not all methods were necessarily effective, but thanks to some of them, an F1 of 80.06% at character level and 34.58% at word level could be obtained.

Zusammenfassung

Dieser Artikel stellt ein Forschungsprojekt in Zusammenarbeit mit dem Bodmer-Lab vor, das darin bestand, ein OCR-software mit lateinischen Drucken aus der Renaissance zu benutzen, um eine Transkription zu erhalten und sie durch Volltextforschung erforschbar zu machen.

Vier freie und open source OCR software wurden getestet, wobei Genauigkeit, Rückruf und F-Messung (F1) auf Zeichen- und Wortebene als Metriken verwendet wurden. Am besten schnitten Tesseract und OCR4all ab, wobei letzteres ein technisches Problem aufwies, das seine Anwendung komplex machte. Daher wurde Tesseract, das zu diesem Zeitpunkt einen F1 von 78,62% (Zeichen) und 31,78% (Wörter) hatte, für den Rest des Projekts gewählt.

Es wurden verschiedene Methoden getestet, um die erzielten Ergebnisse zu verbessern. Nicht alle Methoden waren notwendigerweise effizient, aber dank einiger von ihnen konnte eine F1 von 80,06% auf Zeichenebene und 34,58% auf Wortebene erreicht werden.

Mots-Clés: Humanités numériques, OCR, Reconnaissance Optique de Caractères, Intelligence Artificielle, Bodmer Lab, Tesseract

1. L'imprimeur et la machine

1.1. Les Humanités numériques

« Allier les humanités à l'informatique ou aux technologies numériques. » : comme l'explique Pierre Mounier dans son ouvrage daté de 2018, il ne s'agit pas là d'une idée neuve. On pourrait en effet en observer des applications dans les années 1960-1970 déjà, voire dès la fin de la seconde guerre mondiale (Mounier 2018), mais c'est avec la démocratisation de l'informatique et le développement des sciences de l'information que cette idée et sa mise en application ont commencé à s'imposer globalement. Ces pratiques sont très variées, couvrant toutes les étapes de la recherche, de la création, du traitement et de la gestion de données à la valorisation de résultats et de documents sources. L'expression « Humanités numériques », « Digital Humanities » en anglais, née tardivement, au début des années 2000 (Mounier 2018), apparaît donc comme un « big tent », un vaste terme englobant un très grand nombre d'usages et de techniques (Terras 2016).

En Suisse romande, cette évolution a notamment été marquée par l'ouverture du DHLAB de l'EPFL par Frédéric Kaplan en 2012 (EPFL 2020), auquel sont liés des cursus de formation Bachelor, Master et PhD, ainsi que par la création d'une chaire en Humanités numériques à l'Université de Genève en 2019 (Université de Genève 2020). En parallèle, et depuis 2014, un autre projet fait beaucoup parler de lui : le Bodmer Lab.

1.2. Le Bodmer Lab et la collection de Bry

Le Bodmer Lab se définit comme "un projet de recherche et de numérisation issu d'un partenariat entre l'Université de Genève et la Fondation Martin Bodmer" (Bodmer Lab 2019), dont l'objectif est de rendre accessible à un large public des documents provenant de la Bibliotheca Bodmeriana. Cette collection, que Martin Bodmer voulait "bibliothèque de la littérature mondiale" (Bodmer Lab 2019) au sens où l'entendait Goethe, regroupe en effet de nombreux ouvrages anciens, rares et fragiles, dont la valeur historique est unique. La numérisation de ces documents permet de les faire connaître et de les rendre exploitables par des chercheurs comme par le grand public. Un important travail de mise en valeur et de médiation complète ce processus.

Au sein de la Bibliotheca Bodmeriana, certains ensembles de documents présentent une structure suffisamment cohérente pour pouvoir être considérés comme des sortes de sous-collections. Ce fait a mené le Bodmer Lab à traiter ces ensembles, ou "constellations", de manière indépendante et à en confier l'étude à des spécialistes des domaines concernés (Bodmer Lab 2019).

L'une de ces "constellations" est la collection de Bry, qui rassemble des récits de voyage de l'époque des Grandes découvertes. Cette collection exceptionnelle de vingt-neuf volumes illustrés par des gravures et datant des XVIe et XVIIe siècles est divisée en deux parties : les « Grands Voyages », ou India occidentalis, qui retracent l'exploration des Amériques, et les « Petits Voyages », ou India orientalis, qui concernent essentiellement les voyages en Afrique et en Asie. Ces ouvrages sont l'œuvre de l'éditeur-imprimeur liégeois Théodore de Bry (1528-1598) et de ses descendants. Ces volumes rares sortent de leur atelier de Francfort entre 1590 et 1634 et sont édités en plusieurs langues (Bodmer Lab 2019).

La première édition latine de cette collection, que la Fondation Bodmer possède dans son intégralité, est déjà numérisée et accessible en ligne sur le site du Bodmer Lab (Bodmer Lab 2019), mais il est impossible pour le moment d'y effectuer une recherche plein texte.

1.3. L'océrisation : un enjeu des Humanités numériques

Cette problématique est directement liée à un enjeu majeur des Humanités numériques : l'océrisation. De l'acronyme OCR, pour Optical Character Recognition, reconnaissance optique de caractères en français, il s'agit d'une technologie qui permet d'identifier des caractères dans un document numérisé au format image et de les extraire dans un format texte lisible par un humain comme par une machine, de manière totalement automatisée.

Si certains logiciels permettent aisément cette conversion avec des documents récents, cette tâche est rendue bien plus ardue avec des documents anciens. En effet, l'état du papier et de l'encre, la grande variété des polices de caractères et des mises en page et la méthode de numérisation sont parmi les éléments qui influent sur les performances d'un logiciel.

La collection de Bry est un exemple intéressant de document pour lequel un simple logiciel OCR ne suffit pas. Après un état de l'art de la technologie OCR, le présent article présente les méthodes et résultats d'un projet de recherche visant à tester différents logiciels d'océrisation afin de sélectionner le mieux adapté à ladite collection, Tesseract, puis à l'entraîner et à en optimiser les paramètres dans le but d'obtenir une transcription automatique au plus proche de l'original, avec un objectif de 95% de caractères et mots corrects.

2. OCR : un état de l'art

Le présent état de l'art s'articulera selon deux axes. Le premier présentera la technologie OCR dans son état actuel, en retraçant brièvement son histoire, en expliquant son fonctionnement et en présentant des méthodes d'évaluation des logiciels OCR. Le second axe présentera les développements récents dans le domaine de l'océrisation, et se focalisera plus précisément sur la problématique des documents anciens et des langues anciennes.

2.1. OCR – une technologie mature

2.1.1. Origines et développement de l'OCR

La technologie connue sous le nom d'OCR a vu le jour dans la première moitié du XXème siècle. Malgré de premiers développements intéressants, l'idée de créer des machines capables de lire les caractères et les chiffres est restée un rêve jusque dans les années 1950 (Mori, Suen, Yamamoto 1992). C'est à cette époque que l'OCR trouve son marché et se développe non seulement comme technologie mais

comme produit commercial, sous l'impulsion, entre autres, de David Shepard, fondateur de Intelligent Machines Research Corporation (Nagy 2016).

Les progrès dans ce domaine sont rapides, et il serait impossible de citer les nombreuses recherches entreprises au cours des soixante dernières années. Mentionnons cependant la machine de Jacob Rabinow, développée dans les années 1960 et permettant de lire et trier les adresses postales américaines, et celle de Kurzweil, dans les années 1970, permettant la reconnaissance et la lecture de textes aux aveugles (Nagy 2016). Un historique plus complet des développements dans ce domaine à cette époque se trouve dans l'ouvrage de Herbert F. Schantz, The history of OCR, optical character recognition (Schantz 1982). Notons en outre que, dans un article proposant un panorama de l'évolution des technologies de l'information dans les années 1990, le développement des OCR est mentionné parmi les progrès importants (Bowers 2018).

Plus proche de nous, le projet de numérisation Google Books, entamé en 2004, a permis une grande reconnaissance de la technologie OCR et des possibilités qu'elle offre (Nagy 2016). En 2005, la mise à disposition du premier logiciel OCR libre, Tesseract (Smith 2007), ouvre la voie à une large diffusion de cette technologie (Blanke, Bryant, Hedges 2012).

Depuis, de nombreux projets de recherche dans ce domaine ont vu le jour, mais il est important de mentionner le plus influent au niveau européen, IMPACT. Ce projet à financement européen lancé en 2008 vise à proposer des outils et des méthodes de travail permettant l'océrisation de documents historiques numérisés avec un très haut niveau de précision (Balk, Ploeger 2009). L'aboutissement principal de ce projet est la création du centre de compétences IMPACT (IMPACT 2013) qui propose des outils, des lexiques et des numérisations pouvant servir de données d'entraînement.

2.1.2. Fonctionnement et perfectionnement

Avant l'océrisation, une numérisation de qualité est indispensable. Il est habituellement recommandé de choisir un format TIFF non compressé et de préparer les numérisations, en rognant les parties sans texte des images, par exemple (Zhou 2010), ceci afin de simplifier le travail de l'OCR. En outre, il existe des méthodes d'évaluation automatique des numérisations, permettant d'assurer une précision optimale des OCR (Brener, Iyengar et Pianykh 2005).

L'océrisation à proprement parler se déroule en quatre phases : le prétraitement, ou preprocessing, la segmentation, ou layout analysis, la reconnaissance, ou recognition, et le post-traitement, ou post-processing (Blanke, Bryant, Hedges 2012). La première phase consiste essentiellement à supprimer le bruit puis binariser pour distinguer les caractères (valeur des pixels 1) du fond (valeur des pixels 0). La seconde permet de repérer les lignes de textes et de délimiter les caractères. La troisième implique une extraction et une classification des features pour reconnaître lesdits caractères (Anugrah, Bintoro 2017), et la dernière phase consiste à corriger l'output pour diminuer le taux d'erreurs (Blanke, Bryant, Hedges 2012).

Pour cette dernière phase, plusieurs méthodes sont utilisées. Le machine learning, ou apprentissage supervisé, est rapidement apparu comme une solution efficace. Un article de 1992 précise déjà que l'usage du machine learning a permis aux auteurs de corriger 46% des erreurs d'océrisation avec un taux de précision de 91% sans intervention humaine (Sun et al. 1992).

De nos jours, le machine learning demeure l'une des solutions les plus recommandées pour la correction, souvent couplée avec l'utilisation de modèles de langue et de dictionnaires (Kissos, Dershowitz 2016). Ces deux procédés nécessitent néanmoins la présence de données d'entraînement

fiables, c'est-à-dire des textes numérisés du même type et océrisés parfaitement, les ground truth. Certains chercheurs ont cependant tenté de proposer des méthodes de correction en l'absence de ground truth, basées notamment sur le traitement de l'information contextuelle, avec des résultats plutôt satisfaisants (Ghosh et al. 2016).

Des méthodes plus récentes proposent l'usage d'apprentissage statistique (Mei et al. 2018) ou encore de la distance de Levenshtein, qui permet de mesurer la différence entre des chaînes des caractères (Hládek et al. 2017). Un outil open source développé dans le cadre du projet IMPACT, PoCoTo, permet d'accélérer le repérage et la correction d'erreurs sur la base de modèles de langues correspondants (Vobl et al. 2014).

Des démarches moins techniques, mais non moins demandeuses en matière de ressources humaines, sont également en usage, tel le crowdsourcing, qui permet d'impliquer des volontaires pour corriger les erreurs de l'OCR (Clematide, Furrer, Volk 2016). Ce crowdsourcing peut prendre des formes diverses, à l'image des jeux créés par la Biodiversity Heritage Library et testés avec succès (Seidman et al. 2016) En Suisse, la plateforme e-newspaperarchives.ch propose une fonctionnalité de correction d'OCR.

2.1.3. Évaluation des performances

Dans un projet d'océrisation, il est essentiel d'évaluer les performances des logiciels OCR afin de sélectionner le mieux adapté et/ou d'optimiser celui sélectionné. Dans les années 1990 déjà, une équipe de l'Information Science Research Institute publiait annuellement les résultats de tests de précision des logiciels OCR disponibles sur le marché à l'époque. Leur méthode consistait à océriser un échantillon aléatoire de documents de différentes natures (journaux, textes de lois etc.) et en différentes langues afin d'estimer lequel présentait le moins d'erreurs.

La métrique principale utilisée dans leurs recherches est la précision des caractères, selon le calcul suivant : , où n représente le nombre de caractères de l'output (Rice, Jenkins, Nartker 1996).

Cette métrique est encore utilisée de nos jours et enrichie. Un article de 2018 (Karpinski, Lohani, Belaïd 2018) propose en effet les calculs suivants :

Erreurs = caractères ajoutés + caractères omis + caractères substitués.

Caractères corrects = caractères de l'output – erreurs.

Précision = caractères corrects / caractères dans l'output (Hypothesis zone). Il s'agit de fait de la métrique utilisée par l'Information Science Research Institute présentée ci-dessus

Rappel = caractères corrects / caractères dans l'input (Reference zone).

Ces mêmes calculs peuvent être étendus aux mots entiers afin de déterminer si la segmentation a été effectuée correctement (Saber et al. 2016).

Pour permettre de faire la balance entre précision et rappel, la moyenne harmonique de ces métriques, nommée « F-mesure », ou F1, a également été utilisée. Elle se calcule ainsi : où est la précision et le rappel (Sasaki 2007).

Ceci permet en effet de prendre ces deux métriques en compte au sein d'un seul calcul (Bao, Zhu 2014). Un outil open source, ocrevaluation, a été développé pour permettre d'automatiser ces calculs sur la base des ground truth et des outputs proposés par l'OCR sélectionné (Carrasco 2014).

2.2 OCR - un champ de recherche en mouvement

2.2.1. Réseaux neuronaux artificiels

Parmi les avancées récentes dans le domaine de l'océrisation, l'utilisation de réseaux neuronaux artificiels est en pleine expansion. Le machine learning, au sens d'apprentissage supervisé, est déjà une pratique bien établie dans le domaine, mais l'usage d'apprentissage non-supervisé permet de nouveaux progrès.

En 2014 déjà, un article propose un état de l'art de l'usage des réseaux de neurones pour le prétraitement des documents avant océrisation (Rehman, Saba 2014). Cette technologie permet en effet de faciliter le repérage des lignes de textes, la segmentation et l'extraction de features, mais à l'époque de cet article, de nombreux problèmes se posent encore quant à ses limites, et au temps et à la masse de données d'entraînement nécessaires à son bon fonctionnement.

Un article de 2016 propose l'utilisation de la back propagation avec descente du gradient, qui permet de limiter la répercussion des erreurs. Dans cet article, les réseaux neuronaux sont essentiellement utilisés pour la classification et la reconnaissance des caractères à partir des pixels, et les résultats sont très positifs avec les caractères alphanumériques anglais. Les auteurs soulignent cependant que, dans le cas d'autres écritures, et entre autres celles présentant des ligatures entre les lettres, les résultats sont peu satisfaisants (Afroge, Ahmed, Mahmud 2016).

Dans un article de 2017 cité plus haut, des réseaux neuronaux sont également utilisés pour l'extraction de features et la classification, et les auteurs recommandent de procéder à une réduction du bruit au préalable pour réduire le temps d'entraînement du réseau neuronal et améliorer ses performances (Anugrah, Bintoro 2017).

Enfin, un article de 2019 présente un exemple d'ICR, pour intelligent character recognition – un OCR spécialement entraîné pour reconnaître les textes manuscrits – qui utilise un CNN, ou convolutional neural network, un type de réseau neuronal utilisé surtout pour la reconnaissance d'images non-textuelles. Cette technologie permet en effet de reconnaître une plus grande variété de caractères et de signes de ponctuation, d'après les auteurs (Ptucha et al. 2019).

Bien que les technologies présentées ici n'ont pas pu être utilisée dans le cadre du projet présenté ci-après, elles demeurent un champ de recherche que de futurs projets devront prendre en compte autant que possible.

2.2.2. Multilinguisme et systèmes d'écriture divers

L'un des domaines dans lesquels l'océrisation avance considérablement est la grande variété de langues et de formes d'écritures pour lesquelles un OCR peut proposer des résultats satisfaisants. Certaines langues non-européennes, comme le japonais ou le mandarin, ont très vite trouvé leur place au sein de la recherche dans ce domaine, du fait du vaste marché possible. D'autres, moins répandues ou moins connues des chercheurs, ne reçoivent de l'attention que depuis peu.

C'est le cas, par exemple, des langues d'Inde, comme l'odia (Dash, Puhan, Panda 2017), le bangla, le devanagari, le tamoul etc. (Kumar et al. 2018). D'autres, comme l'arabe et le farsi, sont des objets de recherche depuis longtemps, mais nécessite encore du travail du fait de la complexité de leur système alphabétique et numérique (Amin Shayegan, Aghabozorgi 2014 ; Alghamdi, Teahan 2017). Il en est de

même pour le finnois, dont la richesse des inflexions rend les résultats des OCR encore parfois hasardeux (Järvelin et al. 2016).

Ces langues ont cependant l'avantage d'être dotées d'une production écrite riche, offrant une abondance de données d'entraînement. D'autres, comme le yiddish et l'occitan, présentent une faible quantité de données disponibles. Dans ce type de cas, la création de lexiques et l'établissement de traits spécifiques des langues et des caractères en amont est conseillé, afin d'améliorer les résultats de l'apprentissage supervisé (Urieli, Vergez-Couret 2013).

2.2.3. Documents historiques et langues anciennes

Imprimés anciens

L'océrisation de documents historiques est l'un des champs de recherche phares dans le domaine. En effet, les imprimés anciens présentent de grandes variations quant aux typographies utilisées, et l'usage ou non de ligature ainsi que l'état général du document sont des éléments pouvant limiter les performances des OCR.

Dans un article de 2015, la Bibliothèque Nationale d'Autriche présentait ses projets « Austrian Books Online », « Austrian Newspapers Online » et « Europeana Online », des projets de numérisations et d'océrisation permettant la recherche plein-texte dans des documents historiques. Parmi les problèmes rencontrés, les auteurs notent l'utilisation contrainte de lexiques et modèles de langue modernes, mal adaptés à ce type de documents n'ayant pas de modèles de langues anciennes à disposition. Ils notent cependant que le projet IMPACT, mentionné plus haut, a entre autres permis de reconnaître l'importance de l'implémentation de lexiques et de modèles de langues anciennes adaptés. L'OCR seul ne peut pas tout faire (Kann, Hintersonnleitner 2015).

S'assurer que l'OCR est entraîné avec des données adéquates est donc indispensable, qu'il s'agisse de données linguistiques, au point de faire intervenir la technicité de la linguistique de corpus (Tumbe 2019), ou de signes d'écriture. En effet, certaines langues ont subi une évolution rapide de leur système d'écriture au cours de leur histoire. Un article de 2016 présente le cas d'imprimés roumains produits entre le XVIIIème et le XXème siècle, dont l'écriture a beaucoup évolué, passant du cyrillique à différentes versions simplifiées de cet alphabet, puis enfin à l'écriture latine.

Là aussi, l'entraînement de l'OCR s'est fait à l'aide de données spécifiques – des lettres cyrilliques et latines roumaines des différentes époques concernées. Sans ces données, les performances du logiciel étaient fort limitées (Cojocaru et al. 2016).

Très récemment, un projet de l'Université de Würzburg en Allemagne a abouti à la création d'un logiciel libre, OCR4all, spécialement conçu pour traiter des imprimés anciens (Jost 2019). Cet outil a fait partie de ceux testés dans le cadre du projet présenté ici.

Manuscrits

La problématique des écritures se posent d'autant plus dans le cas de textes manuscrits, qui présentent de nombreuses difficultés pour les chercheurs dans le domaine de l'océrisation, et pour les humanités numériques en général.

Dominique Stutzmann, chargée de recherche à l'Institut de recherche et d'histoire des textes (IRHT), écrivait en 2017 que « [l]es années qui s'ouvrent sont certainement celles d'une interaction intense,

aux bénéfices réciproques, entre l'homme et la machine en paléographie » (Stutzmann 2017), la paléographie étant l'étude et la transcription de manuscrits anciens.

En effet, beaucoup de chercheurs se penchent actuellement sur la question, et une compétition a même été organisée pour stimuler la recherche dans le domaine de la paléographie numérique. Un article de 2017 en retrace le déroulement, les méthodes développées dans ce cadre et les résultats, plutôt positifs (Kestermont, Christlein, Stutzmann 2017).

Un article plus récent encore se penche sur Transkribus, une plateforme libre de HTR, ou handwritten text recognition, et en démontre l'efficacité, dans le cas du corpus testé du moins (Muehlberger et al. 2019). La paléographie numérique a de beaux jours devant elle.

Langues anciennes : le cas du latin

Ne pouvant aborder le cas de toutes les langues anciennes, nous nous focaliserons sur le latin, et plus spécifiquement le latin de l'époque moderne, ou Early Modern Latin, car il s'agit de la langue qui concerne le projet de recherche présenté dans cet article.

Du fait de son corpus extrêmement riche, le latin est une langue ancienne qui a depuis longtemps intéressé les chercheurs dans le domaine de l'océrisation. En 2006 par exemple, un article signale que les OCR de l'époque ne sont pas adaptés au traitement de cette langue, et propose l'implémentation de modèles de langue spécifiques, une solution déjà mentionnée plus haut (Reddy, Crane 2006).

Une problématique propre au latin, qui concerne également notre projet, est celle des abréviations. En effet, il est très fréquent de rencontrer des abréviations dans les textes latins, manuscrits comme imprimés. Par exemple, « dns » peut remplacer dominus, le seigneur, ou encore un tilde sur une voyelle signale généralement qu'elle est suivie d'un « m » ou d'un « n ». Un logiciel OCR ne peut pas a priori traiter ce type de cas. Pourtant, un article de 2003 propose déjà une solution, via un algorithme permettant de déterminer les résolutions possibles d'une abréviation et de sélectionner la meilleure en fonction du contexte (Rydberg-Cox 2003).

Actuellement, certains outils tentent de répondre à ces problèmes en se spécialisant dans le traitement de textes anciens, comme OCR4all mentionné plus haut, voire dans les textes latins, comme Latinocr.org, qui propose des jeux de données d'entraînement.

3. Tests de logiciels OCR

La première partie du projet consistait à tester différents logiciels d'océrisation open source afin de déterminer lequel offrait les meilleurs résultats sans post-correction. Ce chapitre présente cette phase du projet.

3.1 OCR sélectionnés

Pour des raisons de faisabilité, il n'était pas possible de tester tous les logiciels OCR gratuits et open source disponibles, et le choix s'est donc porté sur quatre d'entre eux. Tesseract et OCR4all ont été sélectionnés car ce sont ceux que la littérature récente mentionne le plus, et Kraken et Calamari, car ce sont les forks les plus à jour d'OCRopy, anciennement OCRopus, un projet également très présent dans la littérature.

3.1.1. Tesseract

Tesseract est un logiciel d'océrisation développé initialement par Hewlett Packard entre 1984 et 1994, puis rendu open source en 2005 (Smith 2007). Il a ensuite été repris en 2006 par Google, qui en assure depuis la maintenance et l'a mis à disposition sous la licence Apache-2.0 sur github.com/tesseract-ocr. Il a pour avantage de proposer des modèles pré-entraînés dans de nombreuses langues, avec la possibilité de combiner les modèles entre eux. Il autorise en outre la création de modèles sur la base de numérisations.

3.1.2. Kraken

Kraken est un fork du projet OCRopy, lancé en 2007 par Thomas Breuel, du Deutsches Forschungszentrum für Künstliche Intelligenz, avec le soutien de Google (Breuel 2007). Kraken est supposé rectifier certains problèmes que posent OCRopus, mais présente des fonctionnalités similaires. Comme Tesseract, il propose quelques modèles pré-entraînés et offre la possibilité d'en entraîner soi-même. Il est développé en Python, conçu pour être utilisé sur Linux, et a son site dédié : kraken.re.

3.1.3. Calamari

Le logiciel d'océrisation Calamari, lancé en 2018, est basé sur les projets OCRopy et Kraken. Il est également implémenté en Python et utilise des réseaux neuronaux artificiels pour optimiser ses résultats (Wick, Reul, Puppe 2018). Il est disponible en ligne sur github.com/Calamari-OCR.

3.1.4. OCR4all

OCR4all est un projet de l'Université de Würzburg en Allemagne lancé en 2019. Il a été conçu pour traiter les documents historiques et est doté d'une interface qui facilite son utilisation, sans que des connaissances en informatiques préalables soient nécessaires (Jost 2019). Le projet, qui intègre déjà différents logiciels, tels que Calamari et Kraken, est en cours d'intégration de Tesseract pour la reconnaissance de caractères. Il est à disposition du public sur github.com/OCR4all.

3.2. Méthodologie

3.2.1. Données d'entraînement

Le jeu de données étant composé de 29 livres numérisés contenant plus d'une centaine de pages chacun, 29 images ont été sélectionnées comme données d'entraînement, chacune extraite de l'un des 29 livres. Cette sélection, faite au hasard à l'aide d'un script Python a permis d'obtenir un échantillon de chacun des livres, ceux-ci pouvant présenter des variantes au niveau de la typographie.

Ces numérisations ont ensuite été transcrites manuellement dans des fichiers textes afin d'obtenir le ground truth, la « transcription-témoin », c'est-à-dire l'objectif à atteindre pour l'OCR. Ceci permet de mesurer les performances des différents logiciels à tester.

Ces transcriptions ont en outre été réalisées de deux manières différentes : une première transcription dite « diplomatique », au plus proche du document, et donc au plus proche des résultats qu'un logiciel

OCR devrait pouvoir obtenir, et une seconde transcription dite « normalisée », qui servira de base à la post-correction.

Dans cette dernière les abréviations ont été résolues et des choix ont été faits pour simplifier la recherche et la lecture du texte. Le caractère æ a été remplacé par ae, les i et les j ont tous été remplacés par des i et les u et les v ont tous été remplacés par des u, sauf lorsqu'il s'agissait de chiffres romains. Ces choix ont été faits sur la base d'habitudes de recherches en latin et de règles d'orthographe usuelle de cette langue.

3.2.2. Logiciels et paramétrage

Pour chacun des logiciels d'océrisation choisis, les tests ont été effectués en trois phases.

La première phase consistait à tester les différents logiciels avec leur modèle standard, leurs paramètres par défaut et sans aucune modification de notre part. Ceci a permis, sur un premier test, de voir quel logiciel était le plus performant, avec ses réglages de base. Les résultats sortis étaient alors totalement bruts.

Lors de la seconde phase, d'autres modèles que ceux standards ont été testés avec différents réglages proposés par chaque logiciel. Ceci a ainsi permis de comparer les performances de chaque logiciel avec différents paramètres. Les résultats sortis ont été comparés à ceux de la phase précédente, afin de pouvoir déterminer quels paramètres avaient une influence positive sur les premiers résultats.

Lors de la troisième et dernière phase de l'évaluation des différents logiciels, un pré-traitement a été effectué sur les images de notre jeu de test, afin de les retravailler et de voir si cela permettait d'optimiser les résultats.

Toute cette phase d'évaluation a permis de comparer les modèles d'apprentissage et la qualité des outputs de chacun des logiciels, afin de sélectionner le plus performant. Ces océrisations et calculs de résultats pouvant être longs, un script de threading permettant de lancer l'océrisation de plusieurs images en parallèle a été créé, afin de gagner du temps.

3.2.3. Métriques

Pour mesurer la performance des logiciels testés, des métriques usuelles dans le domaine des Sciences de l'Information ont été utilisées, à savoir la précision et le rappel (Burgy, Gerson, Schüpbach 2020b). La moyenne harmonique, ou F1, a également été utilisée afin de faire la balance entre les deux.

Comme il s'agit de texte, ces métriques ont été utilisées à la fois à l'échelle des caractères et à l'échelle des mots (Burgy, Gerson, Schüpbach 2020b), ceci afin de pouvoir mieux décider quelles stratégies choisir pour l'entraînement des modèles et la post-correction, en vue d'optimiser les résultats. De manière générale, il est fréquent que les résultats au niveau des mots soient moins bons qu'au niveau des caractères, car il suffit qu'un caractère soit incorrect pour que le mot entier soit considéré comme faux.

La distance de Levenshtein, qui permet de comparer deux chaînes de caractères et de repérer le nombre de caractères ajoutés, supprimés ou substitués, a également été utilisée. L'algorithme qui effectue cette opération donne en sortie une somme des erreurs (arvindpdmn, 2019), ce qui a été utile dans le calcul des métriques précédentes.

Pour obtenir un retour visuel, la librairie « difflib » qui affiche les caractères ajoutés, supprimés et substitués, a été utilisée. Cela permet de vérifier quelles parties des outputs présentent des erreurs.

Figure 1 : comparaison d'une transcription (gauche) avec un output de Tesseract (droite) grâce à la librairie "difflib"

Comparaison d'une transcription (gauche) avec un output de Tesseract (droite) grâce à la librairie "difflib"

3.3. Sélection finale

Par suite de la première phase de tests, les résultats obtenus avec Kraken et Calamari n'étaient pas satisfaisants. Avec OCR4all, les résultats étaient très bons, mais l'outil présentait un défaut problématique, à savoir qu'il tournait en boucle infinie lorsqu'il était confronté à une page blanche. Ce problème pouvant être un obstacle de taille et ralentir considérablement le travail, surtout au moment de l'océrisation de l'ensemble de la collection de Bry, le choix s'est finalement porté sur Tesseract. La totalité des tests des trois autres outils et leurs résultats sont disponibles dans le mémoire de recherche (Burgy, Gerson, Schüpbach 2020a).

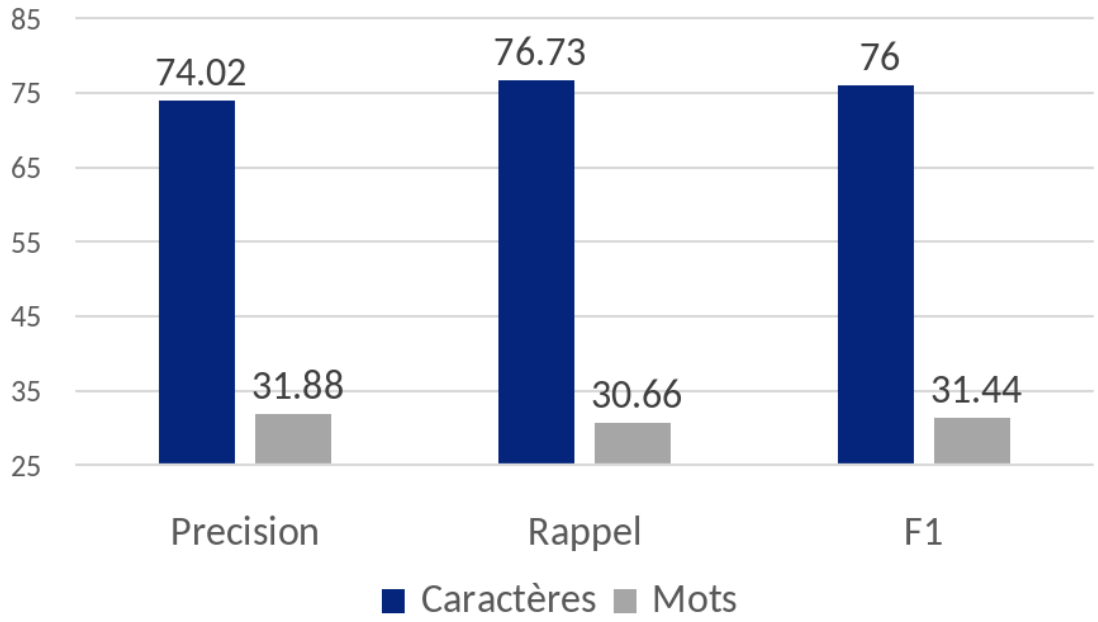
4. Test en trois phases de Tesseract

4.1. Tesseract – phase 1

Dans la première phase de test avec Tesseract, le logiciel a été utilisé avec son modèle standard anglais et ses paramètres par défaut. Les premiers résultats sont déjà encourageants au niveau des caractères avec une F1 de 76%.

Les résultats au niveau des mots sont en revanche bien plus faibles. La F1 est de 31,4%, pour des raisons mentionnées plus haut.

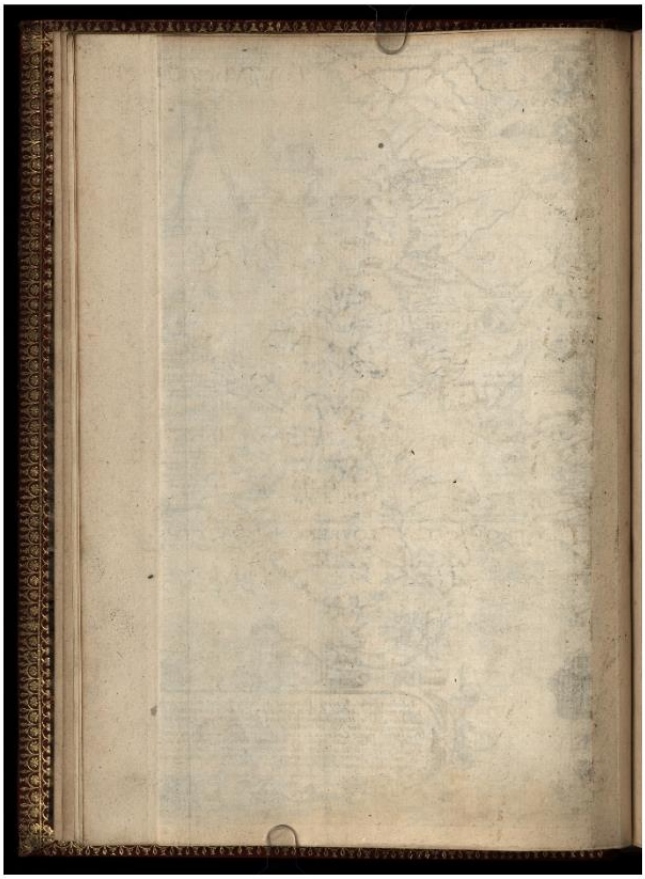
Figure 2 : Tesseract – phase 1 – modèle anglais



Quelques problèmes ont cependant été repérés. Dans le cas de pages vides, par exemple, le script Python de calcul automatique des métriques ne parvient pas à comparer les deux fichiers – la transcription et l'output – et produit des résultats incohérents. Afin de calculer les différentes métriques, il faut en effet connaître le nombre de caractères corrects ainsi que le nombre de caractères du ground truth (la transcription).

Dans le cas d'une transcription ne comportant pas de texte, le nombre de caractères corrects et le nombre de caractères de la transcription seront toujours égaux à 0 et fourniront toujours un résultat de 0 (ou une erreur de division par 0). Il a donc été décidé de définir automatiquement les valeurs de la précision, du rappel et de la F1 dans ce cas précis.

Figure 3 : numérisation d'une page blanche de la collection de Bry et output de Tesseract



t 1 t 1
2
3
4
5 So Ds
6
7

Legends

Colors	Links
Added	(f)irst change
Changed	(n)ext change
Deleted	(t)op

Dans le cas ci-dessus, la transcription contient 0 caractères. L'OCR, lui, a trouvé 10 caractères (espaces compris), tous faux. Les calculs des différentes métriques seront alors les suivants :

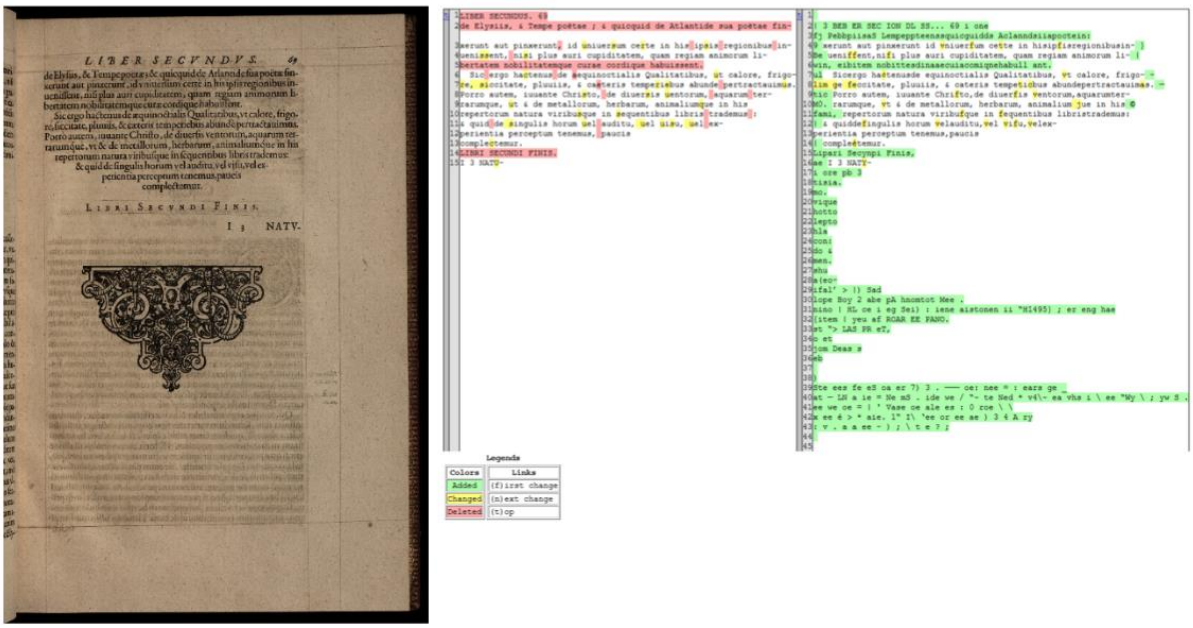
$$P = 0/10 = 0$$

$$R = 0/0 \rightarrow \text{Division par 0 !}$$

Dans tous les cas, les résultats des métriques donneront 0 ou une division par 0. Donc en définissant les résultats à 0, on peut limiter les risques de résultats incalculables sans fausser ces derniers.

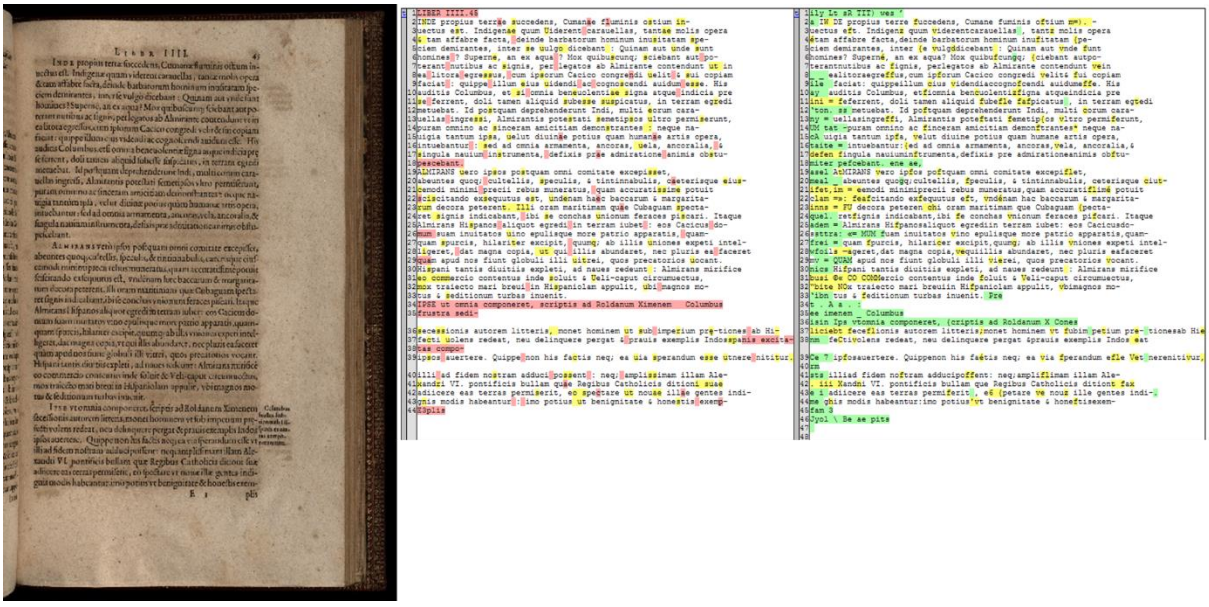
Dans le cas de numérisations comportant des typographies différentes ou des gravures, les résultats tendent à chuter, car le logiciel peine à les traiter et cherche des caractères là où il n'y en a pas. Dans l'exemple ci-dessous, on peut voir que Tesseract a « trouvé » des caractères dans la gravure.

Figure 4 : numérisation d'une page de la collection de Bry comportant une image ; transcription et output de Tesseract



Enfin, dans la plupart des numérisations, une partie du texte de la page adjacente est visible. Tesseract tend à océriser ces caractères également, ce qui influe sur la précision. La figure ci-dessous illustre bien cette problématique, car on voit de nombreux caractères ajoutés – surlignés en vert – au début de la plupart des lignes de l'output.

Figure 5 : numérisation d'une page de la collection de Bry dont la page adjacente est visible ; transcription et output de Tesseract.

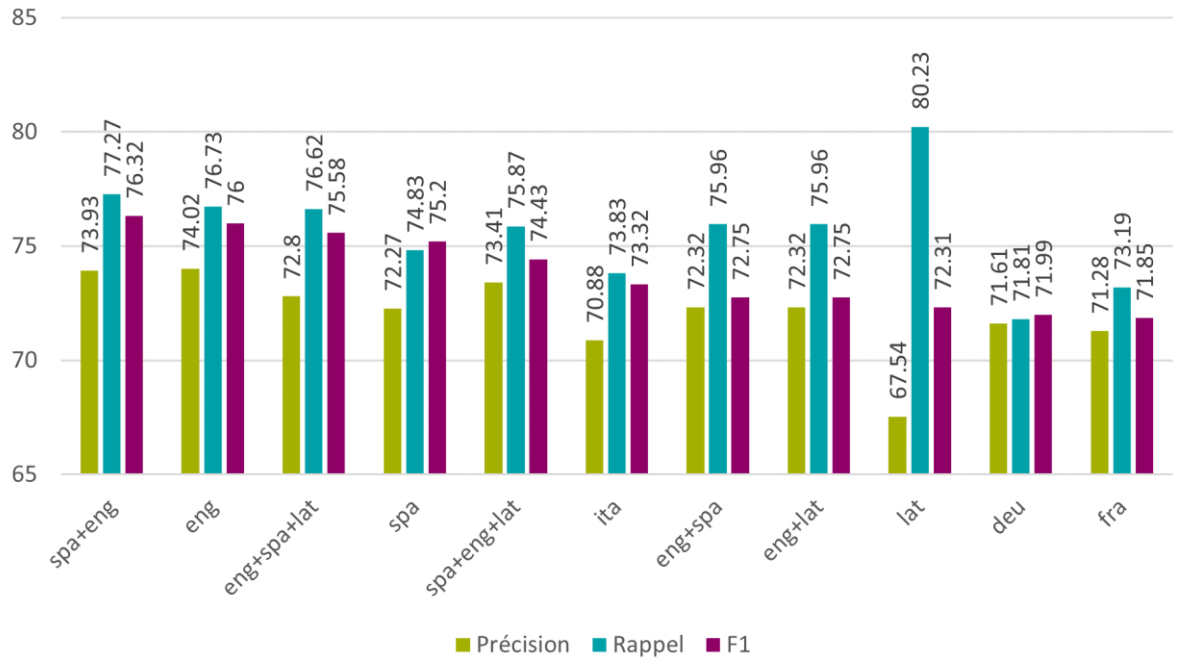


Ces observations ont été très utiles pour les phases suivantes.

4.2. Tesseract – phase 2

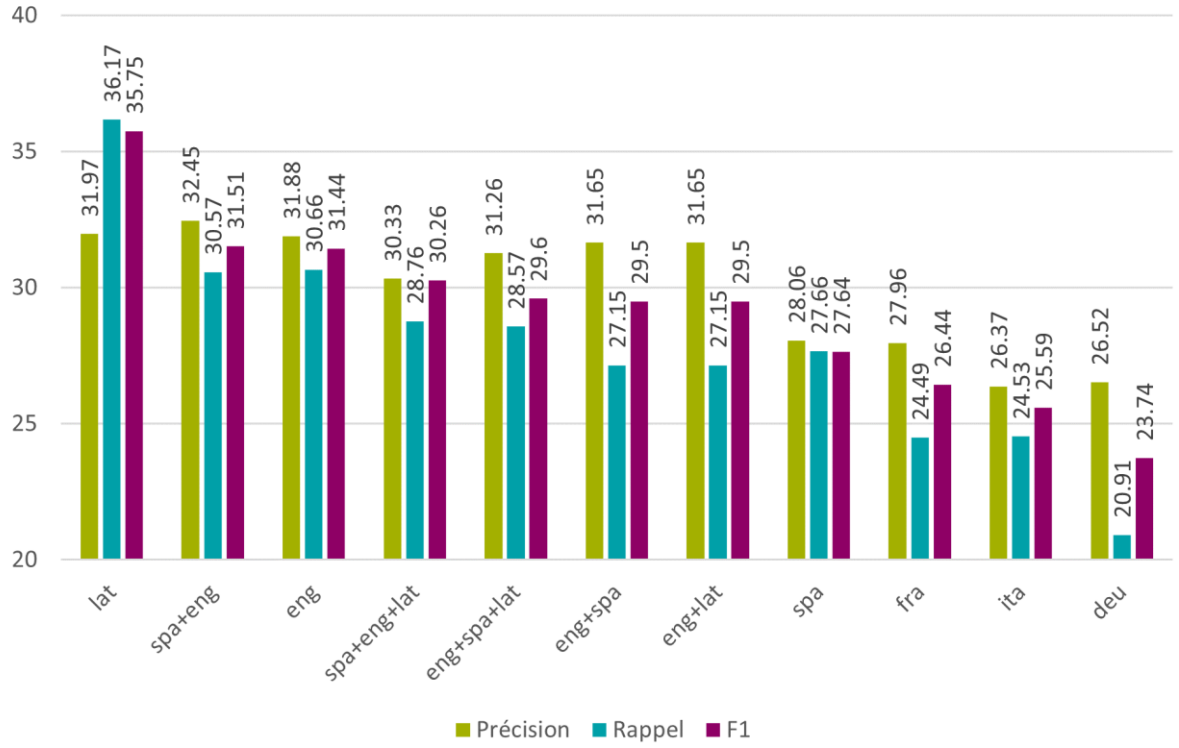
Dans la seconde phase, différents modèles proposés par Tesseract ont été testés, en sélectionnant des langues relativement proches du latin, à savoir les modèles allemand (deu), anglais (eng), espagnol (spa), français (fra), italien (ita), et latin (lat) bien entendu. Plusieurs de ces modèles ont été également combinés entre eux. Les résultats au niveau des caractères montrent que le modèle anglais testé dans la phase 1 se fait légèrement dépasser par la combinaison des modèles espagnol et anglais avec une F1 de 76.32%.

Figure 6 : Tesseract – phase 2 – modèles de langues – caractères



Au niveau des mots, le meilleur modèle est le modèle latin avec une F1 de 35.75%, suivi par la combinaison des modèles espagnol et anglais avec une F1 de 31.51%.

Figure 7 : Tesseract – phase 2 – modèles de langues – mots



Le choix de l'ordre des modèles dans une combinaison est crucial. En effet, la combinaison espagnol/anglais offre de bien meilleurs résultats que la combinaison anglais/espagnol.

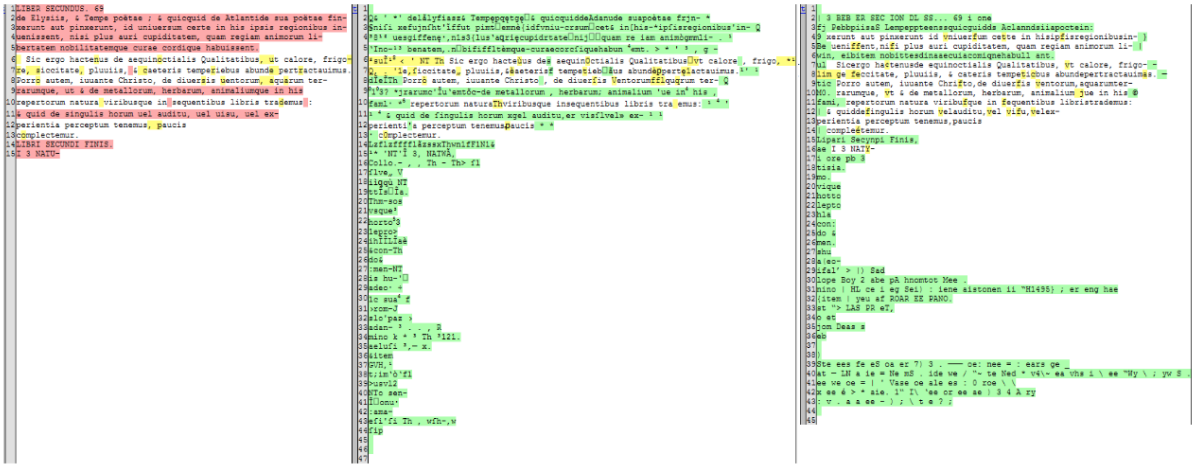
Il est également intéressant de noter que, bien que la F1 de la combinaison espagnol/anglais soit la plus élevée, ce n'est pas forcément le cas pour la précision et le rappel. On peut voir ici que le modèle avec la précision la plus élevée est le modèle anglais tandis que le modèle avec le rappel le plus élevé est le latin.

De fait, il est frappant de voir que le modèle latin, pourtant la langue de cette édition de la collection de Bry, a un taux de rappel si élevé, alors que sa précision est la plus mauvaise de tous les modèles testés. En effet, le modèle latin gère bien moins bien les espaces blancs et les marges ainsi que les images, et ajoute beaucoup de caractères incorrects.

En outre, le modèle latin donne des résultats moins bons lorsqu'il est confronté à des typographies différentes. Cela peut s'expliquer par la manière dont ces différents modèles sont entraînés. En effet, certaines langues vivantes, comme l'anglais, permettent de créer de vastes jeux de données d'entraînement présentant des typographies variées, alors que, pour le latin, la quantité de données à disposition est moindre, et les résultats s'en ressentent (theraysmith, 2017).

Ces deux problématiques sont visibles dans l'exemple ci-dessous.

Figure 8 : comparaison d'une transcription avec un output modèle latin (centre) et modèle anglais (droite)



Dans cette phase, différents paramètres de segmentation de pages (ou psm) qu'implémente Tesseract ont également été testés :

0 = Orientation and script detection (OSD) only.

1 = Automatic page segmentation with OSD.

2 = Automatic page segmentation, but no OSD, or OCR

3 = Fully automatic page segmentation, but no OSD. (Default)

4 = Assume a single column of text of variable sizes.

5 = Assume a single uniform block of vertically aligned text.

6 = Assume a single uniform block of text.

7 = Treat the image as a single text line.

8 = Treat the image as a single word.

9 = Treat the image as a single word in a circle.

10 = Treat the image as a single character.

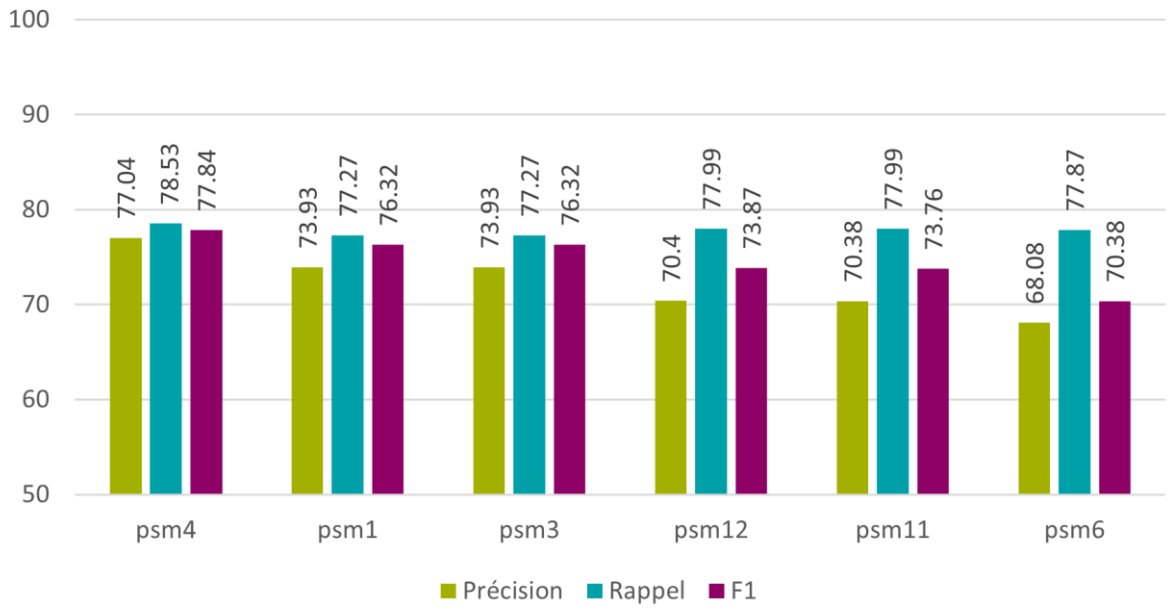
11 = Sparse text. Find as much text as possible in no particular order.

12 = Sparse text with OSD.

13 = Raw line. Treat the image as a single text line

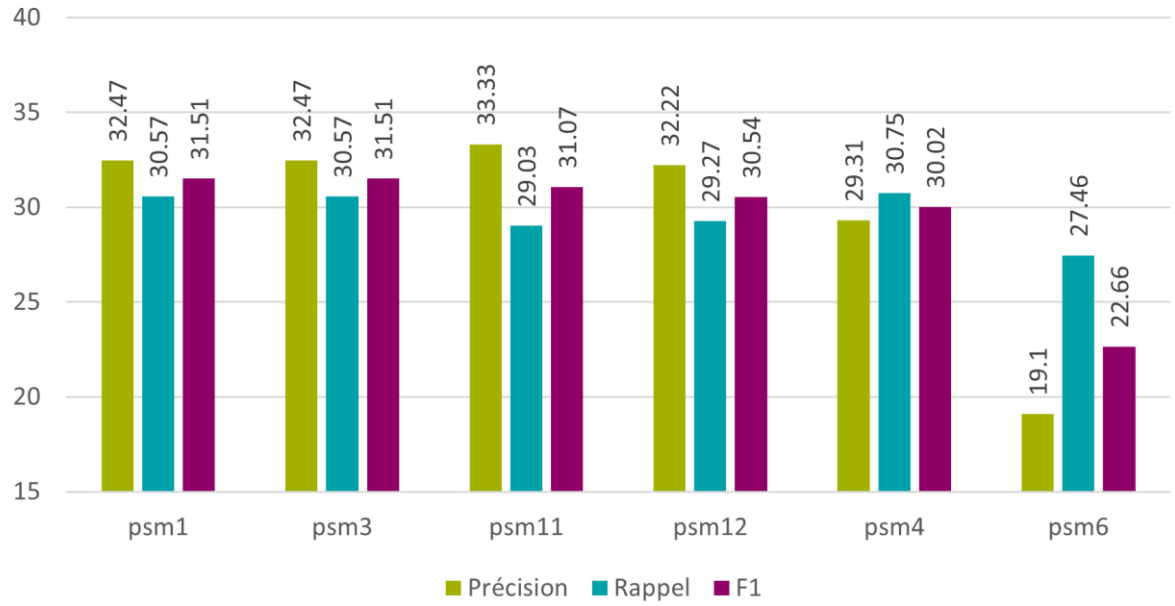
Le graphique ci-dessous présente les résultats obtenus au niveau des caractères avec les six paramètres de segmentation les plus performants et le modèle anglais. On voit bien que la meilleure segmentation pour notre problème est la numéro 4 avec une F1 de 77,84%. La 1 et la 3 viennent ensuite, avec toutes les deux une F1 de 76,32%.

Figure 9 : Tesseract – phase 2 – psm – caractères



Au niveau des mots, les meilleures segmentations sont la 1 et la 3, avec une F1 de 31,51%.

Figure 10 : Tesseract – phase 2 – psm – mots



Ces résultats ont permis d'aiguiller les décisions de la dernière phase.

4.3. Tesseract – phase 3

Pour la dernière phase, les meilleurs paramètres identifiés précédemment (modèle espagnol/anglais et segmentation 1 ou 4) ont été utilisés afin de vérifier si un pré-traitement sur les images permet d'améliorer les résultats.

Trois types de pré-traitements différents ont été testés :

- Modification de l'image en nuance de gris (threshold)
- Modification de la taille de l'image (resample)
- Modification de la taille et modification en nuance de gris (full)

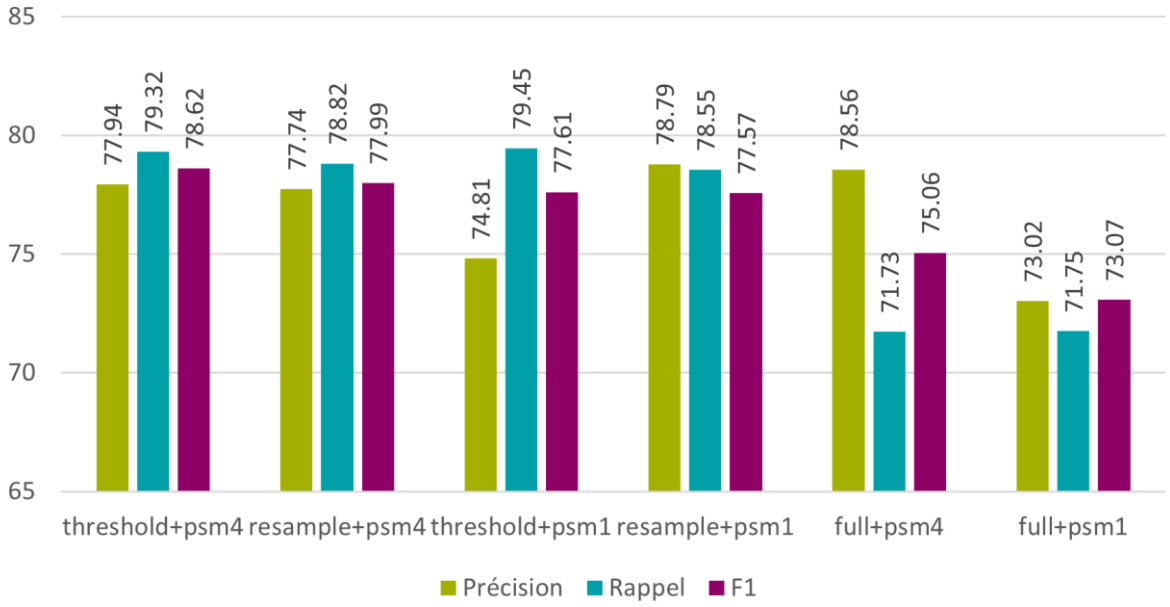
La méthode threshold consiste à modifier tous les pixels dépassant un certain seuil de couleur. Tout pixel plus clair que le seuil donné, 20% dans le cas présent, sera automatiquement transformé en pixel blanc. Tout autre pixel sera modifié en pixel noir.

La méthode resample consiste à rogner l'image en fonction de sa position dans le livre. Si la page est un recto (page de droite dans un livre), 400 pixels sont rognés de la gauche de l'image et 200 de la droite, et inversement si la page est un verso. Cela permet de supprimer assez facilement les pages adjacentes visibles. Néanmoins, comme les valeurs sont fixes, il est possible que le script rogne trop et qu'une partie du texte soit perdu.

La méthode full utilise les deux méthodes ci-dessus. Pour chacune de ces trois méthodes, une bordure blanche a en outre été ajoutée sur l'image, comme la documentation de Tesseract le conseille (Cimon 2019). Toutes les modifications ont été faites à l'aide de la librairie « ImageMagick », disponible sur imagemagick.org.

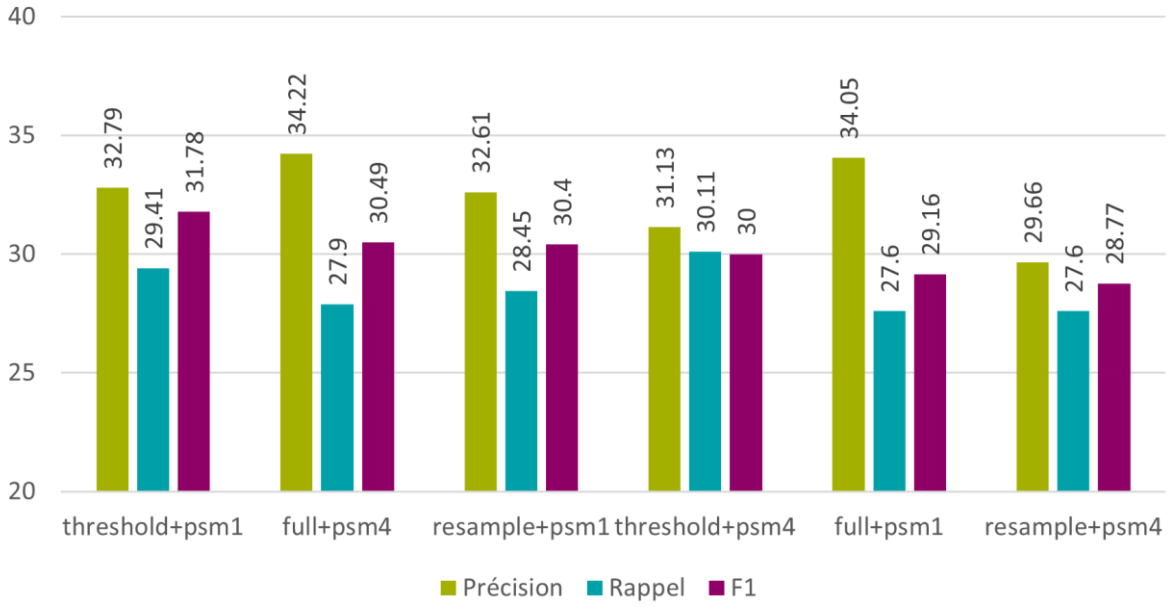
Les résultats obtenus montrent que ces méthodes peuvent améliorer la qualité des outputs. En effet la précision au niveau des caractères augmente de 0.9%, le rappel de 0.79% et la F1 de 0.78%.

Figure 11 : Tesseract – phase 3 – modèle « spa+eng » – pré-traitement des images – caractères



Au niveau des mots, une légère amélioration des résultats est aussi visible. On observe une augmentation de 0.28% en précision, une perte de 1.16% en rappel et une augmentation de 0.27% en F1.

Figure 12 : Tesseract – phase 3 – modèle « spa+eng » – pré-traitement des images – mots



Ces trois méthodes de pré-traitement sont encore naïves et méritent d'être améliorées, mais il est d'ores et déjà possible d'affirmer que le pré-traitement des images augmente effectivement les résultats.

En définitive, la combinaison des modèles espagnol et anglais avec une segmentation de type 1 ou 4 semble être la méthode la plus adaptée au problème. Un pré-traitement sur les images augmente également légèrement les résultats. À ce stade, une F1 de 78.62% au niveau des caractères et de 31.78% au niveau des mots peut être obtenue. Il faut également noter que Tesseract est un logiciel stable, robuste et facile à utiliser qui n'a posé aucun problème de prise en main, ce qui est un avantage non négligeable.

5. Optimisation de l'océrisation

La suite de ce projet consistait à tester différentes méthodes pour améliorer les résultats de l'OCR sélectionné, Tesseract, en utilisant des techniques de pré-traitement des inputs, de post-correction des outputs, ou en utilisant des fonctionnalités du logiciel lui-même.

5.1. Méthodes

Plusieurs méthodes ont été testées :

- Pré-traitement intelligent des images
- Correction brute des outputs
    - Transformation systématique des caractères
    - Suppression de caractères indésirables
    - Suppression des caractères non alphanumériques
- Utilisation d'un corpus latin pour la création d'un dictionnaire
    - Remplacement des mots selon une distance de modification définie
- Utilisation de l'outil de post-correction PoCoTo
- Création d'un modèle Tesseract personnalisé

5.2. Pré-traitement intelligent des images

Comme vu dans Tesseract – phase 3, un pré-traitement intelligent des images peut améliorer les résultats des océrisations.

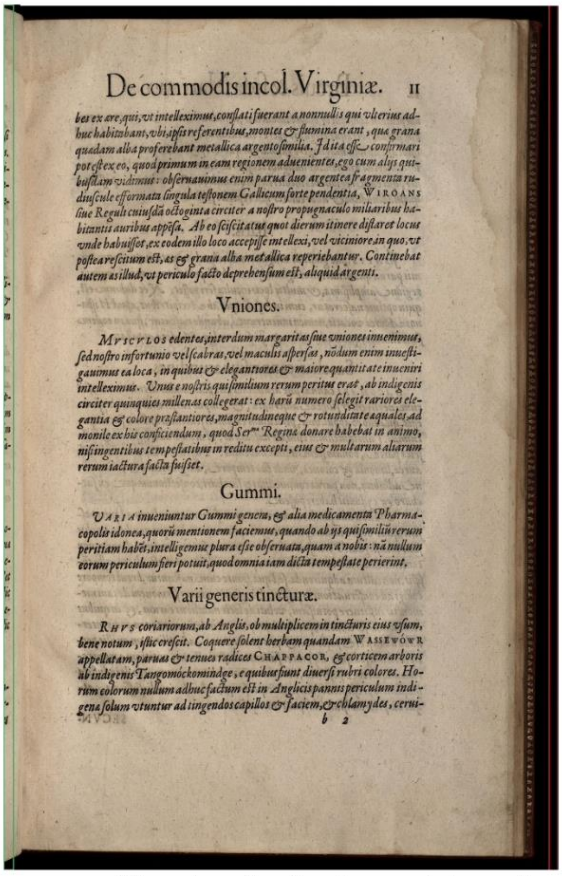
Pour ce faire, un algorithme de recherche permettant de savoir si la page actuellement traitée est un recto ou un verso a été créé dans le cadre du projet. En fonction de cela, un autre algorithme, développé dans ce même cadre, va trouver la position idéale pour rogner l'image et l'effectuer à l'aide de « ImageMagick », mentionné dans Tesseract – phase 3.

L'algorithme de sélection de l'orientation de la page calcule la somme des niveaux de gris de chaque pixel de la colonne de pixels la plus à droite et la plus à gauche de l'image. La somme la plus petite permet d'indiquer si l'image est un recto ou un verso.

A priori, cet algorithme fonctionne uniquement avec le jeu de données du projet, car le Bodmer Lab a pour habitude de cadrer ses numérisations en gardant une partie de la page adjacente ainsi qu'un fond noir sur le bord opposé. Ainsi, en déterminant quel côté est “le plus foncé”, il est possible de savoir si la page est un recto ou un verso.

Sur l’image suivante, la partie droite (en rouge) comporte uniquement des pixels noirs. Un pixel noir ayant une valeur d’environ 0 (dépendant de la luminosité de la pièce au moment de la numérisation), la somme sera petite. Inversement, la partie de gauche (en vert), aura une somme bien plus élevée. On peut donc dire que cette image est un recto car la somme la plus faible est celle de droite.

Figure 13 : détection des zones à rogner selon notre algorithme dans une numérisation extraite de la collection de Bry



De commodis incol. Virginiæ. 11

bes ex ære, qui, vt intelleximus, conflati fuerant a nonnullis qui vlterius adhuc habitabant, vbi ipsa referentibus, montes & flumina erant, qua grana quadam alba proferebant metallica argento similia. Jdita esse confirmari potest ex eo, quod primum in eam regionem aduenientes, ego cum aliis quibusdam vidimus: obseruauimus enim parua duo argentea fragmenta rudiuscule efformata lingula tessonem Gallicum forte pendentia, WIROANS siue Regulus cuiusdã octo vel in a circuiter a nostro propugnaculo miliaribus habitantis auribus appensa. Ab eo scisciitatus quot dierum itinere distaret locus vnde habuisset, ex eodem illo loco accepisse intellexi, vel vicinore in quo, vt postea rescitum est, as & grana alba metallica reperiebantur. Continebat autem as illud, vt periculo facto deprehensum est, aliquid argenti.

Vniones.

MYTYLOS edentes, interdum margaritas siue vniones inueniunt, sed nostro infortunio vel scabras, vel maculis aspersas, nõdum enim inuestigauimus ea loca, in quibus & elegantiores & maiore quantitate inueniri intelleximus. Vnus e nostris quisimilium rerum peritus erat, ab indigenis circiter quinquies millenas colligerat: ex harũ numero selegit rariores elegantia & colore præstantiores, magnitudine quæ & rotunditate æquales, ad monile ex his conficiendum, quod Ser^mæ Reginæ donare habebat in animo, nisi ingentibus tempestatibus in reditu exceptí, eius & multarum aliarum rerum iactura facta fuisset.

Gummi.

VARIA inueniuntur Gummi genera, & alia medicamenta Pharmacopolis idonea, quorũ mentionem faciemus, quando ab ijs quisimiliũ rerum peritiam habet, intelliget nos plura esse obseruata, quam a nobis: nã nullum eorum periculum fieri potuit, quod omnia iam dicta tempestate perierint.

Varii generis tincturæ.

RUFUS coriariorum, ab Anglis, ob multiplicem in tincturis eius vsum, bene notum, sic crescit. Coquere solent herbam quandam WASSEWOWR appellatam, paruas & tenues radices CHAPACOR, & corticem arboris ab indigenis Tangomõckomindge, e quibus sunt diuersi rubri colores. Horum colorum nullum adhuc factum est in Anglicis pannis periculum indigena solum vtuntur ad ingendos capillos & faciem, & eblamydes, cerui-

b 2

Algorithm 1 Page recto / verso

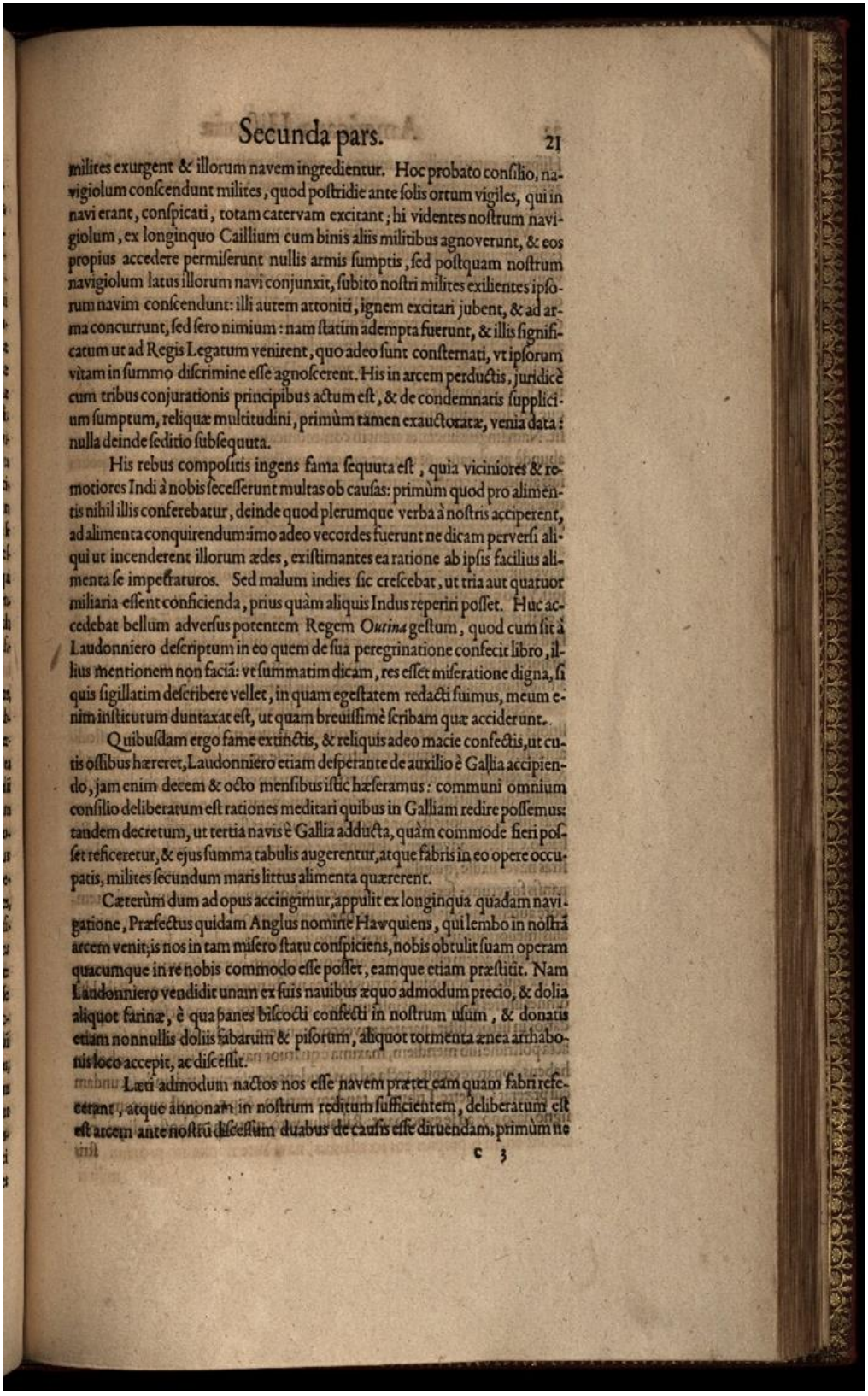
Input: Chemin de l'image
Ouvrir l'image
Convertir l'image en pixels noirs et blancs
Initialiser right=0, left=0, w=image width et h=image height
for $i = 0$ **to** $h - 1$ **do**
  Initialiser r comme la valeur rouge du pixel$[w - 1, i]$
  Initialiser g comme la valeur verte du pixel$[w - 1, i]$
  Initialiser b comme la valeur bleu du pixel$[w - 1, i]$
  $right+ = r + g + b$

  Définir r comme la valeur rouge du pixel$[0, i]$
  Définir g comme la valeur verte du pixel$[0, i]$
  Définir b comme la valeur bleu du pixel$[0, i]$
  $left+ = r + g + b$
end for
if $right < left$ **then**
  Output: L'image est un recto
else
  Output: L'image est un verso
end if

Lorsque l’on sait si l’image est un recto ou un verso, il faut déterminer quel pourcentage de l’image doit être rogné pour enlever le surplus de la page adjacente. Pour ce faire, un algorithme dont c’est l’objectif a été créé. Si la page est recto, l’algorithme va calculer la somme des couleurs de chaque colonne entre la gauche et le centre de l’image. Ces sommes sont stockées dans un tableau. Par la suite, l’algorithme va récupérer l’indice de la valeur la plus faible dans ce tableau. Cet indice signale l’endroit où l’image doit être rognée.

Le processus est presque le même si l’on travaille sur un verso. La seule différence réside dans le calcul des sommes des colonnes de pixels. L’algorithme ne partira pas de la gauche vers le centre mais de la droite vers le centre.

Figure 14 : image non rognée et image rognée avec bordure blanche



milites exurgent & illorum navem ingredientur. Hoc probato consilio, navigiolum conscendunt milites, quod postridie ante solis ortum vigiles, qui in navi erant, conspicati, totam catervam excitant; hi videntes nostrum navigiolum, ex longinquo Caillium cum binis aliis militibus agnoverunt, & eos propius accedere permiserunt nullis armis sumptis, sed postquam nostrum navigiolum latus illorum navi conjunxit, subito nostri milites exilientes ipsorum navim conscendunt: illi autem attoniti, ignem excitari jubent, & ad arma concurrunt, sed sero nimium: nam statim adempta fuerunt, & illis significatum ut ad Regis Legatum venirent, quo adeo sunt consternati, vt ipsorum vitam in summo discrimine esse agnoscerent. His in arcem perductis, juridicè cum tribus conjurationis principibus actum est, & de condemnatis supplicium sumptum, reliquæ multitudini, primùm tamen exauctoratæ, venia data: nulla deinde seditio subsequuta.

His rebus compositis ingens fama sequuta est, quia viciniores & remotiores Indi à nobis secesserunt multas ob causas: primùm quod pro alimentis nihil illis conferebatur, deinde quod plerumque verba à nostris acciperent, ad alimenta conquirendum: imo adeo vecordes fuerunt ne dicam perversi aliqui ut incenderent illorum ædes, existimantes ea ratione ab ipsis facilius alimenta se impetraturos. Sed malum indies sic crescebat, ut tria aut quatuor miliaria essent conficienda, prius quàm aliquis Indus reperiri posset. Huc accedebat bellum adversus potentem Regem *Outina* gestum, quod cum sit à Laudonniero descriptum in eo quem de sua peregrinatione confecit libro, illius mentionem non facio: vt summatim dicam, res esset miseratione digna, si quis sigillatim describere vellet, in quam egestatem redacti fuimus, meum enim institutum duntaxat est, ut quam breuissimè scribam quæ acciderunt.

Quibusdam ergo fame extinctis, & reliquis adeo macie confectis, ut cutis ossibus hæreret, Laudonniero etiam desperante de auxilio è Gallia accipiendo, jam enim decem & octo mensibus istic hæseramus: communi omnium consilio deliberatum est rationes meditari quibus in Galliam redire possemus: tandem decretum, ut tertia navis è Gallia adducta, quàm commode fieri posset reficeretur, & ejus summa tabulis augerentur, atque fabris in eo opere occupatis, milites secundum maris littus alimenta quærerent.

Cæterùm dum ad opus accingimur, appulit ex longinqua quadam navigatione, Præfectus quidam Anglus nomine Hawquiens, qui lembo in nostrã arcem venit; is nos in tam misero statu conspiciens, nobis obtulit suam operam quacumque in re nobis commodo esse posset, eamque etiam præstitit. Nam Laudonniero vendidit unam ex suis nauibus æquo admodum precio, & dolia aliquot farinæ, è qua panes biscocti confecti in nostrum usum, & donatis etiam nonnullis doliis fabarum & pisorum, aliquot tormenta ænea arrhabonis loco accepit, ac discessit.

Læti admodum nactos nos esse navem præter eam quam fabri refecerant, atque annonam in nostrum reditum sufficientem, deliberatum est est arcem ante nostrũ discessum duabus de causis esse diruendam, primùm ne

Ces deux algorithmes ont également été utilisés avec différents paramètres pour en créer plusieurs versions. En effet, dans certains cas, le rognage était trop important et une partie du texte était alors perdue. Pour éviter cela, il a fallu faire des tests en divisant la valeur du rognage (crop) par son quart, son tiers et sa moitié.

Le script permettant l'appel à ImageMagick est le suivant :

Dans le cas d'une image recto :

convert *imagePath* -gravity West -chop *chopx*0 -trim -trim -resample -bordercolor white -border 20x20 *savePath*

Dans le cas d'une image verso :

convert *imagePath* -gravity East -chop *chopx0* -trim -trim -resample -bordercolor white -border 20x20 *savePath*

Il suffit de remplacer les valeurs italiques soulignées par les valeurs récupérées dans l'algorithme.

Grâce à ces trois algorithmes, il est possible de supprimer la page adjacente sur l'image actuellement traitée. Avec la meilleure version de l'algorithme, la F1 passe de 78.62% à 79.23% au niveau des caractères et de 31.78% à 32.14% au niveau des mots.

Figure 16 : Tesseract – pré-traitement intelligent – caractères

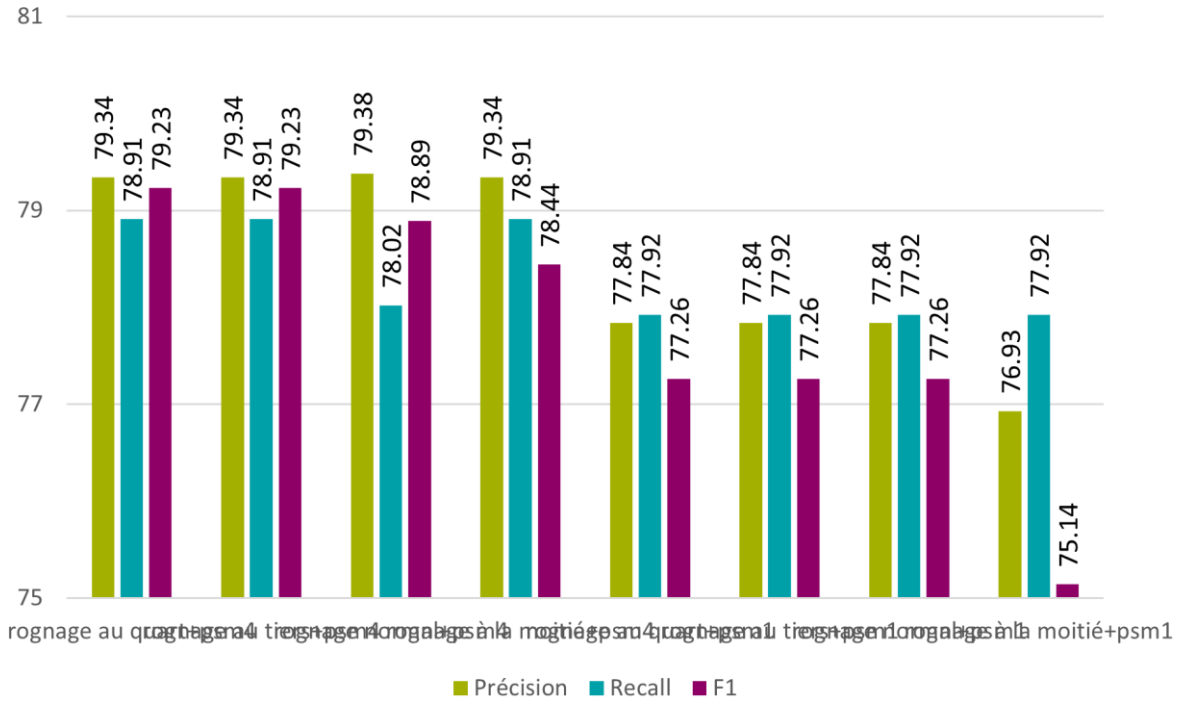
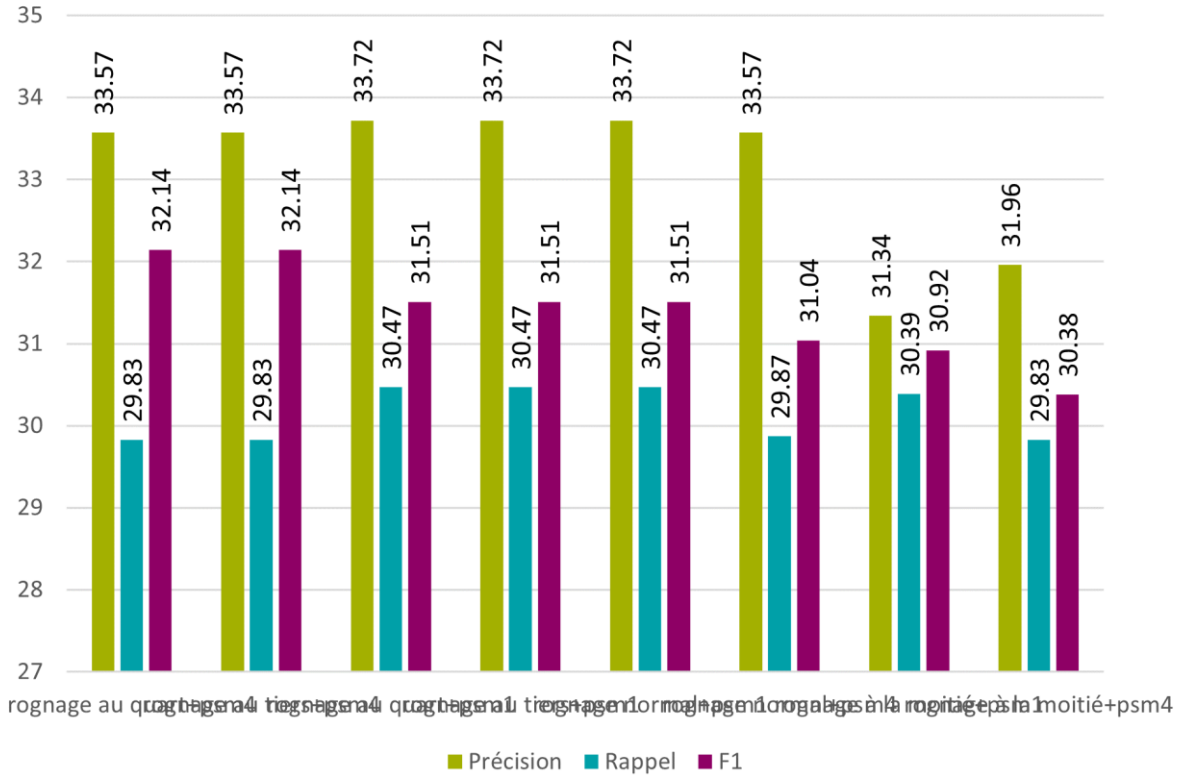


Figure 17 : Tesseract – pré-traitement intelligent – mots



5.3. Correction brute des outputs

Après avoir appliqué les algorithmes de pré-traitement et océrisé l'ensemble des images rognées dans Tesseract, il est possible d'améliorer encore les résultats en effectuant une post-correction sur les outputs.

Pour ce faire, plusieurs algorithmes ont été testés :

1. Suppression de tous les caractères différents d'un caractère d'espacement (vertical ou horizontal), d'une lettre (minuscule ou majuscule) ou d'un chiffre
- Par la suite, remplacement de tous les v par des u et les j par des i, du fait de l'équivalence de ces lettres en latin

1. Uniquement de tous les v par des u et les j par des i
2. Suppression de tous les caractères différents d'une lettre (minuscule ou majuscule), d'un chiffre, d'un point, d'une virgule, d'un espace ou d'un retour à la ligne
- Par la suite, remplacement de tous les v par des u et les j par des i

1. Suppression de tous les caractères différents d'une lettre (minuscule ou majuscule), d'un chiffre, d'un double point, d'un tiret, d'un espace ou d'un retour à la ligne
- Par la suite, remplacement de tous les v par des u et les j par des i

1. Suppression de tous les caractères différents d'une lettre (minuscule ou majuscule), d'un double point, d'un tiret, d'un espace ou d'un retour à la ligne
- Par la suite, remplacement de tous les v par des u et les j par des i

Les résultats montrent qu'un simple remplacement des lettres v et j améliore les résultats. En effet, la F1 passe de 79.23% à de 80.06% au niveau des caractères, et de 32.14% à 34.58% au niveau des mots.

Figure 18 : Tesseract – correction brute – caractères

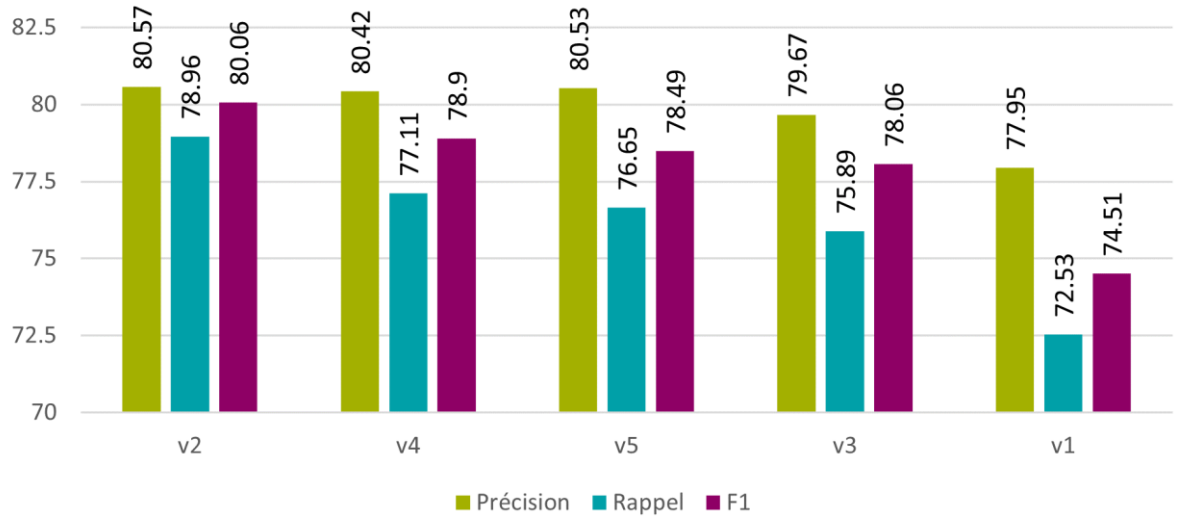
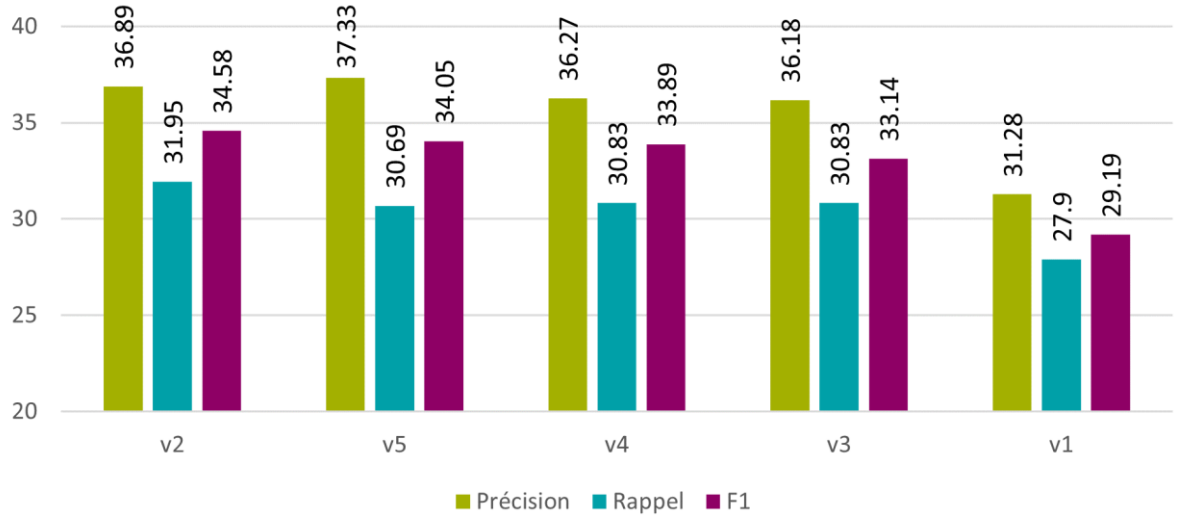


Figure 19 : Tesseract – correction brute – mots



5.4. Utilisation d'un corpus latin pour la création d'un dictionnaire

Une autre méthode de post-correction envisageable était la modification directe des mots sur la base d'un corpus de textes latins. La librairie « symspellpy », disponible sur pypi.org/project/symspellpy, a été utilisée afin de créer le dictionnaire et de calculer la distance d'édition des mots. Pour le corpus, celui de Latinocr.org, mentionné dans l'état de l'art et qui se trouve sur ryanfb.github.io/latinocr/resources.html, a été utilisé.

L'algorithme suivant a ensuite été créé :

Algorithm 4 Remplacement par dictionnaire

Input: Text a corriger
Instancier Symspellpy
Télécharger ou créer le dictionnaire latin en se basant sur un corpus de textes latins
Initaliser *lines*=tableau vide
Initialiser *text_corrected*=""
Couper notre text à chaque retour de ligne et mettre chaque valeur dans le tableau *lines*
for $i = 0$ **to** longueur de *lines* - 1 **do**
  Initialiser *words*=tableau vide
  Couper *lines*[i] à chaque espace et mettre chaque valeur dans le tableau *words*
  while $j <$ longeur de *words* - 2 **do**
    Initialiser $s1$ comme étant la suggestion la plus proche de *words*[j] en utilisant symspellpy
    Initialiser $s2$ comme étant la suggestion la plus proche de *words*[j] + *words*[$j + 1$] en utilisant symspellpy
    if distance de $s1 <$ distance de $s2$ **then**
      Définir *text_corrected*=$s1$ + " "
      Définir $j = j + 1$
    else
      Définir *text_corrected*=$s2$ + " "
      Définir $j = j + 2$
    end if
  end while
Définir *text_corrected*=*text_corrected* + "\n"
end for
Output: Le texte corrigé "*text_corrected*"

Cet algorithme utilise la librairie « symspellpy » pour comparer chaque mot avec ceux du dictionnaire. Pour ce faire, il est nécessaire de définir une distance de recherche maximale (2, dans notre cas). Cette distance limite les résultats aux mots qui ont une distance d'édition de maximum deux caractères.

Tesseract trouve parfois des espaces au milieu des mots. Afin de pouvoir les corriger automatiquement avec l'algorithme, il a fallu comparer chaque mot et celui qui le suit avec le dictionnaire. De cette manière, si un mot a été coupé en deux, il est possible de le traiter.

Plusieurs versions de cet algorithme ont été testées :

2. Distance de suggestion du mot < distance de suggestion du mot et de celui qui le suit
3. Distance de suggestion du mot <= distance de suggestion du mot et de celui qui le suit
4. Distance de suggestion du mot et de celui qui le suit < distance de suggestion du mot
5. Distance de suggestion du mot et de celui qui le suit <= distance de suggestion du mot

Aucune de ces versions n'a pu améliorer les résultats. Au contraire, ils baissent d'environ 5%. Cela est dû au fait que chaque mot va essayer d'être corrigé par rapport à un mot du corpus – même s'il est correct, pour autant que la distance d'édition soit inférieure ou égale à 2. De ce fait, la précision, le rappel et la F1, que ce soit au niveau des caractères ou des mots, baissent.

Figure 20 : Tesseract – corrections par dictionnaire – caractères

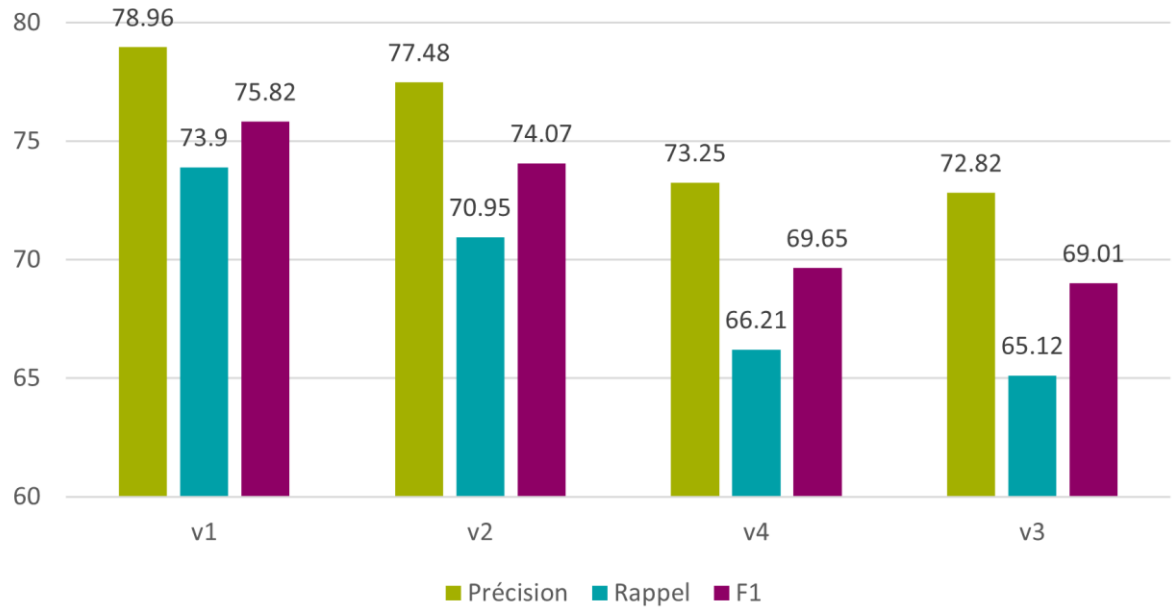
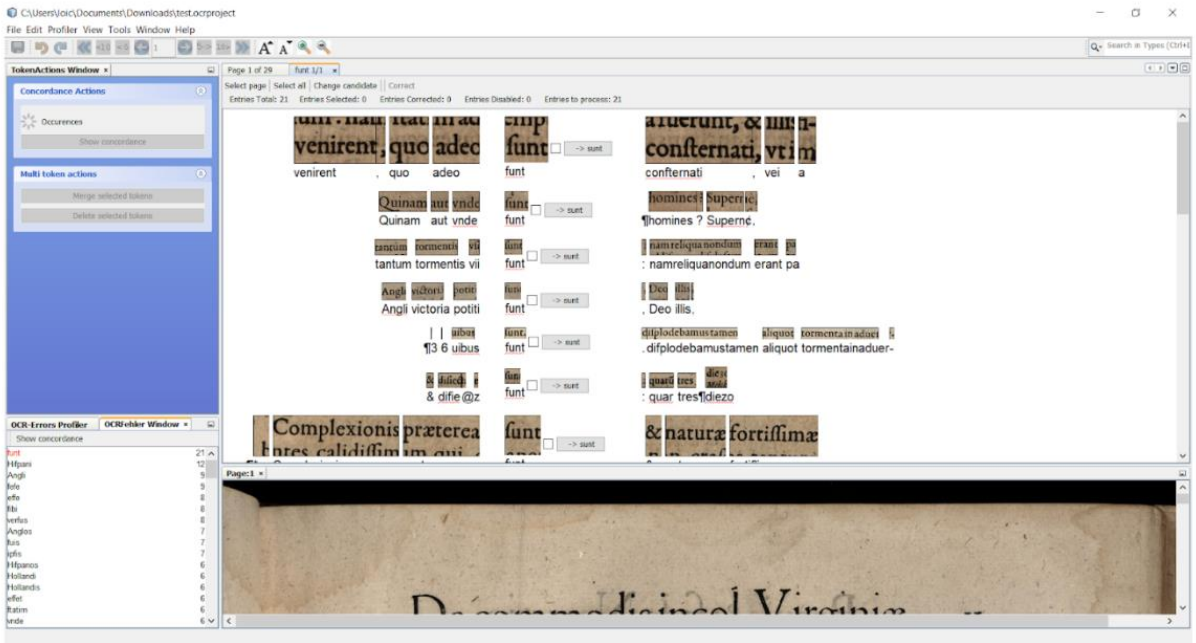


Figure 21 : Tesseract – corrections par dictionnaire – mots



Ce problème peut s'expliquer entre autres par la complexité du latin, qui est une langue à cas. Dans la plupart des dictionnaires latins, un nom va être présenté sous la forme consul, -is, m., ce qui veut dire que c'est un mot masculin de la 3ème déclinaison, qui prend donc -is comme terminaison au génitif. Ce mot pourrait cependant apparaître sous des formes comme consules ou consulibus, mais le dictionnaire ne contient pas ces formes, évidentes pour un latiniste, mais non pour un ordinateur.

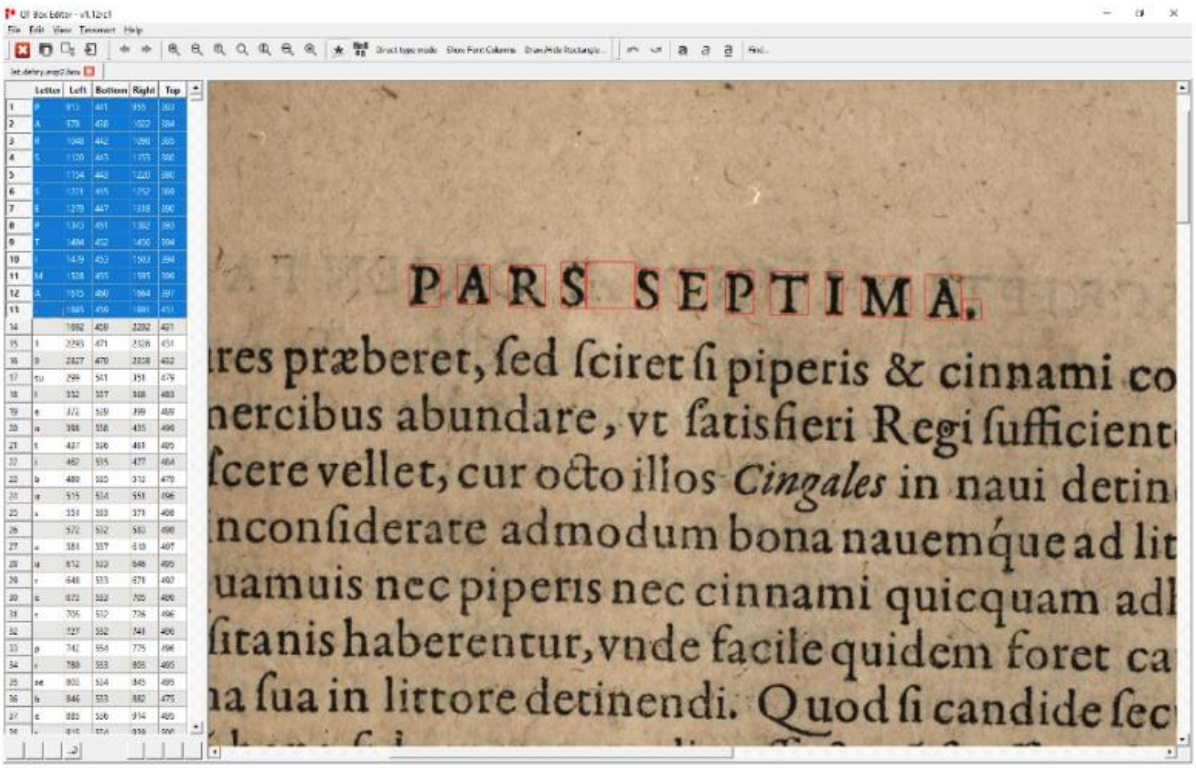
Il aurait cependant été intéressant de pousser plus loin l'expérience des dictionnaires, méthode de post-correction reconnue dans le domaine de l'océrisation, mais, pour des raisons de faisabilité, il a fallu s'en tenir là.

5.5. Utilisation de l'outil de post-correction PoCoTo

Une autre méthode de post-correction testée consistait à utiliser le logiciel PoCoTo. Il s'agit d'un logiciel de post-correction développé dans le cadre du projet IMPACT et permettant de corriger les erreurs des logiciels OCR (Vobl et al. 2014). Les avantages et limites de ce logiciel ont rapidement pu être repérées.

PoCoTo prend en input les images a océriser ainsi que leur océrisation au format HOCR. Ce format enregistre l'océrisation de chaque caractère (comme avec le format texte) mais également la position de la portion d'image qui lui a fait découvrir ce caractère. Il est donc possible par la suite, grâce à PoCoTo, de vérifier manuellement si l'output correspond au ground truth et de la corriger au besoin. Ceci est cependant long, car l'on corrige chaque mot un à un.

Figure 22 : capture d'écran du logiciel PoCoTo en cours d'utilisation



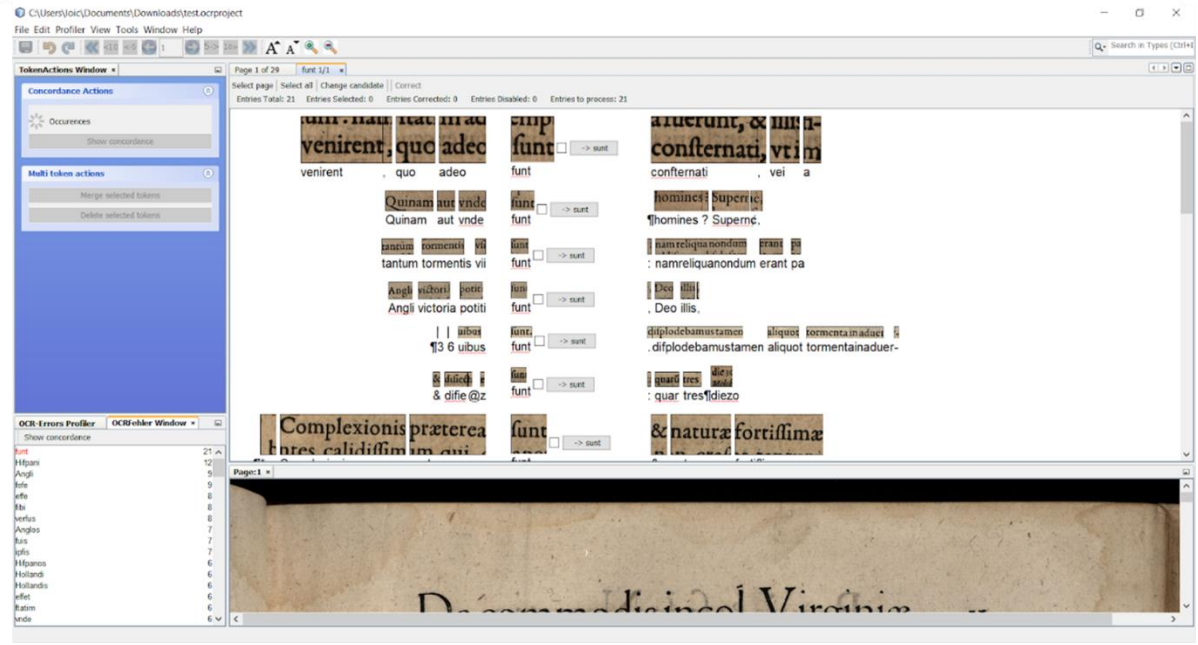
Une seconde option de PoCoTo permet de télécharger des profilers. Actuellement, il est possible de télécharger des profilers en latin, en grec ou en allemand pré-entraînés. Il est également possible de créer son propre profiler.

Ces derniers stockent les erreurs courantes des OCR pour un langage en particulier, afin d'effectuer une correction « semi-automatique » des outputs. Le profiler va détecter si un mot souvent reconnu

incorrectement par les OCR se trouve dans l'un des outputs et propose une correction. Néanmoins, tout se fait depuis une interface graphique et une intervention humaine est alors obligatoire.

Figure 23 : capture d'écran du logiciel PoCoTo en cours d'utilisation avec le système de profiler latin

Figure 23 : capture d'écran du logiciel PoCoTo en cours d'utilisation avec le système de profiler latin



Cette obligation d'avoir une intervention humaine est chronophage et, au vu du temps disponible pour ce projet, il n'a pas été possible d'intégrer cet outil dans la post-correction.

5.6. Création d'un modèle Tesseract personnalisé

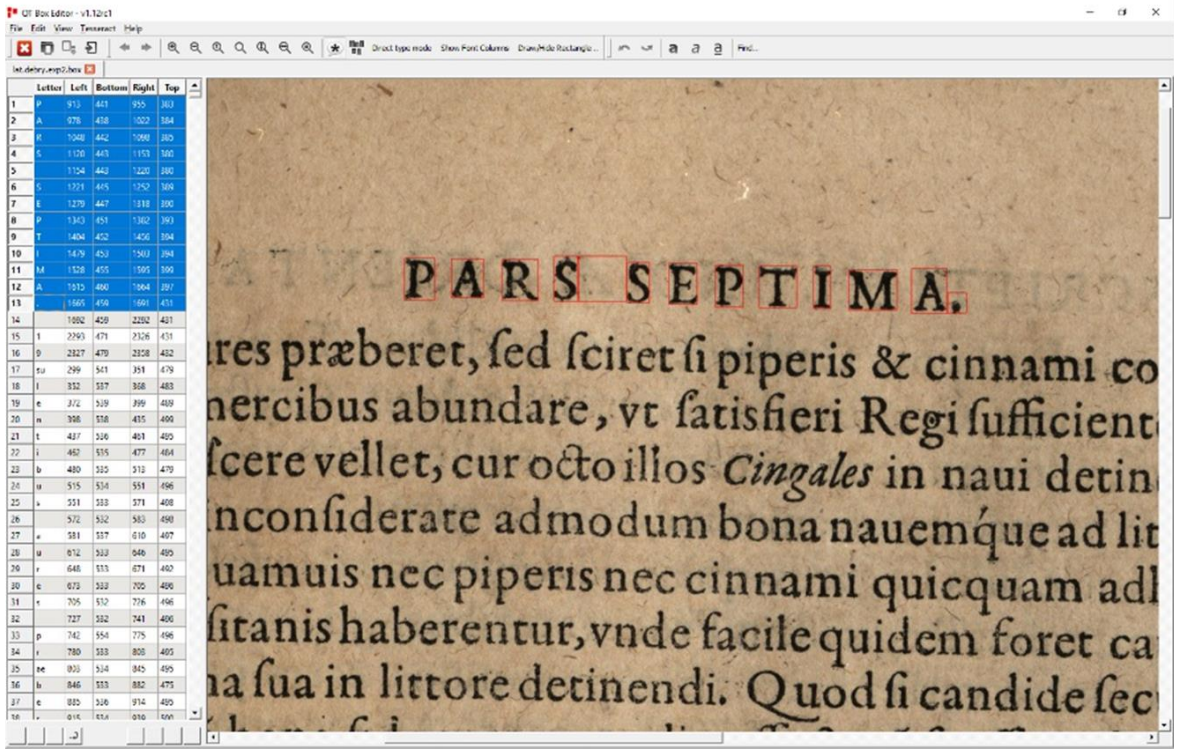
La dernière méthode d'optimisation des résultats testée est la création d'un modèle Tesseract personnalisé. Cette fonctionnalité est rendue possible par l'outil open source multi-plateforme, QT Box Editor, disponible sur github.com/zdenop/qt-box-editor. Cet outil offre la possibilité de corriger manuellement la segmentation des caractères sur la numérisation ainsi que chaque caractère identifié, afin de pouvoir créer un modèle basé sur les typographies de la collection de Bry.

Cette méthode a été testée en dernier car la création d'un modèle personnalisé prend un temps considérable mais n'assure pas pour autant d'améliorer les résultats. En outre, il y a de fortes chances que ce modèle ne soit pas réutilisable pour d'autres collections, étant donné que le logiciel s'entraîne à reconnaitre les typographies spécifiques de cette collection.

Tesseract préconise d'avoir au minimum trois fois chaque caractère dans un jeu d'entraînement, et trois images contenant la plupart des caractères ont alors été sélectionnées pour créer notre jeu de données. Une de ces images comporte une gravure afin que Tesseract apprenne également à ne pas y reconnaître de texte. Finalement, le travail fourni sur ces trois numérisations correspond à une vérification et correction manuelle d'environ 6'200 caractères.

Figure 24 : capture d'écran de QT Box Editor pendant la vérification et correction des caractères

Figure 24 : capture d'écran de QT Box Editor pendant la vérification et correction des caractères



Cette ébauche de modèle a permis d'obtenir des résultats relativement positifs, au vu du peu de données ayant servi à sa création, mais c'est bien sûr insuffisant par rapport aux autres tests. L'idée reste cependant intéressante, et, si le temps permet de traiter quelques images de plus, les résultats pourraient peut-être dépasser ceux des modèles testés auparavant.

Figure 25 : Tesseract – modèle personnalisé – caractères

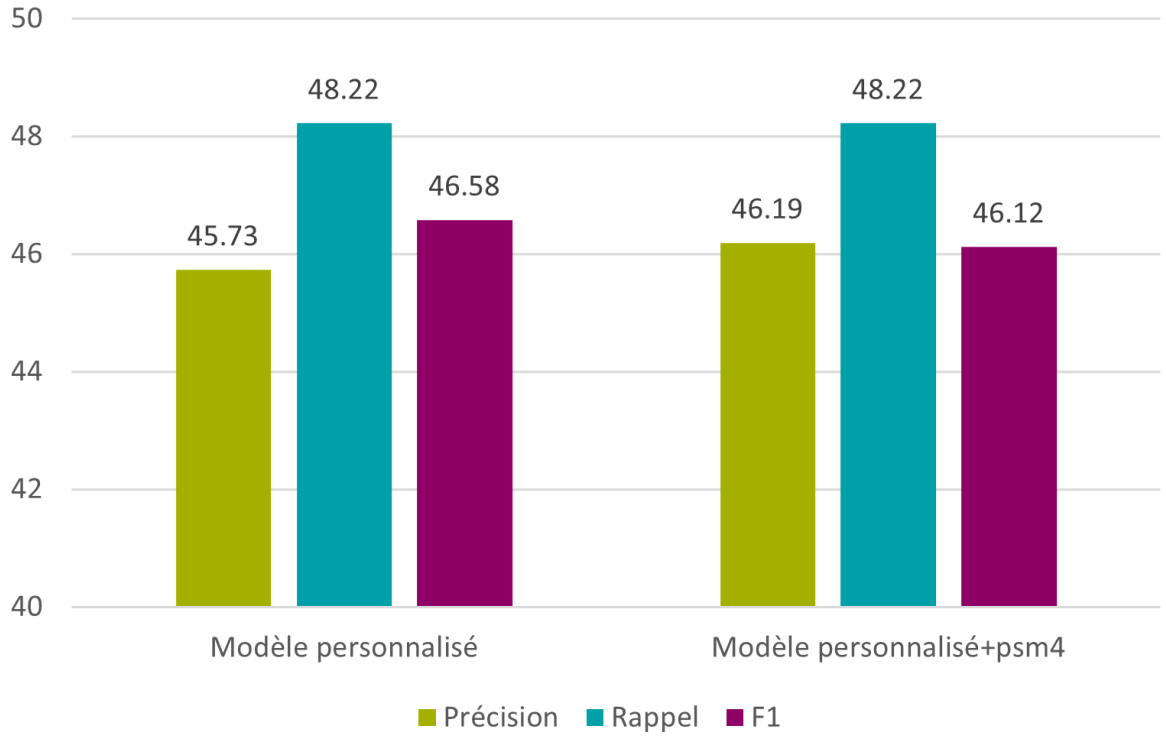
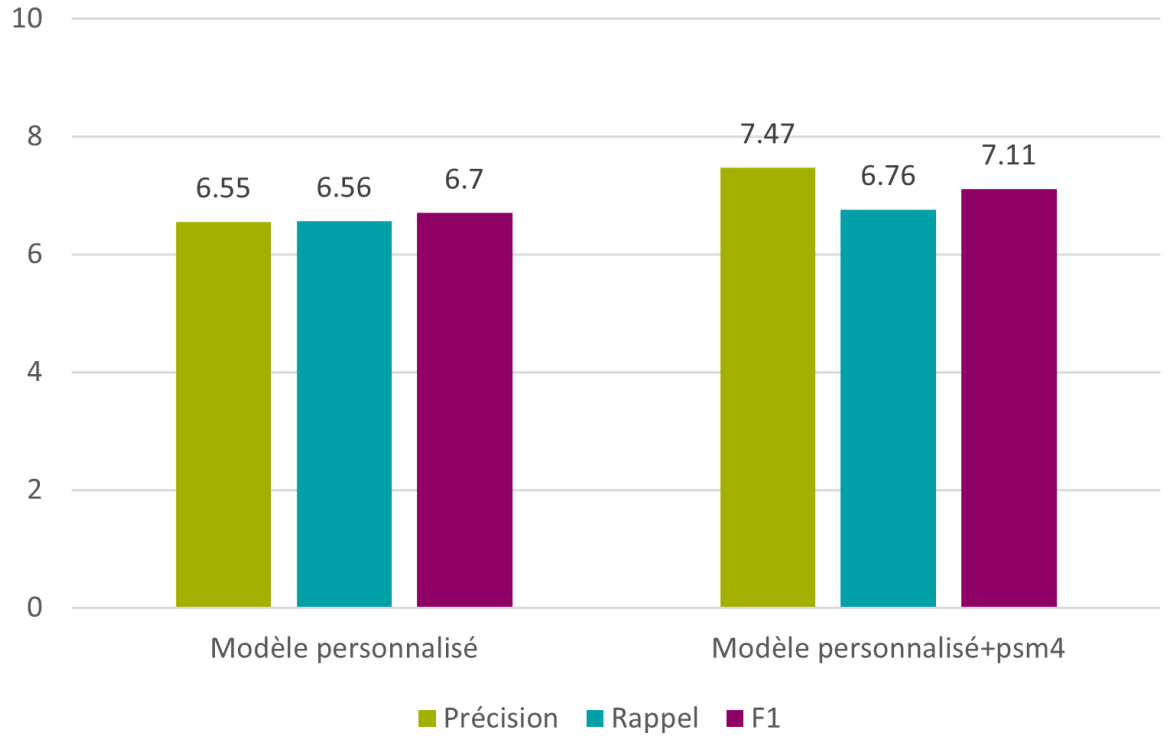


Figure 26 : Tesseract – modèle personnalisé – mots



5.7. Résultats finaux

À la suite de tous ces tests, la F1 maximale obtenue est de 80.06% au niveau des caractères, et de 34.58% au niveau des mots. L'objectif de 95% au niveau des caractères et des mots n'est pas atteint, mais il a tout de même été possible de s'en rapprocher.

6.Conclusion et perspectives futures

Après avoir testé quatre logiciels d'océrisation, dont deux ont très vite posé des problèmes techniques (Kraken et Calamari), deux logiciels présentent de bonnes performances. Tesseract, lors du meilleur test, atteint une F1 de 78.62% au niveau des caractères et de 31.78% au niveau des mots (voir Tesseract – phase 3). OCR4all est également performant, mais présente un problème technique qui le met malheureusement hors course. Il est cependant recommandé de suivre l'évolution du problème technique posé par OCR4all car, s'il est réglé, ce logiciel pourraient alors devenir un excellent choix.

En utilisant différentes méthodes de pré-traitement et de post-correction, il a été possible de faire monter les résultats de Tesseract à une F1 de 80.06% au niveau des caractères et de 34.58% au niveau des mots (voir Résultats finaux). Le chemin est encore long jusqu'au 95%, mais la voie est à présent ouverte pour de futurs essais.

Ce projet était limité dans le temps, et il a été frappant de découvrir la durée nécessaire à ce type de travail, chaque paramètre modifié nécessitant une nouvelle itération et un nouveau temps de calcul. Il est de ce fait compréhensible qu'une technologie aussi ancienne que l'océrisation soit toujours en développement, du fait de sa complexité et de l'immense variété des données qu'elle traite.

Ce projet d'océrisation est un exemple parmi tant d'autres, mais il permet de donner un aperçu des logiciels OCR, des technologies et méthodes de travail qui leur sont liées, du traitement des imprimés anciens, de la complexité de la langue latine etc., et ainsi de mieux comprendre en quoi l'océrisation est un enjeu des Humanités numériques… et des sciences de l'information. Un tel projet nécessite en effet des connaissances et compétences à la fois en sciences humaines et en informatique, et c'est au cœur des Humanités numériques ainsi que des sciences de l'information que l'on peut trouver des profils de chercheurs correspondant aux besoins du domaine.

Bibliographie

AFROGE, Shyla, AHMED, Boshir et MAHMUD, Firoz, 2016. Optical character recognition using back propagation neural network. In : 2nd International Conference on Electrical, Computer Telecommunication Engineering (ICECTE), Rajshahi, 8-10 décembre 2016 [en ligne]. Décembre 2016. pp. 1–4. [Consulté le 28 août 2019]. Disponible à l'adresse : https://ieeexplore.ieee.org/document/7879615

ALGHAMDI, Mansoor et TEAHAN, William, 2017. Experimental evaluation of Arabic OCR systems. PSU Research Review [en ligne]. 28 novembre 2017. Vol. 1, no. 3, pp. 229–241. [Consulté le 28 août 2019]. Disponible à l'adresse : http://www.emeraldinsight.com/doi/10.1108/PRR-05-2017-0026

AMIN SHAYEGAN, Mohammad et AGHABOZORGI, Saeed, 2014. A new method for Arabic/Farsi numeral data set size reduction via modified frequency diagram matching. Kybernetes [en ligne]. 29 avril 2014. Vol. 43, n°5, pp. 817–834. [Consulté le 28 août 2019]. Disponible à l'adresse : http://www.emeraldinsight.com/doi/10.1108/K-10-2013-0226

ANUGRAH, Rio et BINTORO, Ketut Bayu Yogha, 2017. Latin letters recognition using optical character recognition to convert printed media into digital format. Jurnal Elektronika Dan Telekomunikasi [en ligne]. Décembre 2017. Vol. 17, n°2, pp. 56–62. [Consulté le 28 août 2019]. Disponible à l'adresse : https://www.jurnalet.com/jet/article/view/163

ARVINDPDMN [pseudonyme], 2019. Levenshtein distance. Developedia [en ligne]. 3 septembre 2019. Mis à jour le 4 septembre 2019. [Consulté le 11 novembre 2019]. Disponible à l'adresse : https://devopedia.org/levenshtein-distance

BALK, Hildelies et PLOEGER, Lieke, 2009. IMPACT : working together to address the challenges involving mass digitization of historical printed text. OCLC Systems & Services: International digital library perspectives [en ligne]. 30 octobre 2009. Vol. 25, n°4, pp. 233–248. [Consulté le 28 août 2019]. Disponible à l'adresse : https://www.emeraldinsight.com/doi/full/10.1108/10650750911001824

BAO, Ping et ZHU, Suoling, 2014. System design for location name recognition in ancient local chronicles. Library Hi Tech [en ligne]. 10 juin 2014. Vol. 32, n°2, pp. 276–284. [Consulté le 28 août 2019]. Disponible à l'adresse : http://www.emeraldinsight.com/doi/10.1108/LHT-07-2013-0101

BEINERT, Wolfgang 2018. Antiqua. Typolexicon [en ligne]. 1er avril 2018. [Consulté le 6 janvier 2020]. Disponible à l'adresse : https://www.typolexikon.de/antiqua/

BEINERT, Wolfgang 2019. Fraktur. Typolexicon [en ligne]. 1er août 2019. [Consulté le 6 janvier 2020]. Disponible à l'adresse : https://www.typolexikon.de/fraktur-schrift/

BLANKE, Tobias, BRYANT, Michael et HEDGES, Mark, 2012. Open source optical character recognition for historical research. Journal of Documentation [en ligne]. Août 2012. Vol. 68, n°5, pp. 659–683. [Consulté le 28 août 2019]. Disponible à l'adresse : https://www.emeraldinsight.com/doi/full/10.1108/00220411211256021

BODMER LAB, 2019. Bodmer Lab [en ligne]. 2019. Mis à jour le 9 janvier 2020. [Consulté le 9 janvier 2020]. Disponible à l'adresse : https://bodmerlab.unige.ch/fr

BOWERS, Steven K., 2018. Information Technology and Libraries at 50 : The 1990s in Review. Information Technology & Libraries [en ligne]. Décembre 2018. Vol. 37, n°4, pp. 9–14. [Consulté le 28 août 2019]. Disponible à l'adresse : http://search.ebscohost.com/login.aspx?direct=true&db=lih&AN=133718523&site=ehost-live

BRENER, Nathan E., IYENGAR, S. S. et PIANYKH, O. S., 2005. A conclusive methodology for rating OCR performance. Journal of the American Society for Information Science & Technology [en ligne]. Juillet 2005. Vol. 56, n°12, pp. 1274–1287. [Consulté le 28 août 2019]. Disponible à l'adresse : http://search.ebscohost.com/login.aspx?direct=true&db=lih&AN=18172083&site=ehost-live

BREUEL, Thomas M., 2007. Announcing the OCRopus Open Source OCR system. Google developpers [en ligne]. 9 avril 2007. [Consulté le 9 janvier 2020]. Disponible à l'adresse : https://developers.googleblog.com/2007/04/announcing-ocropus-open-source-ocr.html

BURGY, Florence, GERSON, Steeve, SCHÜPBACH, Loïc, 2020a. Ex imagine ad litteras : Projet d'océrisation de la collection de Bry [en ligne]. Genève : Haute école de gestion de Genève. Mémoire de recherche. [Consulté le 25 novembre 2020]. Disponible à l'adresse : https://doc.rero.ch/record/328465?ln=fr

BURGY, Florence, GERSON, Steeve, SCHÜPBACH, Loïc, 2020b. Ex imagine ad litteras : résultats actuels et espoirs futurs. Recherche d'IdéeS [en ligne]. 3 mars 2020. [Consulté le 30 mars 2020]. Disponible à

l'adresse : https://campus.hesge.ch/blog-master-is/ex-imagine-ad-litteras-resultats-actuels-et-espoirs-futurs/

CARRASCO, Rafael C., 2014. An Open-source OCR Evaluation Tool. In : Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage, Madrid, 19-20 mai 2014 [en ligne]. New York : ACM. 2014. pp. 179–184. [Consulté le 28 août 2019]. Disponible à l'adresse :

https://dl.acm.org/citation.cfm?doid=2595188.2595221

CIMON, Lucas, 2019. ImproveQuality. Tesseract Wiki [en ligne]. 25 novembre 2019. [Consulté le 9 janvier 2020]. Disponible à l'adresse : https://github.com/tesseract-ocr/tesseract/wiki/ImproveQuality

CLEMATIDE, Simon, FURRER, Lenz et VOLK, Martin, 2016. Crowdsourcing an OCR Gold Standard for a German and French Heritage Corpus. In : Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016), Portorož, 23-28 mai 2016 [en ligne]. 2016. pp. 975-982. [Consulté le 28 août 2019]. Disponible à l'adresse : https://www.zora.uzh.ch/id/eprint/124786

COJOCARU, Svetlana et al., 2016. Optical Character Recognition Applied to Romanian Printed Texts of the 18th–20th Century. Computer Science Journal of Moldova [en ligne]. 2016. Vol. 24, n°1 (70), pp. 106-117. [Consulté le 28 août 2019]. Disponible à l'adresse : http://www.math.md/files/csjm/v24-n1/v24-n1-(pp106-117).pdf

DASH, Kalyan S., PUHAN, N. B. et PANDA, G., 2017. Odia character recognition : a directional review. The Artificial Intelligence Review [en ligne]. 2017. Vol. 48, n°4, pp. 473–497. [Consulté le 28 août 2019]. Disponible à l'adresse : https://search.proquest.com/lisa/docview/1961506152/abstract/35B11DA70B14444EPQ/2

EPFL, 2020. DHLAB. EPFL.ch [en ligne]. 3 décembre 2020. [Consulté le 3 décembre 2020]. Disponible à l'adresse : https://www.epfl.ch/labs/dhlab/

GHOSH, Kripabandhu et al., 2016. Improving Information Retrieval Performance on OCRed Text in the Absence of Clean Text Ground Truth. Information Processing & Management [en ligne]. 1 septembre 2016. Vol. 52, n°5, pp. 873–884. [Consulté le 28 août 2019]. Disponible à l'adresse : http://www.sciencedirect.com/science/article/pii/S030645731630036X

HLÁDEK, Daniel et al., 2017. Learning string distance with smoothing for OCR spelling correction. Multimedia Tools and Applications [en ligne]. Novembre 2017. Vol. 76, n°22, pp. 24549–24567. [Consulté le 28 août 2019]. Disponible à l'adresse : http://link.springer.com/10.1007/s11042-016-4185-5

IMPACT, 2013. IMPACT Centre of Competence [en ligne]. 2013. Mis à jour le 9 janvier 2020. [Consulté le 9 janvier 2020]. Disponible à l'adresse : https://www.digitisation.eu/

JÄRVELIN, Anni et al., 2016. Information retrieval from historical newspaper collections in highly inflectional languages: a query expansion approach. Journal of the Association for Information Science & Technology [en ligne]. Décembre 2016. Vol. 67, n° 12, pp. 2928–2946. [Consulté le 28 août 2019]. Disponible à l'adresse : http://search.ebscohost.com/login.aspx?direct=true&db=lih&AN=119478036&site=ehost-live

JOST, Clémence, 2019. Lancement d'OCR4all, un outil open source et gratuit de reconnaissance de caractères anciens pour les chercheurs en histoire et les archivistes. Archimag [en ligne]. 24 avril 2019.

[Consulté le 5 septembre 2019]. Disponible à l'adresse : https://www.archimag.com/archives-patrimoine/2019/04/24/ocr4all-open-source-gratuit-reconnaissance-caracteres-anciens

KANN, Bettina et HINTERSONNLEITNER, Michael, 2015. Volltextsuche in historischen Texten. Bibliothek Forschung und Praxis [en ligne]. Avril 2015. Vol. 39, n° 1, pp. 73–79. [Consulté le 28 août 2019]. Disponible à l'adresse : https://www.degruyter.com/downloadpdf/j/bfup.2015.39.issue-1/bfp-2015-0004/bfp-2015-0004.pdf

KARPINSKI, R., LOHANI, D. et BELAÏD, A., 2018. Metrics for Complete Evaluation of OCR Performance. In : IPCV'18 - The 22nd Int'l Conf on Image Processing, Computer Vision, & Pattern Recognition, Las Vegas, juillet 2018 [en ligne]. 2018. pp. 23-29. [Consulté le 28 août 2019]. Disponible à l'adresse : https://csce.ucmss.com/cr/books/2018/LFS/CSREA2018/IPC3481.pdf

KESTEMONT, Mike, CHRISTLEIN, Vincent et STUTZMANN, Dominique, 2017. Artificial Paleography: Computational Approaches to Identifying Script Types in Medieval Manuscripts. Speculum [en ligne]. 2 octobre 2017. Vol. 92, S1, pp. S86–S109. [Consulté le 28 août 2019]. Disponible à l'adresse : https://www.journals.uchicago.edu/doi/10.1086/694112

KISSOS, Ido et DERSHOWITZ, Nachum, 2016. OCR Error Correction Using Character Correction and Feature-Based Word Classification. In : 12th IAPR Workshop on Document Analysis Systems (DAS), Santorini, 11-14 avril 2016 [en ligne]. Avril 2016. pp. 198–203. [Consulté le 28 août 2019]. Disponible à l'adresse : http://ieeexplore.ieee.org/document/7490117/

KUMAR, Munish et al., 2018. Character and numeral recognition for non-Indic and Indic scripts : a survey. The Artificial Intelligence Review [en ligne]. 2018. pp. 1–27. [Consulté le 28 août 2019]. Disponible à l'adresse : https://search.proquest.com/lisa/docview/1984338483/abstract/35B11DA70B14444EPQ/1

MEI, Jie et al., 2018. Statistical learning for OCR error correction. Information Processing & Management [en ligne]. 1 novembre 2018. Vol. 54, n° 6, pp. 874–887. [Consulté le 28 août 2019]. Disponible à l'adresse : http://www.sciencedirect.com/science/article/pii/S0306457317307823

MORI, Shunji, SUEN, Ching Y. et YAMAMOTO, Kazuhiko, 1992. Historical review of OCR research and development. Proceedings of the IEEE [en ligne]. Juillet 1992. Vol. 80, n° 7, pp. 1029-1058. [Consulté le 18 décembre 2019]. Disponible à l'adresse : http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=156468&isnumber=4050

MOUNIER, Pierre, 2018. Les humanités numériques : une histoire critique [en ligne]. Paris : Éditions de la Maison des sciences de l'homme. Interventions. [Consulte le 3 décembre 2020]. Disponible à l'adresse : https://books.openedition.org/editionsmsh/12006

MUEHLBERGER, Guenter et al., 2019. Transforming scholarship in the archives through handwritten text recognition: Transkribus as a case study. Journal of Documentation [en ligne]. 24 juillet 2019. [Consulté le 28 août 2019]. Disponible à l'adresse : https://www.emeraldinsight.com/doi/10.1108/JD-07-2018-0114

NAGY, George, 2016. Disruptive developments in document recognition. Pattern Recognition Letters [en ligne]. 1 août 2016. Vol. 79, pp. 106–112. [Consulté le 28 août 2019]. Disponible à l'adresse : http://www.sciencedirect.com/science/article/pii/S0167865515004109

PTUCHA, Raymond, et al., 2019. Intelligent character recognition using fully convolutional neural networks. Pattern Recognition [en ligne]. Avril 2019. Vol. 88, pp. 604–613. [Consulté le 28 août 2019]. Disponible à l'adresse : http://www.sciencedirect.com/science/article/pii/S0031320318304370

REDDY, Sravana et CRANE, Gregory, 2006. A Document Recognition System for Early Modern Latin. In: Chicago Colloquium on Digital Humanities and Computer Science: What Do You Do With A Million Books [en ligne]. 2006. Vol. 23, pp. 1-4. [Consulté le 28 août 2019]. Disponible à l'adresse : https://dl.tufts.edu/concern/pdfs/kd17d4036

REHMAN, Amjad et SABA, Tanzila, 2014. Neural networks for document image preprocessing: state of the art. The Artificial Intelligence Review [en ligne]. 2014. Vol. 42, n° 2, pp. 253–273. [Consulté le 28 août 2019]. Disponible à l'adresse : https://search.proquest.com/lisa/docview/1542796407/abstract/EC9B8EB6A5EF463APQ/41

RICE, Stephen V., JENKINS, Frank R. et NARTKER, Thomas A., 1996. The Fifth Annual Test of OCR Accuracy. Information Science Research Institute [en ligne]. 1996. pp. 1-46. [Consulté le 28 août 2019]. Disponible à l'adresse : http://stephenvrice.com/images/AT-1996.pdf

REUL, Christian, 2020. @chreul. thx for the hint and sorry […]. line segmentation hangs on empty pages · Issue #45 [en ligne]. 13 janvier 2020. [Consulté le 14 janvier 2020]. Disponible à l'adresse : https://github.com/OCR4all/OCR4all/issues/45

RYDBERG-COX, Jeffrey A., 2003. Automatic Disambiguation of Latin Abbreviations in Early Modern Texts for Humanities Digital Libraries. In : Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital Libraries, Houston, 27-31 mai 2003 [en ligne]. Washington DC: IEEE Computer Society. 2003. pp. 372–373. [Consulté le 28 août 2019]. Disponible à l'adresse : http://dl.acm.org/citation.cfm?id=827140.827207

SABER, Shimaa et al., 2016. Performance Evaluation of Arabic Optical Character Recognition Engines for Noisy Inputs. In : Gaber T., Hassanien A., El-Bendary N. et Dey N. The 1st International Conference on Advanced Intelligent System and Informatics (AISI2015), Beni Suef, 28-30 novembre 2015. Cham : Springer, pp. 449-459. [Consulté le 28 août 2019]. Advances in Intelligent Systems and Computing, 407. Disponible à l'adresse : https://link.springer.com/chapter/10.1007/978-3-319-26690-9_40

SASAKI, Yutaka, 2007. The truth oft he F-measure. Teach Tutor mater [en ligne]. 26 Octobre 2007. Vol. 1, n° 5, pp. 1-5. [Consulté le 8 décembre 2019]. Disponible à l'adresse : https://www.researchgate.net/publication/268185911_The_truth_of_the_F-measure

SCHANTZ, Herbert F., 1982. The history of OCR, optical character recognition [en ligne]. Manchester Center, Vt. : Recognition Technologies Users Association. [Consulté le 18 décembre 2019]. Disponible à l'adresse : https://archive.org/details/historyofocropti0000scha

SEIDMAN, Max J., et al., 2016. Are games a viable solution to crowdsourcing improvements to faulty OCR ? - The Purposeful Gaming and BHL experience. Code4Lib Journal [en ligne]. Juillet 2016. Vol. 33, p. 1. [Consulté le 28 août 2019]. Disponible à l'adresse : http://search.ebscohost.com/login.aspx?direct=true&db=lih&AN=116963678&site=ehost-live

SMITH, Ray, 2007. An overview of the Tesseract OCR engine. In : Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), Curitiba, Paraná, Brésil, 23-26 septembre 2007 [en ligne]. Septembre 2007. Vol. 2, pp. 629-633. [Consulté le 26 octobre 2019]. Disponible à l'adresse : https://ieeexplore.ieee.org/document/4376991?arnumber=4376991

STUTZMANN, Dominique, 2017. Paléographie : la révolution numérique. L'Histoire [en ligne]. Septembre 2017. Vol. 439, p. 30. [Consulté le 28 août 2019]. Disponible à l'adresse : https://www.lhistoire.fr/irht-dans-le-secret-des-manuscrits/paléographie-la-révolution-numérique

SUN, Wei et al., 1992. Intelligent OCR Processing. Journal of the American Society for Information Science [en ligne]. Juillet 1992. Vol. 43, n°6, pp. 422–431. [Consulté le 28 août 2019]. Disponible à l'adresse : http://search.ebscohost.com/login.aspx?direct=true&db=lih&AN=16918942&site=ehost-live

TERRAS, Melissa, 2016. A Decade in Digital Humanities. Journal of Siberian Federal University [en ligne]. 2016. Vol. 9, pp.1637-1650. [Consulté le 3 décembre 2020]. Disponible à l'adresse: https://www.researchgate.net/publication/309217683_A_Decade_in_Digital_Humanities

THERAYSMITH [pseudonyme], 2017. The text corpus is from *all* the www, […]. Q&A : Indic - length of the compressed codes · Issue #654 [en ligne]. 23 janvier 2017. [Consulté le 22 décembre 2019]. Disponible à l'adresse : https://github.com/tesseract-ocr/tesseract/issues/654#issuecomment-274574951

TUMBE, Chinmay, 2019. Corpus linguistics, newspaper archives and historical research methods. Journal of Management History [en ligne]. 30 mai 2019. [Consulté le 28 août 2019]. Disponible à l'adresse : https://www.emeraldinsight.com/doi/10.1108/JMH-01-2018-0009

UNIVERSITÉ DE GENÈVE, 2020. Les missions de la chaire – Humanités numériques. Unige.ch [en ligne]. 3 décembre 2020. [Consulté le 3 décembre 2020]. Disponible à l'adresse : https://www.unige.ch/lettres/humanites-numeriques/fr/la-chaire/les-missions-de-la-chaire/

URIELI, Assaf et VERGEZ-COURET, Marianne, 2013. Jochre, océrisation par apprentissage automatique : étude comparée sur le yiddish et l'occitan. In : TALARE 2013 : Traitement automatique des langues régionales de France et d'Europe, Les Sables d'Olonne, juin 2013 [en ligne]. 21 juin 2013. pp. 221-234. [Consulté le 28 août 2019]. Disponible à l'adresse : https://hal-univ-tlse2.archives-ouvertes.fr/hal-00979665

VOBL, Thorsten et al., 2014. PoCoTo - an Open Source System for Efficient Interactive Postcorrection of OCRed Historical Texts. In: Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage, Madrid, 19-20 mai 2014 [en ligne]. New York : ACM. 2014. pp. 57–61. [Consulté le 28 août 2019]. Disponible à l'adresse : https://dl.acm.org/citation.cfm?id=2595197

WICK, Christoph, REUL, Christian et PUPPE, Frank, 2018. Calamari - A High-Performance Tensorflow-based Deep Learning Package for Optical Character Recognition. [En ligne]. Preprint. 6 août 2018. [Consulté le 09 janvier 2020]. Disponible à l'adresse : https://arxiv.org/abs/1807.02004

ZHOU, Yongli, 2010. Are Your Digital Documents Web Friendly? : Making Scanned Documents Web Accessible. Information Technology & Libraries [en ligne]. Septembre 2010. Vol. 29, n°3, pp. 151–160. [Consulté le 28 août 2019]. Disponible à l'adresse : http://search.ebscohost.com/login.aspx?direct=true&db=lih&AN=52871764&site=ehost-live